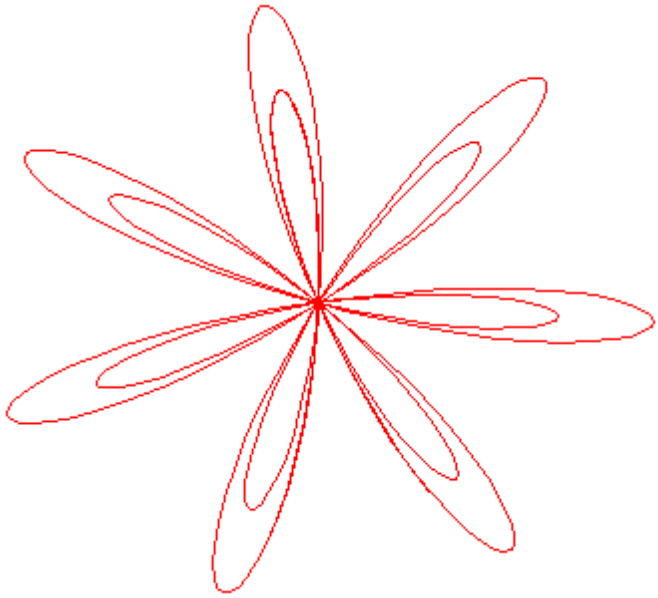
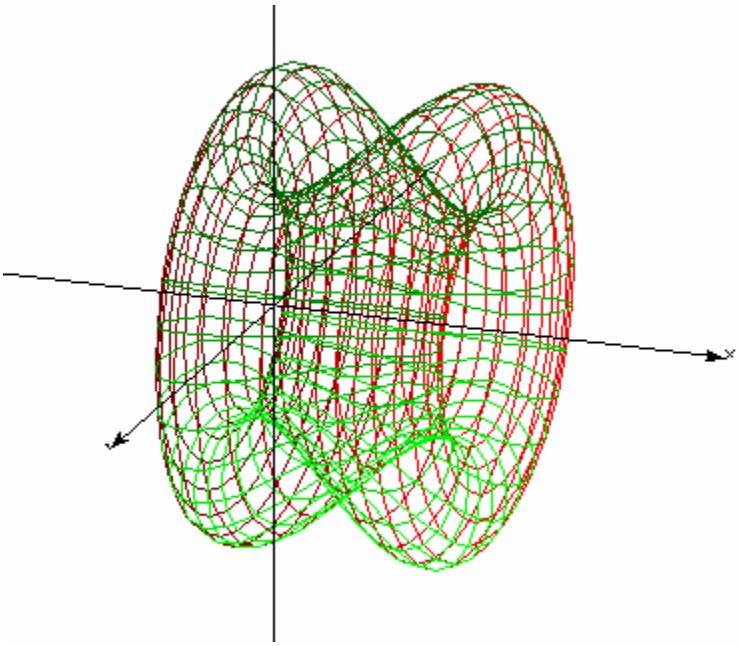


Numerieke wiskunde met Python

v 150923



De Meester André

Inhoudsopgave

Numerieke wiskunde – Python

1.Inleiding

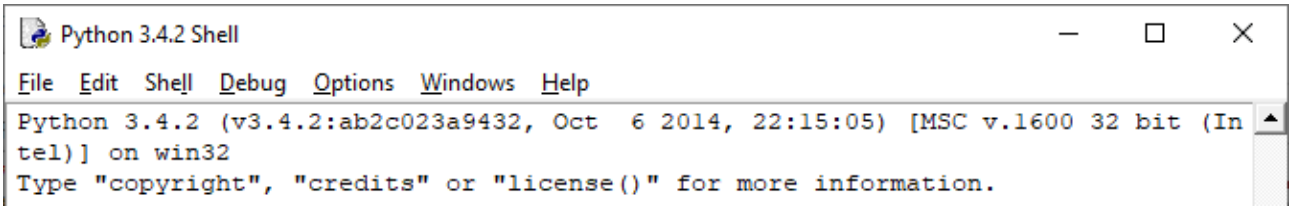
Waarom zou je computerprogramma's maken voor wiskunde? Welk nut heeft dat? Reeds in de oudheid werden er methodes gebruikt om “bijzondere getallen” te berekenen. Een voorbeeld is de methode van Heroon om de vierkantswortel te benaderen. Deze methodes vergden vroeger veel manueel rekenwerk. Nu doen computer of rekentoestel het rekenwerk en is de “numerieke wiskunde” veel interessanter geworden. Bovendien kan voor bijna elk onderdeel van het wiskunde-onderwijs een geschikt computerprogramma gemaakt worden. Deze handleiding behandelt een groot aantal van deze programma's. Op de PC gebruiken we de PYTHON-taal. (Python 3). Python is een computertaal die zeer geschikt is voor wiskundige toepassingen. Ze heeft een eenvoudig aan te leren syntax en ze laat toe om gestructureerde en goed leesbare programma's te schrijven. Je zult merken dat er voor elk deelprobleem een oplossing of 'procedure' gezocht en gedefinieerd wordt. Deze kan dan ook in andere programma's gebruikt worden.

Behalve deze handleiding, moet je ook een gecomprimeerde map met alle beschreven programma's downloaden. Bij elk onderdeel staat de naam van het bijhorend programma in het blauw afgedrukt.

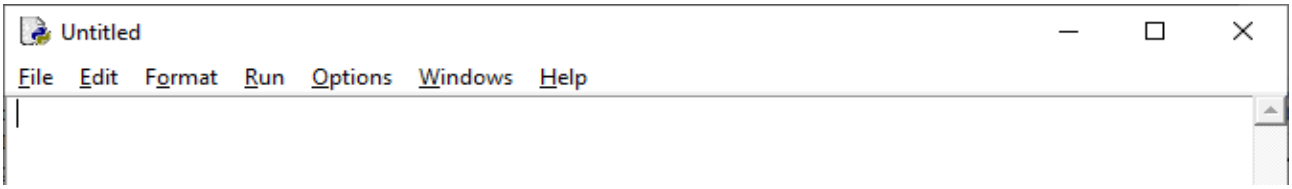
2. Berekenen van de schuine zijde in een rechthoekige driehoek pythagoras

In een rechthoekige driehoek ABC geldt de stelling van Pythagoras: $a^2=b^2+c^2$

Als je Python opstart, kom je in de Shell terecht ,dwz. het scherm waarin je programma's uitgevoerd worden, of waarin je ook rechtstreeks enkelvoudige Python- instructies geeft.



Om een programma te maken moet je naar de editor,
In de menu-balk kies je File - New File

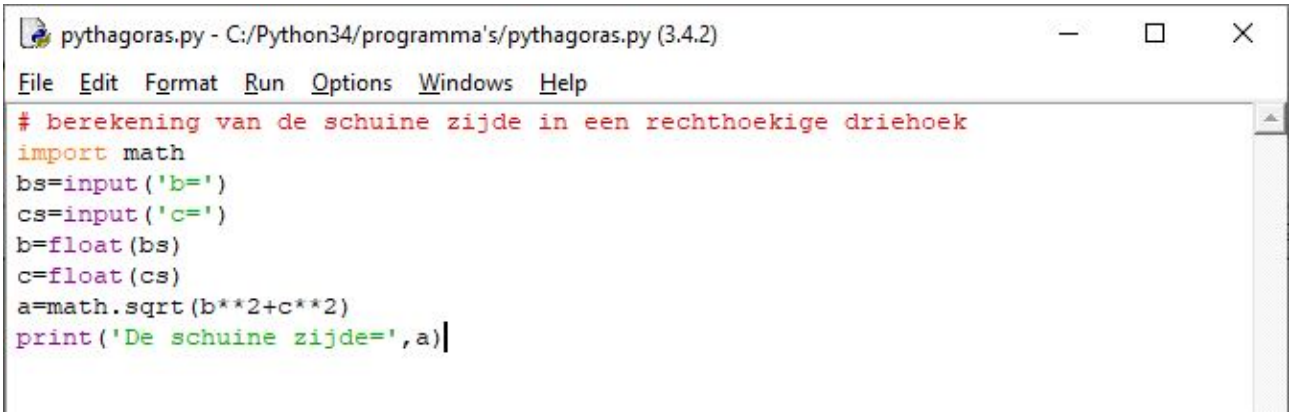


Nu ben je in de editor waarin je het programma kunt invoeren.
Tik nu:

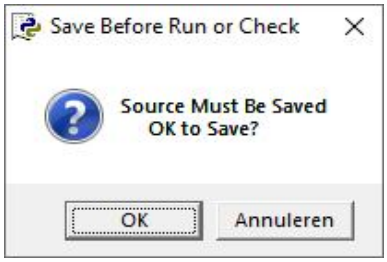
```
# berekening van de schuine zijde in een rechthoekige driehoek
import math
bs=input('b=')
cs=input('c=')
b=float(bs)
c=float(cs)
a=math.sqrt(b**2+c**2)
print('De schuine zijde=',a)
```

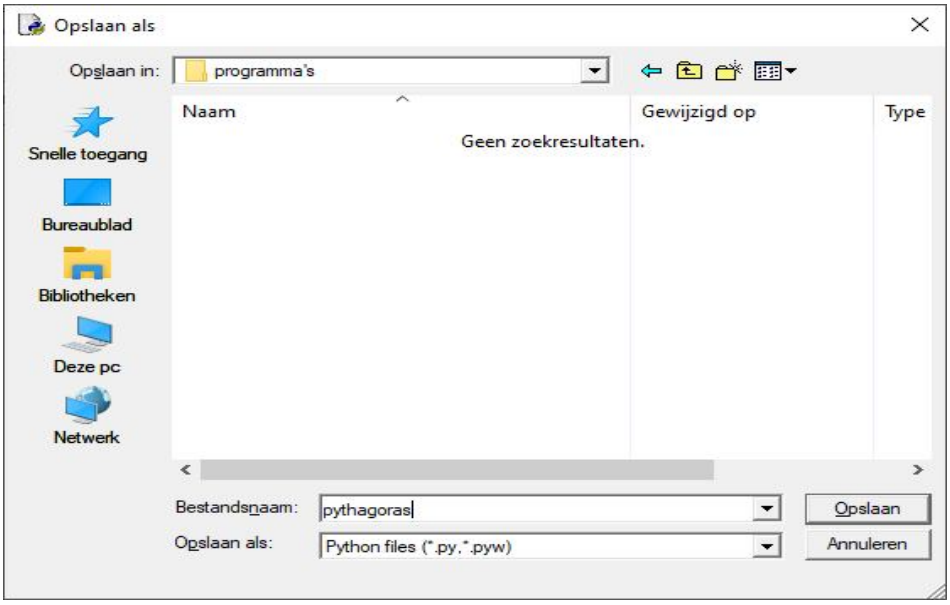
We overlopen de programma- listing:
 # berekenen.... : elke lijn die begint met # :commentaar die de programmeur overal mag toevoegen ter verduidelijking van het programma.
import math :de module math wordt geïmporteerd zodat je gebruik kunt maken van wiskundige functies.
bs=input('b=') : op het scherm komt **b= .** De computer wacht tot de gebruiker iets invult,.
Als je bvb. 12 invult, dan is bs de string '12' (het woordje 12)
b=float(bs) : float(bs) zet '12' om naar het getal 12. b wordt dus 12.

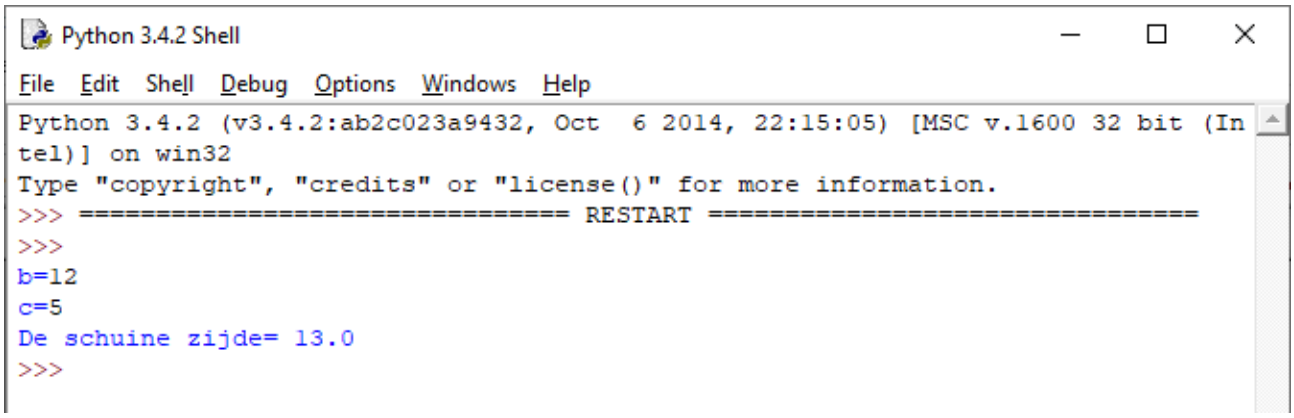
a=math.sqrt(b2+c**2)** : dit is pythontaal voor $a = \sqrt{b^2 + c^2}$
sqrt =square root = vierkantswortel.



Om dit programma uit te voeren, kies Run – Run Module, je krijgt dan een dialoogvenster.
Klik OK, en je krijgt gelegenheid om het programma op te slaan, doe dit in de gepaste directory.







3. Python taal

Dit is geen zuivere cursus programmeren. Het gaat hier veeleer over het numeriek oplossen van wiskundige problemen, vooral in die gevallen waarin het manuele rekenwerk hopeloos lang duurt of ook voor het nauwkeurig tekenen van de grafiek van wiskundige functies. In deze notities zijn er problemen opgenomen uit de getallenleer, oplossen van vergelijkingen, ontbinden in factoren, driehoeksmeting, goniometrie, statistiek, afgeleiden en integralen, vlakke en ruimtemeetkunde. Omdat we deze problemen willen omzetten naar een computerprogramma, is een minimale kennis van de taal wel nodig.

Enkele taalelementen:

Bewerkingen: + – * / de 4 hoofdbewerkingen

** machtsverheffing % modulo // geheel quotiënt

Bewerkingen in voorwaarden: > < >= <= == is gelijk aan != is verschillend

Er zijn 4 basistypes die we veel zullen gebruiken:
(er zijn nog andere types die we hier niet nodig hebben)

1. **string** (str) woord of zin vb. 'Geef een functie:' "x="
2. **geheel getal** (int) vb. 45 -130
3. **reëel getal** (float) vb. 2.3541
4. **lijst** (list) vb. [1,2,3] ['appel,'peer','kers']

Omzetten naar een ander type: str(..) int(..) float(..)

Elke letter of woord kan je in principe gebruiken als variabele van één van deze 4 types. In deze notities worden steeds kleine letters gebruikt alhoewel dit geen noodzaak is.

Een lege lijst kan je initialiseren met lijstvb=[] of lijstvb=list()
Aanvullen van de lijst kan je met lijstvb.append(..)
Je kan elk van de 4 types toevoegen.
Matrices bijvoorbeeld kan je definiëren als lijsten van lijsten. Noemt zo een matrix a, dan kan je het element op de i-de rij en de j-de kolom
aanspreken met a[i][j]

Uitvoer van de 4 types :

```
ant=2
print('Het antwoord is ',ant)
```

Dit is de meest elementaire uitvoerinstructie. Als je dit programma uitvoert, krijg je:

Het antwoord is 2

Invoer:
Bij invoer -input- kunnen enkel strings ingevoerd worden. Deze kunnen met int of float omgezet worden naar een geheel of reëel getal.
Voorbeelden:

```
xs =input('Geef een getal:')
x  =int(xs)                          of korter

x=int(input('Geef een getal:')
```

Structuur:
Er mogen meerdere instructies op één lijn, gescheiden door **;**

Een programma kan opgedeeld worden in deeltjes:
Er zijn 5 basisinstructies (deelprogramma's)

1. een functie definitie,

```
def funvb :
    .......
    return ..
```

2. een voorwaardelijke instructie

```
if..voorwaarde :
    .........
elif..voorwaarde :
    .........
else :
    ........
```

Drie soorten loops

3. for .. in range(..,..) :
    ....
4. for .. in (..,..) :
    ....
5. while..voorwaarde..... :
    ....

Let vooral op de structuur die steeds gelijkaardig is:
na de voorwaarde of instructie of definitie tik je een **:** <Enter>
De volgende lijn springt een welbepaald stukje in (indent).
Zorg er voor dat alle lijnen waarop het deelprogramma toepasselijk is, evenveel inspringen. Vanaf dat je de indent ongedaan maakt, is dit deelprogramma afgelopen.
In het volgende programmaatje zit er slechts één instructie in de for-loop, nl. dlg.append()

Voorbeeld van het opstellen van een lijst:

```
dig=[]
for i in range(0,10):
      dig.append(i)
print(dig)
print(dig[3:5])
```

```
>>>
[0, 1, 2, 3, 4, 5, 6, 7, 8, 9]
[3, 4]
>>>
```

Aan een functiedefinitie kan je elk van de 4 types variabelen als parameter doorgeven of van de functie terugkrijgen. Ben je een stuk van het programma vergeten, geen probleem, de editor is een goede tekstverwerker. Je kunt op elk moment overal wijzigingen aanbrengen. Je mag het programma ook met een andere eenvoudige tekstverwerker invoeren en nadien kopiëren en plakken in de python- editor. Let dan wel op dat de indents correct zijn.

Getallen kunnen afgerond worden tot een aantal cijfers na de komma
vb.

```
>>>print(round(3.0455,3))
3.046
>>>
```

Ze kunnen ook omgezet worden naar een string met een vast aantal cijfers na de komma:

```
>>>print(format(3.0455,'.3f'))
3.046
>>>
```

De 2 resultaten lijken maar zijn niet dezelfde: round rond af maar blijft een getal.
format rond ook af maar zet om naar een string met verplicht 3 cijfers na de komma: interessant voor tabellen.

```
>>>print(round(2.0004,3))
2.0
>>>print(format(2.0004,'.3f'))
2.000
>>>
```

Om wiskundige functies te gebruiken moet de module **math** geïmporteerd worden.Aan het begin van het programma moet er dan komen **import math**
Alle wiskundige functies die je gebruikt, moeten dan voorafgegaan worden door **math.**
Enkele veelgebruikte functies zijn
exp (e**) sin cos tan acos (bgcos) asin (bgsin) atan (bgtan) sqrt (vkw) log (ln) log10 (log) fabs (abs)
radians (decim graden ==> radialen) degrees (radialen ==> decim graden vb.

```
import math
hdeg=float(input('Geef een hoek'))
hrad=math.radians(hdeg)
si=math.sin(hrad)
print('De sinus is ',si,sep='')
```

Beter nog is:

```
from math import *
hdeg=float(input('Geef een hoek:'))
print('De sinus is ',sin(radians(hdeg)),sep=' ')
```

```
>>>
Geef een hoek:30
De sinus is 0.49999999999999994
>>>
```

Met 'from math import *' moet je elke functie niet laten voorafgaan door **math.** sep in een print-instructie betekent seperator.

Grafieken

Om grafieken te tekenen, zullen we eerst de module **graphics** (Zelle) gebruiken. (Je kan ook met de module **tkinter** werken: zie later)
Aan het begin van het programma tik je: **from graphics import ***

Lukt dit niet, dan moet je graphics nog installeren
Tik in de path – balk van de verkenner **cmd** <enter> (command)
Je komt dan in een zwart venster waarin je Msdos commando's kunt intikken
Begeef je eerst naar de juiste directory met cd (change directory):tik
cd c:\python34\scripts <enter>
en tik dan
pip install graphics.py <enter>
of
pip3 install --user http://bit.ly/csc161graphics <enter>
(zie hiervoor eventueel ook op het internet)

Je kan ook meteen de module **tabulate** installeren. Die module gebruiken we voor financiële tabellen en ook voor matrices.
pip install tabulate <enter>

Nog een opmerking: in de programma's is er geen beveiliging tegen foutieve invoer: doe je dit wel, dan crasht het programma, dwz. je krijgt een foutmelding. Dus, als een programma crasht, gewoon terug runnen.

Oefening: afstand tussen 2 punten

De afstand tussen 2 punten $P(x_1,y_1)$ en $Q(x_2,y_2)$ wordt gegeven door de formule:

$$d(P,Q)=\sqrt{(x_2-x_1)^2+(y_1-y_2)^2}$$

Maak een programma die vraagt om de coördinaten van 2 punten in te voeren en die de afstand tussen deze punten berekent en afdrukt.
Gebruik bvb. x1=float(input('x1='))
Het programma drukt dan:
De afstand tussen P en Q =.....
Gebruik hier ook de functie math.sqrt

4. 1 Grootste gemene deler en kleinste gemeen veelvoud van 2 getallen
ggd_kgv breukvereenvoudigen
De ggd is het product van alle gemeenschappelijke factoren, elk met de kleinst voorkomende exponent. Het kgv is het product van alle factoren, elk met de grootst voorkomende exponent.
Vb; a= 395136=2^7. 3^2. 7^3 b= 190512 = 2^4.3^5.7^2
dan is ggd = 2^4. 3^2. 7^2 = 7056 en kgv= 2^7. 3^5. 7^3 = 10668672
Merk op dat steeds geldt: a.b =ggd. kgv
Je kan de ggd ook berekenen door het grootste getal te delen door het kleinste:
395136/ 190512 = 2 met rest 14112

Dan vervangen we het grootste door het kleinste en het kleinste door de rest :
a wordt 190512 en b wordt 14112
Nu doen we terug hetzelfde
190512/ 14112 = 13 met rest 7056
a wordt 14112 en b wordt 7056
14112/ 7056 = 2 met rest 0
Als de rest 0 is, is de deler van de laatste deling de ggd van de oorspronkelijke getallen. Deze eigenschap kan gemakkelijk bewezen worden: een gemeenschappelijke deler van deeltal en deler is ook een gemeenschappelijke deler van deler en rest. Dit geeft ons een eenvoudig algoritme om de ggd te bepalen. En het kgv vinden we door het product van beide getallen te delen door de ggd.
In Python bestaan eenvoudige functies om de rest en het quotiënt van de deling van 2 gehele getallen te bepalen, nl. % = modulo (rest) en // = div (quotiënt) .

Vb: 17 % 5 = 2 en 17//5 =3

Programma:

```
# ggd en kgv ( % = mod, // = div )
a=int(input('Geef a='));b=int(input('Geef b='))
p=a*b;c=1
while c != 0:
    c=a%b;a=b;b=c
print('ggd=',a,'kgv=',p//a)
>>>
Geef a=864
Geef b=324
ggd= 108 kgv= 2592
>>>
```

4.2 **Herleiden naar onvereenvoudigbare breuken**

Het volgende programma vraagt om 2 gehele getallen a en b in te voeren. Het programma moet de breuk a/b vereenvoudigen tot een onvereenvoudigbare breuk door teller a en noemer b te delen door de ggd van teller en noemer.

Programma:

```
# vereenvoudigen van breuken
def ggd(a,b):
    c=1
    while c != 0:
        c=a%b;a=b;b=c
    return a
a=int(input('Geef a='));b=int(input('Geef b='))
deler=ggd(a,b)
print(a,'/', b,"=",a//deler,'/',b//deler,sep='')
>>>
Geef a=232848
Geef b=548856
232848/548856=14/33
>>>
```

5. Getallen van Fibonacci fibonacci

De getallenrij van Fibonacci is zeer eenvoudig op te stellen, de eerste 2 getallen zijn allebei =1
f_0=1 f_1=1 en elk van de volgende getallen is de som van de 2 vorige:
$f_n = f_{n-1} + f_{n-2}$
f_2 =2 f_3 =3 f_4 =5 f_5 =8 f_6 =13 ....
Merkwaardig is nu dat de verhouding van 2 opeenvolgende getallen nadert tot een welbepaald irrationaal getal.

Programma:

```
# fibonacci
a=1;b=1;i=1
while i<10:
    i=i+1;c=a+b;a=b;b=c
    print('f(',i,')=',c,'  ',c/a)
```

```
>>>
f( 2 )= 2    2.0
f( 3 )= 3    1.5
f( 4 )= 5    1.6666666666666667
f( 5 )= 8    1.6
f( 6 )= 13    1.625
f( 7 )= 21    1.6153846153846154
f( 8 )= 34    1.619047619047619
f( 9 )= 55    1.6176470588235294
f( 10 )= 89    1.6181818181818182
>>>
```

Ogenschijnlijk nadert de verhouding $\frac{f_n}{f_{n-1}}$ tot een zeker getal t

dwz. voor grote waarden van n is $f_n \cong t.f_{n-1}$ en $f_{n-1} \cong t.f_{n-2}$

We vervangen in de formule $f_n = f_{n-1} + f_{n-2}$

$t.t.f_{n-2} \cong t.f_{n-2} + f_{n-2}$

Hieruit volgt, als we delen door f_{n-2}, dat in de limiet $t^2 = t+1$.
t is dus de positieve oplossing van de vierkantsvergelijking $t^2-t-1=0$.

$$t = \frac{-b+\sqrt{d}}{2a} = \frac{1+\sqrt{5}}{2} = 1{,}6180239...$$

6. Oplossen van vierkantsvergelijkingen in IR vierkantsvergel_reeel

Het volgende programma vraagt om 3 x de coëfficiënten van een vierkantsvergelijking $ax^2+bx+c=0$ in te voeren. Telkens worden de oplossingen afgedrukt.
Let op de if..elif..else constructie in het geval dat d>,= of <0 is.
De wortelformules zijn: $d=b^2-4ac$

als d>0: $x_1 = \frac{-b-\sqrt{d}}{2a}$ en $x_1 = \frac{-b+\sqrt{d}}{2a}$

als d=0: $x_1 = x_2 = \frac{-b}{2a}$

als d<0 Geen oplossingen

In het programma is een while-loop ingebouwd die met behulp van een teller (1-2-3) ons 3 x vraagt om de coëfficiënten a,b,c in te voeren. Nadien wordt de oplossing afgedrukt.

Programma:

```
# oplossen van vierkantsvergelijkingen in R
import math
print('Oplossen van de vierkantsvergelijking ax²+bx+c=0 in R')
teller=0
while teller<3:
    teller=teller+1
    print('Vierkantsvergelijking',teller)
    a=float(input('a='))
    b=float(input('b='))
    c=float(input('c='))
    d=b*b-4*a*c
    if d>0:
        root=math.sqrt(d)
        x1=(-b-root)/2/a
        x2=(-b+root)/2/a
        print('x1=',x1,'  x2=',x2)
    elif d==0:
        x1=x2=-b/2/a
        print('x1= x2=',x1)
    else:
        print('Geen oplossingen')
```

```
>>>
Oplossen van de vierkantsvergelijking ax²+bx+c=0 in R
Vierkantsvergelijking 1
a=2
b=-7
c=3
x1= 0.5   x2= 3.0
Vierkantsvergelijking 2
a=1
b=-4
c=4
x1= x2= 2.0
Vierkantsvergelijking 3
a=5
b=3
c=11
Geen oplossingen
>>>
```

Het is hier wel een beetje jammer dat we de getallen één voor één moeten invoeren. Daar is een mouw aan te passen, die het programma iets gecompliceerder maakt, maar die aan de gebruiker een groter invulcomfort geeft. We komen hier later op terug.

7. Combinatieleer
permutaties variaties combinaties

Met de volgende programma's kan je permutaties, variaties en combinaties berekenen. Merk het recursieve voorschrift op in de procedures van permutaties.

7.1 Permutaties n!

Formule: n! = n.(n-1).(n-2).....1

Let op het recursieve voorschrift:
vanuit de functie fac wordt fac zelf opgeroepen.
Als n>1, dan is fac(n)=n.fac(n-1), anders, dus als n=1 dan is fac(n)=1

Programma:

```
# permutaties
def fac(n):
    if n==1 or n==0:
        return 1
    else:
        return n*fac(n-1)
p=int(input('Geef een getal:'))
print(p,'!=',fac(p))
```

```
>>>
Geef een getal:12
12 != 479001600
>>>
```

7.2 Variaties: programma

```
# variaties
print('Berekenen van V(n,p)')
n=int(input('Geef n:'));p=int(input('Geef p:'))
v=1
for x in range(n-p+1,n+1):
    v=v*x
print('V(',n,',',p,')=',v)
```

```
>>>
Berekenen van V(n,p)
Geef n:11
Geef p:5
V( 11 , 5 )= 55440
>>>
```

7.3 Combinaties:

Programma:

```
# combinaties ( formule van pascal)
def comb(n,p):
    c=1;n=int(n);p=int(p)
    for i in range(0,p):c=c*(n-i)//(i+1)
    return c
print('Berekenen van C(n,p)')
n=int(input('Geef n:'));p=int(input('Geef p:'))
print('C(',n,',',p,',)=',comb(n,p))

>>>
Berekenen van C(n,p)
Geef n:15
Geef p:6
C( 15 , 6 )= 5005
>>>
```

8. Priemgetallen berekenen priemgetallen

Dit is een kort maar pittig programmaatje. We willen een lijst priem maken waarin alle priemgetallen opgeslagen worden.
We beginnen met priem=[2,3]. Dus priem[0]=2 en priem[1]=3.
Stel dat we een rij priemgetallen priem[0]....priem[i] gevonden hebben.
Dan tellen we telkens 2 op vanaf priem[i] : a=priem[i]+2..? +2..? en onderzoeken deelbaarheid door alle vorige priemen waarvan het kwadraat niet groter is dan a.
Waarom hoeven we niet verder te zoeken?: omdat bij deelbaarheid door zo een priem het quotiënt zelf een deler is waarvan het kwadraat kleiner is dan a. Is bij de 3de while-loop a mod b=0 dan is a geen priemgetal : we stellen t=0 en breken (break) uit deze loop,dan gaan we dus verder met de volgende a.
Blijft t=1, bij het doorlopen van de 3de while-loop, dan is a modulo b telkens verschillend van 0 geweest, voor alle priemen b waarvan het kwadraat < a. Bijgevolg is a een priemgetal.

Programma:

```
#priemgetallen
priem=[2,3]; i=1
while i<100:
    a=priem[i];t=0;
    while t==0:
        a=a+2;t=1;j=0;b=2
        while b*b<a:
            b=priem[j]
            if a%b ==0:
                t=0;break
            j=j+1
    i=i+1;priem.append(a)
print(priem)

>>>
```

[2, 3, 5, 7, 11, 13, 17, 19, 23, 29, 31, 37, 41, 43, 47, 53, 59, 61, 67, 71, 73, 79, 83, 89, 97, 101, 103, 107, 109, 113, 127, 131, 137, 139, 149, 151, 157, 163, 167, 173, 179, 181, 191, 193, 197, 199, 211, 223, 227, 229, 233, 239, 241, 251, 257, 263, 269, 271, 277, 281, 283, 293, 307, 311, 313, 317, 331, 337, 347, 349, 353, 359, 367, 373, 379, 383, 389, 397, 401, 409, 419, 421, 431, 433, 439, 443, 449, 457, 461, 463, 467, 479, 487, 491, 499, 503, 509, 521, 523, 541, 547]
>>>

9. Alle priemgetallen tussen 2 opgegeven getallen
priemgetallen_tussen_2_getallen

De kern van het programma blijft hetzelfde als bij het vorige maar **while i<100** wordt hier vervangen door **while priem[i] < boven**.
Aan het einde staat een if- instructie die bijhoudt wat de index ind_onder is van de eerste priem die boven de ondergrens ligt. Tot slot wordt een slice afgedrukt, tussen ind_onder en de laatste i van de lijst van alle priemgetallen.

Programma:

```
# berekenen van alle priemgetallen tussen 2 grenzen 'onder' en 'boven'
priem=[2,3];i=1
print('Priemgetallen berekenen tussen onder en boven')
onder=int(input('onder='));boven=int(input('boven='))
ind_onder=0
while priem[i]<boven:
   a=priem[i];t=0;
   while t==0:
      a=a+2;t=1;j=0;b=2
      while b*b<a:
         b=priem[j]
         if a%b ==0:
            t=0;break
         j=j+1
   i=i+1;priem.append(a)
   if ind_onder==0 and priem[i]>onder:
      ind_onder=i
print(priem[ind_onder:i])

>>>
Priemgetallen berekenen tussen onder en boven
onder=2000
boven=2500
[2003, 2011, 2017, 2027, 2029, 2039, 2053, 2063, 2069, 2081, 2083, 2087, 2089, 2099, 2111, 2113, 2129,
2131, 2137, 2141, 2143, 2153, 2161, 2179, 2203, 2207, 2213, 2221, 2237, 2239, 2243, 2251, 2267, 2269,
2273, 2281, 2287, 2293, 2297, 2309, 2311, 2333, 2339, 2341, 2347, 2351, 2357, 2371, 2377, 2381, 2383,
2389, 2393, 2399, 2411, 2417, 2423, 2437, 2441, 2447, 2459, 2467, 2473, 2477]
>>>
```

10. Een getal ontbinden in priemfactoren ontbinden_in_priemgetallen

Eerst wordt gevraagd wat het grootste getal is dat ingevoerd mag worden. Dat is nodig om een lijst aan te maken van alle priemgetallen waarvan het kwadraat kleiner is dan dit getal.

Met de while loop kan je getallen c invoeren, zoveel je wilt, als ze maar kleiner zijn dan deze bovengrens.
Er wordt dan een antwoordstring opgebouwd. **i** is de index van de priem waardoor deelbaarheid onderzocht wordt. Is er deelbaarheid door priem[i], dan wordt c=c//priem[i] en j=j+1. Dit blijft gebeuren in de 2de while loop tot als er geen deelbaarheid meer is.
Pas nadien wordt er overgegaan naar het volgende priemgetal.
Maar ondertussen is de antwoordstring wel aangevuld met priem[i]**j.
(als j=1 wordt enkel priem[i] afgedrukt).
En dit blijft zo doorgaan tot c=1

Programma:

```
# ontbinden van een getal in priemfactoren
import math
# eerst vragen naar een bovengrens van de in te voeren getallen
bovengrens=int(input('Geef de bovengrens van de getallen die je gaat invoeren:'))
grootstepriem=round(math.sqrt(bovengrens))
priem=[2,3];i=1
# berekent een rij priemgetallen tot de vkw van de bovengrens
while priem[i]<grootstepriem:
   a=priem[i];t=0;
   while t==0:
      a=a+2;t=1;j=0;b=2
      while b*b<a:
         b=priem[j]
         if a%b ==0:
               t=0;break
         j=j+1
   i=i+1;priem.append(a)
# een rij getallen invoeren waarvan de ontbinding wordt berekend
getal=1
while getal<=bovengrens and getal>0:
   getal=int(input('getal='));c=getal
   if getal>bovengrens:
      print('Getal te groot')
   else:
      antwstring=str(getal)+"=";i=-1
      while c>1 and i<len(priem)-1:
         i=i+1;j=0
         while c%priem[i]==0:
            c=c//priem[i];j=j+1
         if j>1:antwstring=antwstring+str(priem[i])+"^"+str(j)+"*"
         if j==1:antwstring=antwstring+str(priem[i])+"*"
      antwstring=antwstring+str(c)
      print(antwstring)
```

```
>>>
Geef de bovengrens van de getallen die je gaat invoeren:10000
getal=4502
4502=2*2251
getal=1024
1024=2^10*1
```

```
getal=7365
7365=3*5*491
getal=432
432=2^4*3^3*1
getal=10001
Getal te groot
>>>
```

11. Een hoek in decimale graden omzetten naar een hoek met graden, minuten, seconden

decimgrad_gradminsec

Om rechthoekige en willekeurige driehoeken te kunnen oplossen, moeten decimale hoeken (getal) kunnen omgezet worden naar gms hoeken (string) en omgekeerd. Daartoe dienen de volgende 2 functie definities.

Programma:

```
# dd ==>gms vb. het getal 34.375  wordt omgezet naar de string '34g22m30s'
def gms(hoekdd):
    hoekstr=''
    gr=int(hoekdd);hoekstr=hoekstr+str(gr)+'g';decmi=60*(hoekdd-gr)
    mi=int(decmi);hoekstr=hoekstr+str(mi)+'m';decs=60*(decmi-mi)
    se=round(decs);hoekstr=hoekstr+str(se)+'s'
    return hoekstr
# hoek invoeren
h=float(input("Geef een hoek in decimale vorm:"))
print (gms(h))

>>>
Geef een hoek in decimale vorm:37.234
37g14m2s
>>>
```

12. Een hoek in graden,minuten,seconden omzetten naar een hoek met decimale graden

Programma:

```
# gms ==>dd vb. string '34g22m30s' wordt omgezet naar een getal 34.375
def dd(hoek):
    gpos=hoek.index("g");gr=float(hoek[0:gpos]);hoek=hoek[gpos+1:]
    mpos=hoek.index("m");mi=float(hoek[0:mpos]);hoek=hoek[mpos+1:]
    spos=hoek.index("s");se=float(hoek[0:spos])
    dechoek=gr+mi/60+se/3600
    return dechoek
# hoek invoeren
h=input("Geef een hoek ..g..m..s: ")
print(dd(h))

>>>
Geef een hoek ..g..m..s: 37g14m2s
37.23388888888889
>>>
```

13.1 Hoe gebruiken we functies uit een module?

Er zijn 3 methodes om de functies uit een module te gebruiken.
Vb 1. heb je enkel de sqrt-functie uit math nodig:

```
from math import sqrt
...
print(sqrt(2))
```

Vb 2. heb je meerdere functies nodig, bvb. sin en radians:

```
import math
...
print(math.sin(math.radians(45)))
```

Vb 3. je kan ook alle functies invoeren, door de 'wildcard' * te gebruiken

```
from math import *
...
print(sin(radians(45)))
```

13.2 Keuzemenu's

Als je rechthoekige of willekeurige driehoeken oplost, zijn er 4 verschillende gevallen. Om die door de gebruiker te laten kiezen, gebruik je een menu met deze 4 mogelijkheden. In dit menu voorzie je ook een mogelijkheid om het programma te beëindigen. Het menu ziet er zo uit:

```
keuze=1
while keuze in [1,2,3,4]:
   print('Kies uit: \n 1.... \n 2....')
   keuze=int(input('Maak uw keuze:')
   if keuze==1:
            ...
   if keuze==2:
            ...
```

14. Oplossen van rechthoekige driehoeken. rechthoekige_driehoek_oplossen

De formules van goniometrie moeten omgezet worden naar python formules:
Vb.
B=Bgsin(b/a) wordt: B=math.degrees(math.asin(b/a))
ofwel: B=degrees(asin(b/a))
c=a.sin(C) wordt: c=a*math.sin(math.radians(C))
ofwel: ...

Programma:

```
# Oplossen van rechthoekige driehoeken
import math
# dms ==>dd
def dg(hoek):
   gpos=hoek.index("g");gr=float(hoek[0:gpos]);hoek=hoek[gpos+1:]
   mpos=hoek.index("m");mi=float(hoek[0:mpos]);hoek=hoek[mpos+1:]
   spos=hoek.index("s");se=float(hoek[0:spos])
```

```
    dechoek=gr+mi/60+se/3600
    return dechoek
# dd ==>dms
def gms(hoekdd):
    hoekstr="
    gr=int(hoekdd);hoekstr=hoekstr+str(gr)+'g';decmi=60*(hoekdd-gr)
    mi=int(decmi);hoekstr=hoekstr+str(mi)+'m';decs=60*(decmi-mi)
    se=round(decs);hoekstr=hoekstr+str(se)+'s'
    return hoekstr
# menu
k1="1. 2 rechthoekszijden"
k2="2. schuine zijde en een rechthoekszijde"
k3="3. rechthoekszijde en een hoek"
k4="4. schuine zijde en een hoek"
k5="5. einde"
keuze=1
while keuze in [1,2,3,4]:
    print(k1+'\n'+k2+'\n'+k3+'\n'+k4+'\n'+k5 +'\n' )
    keuze=int(input("Maak uw keuze:"))
    print(keuze,end=' ')
    if keuze==1:
        print(":geef b en c")
        b=float(input("b=")); c=float(input("c="))
        a=math.sqrt(b*b+c*c);B=math.degrees(math.asin(b/a));C=90-B
        print("a=",a," B=",gms(B)," C=",gms(C),"\n")
    if keuze==2:
        print(":geef a en c")
        a=float(input("a=")); c=float(input("c="))
        b=math.sqrt(a*a-c*c);B=math.degrees(math.asin(b/a));C=90-B
        print("b=",b," B=",gms(B)," C=",gms(C),"\n")
    if keuze==3:
        print(":geef b en C")
        b=float(input("b=")); Cst=(input("C (..g..m..s)="));C=dg(Cst)
        B=90-C;c=b*math.tan(math.radians(C));a=math.sqrt(b*b+c*c)
        print("B=",gms(B)," c=",c," a=",a,"\n")
    if keuze==4:
        print(":geef a en C")
        a=float(input("a=")); Cst=(input("C (..g..m..s)="));C=dg(Cst)
        B=90-C;c=a*math.sin(math.radians(C));b=math.sqrt(a*a-c*c)
        print("B=",gms(B)," c=",c," b=",b,"\n")
print('Einde')

>>>
1. 2 rechthoekszijden
2. schuine zijde en een rechthoekszijde
3. rechthoekszijde en een hoek
4. schuine zijde en een hoek
5. einde

Maak uw keuze:2
2 :geef a en c
```

a=11
c=8
b= 7.54983443527075 B= 43g20m30s C= 46g39m29s

1. 2 rechthoekszijden
2. schuine zijde en een rechthoekszijde
3. rechthoekszijde en een hoek
4. schuine zijde en een hoek
5. einde

Maak uw keuze:3
3 :geef b en C
b=7
C (..g..m..s)=42g15m46s
B= 47g44m13s c= 6.361202026192962 a= 9.458588225419344

1. 2 rechthoekszijden
2. schuine zijde en een rechthoekszijde
3. rechthoekszijde en een hoek
4. schuine zijde en een hoek
5. einde

Maak uw keuze:5
5 Einde
>>>

15. Oplossen van willekeurige driehoeken willekeurige_driehoek_oplossen

We gebruiken de sinusregel en de cosinusregel

$\cos(A) = \frac{b^2 + c^2 - a^2}{2.b.c}$ in Python: A=degrees(acos((b*b+c*c-a*a)/2/b/c))

$c = \sqrt{a^2 + b^2 - 2 \cdot a \cdot b \cdot \cos C}$

$b = a.\frac{\sin B}{\sin A}$ $B = \text{asin}(\frac{b.\sin A}{a})$

Zoals bij rechthoekige driehoeken kunnen we ook hier uitgaan van een keuzemenu

Programma:

```
# Oplossen van willekeurige driehoeken
import math
# dms ==>dd
def dg(hoek):
    gpos=hoek.index("g");gr=float(hoek[0:gpos]);hoek=hoek[gpos+1:]
    mpos=hoek.index("m");mi=float(hoek[0:mpos]);hoek=hoek[mpos+1:]
    spos=hoek.index("s");se=float(hoek[0:spos])
    dechoek=gr+mi/60+se/3600
    return dechoek
# dd ==>dms vb. het getal 34.375  wordt omgezet naar de string '34g22m30s'
def gms(hoekdd):
```

```
    hoekstr="
    gr=int(hoekdd);hoekstr=hoekstr+str(gr)+'g';decmi=60*(hoekdd-gr)
    mi=int(decmi);hoekstr=hoekstr+str(mi)+'m';decs=60*(decmi-mi)
    se=round(decs);hoekstr=hoekstr+str(se)+'s'
    return hoekstr
# goniometrische functies
def invcos(p,q,r):
    return math.degrees(math.acos((q*q+r*r-p*p)/2/q/r))
def sinreg(p,Q,R):
    return p*math.sin(math.radians(Q))/math.sin(math.radians(R))
def cosreg(p,q,R):
    return math.sqrt(p*p+q*q-2*p*q*math.cos(math.radians(R)))
def invsin(p,q,P):
    return math.degrees(math.asin(q*math.sin(math.radians(P))/p))
def r(z):return round(z,4)
def oplossing():
    print(uitleg,'\n','abc=',r(a),r(b),r(c),'\n')
    print('ABC=',gms(A),gms(B),gms(C),'\n')
# menu
k1="1. de 3 zijden"
k2="2. 2 zijden en de ingesloten hoek"
k3="3. 1 zijde en 2 hoeken"
k4="4. 2 zijden en een overstaande hoek"
k5="5. einde"
keuze=1;uitleg="
while keuze in [1,2,3,4]:
    print('Oplossen van willekeurige driehoeken')
    print(k1+'\n'+k2+'\n'+k3+'\n'+k4+'\n'+k5)
    keuze=int(input("Maak uw keuze:"))
    if keuze==1:
        print(k1+"; geef a, b , c:")
        a=float(input("a="));b=float(input("b=")); c=float(input("c="))
        A=invcos(a,b,c);B=invcos(b,c,a);C=invcos(c,a,b)
        uitleg='De 3 hoeken worden berekend met de omgekeerde cosinusregel'
        oplossing()
    if keuze==2:
        print(k2+"; geef a, b , C:")
        a=float(input("a=")); b=float(input("b="))
              Cst=input("C (..g..m..s)=");C=dg(Cst)
        c=cosreg(a,b,C);A=invcos(a,b,c);B=invcos(b,c,a)
        uitleg='De 3de zijde c met de cosinusregel, A en B met de omgekeerde cosinusregel'
        oplossing()
    if keuze==3:
        print(k3+"; geef a, B , C:")
        a=float(input("a=")); Bst=input("B(..g..m..s)=")
        B=dg(Bst);Cst=input("C (..g..m..s)=");C=dg(Cst)
        A=180-B-C;c=sinreg(a,C,A);b=sinreg(a,B,A)
        uitleg='A = 180-(B+C), b en c met de sinusregel'
        oplossing()
    if keuze==4:
        print(k4+"; geef a, b , A:")
```

```
        a=float(input("a="));b=float(input("b="))
              Ast=input("A (..g..m..s)=");A=dg(Ast)
        if a<b*math.sin(math.radians(A)):
           print('Omdat a< b.sinA is er geen oplossing\n')
        else:
           B=invsin(a,b,A);C=180-A-B;c=cosreg(a,b,C)
           uitleg='"B met de omgekeerde sinusregel, C = 180 -(A+B), c met de cosinusregel"'
           oplossing()
           if a<b:
              B=180-B;C=180-A-B;c=cosreg(a,b,C)
              uitleg='Omdat a<b maar >b.sinA is er een 2de oplossing'
              oplossing()
print('Einde')

>>>
Oplossen van willekeurige driehoeken
1. de 3 zijden
2. 2 zijden en de ingesloten hoek
3. 1 zijde en 2 hoeken
4. 2 zijden en een overstaande hoek
5. einde
Maak uw keuze:2
2. 2 zijden en de ingesloten hoek; geef a, b , C:
a=3
b=4
C (..g..m..s)=74g25m46s
De 3de zijde c met de cosinusregel, A en B met de omgekeerde cosinusregel
 abc= 3.0 4.0 4.3079
 ABC= 42g7m55s 63g26m19s 74g25m46s

Oplossen van willekeurige driehoeken
1. de 3 zijden
2. 2 zijden en de ingesloten hoek
3. 1 zijde en 2 hoeken
4. 2 zijden en een overstaande hoek
5. einde
Maak uw keuze:5
Einde
>>>
```

16 Benaderen van de vierkantswortel met de methode van Heroon heroon

Wiskundige benadering

Veronderstel dat we intuïtief een eerste benadering van de vierkantswortel van een getal bepaald hebben.
Voorbeeld: voor de vierkantswortel van 23 kunnen we 5 nemen.
5 is groter dan vierkantswortel van 23, dus 23/5 zal een benadering zijn die kleiner is dan de vierkantswortel van 23.
Een betere benadering is het rekenkundig gemiddelde van 5 en 23/5,

dwz. $\frac{5+\frac{23}{5}}{2}$ = 4,8. Meer algemeen, als x een benadering is , dan is $\frac{x+\frac{a}{x}}{2}$ een betere benadering. Dit laatste getal kunnen we dan als x nemen om terug een betere benadering te vinden, enz....Dit kunnen we verder zetten tot als we de vierkantswortel van a zeer dicht benaderd hebben

Programma met TI84

Hierna volgt een programma dat je kunt gebruiken op de TI84 om vierkantswortels te benaderen:
a is het getal waarvan we de vierkantswortel willen benaderen. x is een eerste benadering. In de While….
End-loop wordt steeds een betere benadering berekend met de formule

In het vorige programma werd een While <voorwaarde> loop gebruikt.
Een andere loop is de For(I ..-loop. Alle loops in TI84-basic eindigen met End

Programma in Python

In de eerste loop wordt het kleinste geheel getal bepaald waarvan het kwadraat > a.
In de 2de loop wordt de vierkantswortel benaderd met x=(x+a/x)/2 tot als het verschil tussen x^2 en a kleiner is dan een zeer klein getal eps.

Programma:

```
# benaderen van vierkantswortels met de methode van Heroon
eps=0.00001
print("Benaderen van een vierkantswortel")
a=float(input("Geef een getal:"))
x=0
while x*x<a:
    x=x+1
while abs(x*x-a)>eps:
    print(x)
    x=(x+a/x)/2
>>>
Benaderen van een vierkantswortel
Geef een getal:31
6
5.583333333333334
5.567786069651741
>>>
```

Dit is slechts een benadering, de TI84 geeft 5.567764363

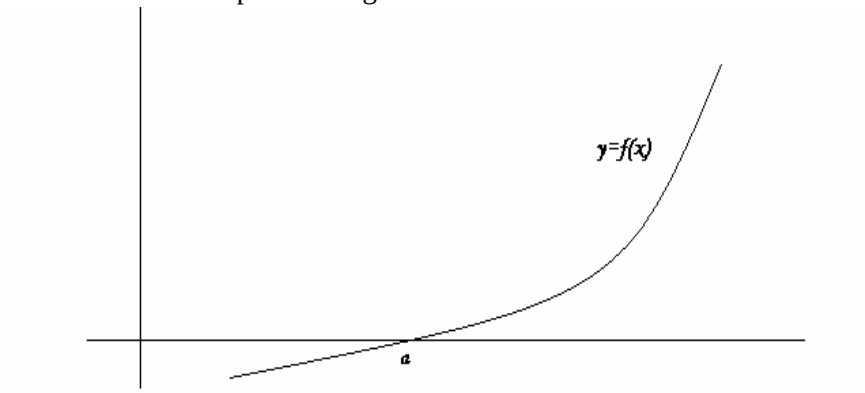
17. De methode van Newton

Inleiding

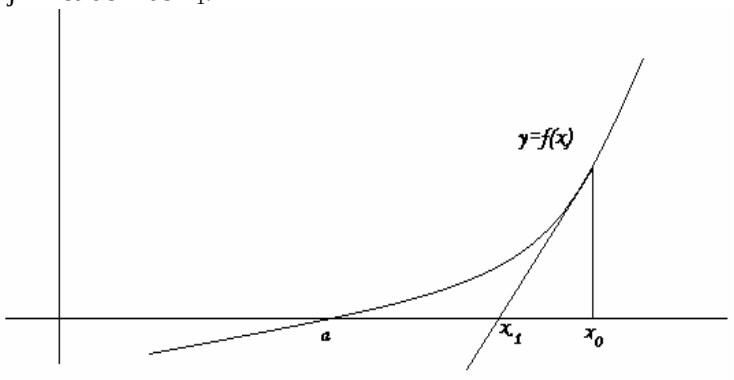
Een nulpunt van een functie berekenen, kan bij slechts een zeer klein aantal functies gebeuren op een analytische manier:
Bij lineaire functies f(x)=ax+b is het nulpunt -b/a,
Bij de kwadratische functie $f(x)=ax^2+bx+c$ worden de nulpunten bepaald met de wortelformules.
Bij andere functies[1] bestaat er geen algemeen systeem om het nulpunt te vinden. In dat geval moeten we het nulpunt vinden door middel van een zogenaamde benaderingsmethode.
Eén der beste methodes maakt gebruik van de theorie van de afgeleiden:

Benaderen van een nulpunt met de methode van Newton:

Grafische bepaling van een benaderingsfunctie
Opdracht:
1. Beschouw een willekeurige functie die een nulpunt heeft.
Wat betekent dit nulpunt in de grafiek?



Probeer dit nulpunt nu te benaderen door raaklijnen aan de kromme te tekenen.
Begin in een punt x_0 en trek de raaklijn in het punt P(x_0, $f(x_0)$) aan de kromme. Noem het snijpunt van de raaklijn met de x-as x_1.



Herhaal nu hetzelfde om van x_1 naar x_2 te gaan.
En om van x_2 naar x_3 te gaan, enz….
Wat merk je van de rij x_0, x_1, x_2, x_3, x_4, …..?
Zal je steeds vanuit een willekeurig punt naderen tot het nulpunt van de functie?

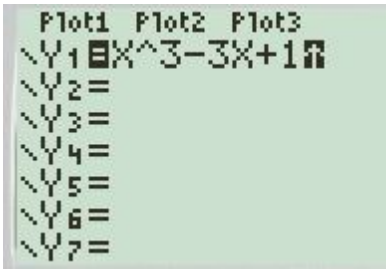
---

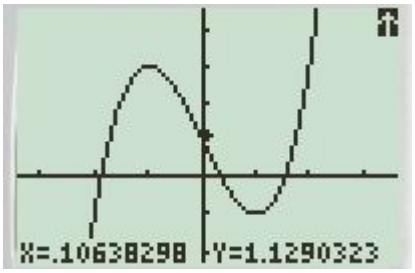
[1] Bij 3de graadsfunctie bestaat de methode van Cardano. Te ingewikkeld en niet nodig. Newton is eenvoudiger en toepasbaar op alle soorten functies.

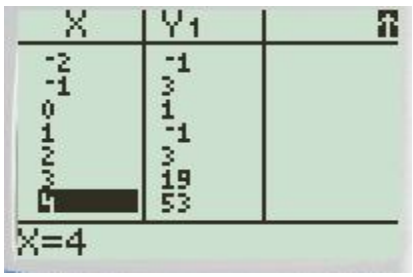
Probeer een situatie te tekenen waar dit niet zo is.
Hoe kan je weten dat een gegeven startwaarde een convergerende rij oplevert?
Een grafiek tekenen? Een waardetabel opstellen?
Bedenk dat de meeste functies meerdere nulpunten hebben. Om ze allemaal te benaderen, zullen we voor elk een geschikte startwaarde nodig hebben.
Hoe kunnen we nu analytisch x_1 berekenen, uitgaande van x_0 ?
x_1 is het snijpunt van de raaklijn met de x-as .
We moeten eerst de vergelijking van de raaklijn opstellen:
$y-f(x_0) = f'(x_0).(x-x_0)$
$(x_1,0)$ is het snijpunt van deze raaklijn met de x-as: we vullen in:
$0-f(x_0) = f'(x_0).(x_1-x_0)$ waaruit we x_1 kunnen berekenen in functie van x_0:
$x_1 = x_0 - f(x_0)/ f'(x_0)$
Hoe zullen we nu x_2 berekenen uitgaande van x_1?
Natuurlijk op dezelfde manier.
Vergelijking raaklijn:
$y-f(x_1) = f'(x_1).(x-x_1)$
snijpunt met de x-as $(x_2,0)$ invullen:
$0-f(x_1) = f'(x_1).(x_2-x_0)$ waaruit volgt $x_2 = x_1 - f(x_1)/ f'(x_1)$
Deze functie die uitgaande van x_0 achtereenvolgens x_1, x_2, x_3 … berekent, noemen we de benaderingsfunctie: ze is van de vorm **x - f(x) /f '(x)**
Uitgewerkt voorbeeld: we zoeken een nulpunt van de functie x^3-x-2
f(0)= -2 , f(1)= -2, f(2)= 4, dus tussen 1 en 2 zit een nulpunt.
1 is waarschijnlijk een goede startwaarde. We stellen de benaderingsfunctie op

$$x - \frac{f(x)}{f'(x)} = x - \frac{x^3 - x - 2}{3x^2 - 1} = \frac{x(3x^2 - 1) - (x^3 - x - 2)}{3x^2 - 1} = \frac{2x^3 + 2}{3x^2 - 1}$$

.
Op de TI84 voeren we achtereenvolgens in
1 STO X ENTER
$(2X^3+2)/(3X^2-1)$ STO X ENTER
We krijgen als we telkens op ENTER drukken :
$x_2= 2$, $x_3= 1.636...$, $x_4= 1.5303..$, $x_5= 1.52144..$,
$x_6= 1.52137971$, $x_7= 1.521379707$, $x_8= 1.521379707$
Voeren we nu $x^3 - x - 2$ in, dan vinden we 0, wat bewijst dat we het nulpunt gevonden hebben. We hadden hier het geluk van een goede startwaarde te nemen. Om daar zeker van te zijn, is een grafisch onderzoek van de functie nodig.
2. Als je ook de grafiek tekent, kan je zonder moeite goede startwaarden vinden.
Stel bvb. $f(x)= x^3 - 3x + 1$
Voer de functie in en teken de grafiek met Graph, ZBox en/of 2nd Table

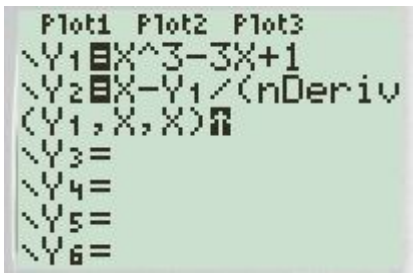






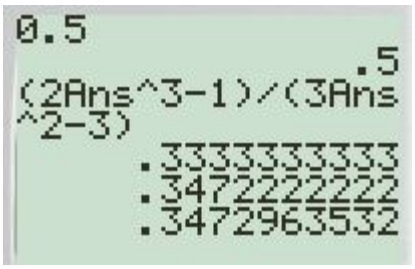
Er zitten nulpunten dicht bij -2, tussen 0 en 1 en tussen 1 en 2
Opgepast, de functie heeft een maximum in -1 en een minimum in 1. Dit kan je gemakkelijk analytisch vinden: het zijn de nulpunten van de afgeleide functie $3x^2-3$

Als je deze punten als startwaarde zou nemen, waar kom je dan terecht?
Kies nu 3 goede startwaarden.
Je kunt nu bvb. voor Y_2 de benaderingsfunctie invullen



Eerste startwaarde x_0 invullen -2 ENTER Y_2(ANS) ENTER ENTER ENTER ..en we krijgen achtereenvolgens waarden x_1, x_2 , x_3
Reeds bij x_3 vinden we een goede benadering van het kleinste nulpunt -1,879385242

Opmerking: je kan natuurlijk ook zelf analytisch het voorschrift van de benaderingsfunctie opstellen, voor $f(x) = x^3-3x+1$ vind je
$x-f(x)/f\,'(x) = x-(x^3-3x+1)/(3x^2-3) = (2x^3-1)/(3x^2-3)$
Omdat we als startwaarde 0.5 hebben gekozen, vinden we een ander nulpunt. Zoek nu ook het 3^{de} nulpunt.

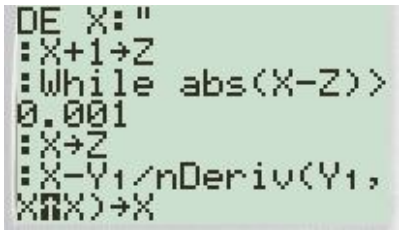


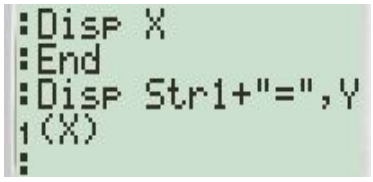
3. Nog leuker is het opstellen van een programma dat ons vraagt om een functie in te voeren, daarvan eerst een tabel toont, ons vraagt om een startwaarde op te geven en dan automatisch het nulpunt berekent.

PROGRAM NEW <E>	Nieuw programma invoeren
Name= NEWTON <E>	Titel programma
: Input"F(x)=",Str1	Functievoorschrift invoeren
: Str1 STO Y1	Voorschrift toekennen aan Y_1
:DispTable	Tabel afdrukken
:Input "START=",X	Startwaarde invoeren
:Disp"BENADERENDE X:"	Titel afdrukken
:X+1 STO Z	(1)
:While abs(X-Z)>0.001	(2)
:X STO Z	(3)
:X-Y_1/nDeriv(Y_1,X,X) STO X	(4)
:Disp X	Berekende X afdrukken
:End	Einde While Loop
:Disp Str1+"=",Y_1(X)	Afdrukken functiewaarde in de laatste X

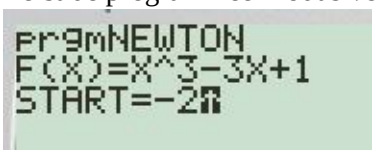
(1): Z is een variabele die in de While loop de oude waarde van X bevat, daarom moet vooraf het verschil tussen X en Z > 0.001, anders zou de While voorwaarde meteen voldaan zijn, dus door Z=X+1 te stellen is abs(X-Z)=1 >0.001
(2): Zolang de oude waarde en de nieuwe waarde van X meer dan 0.001 verschillen moet de loop tussen While en End uitgevoerd worden.
(3) de waarde van X in Z stoppen zodat er een

(4) nieuwe waarde voor X kan berekend worden met de benaderingsfunctie





We voeren nu het programma uit.
Eerst de programmeermodus verlaten: 2nd QUIT, dan PRGM EXEC NEWTON



Hoe moet je sommige elementen terugvinden op de TI84?:

=, >,< in het menu 2nd TEST
Input, Disp,DispTable in het menu PRGM I/O
Abs in het menu MATH NUM 1
Str1 in het menu VARS VARS String 1

En nu het programma van de methode van Newton in Python newton

Programma:

```
# methode van Newton
def f(x):
    return eval(fs)
def df(x):
    dx=0.00001
    return (f(x+dx)-f(x))/dx
fs=input("Geef een functie:")
ts=input("Geef een getal:")
x=eval(ts)
ox=x+1
while abs(ox-x)>0.0001:
    ox=x
    x=x-f(x)/df(x)
    print(x,' ',f(x))
print("Nulpunt=",x)
>>>
Geef een functie:x**5-3*x**3-6
Geef een getal:2
1.9545460950365017   0.12469155298873247
1.9513148971038616   0.0005965848896813952
1.951299288737924   2.2677976829754698e-08
Nulpunt= 1.951299288737924
>>>
```

Opmerking: ook hier kan je vooraf een loop programmeren die een tabel van functiewaarden geeft, zodat je een goede startwaarde kunt kiezen. Probeer maar.

Numerieke integraalberekening

Om de oppervlakte onder een kromme numeriek te benaderen, kunnen we het integratieinterval b-a vermenigvuldigen met een gewogen gemiddelde functiewaarde over dit interval.
Deze benadering is enkel nauwkeurig bij continue functies indien de grafiek in het interval nergens 'te verticaal' is. We bespreken 2 methodes.
Voor beide methodes kan je een programma schrijven op de PC en de TI84.
Het programma in Python wordt bij beide methodes gegeven.
Hieruit kan je zelf een gelijkaardig programma op de TI84 afleiden.

18. Trapeziummethode trapezium_integraal

Om de oppervlakte onder een kromme te benaderen, verdelen we het interval [a,b] in n stukken en berekenen de som van de oppervlaktes van de trapeziums die we op deze wijze bekomen.
(oppervlakte trapezium = (grote basis +kleine basis) x hoogte /2)
Als we het interval bvb. verdelen in 8 stukken dan is de som van deze oppervlaktes=
$\mathbf{T_1 + T_2 + ... + T_8}$ =

$$\frac{f(x_0)+f(x_1)}{2}\cdot h+\frac{f(x_1)+f(x_2)}{2}\cdot h+\cdots+\frac{f(x_7)+f(x_8)}{2}\cdot h=$$

$$\frac{f(x_0)+2f(x_1)+\cdots+2f(x_7)+f(x_8)}{2}\cdot h=\left[f(x_1)+\cdots+f(x_7)+\frac{f(a)+f(b)}{2}\right]\cdot h$$

met $x_0 = a$ en $x_8 = b$

Algemener, als we het interval [a,b] in n stukken verdelen, is de som van de oppervlaktes =

$$\left[\sum_{j=1}^{n-1} f(x_j)+\frac{f(a)+f(b)}{2}\right]\cdot h$$ waarbij $x_0 = a$, $x_n = b$, $h = (b-a) / n$

Programma met Python

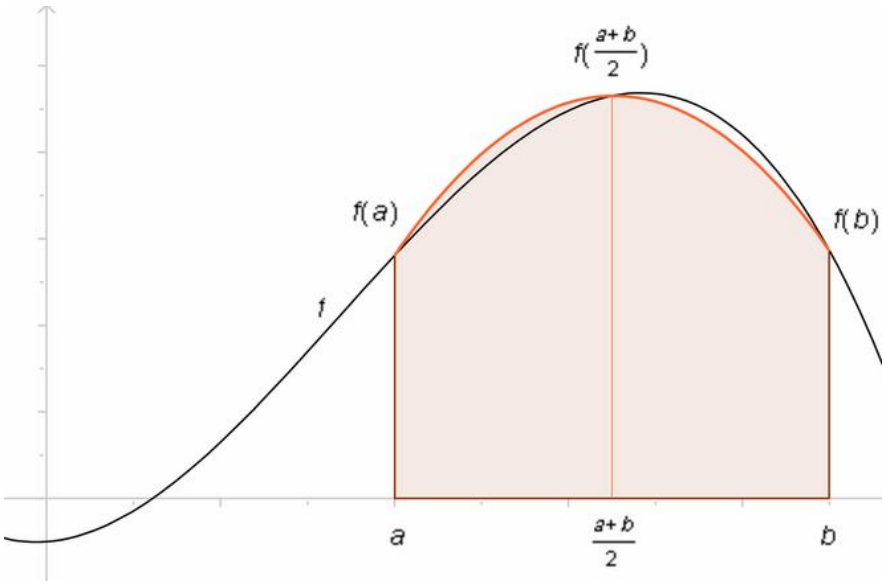
```
# Trapezium
def f(x):
    return eval(fs)
fs=input("Geef een functie:")
a=float(input("Geef ondergrens:"))
b=float(input("Geef bovengrens:"))
n= int(input("Geef aantal deelintervallen:"))
h=(b-a)/n
x=a
som=0
for j in range(1,n):
    x=x+h
    som=som+f(x)
som=som+(f(a)+f(b))/2
integ=som*h
print('Integraal=',integ)
```

```
>>>
Geef een functie:x*x-3*x+2
Geef ondergrens:1
Geef bovengrens:6
Geef aantal deelintervallen:10
Integraal= 29.375
>>>
```

OPGAVE: Schrijf nu zelf een programma met de TI84. Probeer het uit met enkele bepaalde integralen die je ook analytisch kunt berekenen.

19. De regel van Simpson

simpson_integraal



De regel van Simpson benadert de integraal van een functie f over een interval [a,b] als de integraal van de 2^{de} graadsfunctie (parabool) P(x) die in de eindpunten en het midden van het interval met de functie overeenkomt

dwz. P(a)=f(a), P(b)=f(b) en $P\left(\frac{a+b}{2}\right)=f\left(\frac{a+b}{2}\right)$.

Stel $P(x)=px^2+qx+r$, dan geldt

$pa^2+qa+r=f(a)$ (1)

$pb^2+qb+r=f(b)$ (2)

$$p\left(\frac{a+b}{2}\right)^2+q\left(\frac{a+b}{2}\right)+r=f\left(\frac{a+b}{2}\right) \quad (3)$$

OPGAVE:

1. Bereken $f(a)+f(b)+4f(\frac{a+b}{2})$ (4)

2. Toon aan dat

$$\int_a^b \left(px^2+qx+r\right)dx=\frac{b-a}{6}.\left(f(a)+f(b)+4f(\frac{a+b}{2})\right) \quad (5)$$

In plaats van één benaderende parabool op het volledige interval [a,b]

zullen we nu [a,b] verdelen in n stukken met lengte h, dus $\frac{b-a}{n}=h$.

$x_0 = a$ $x_1 = x_0 + h$ $x_2 = x_1+h$ $x_3 = x_2+h$ …

Voor elk van de punten van deze verdeling nemen we de oppervlakte van een benaderende parabool.
De 1ste parabool zit in het interval (a, a+2h), de 2de parabool zit in (a+2h, a+4h) , …

We krijgen op die manier $\frac{n}{2}$ parabolen.

Merk hier op dat voor de methode van Simpson n een even getal moet zijn.
Om formule (5) toe te passen voor de 1ste parabool, vervangen we a en b door x_0 en x_2 ,

b-a wordt dan 2h en $x_1 = x_0 + h = \frac{x_0 + x_2}{2}$.

OPGAVE: Toon aan dat

$P_1 = \frac{h}{3} \cdot [f(x_0)+4f(x_1)+f(x_2)]$

$P_2 = \frac{h}{3} \cdot [f(x_2)+4f(x_3)+f(x_4)]$

...

$p_{\frac{n}{2}} = \frac{h}{3} \cdot [f(x_{n-2})+4f(x_{n-1})+f(x_n)]$

Tellen we al deze oppervlaktes op, dan krijgen we een benadering van de oppervlakte onder de kromme y=f(x) in [a,b]

OPGAVE: Toon aan dat

$$\int_a^b f(x)dx = \frac{h}{3} \cdot [\, f(x_0)+4f(x_1)+2f(x_2)+4f(x_3)+2f(x_4)+\ldots+2f(x_{n-2})+4f(x_{n-1})+f(x_n)\,]$$

Programma:

```
# Simpson
def f(x):
   return eval(fs)
fs=input("Geef een functie:")
a=eval(input("Geef ondergrens:"))
b=eval(input("Geef bovengrens:"))
n= eval(input("Geef aantal deelintervallen:"))
h=(b-a)/n
x=a
som=0
for j in range(1,n):
   x=x+h
   if j%2==0:
      t=2
   else:
      t=4
   som=som+t*f(x)
som=som+f(a)+f(b)
integ=som*h/3
print('Integraal=',integ)
>>>
Geef een functie:x**3-3*x+1
Geef ondergrens:2
Geef bovengrens:4
```

Geef aantal deelintervallen:10
Integraal= 44.00000000000006
>>>
Wil je het programma ook laten werken voor wiskundige functies sin, cos, exp,... dan volstaat het om als eerste lijn **import math** toe te voegen.

OPGAVE: Schrijf nu zelf een programma met de TI84.
Als je dit programma uitvoert met dezelfde functies als bij de trapeziummethode zal je merken dat Simpson veel nauwkeuriger is.
Een paar hintjes voor het programmeren op de TI84:
- **j %2 ==0** kan je bij de TI84 vervangen door **fPart (J/2)=0**
- Een **if……else…**constructie wordt bij de TI84
  If …..:Then : ……: Else:…..: End
  Je kan wel alles op 1 lijn schrijven, maar dan gescheiden door een **:**
- **x=a** wordt bij de TI84 **A STO X**
- Je kan een functie laten invoeren in een programma met **Input"FUNCTIE:",Y_1**
  Nadeel hiervan is dat je bij het invoeren van de functie, aanhalingstekens moet gebruiken, bvb.**"SIN(X)"**.
  Beter is om de functie met een stringvariabele in te voeren, en deze in een volgende lijn om te zetten naar de functie Y_1.
  Input"FUNCTIE:",Str_1
  Str_1 STO Y_1

20. Oplossen van vierkantsvergelijkingen in C vierkantsvergel_complex

Om vierkantsvergelijkingen in C op te lossen, weten we dat als d<0 er 2 complex toegevoegde oplossingen zijn: $x_1 = \frac{-b-\sqrt{-d}\cdot j}{2a}$ en $x_2 = \frac{-b+\sqrt{-d}\cdot j}{2a}$

in Python:
root=math.sqrt(-d)
x1=(-b-root*(1j))/2
x2=(-b+root*(1j))/2

Programma:

```
# oplossen van vierkantsvergelijkingen in C
import math
print('Oplossen van de vierkantsvergelijking ax²+bx+c=0 in C')
teller=0
while teller<3:
    teller=teller+1
    print('Vierkantsvergelijking',teller)
    a=float(input('a='))
    b=float(input('b='))
    c=float(input('c='))
    d=b*b-4*a*c
    if d>0:
        root=math.sqrt(d)
        x1=(-b-root)/2/a
        x2=(-b+root)/2/a
    elif d==0:
```

```
        x1=x2=-b/2/a
    else:
        root=math.sqrt(-d)
        x1=(-b-root*(1j))/2
        x2=(-b+root*(1j))/2
    print('x1=',x1,'  x2=',x2)

>>>
```

Oplossen van de vierkantsvergelijking $ax^2+bx+c=0$ in C
Vierkantsvergelijking 1
a=2
b=5
c=8
x1= (-2.5-3.122498999199199j) x2= (-2.5+3.122498999199199j)
Vierkantsvergelijking 2
a=1
b=4
c=29
x1= (-2-5j) x2= (-2+5j)
Vierkantsvergelijking 3
a=2
b=7
c=-9
x1= -4.5 x2= 1.0
>>>

21. Ontbinden van een kwadratische functie in C ontbinden_kwadrat_fie_inC

Is d>0 dan moet de ontbinding er zo uitzien: $3x^2+3x-18= 3(x+3)(x-2)$
De oplossingen van de overeenkomstige vierkantsvergelijking zijn -3 en 2
In de ontbinding moeten we dus $-x_1$ en $-x_2$ afdrukken.
$-x_2 = -2$ zal correct afgedrukt worden, maar bij $-x_1 = 3$ moet een '+' voorgevoegd worden. Hiervoor dient de functie v(x).

Hebben we complex toegevoegde oplossingen, dan kiezen we er best voor om de haakjes te behouden: vb. $5x^2-10x+50=5[x-(1-3j)][x-(1+3j)]$.
Hiervoor dient de functie vc(x)

Programma:

```
# Ontbinden van een kwadratische functie ax²+bx+c in C
import math
def v(x):
    if x>0:
        sgn='+'
    else:
        sgn=''
    return sgn+format(x,'.3f')
def vc(x):
    return "-("+format(x,'.3f')+")"
print('Ontbinden van een kwadratische functie ax²+bx+c in C')
teller=0
while teller<3:
```

```
    teller=teller+1
    a=float(input('a='))
    b=float(input('b='))
    c=float(input('c='))
    d=b*b-4*a*c
    if d>0:
        root=math.sqrt(d)
        x1,x2=(-b-root)/2/a,(-b+root)/2/a
    elif d==0:
        x1=x2=-b/2/a
    else:
        root=math.sqrt(-d)
        x1,x2=(-b-root*(1j))/2/a,(-b+root*(1j))/2/a
     if d>0:
        print(str(a)+'(x'+v(-x2)+')(x'+v(-x1)+')')
    elif d==0:
        print(str(a)+'(x'+v(-x2)+')²')
    else:
        print(str(a)+'[x'+vc(x2)+'][x'+vc(x1)+']')
>>>
```

Ontbinden van een kwadratische functie ax^2+bx+c in C

```
a=3
b=-30
c=75
3.0(x-5.000)²
a=2
b=12
c=26
2.0[x-(-3.000+2.000j)][x-(-3.000-2.000j)]
a=1
b=-7
c=11
1.0(x-4.618)(x-2.382)
>>>
```

De functies v(x) en vc(x) zetten de oplossingen om naar strings met 3 cijfers na de komma, eventueel voorzien van een + of -teken: bij reële getallen , een + bij indien positief, bij complexe getallen steeds –, vermits dat de complexe oplossing tussen haakjes staat.

22. Meervoudige invoer

Sommige programma's vragen om verschillende getallen in te voeren. Je kan de getallen één voor één laten invoeren. Maar het zou voor de gebruiker meer comfort bieden, als alle getallen ineens kunnen ingevoerd worden, gewoon gescheiden door komma's.
Beschouw eerst het volgende voorbeeld:

```
l=(input('Geef a0,a1,a2:')).split(',')
print(l)
a=[]
for i in l:a.append(float(i))
print( a[0]+a[1]+a[2])
```

We runnen dit progamma en we geven als invoer 2,4,7
De inputstring input(...) is dan '2,4,7'
Door de instructie l=(input....).split(,) wordt een list opgesteld van 3 deelstrings '2' '4' en '7' , waarbij het splitkarakter de string ',' is.
Met de for-loop wordt dan een list a opgebouwd van getallen a[0]=2, a[1]=4 en a[2]=7.
l is dus de list ['2',' 4','7'] en a is de list [2,4,7]
We hebben die dus in één inputstring kunnen invoeren

```
>>>
Geef a0,a1,a2:2,4,7
['2', '4', '7']
13.0
>>>
```

l is een lijst van strings maar a is een lijst van getallen.
Als we de 2 print-instructies weglaten, krijgen we een eenvoudige methode van meervoudige invoer die overal toepasbaar is.
We kunnen natuurlijk ook een functie gebruiken:

```
def mult(s):
    l=s.split(',')
    co=[]
    for i in l:co.append(float(i))
    return co
#hoofdprogramma
a,b,c,d=mult(input('a,b,c,d='))
......
```

Stel dat we invoeren

```
>>>
a,b,c,d=2,3,4,5
```

Dan is input('a,b,c,d=') de string '2,3,4,5'
In def mult(s) wordt s='2,3,4,5'. Hiervan wordt met split een lijst l=['2','3','4','5'] gemaakt.
Met de for- loop wordt co opgevuld met de floats van de elementen van l.
co wordt dus de lijst [2,3,4,5] die teruggegeven wordt en waaraan a,b,c,d gelijk is. Dus a=2, b=3, c=4, d=5
Op deze manier is meervoudige invoer van reële getallen mogelijk.
Vervang je float door int in de mult-functie, dan wordt dit een functie voor de meervoudige invoer van gehele getallen.

23. Toepassing van numerieke integratie in de statistiek normale_verdeling_cdf

Numerieke integratie is eigenlijk niet nodig voor integralen die je ook analytisch kunt berekenen. Maar voor heel wat functies is dit niet mogelijk.
Zo bijvoorbeeld ook de normale verdelingsfunctie nd :

$$f(x) = \frac{1}{\sigma\sqrt{2\pi}} \cdot e^{-\frac{1}{2}\left(\frac{x-\mu}{\sigma}\right)^2}$$ en de vereenvoudigde vorm,

de standaardnormale verdelingsfunctie snd (als $\mu = 0$ en $\sigma = 1$)

$$f(x) = \frac{1}{\sqrt{2\pi}} \cdot e^{\frac{-x^2}{2}}$$

In het 1ste programma wordt met **Simpson** de kans berekend dat een stochastische variabele tussen 2 opgegeven grenzen ligt, bij gegeven gemiddelde en standaardafwijking.
μ en σ worden respectievelijk vervangen door m en s. Vergelijk de functiedefinitie van nd met het voorschrift van de normale verdelingsfunctie. Omdat voor de normale verdelingsfunctie geen elementaire primitieve functie kan bepaald worden, is de numerieke berekening van de oppervlakte onder de nd-functie noodzakelijk.

Programma:

```
# simulatie normalcdf TI84
import math
def mult(s):
    l=s.split(',')
    co=[]
    for i in l:co.append(float(i))
    return co
def nd(x):
    return math.e**(-(x-m)**2/2/s/s)/s/math.sqrt(2*math.pi)
def integr(a,b,n):
    h=(b-a)/n
    x=a
    som=0
    for j in range(1,n):
        x=x+h
        if j%2==0:
            t=2
        else:
            t=4
        som=som+t*nd(x)
    som=som+nd(a)+nd(b)
    integ=som*h/3
    return(integ)
m,s=mult(input('Geef rekenkundig gemiddelde en standaardafwijking:'))
a,b=mult(input('Geef ondergrens en bovengrens van de gegevens:'))
n=int(input('Aantal deelintervallen:'))
percentgeg=round(100*integr(a,b,n),2)
print('Tussen',a,'en',b,'liggen',percentgeg,'% van de gegevens')

>>>
Geef rekenkundig gemiddelde en standaardafwijking:8.4,2.3
Geef ondergrens en bovengrens van de gegevens:6,10
Aantal deelintervallen:50
Tussen 6.0 en 10.0 liggen 60.83 % van de gegevens
>>>
```

24. Tabel van Zi-waarden berekenen tabel_Zi_norm_verdel

In het 2de programma wordt een tabel opgesteld voor de kans dat een gestandaardiseerde variabele kleiner is dan een opgegeven waarde.
De kans dat Zi <0 is 50%=0,5. Dit is de oppervlakte onder de kromme tussen -∞ en 0.

Met stappen van 0,01 wordt telkens de oppervlakte bijgeteld onder de kromme.
Al deze oppervlaktes worden opgeslagen in een lijst s.
Dus s(0) = opp. onder de dichtheidskromme tussen -∞ en 0 = 0,5
s(1)= opp. onder de dichtheidskromme tussen -∞ en 0,01
= s(0)+opp. onder de dichtheidskromme tussen 0 en 0,01
s(262)= opp. onder de dichtheidskromme tussen -∞ en 2,62
In de 1ste FOR..NEXT-loop worden al deze s(i) berekend.
De tabel wordt zo opgesteld dat in de bovenste rij per kolom de 0,0c waarde komt (dus vanaf 0,00 tot 0,09). Daaronder komt per rij eerst a,b (van 0,0 tot 3,4), nadien in de juiste kolom de s(abc)-waarde. De laatste waarde die wordt afgedrukt, is s(349) = 0,9998, dwz.: P(-∞< Zi< 3,49) =0,9998.

Programma:

```
# Tabel van de Zi vanaf 0 tot 3,49
import math
def snd(x):
   return math.e**(-x*x/2)/math.sqrt(2*math.pi)
def integr(a,b):
   n=10;h=(b-a)/n
   x=a
   som=0
   for j in range(1,n):
      x=x+h
      if j%2==0:
         t=2
      else:
         t=4
      som=som+t*snd(x)
   som=som+snd(a)+snd(b)
   integ=som*h/3
   return(integ)
# s(abc) staat voor P(-oo < Zi <a,bc)
# vb: s(262)=0,9955 betekent P(-oo < Zi <2,62)=0,9955
s=['0.5000'];totint=0.5
for i in range(0,349):
   totint=totint+integr(i/100,(i+1)/100)
   s.append(format(totint,'.4f'))
hoofd='Zi'
for i in range(0,10):
   hoofd=hoofd+'   '+format(i/100,'.2f')
print(hoofd)
for i in range(0,35):
   lijn=str(i/10)
   for j in range(0,10):
      lijn=lijn+' '+s[10*i+j]
   print(lijn)
```

>>>

Zi	0.00	0.01	0.02	0.03	0.04	0.05	0.06	0.07	0.08	0.09
0.0	0.5000	0.5040	0.5080	0.5120	0.5160	0.5199	0.5239	0.5279	0.5319	0.5359
0.1	0.5398	0.5438	0.5478	0.5517	0.5557	0.5596	0.5636	0.5675	0.5714	0.5753
0.2	0.5793	0.5832	0.5871	0.5910	0.5948	0.5987	0.6026	0.6064	0.6103	0.6141
0.3	0.6179	0.6217	0.6255	0.6293	0.6331	0.6368	0.6406	0.6443	0.6480	0.6517
0.4	0.6554	0.6591	0.6628	0.6664	0.6700	0.6736	0.6772	0.6808	0.6844	0.6879
0.5	0.6915	0.6950	0.6985	0.7019	0.7054	0.7088	0.7123	0.7157	0.7190	0.7224
0.6	0.7257	0.7291	0.7324	0.7357	0.7389	0.7422	0.7454	0.7486	0.7517	0.7549
0.7	0.7580	0.7611	0.7642	0.7673	0.7704	0.7734	0.7764	0.7794	0.7823	0.7852
0.8	0.7881	0.7910	0.7939	0.7967	0.7995	0.8023	0.8051	0.8078	0.8106	0.8133
0.9	0.8159	0.8186	0.8212	0.8238	0.8264	0.8289	0.8315	0.8340	0.8365	0.8389
1.0	0.8413	0.8438	0.8461	0.8485	0.8508	0.8531	0.8554	0.8577	0.8599	0.8621
1.1	0.8643	0.8665	0.8686	0.8708	0.8729	0.8749	0.8770	0.8790	0.8810	0.8830
1.2	0.8849	0.8869	0.8888	0.8907	0.8925	0.8944	0.8962	0.8980	0.8997	0.9015
1.3	0.9032	0.9049	0.9066	0.9082	0.9099	0.9115	0.9131	0.9147	0.9162	0.9177
1.4	0.9192	0.9207	0.9222	0.9236	0.9251	0.9265	0.9279	0.9292	0.9306	0.9319
1.5	0.9332	0.9345	0.9357	0.9370	0.9382	0.9394	0.9406	0.9418	0.9429	0.9441
1.6	0.9452	0.9463	0.9474	0.9484	0.9495	0.9505	0.9515	0.9525	0.9535	0.9545
1.7	0.9554	0.9564	0.9573	0.9582	0.9591	0.9599	0.9608	0.9616	0.9625	0.9633
1.8	0.9641	0.9649	0.9656	0.9664	0.9671	0.9678	0.9686	0.9693	0.9699	0.9706
1.9	0.9713	0.9719	0.9726	0.9732	0.9738	0.9744	0.9750	0.9756	0.9761	0.9767
2.0	0.9772	0.9778	0.9783	0.9788	0.9793	0.9798	0.9803	0.9808	0.9812	0.9817
2.1	0.9821	0.9826	0.9830	0.9834	0.9838	0.9842	0.9846	0.9850	0.9854	0.9857
2.2	0.9861	0.9864	0.9868	0.9871	0.9875	0.9878	0.9881	0.9884	0.9887	0.9890
2.3	0.9893	0.9896	0.9898	0.9901	0.9904	0.9906	0.9909	0.9911	0.9913	0.9916
2.4	0.9918	0.9920	0.9922	0.9925	0.9927	0.9929	0.9931	0.9932	0.9934	0.9936
2.5	0.9938	0.9940	0.9941	0.9943	0.9945	0.9946	0.9948	0.9949	0.9951	0.9952
2.6	0.9953	0.9955	0.9956	0.9957	0.9959	0.9960	0.9961	0.9962	0.9963	0.9964
2.7	0.9965	0.9966	0.9967	0.9968	0.9969	0.9970	0.9971	0.9972	0.9973	0.9974
2.8	0.9974	0.9975	0.9976	0.9977	0.9977	0.9978	0.9979	0.9979	0.9980	0.9981
2.9	0.9981	0.9982	0.9982	0.9983	0.9984	0.9984	0.9985	0.9985	0.9986	0.9986
3.0	0.9987	0.9987	0.9987	0.9988	0.9988	0.9989	0.9989	0.9989	0.9990	0.9990
3.1	0.9990	0.9991	0.9991	0.9991	0.9992	0.9992	0.9992	0.9992	0.9993	0.9993
3.2	0.9993	0.9993	0.9994	0.9994	0.9994	0.9994	0.9994	0.9995	0.9995	0.9995
3.3	0.9995	0.9995	0.9995	0.9996	0.9996	0.9996	0.9996	0.9996	0.9996	0.9997
3.4	0.9997	0.9997	0.9997	0.9997	0.9997	0.9997	0.9997	0.9997	0.9997	0.9998

>>>

Uit deze tabel kan je bvb. aflezen dat 99% van de gegevens een Z_i hebben die kleiner is dan 2,33.

Financiële algebra

25. Samengestelde intrest : menu samengestelde_intrest_menu

De formule voor het eindkapitaal is $K_n = K_0 \cdot \left(1 + \frac{R}{100}\right)^n$

Afhankelijk van de gestelde vraag kan je hieruit elke onbekende berekenen.

Zo is $K_0 = \frac{K_n}{\left(1+\frac{R}{100}\right)^n}$ $R = 100 \cdot \left(\sqrt[n]{\frac{K_n}{K_0}} - 1\right)$ $n = \frac{\log(K_n) - \log(K_0)}{\log(1+\frac{R}{100})}$

Het programma geeft je een keuzemenu met deze 4 mogelijkheden.

Programma:

```
# menu samengestelde intrest
import math
def mult(s):
   l=s.split(',')
   co=[]
   for i in l:co.append(float(i))
   return co
print('Samengestelde intrest: kn=k0(1+r/100)**n')
print('Beginkapitaal=k0, Eindkapitaal=kn, Periode=n, Rentevoet=r')
print('1. kn    2. k0    3. n    4. r    5. einde')
keuze=1
while keuze in [1,2,3,4]:
   keuze=int(input('Kies: 1 2 3 4 5  :'))
   if keuze==1:
      k0,n,r=mult(input('Geef k0, n, r:'))
      kn=k0*(1+r/100)**n
      print('Eindkapitaal kn= ',format(kn,'.2f'))
   if keuze==2:
      kn,n,r=mult(input('Geef kn, n, r:'))
      k0=kn/(1+r/100)**n
      print('Beginkapitaal k0= ',format(k0,'.2f'))
   if keuze==3:
      kn,k0,r=mult(input('Geef kn, k0, r:'))
      n=(math.log(kn)-math.log(k0))/(math.log(1+r/100))
      print('Periode n= ',format(n,'.2f'))
   if keuze==4:
      kn,k0,n=mult(input('Geef kn, k0, n:'))
      r=100*((kn/k0)**(1/n)-1)
      print('Rentevoet r= ',format(r,'.2f'))
print('Einde')
>>>
Samengestelde intrest: kn=k0(1+r/100)**n
Beginkapitaal=k0, Eindkapitaal=kn, Periode=n, Rentevoet=r
1. kn    2. k0    3. n    4. r    5. einde
Kies: 1 2 3 4 5  :1
Geef k0, n, r:5000,10,3
Eindkapitaal kn=  6719.58
Kies: 1 2 3 4 5  :2
Geef kn, n, r:8000,5,4
Beginkapitaal k0=  6575.42
Kies: 1 2 3 4 5  :3
Geef kn, k0, r:7000,5000,4
```

Periode n= 8.58
Kies: 1 2 3 4 5 :4
Geef kn, k0, n:7000,5000,10
Rentevoet r= 3.42
Kies: 1 2 3 4 5 :5
Einde
>>>

26. Lening met constante annuïteiten annuiteitstabel

Hypothecaire leningen (voor de aankoop van een woning) worden meestal afgesloten met het principe van de constante annuïteiten.
Een lening wordt steeds afbetaald door periodiek (jaarlijks of maandelijks) een gedeelte van het kapitaal plus de intrest op het schuldsaldo te betalen. Deze som moet constant blijven.
Stel geleend bedrag=K periode=N rentevoet=R

Ter vereenvoudiging stellen we $\frac{R}{100} = i \quad en\ 1+i = u$

Op het einde van elke periode (jaar of maand) wordt een deel van het kapitaal terugbetaald ($K_1 \ldots K_n$) en intrest ($I_1 \ldots I_n$) . Er blijft dan telkens een schuldsaldo over ($K-K_1$, $K-K_1-K_2$,.. 0)
Er wordt steeds intrest betaald op het vorige schuldsaldo.
De 1ste keer is het schuldsaldo nog het ganse kapitaal = K, de 2de keer = $S_1= K-K_1$, de 3de keer = $S_2= K-K_1-K_2$
De intrest is dus
$I_1 = K.I$ $I_2 = S_1.i = (K-K_1).i$ $I_3 = S_2.i = (K-K_1-K_2).i$ enz..
Omdat de afbetaling steeds gelijk blijft is $K_1+I_1 = K_2+I_2 = K_3+I_3 = \ldots.$
Vul in: $K_1+ K.i = K_2+(K-K_1).i = K_3+(K-K_1-K_2).i$
Hieruit volgt $K_2= K_1(1+i)$ $K_3=K_2(1+i)$ $K_4=K_3(1+i) \ldots.$
Of $K_2= K_1.u$ $K_3=K_2.u$ $K_4=K_3.u$
We hebben nu al formules om $I_1 \ldots I_n$ te berekenen en formules om $K_2 \ldots K_n$ te berekenen, uitgaande van K_1.
Enkel een formule voor K_1 ontbreekt nog.
Om deze te vinden zullen we alle K_i uitdrukken in functie van K_1 en u.
$K_2= K_1.u$ $K_3=K_2.u = K_1.u^2$ $K_4=K_3.u = K_1.u^3 \ldots$ $K_n=K_1.u^{n-1}$
De som van alle terugbetaalde kapitalen = geleende kapitaal, dus
$K_1+K_2+ \ldots K_n = K$
Hierin kunnen we invullen en K_1 buiten haakjes brengen:
$K_1(1 +u +u^2+u^3+\ldots+u^{n-1}) = K$ (1)
Vermenigvuldig beide leden met u
$K_1(u +u^2+u^3+u^4+\ldots+u^n) = K.u$ (2)
Trek vergelijking (1) lid aan lid af van vergelijking (2):
$K_1 \cdot (u^n-1) = K \cdot (u-1)$ waaruit we K_1 berekenen.

$$K_1 = \frac{K \cdot (u-1)}{u^n - 1}$$

We brengen alle formules over naar een tabel

Periode	Intrest	Kapitaal	Saldo
1	K.i	$K.(u-1)/(u^n-1)$	$K-K_1$
2	$(K-K_1).i$	$K_1.u$	$K-K_1-K_2$
3	$(K-K_1-K_2).i$	$K_2.u$	$K-K_1-K_2-K_3$
…		……	…
n	……		0

Je merkt dat je enkel de 1ste lijn afzonderlijk moet berekenen.

Voor alle volgende lijnen geldt steeds:
Intrest = vorig schuldsaldo . i
Kapitaalsaflossing = vorige kapitaalsaflossing .u
Schuldsaldo= vorig schuldsaldo- laatste kapitaalsaflossing

Het volgende programma berekent de aflossingstabel van een lening met constante annuïteiten.

Programma:

```
# annuiteiten
from tabulate import tabulate
def lijn(i):
    return [round(intrest[i],2),round(kapafl[i],2),round(saldo[i],2)]
k=float(input("Leenkapitaal:"))
r=float(input("Rentevoet:"))
n=int(input("Periode:"))
intrest=[];kapafl=[];saldo=[]
i=r/100;u=1+i
kapafl.append(i*k/(u**n-1))
intrest.append(i*k)
saldo.append(k-kapafl[0])
print("Intrest   Kapitaal  Saldo")
tabel=[]
tabel.append(lijn(0))
for j in range(1,n):
    intrest.append(i*saldo[j-1])
    kapafl.append(u*kapafl[j-1])
    saldo.append(saldo[j-1]-kapafl[j])
    tabel.append(lijn(j))
print(tabulate(tabel))
print("Periodieke afbetaling=",format(intrest[0]+kapafl[0],'.2f'))
>>>
Leenkapitaal:80000
Rentevoet:4
Periode:5
Intrest  Kapitaal Saldo
-------  -------- -------
3200     14770.2  65229.8
2609.19  15361    49868.9
1994.75  15975.4  33893.4
1355.74  16614.4  17279
 691.16  17279        0
-------  -------  -------
Periodieke afbetaling= 17970.17
>>>
```

27. Annuïteitsmenu annuiteitsmenu

De afbetaling **A** van een lening met constante annuïteiten blijft steeds gelijk, m.a.w.

$A=I_1+K_1$ $\qquad I_1=\frac{K.R}{100}$ $\qquad u=1+\frac{r}{100}$

$$K_1 = \frac{K(u-1)}{(u^n-1)} = \frac{K.R}{100}.\frac{1}{\left(\left(1+\frac{R}{100}\right)^n-1\right)} \qquad A = \frac{K.R}{100}.\left(1+\frac{1}{\left(\left(1+\frac{R}{100}\right)^n-1\right)}\right)$$

Omgekeerd kan je hier formules uit afleiden voor K en n:

$$K = \frac{100.A}{R.\left(1+\frac{1}{\left(\left(1+\frac{R}{100}\right)^n-1\right)}\right)} \qquad n = \frac{\log\left(\frac{1}{\frac{100.A}{K.R}-1}+1\right)}{\log\left(1+\frac{R}{100}\right)}$$

Een formule om R te berekenen is onmogelijk want R is een onbekende in een niet lineaire vergelijking. Maar we benaderen R met de methode van Newton als nulpunt van de verschilfunctie:

$$\frac{K.R}{100}.\left(1+\frac{1}{\left(\left(1+\frac{R}{100}\right)^n-1\right)}\right)-A$$

Programma:

```
# menu annuïteiten
import math
def mult(s):
   l=s.split(',');co=[]
   for i in l:co.append(float(i))
   return co
def f(r):return a-k*r/100*(1+(1/((1+r/100)**n-1)))
def df(r):
   dr=0.00001
   return (f(r+dr)-f(r))/dr
print('Constante annuïteiten:\nGeleend kapitaal=k, Afbetaling=a, Periode=n, Rentevoet=r, Einde=e')
print('a = k.(r/100).(1+1/((1+r/100)**n-1))  wat wil je berekenen?')
keuze='k'
while keuze in ['k','a','n','r']:
   keuze=input('Kies: k  a  n  r  e :')
   if keuze=='k':
      a,n,r=mult(input('Geef a, n, r:'))
      k=100*a/r/(1+(1/((1+r/100)**n-1)))
      print('Geleend kapitaal k= ',format(k,'.2f'))
   if keuze=='a':
      k,n,r=mult(input('Geef k, n, r:'))
      a=k*r/100*(1+(1/((1+r/100)**n-1)))
      print('Periodieke afbetaling a= ',format(a,'.2f'))
   if keuze=='n':
```

```
        k,a,r=mult(input('Geef k, a, r:'))
        n=math.log(1/(100*a/k/r-1)+1)/math.log(1+r/100)
        print('Periode n= ',format(n,'.2f'))
    if keuze=='r':
        k,a,n=mult(input('Geef k, a, n:'))
        # newton
        r=-0.9
        while f(r)*f(r+1)>0:r=r+1
        oudr=r+1
        while abs(oudr-r)>0.0001:
            oudr=r;r=r-f(r)/df(r)
        print('Rentevoet r= ',format(r,'.2f'))
print('Einde')
>>>
Constante annuïteiten:
Geleend kapitaal=k, Afbetaling=a, Periode=n, Rentevoet=r, Einde=e
a = k.(r/100).(1+1/((1+r/100)**n-1))  wat wil je berekenen?
Kies:  k   a   n   r   e  :k
Geef a, n, r:5000,25,3
Geleend kapitaal k=  87065.74
Kies:  k   a   n   r   e  :a
Geef k, n, r:80000,20,2.5
Periodieke afbetaling a=  5131.77
Kies:  k   a   n   r   e  :n
Geef k, a, r:80000,5132,2.5
Periode n=  20.00
Kies:  k   a   n   r   e  :r
Geef k, a, n:80000,5132,20
Rentevoet r=  2.50
Kies:  k   a   n   r   e  :e
Einde
```

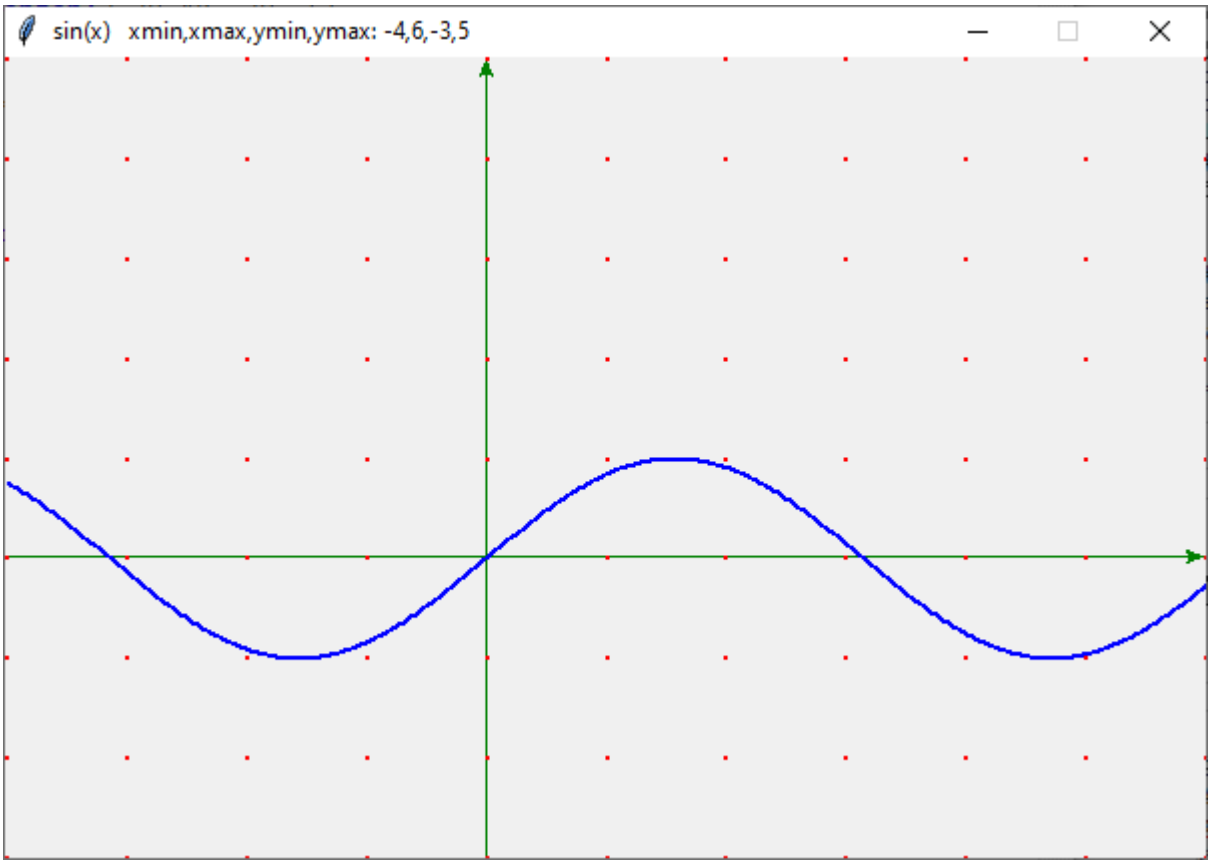
Grafische programma's

28. Basisprogramma: tekenen van een functie. grafiek_basis

Met Python kunnen we relatief eenvoudig grafieken van wiskundige functies tekenen.
Het grafische venster zelf is een variabele,
bvb. win = GraphWin(hoofding,600,400)
hoofding is ook een zelfgekozen variabele die je kunt opvullen met een string die informatie geeft over de grafiek.
Deze hoofding komt letterlijk boven het venster.
600 en 400 zijn de breedte en hoogte van het grafisch venster.
Dit kan je zelf kiezen, of met variabelen door de gebruiker laten kiezen.
Je hoeft geen transformatieformules te gebruiken om van de wiskundige coördinaten over te gaan naar de vensterco̲ördinaten: win.setCoords(xmin,ymin,xmax,ymax) zorgt hiervoor.
(coördinaten linksonder en rechtsboven). Hoe de grafiek van assen, rooster en functie getekend wordt, is gemakkelijk leesbaar in het programma zelf.

Programma:

```
# grafiek van een wiskundige functie
from graphics import *
from math import *
# meervoudige invoer
def mult(s):
    l=s.split(',')
    co=[]
    for i in l:co.append(float(i))
    return co
def f(x):
    return eval(fs)
# hoofdprogramma
fs=input("Geef een functie:")
print("Assenstelsel: ")
xy='xmin,xmax,ymin,ymax:';assen=input(xy)
xmin,xmax,ymin,ymax=mult(assen)
hoofding=fs+'   '+xy+' '+assen
win = GraphWin(hoofding,600,400)
win.setCoords(xmin,ymin,xmax,ymax)
#assen
pl=Point(xmin,0);pr=Point(xmax,0);lix=Line(pl,pr);lix.setFill('green')
po=Point(0,ymin);pb=Point(0,ymax);liy=Line(po,pb);liy.setFill('Green')
lix.setArrow('last');liy.setArrow('last');lix.draw(win);liy.draw(win)
# rooster
for x in range(int(xmin),int(xmax+1)):
    for y in range(int(ymin),int(ymax+1)):
        pt=Point(x,y);pt.setFill('Red');pt.draw(win)
# grafiek
stap=0.02;x=xmin
while x<xmax:
    x=x+stap;y=f(x);pt=Point(x,y);pt.setFill('Blue');pt.draw(win)
>>>
Geef een functie:sin(x)
Assenstelsel:
xmin,xmax,ymin,ymax:-4,6,-3,5
>>>
```



29. Grafieken van een functie en de 1ste en 2de afgeleide functie

grafiekfuncties_f_df_d2f

Je kan op een eenvoudige manier een benadering van de 1ste en 2de afgeleide definiëren van een gegeven functie. Stel dx =0.00001 (klein)

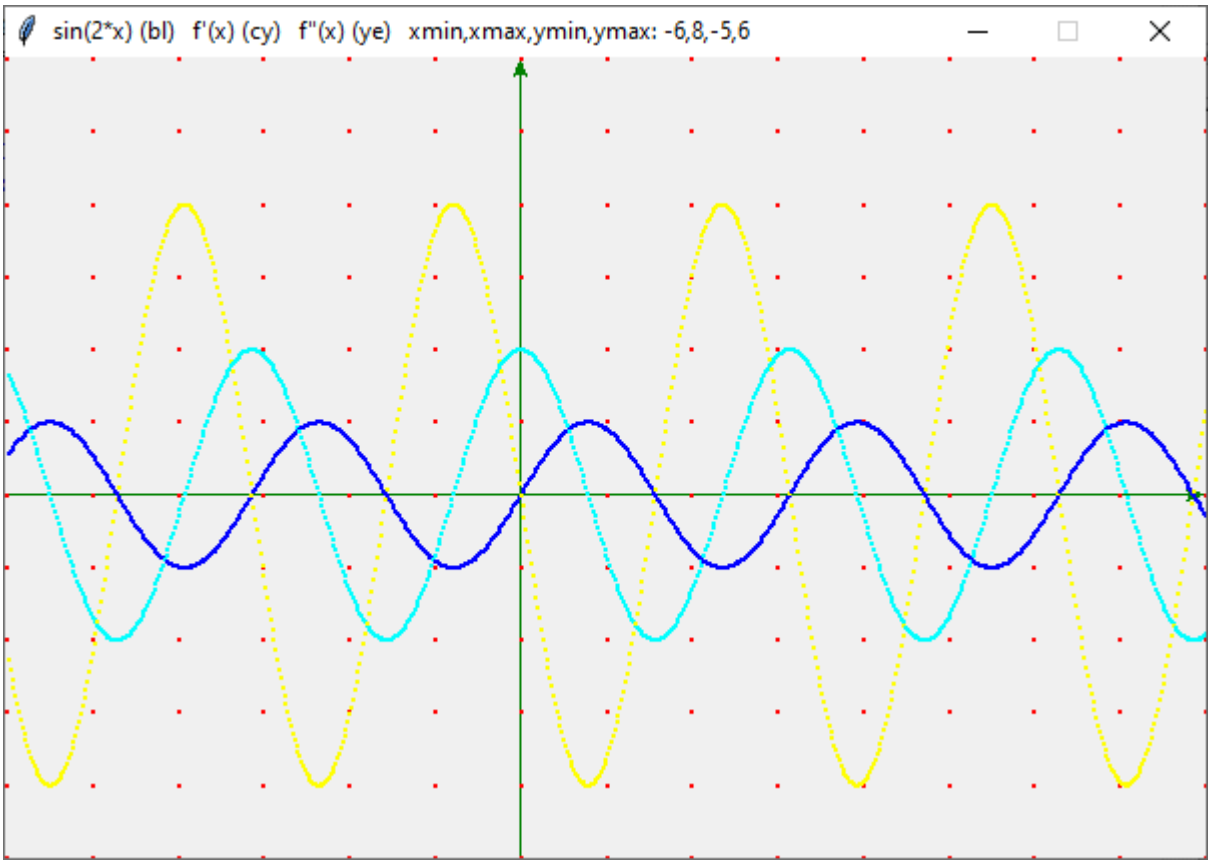
$$f'(x) \cong df(x) = \frac{f(x+dx)-f(x)}{dx} \quad \text{en} \quad f''(x) \cong d2f(x) = \frac{df(x+dx)-df(x)}{dx}$$

Deze benadering is voldoende nauwkeurig om de grafiek te tekenen.
Voegen we nu enkele lijntjes toe in het vorige programma.

Programma:

```
#grafieken van een functie en haar afgeleide functies
from graphics import *
from math import *
# meervoudige invoer
def mult(s):
    l=s.split(',')
    co=[]
    for i in l:co.append(float(i))
    return co
def f(x):
    return eval(fs)
def df(x):
    dx=0.00001
    return (f(x+dx)-f(x))/dx
```

```
def d2f(x):
    dx=0.00001
    return (df(x+dx)-df(x))/dx
# hoofdprogramma
fs=input("Geef een functie:")
print("Assenstelsel: ")
xy='xmin,xmax,ymin,ymax:';assen=input(xy)
xmin,xmax,ymin,ymax=mult(assen)
hoofding=fs+' (bl)   f\'(x) (cy)   f\''(x) (ye)   '+xy+' '+assen
win = GraphWin(hoofding,600,400)
win.setCoords(xmin,ymin,xmax,ymax)
#assen
pl=Point(xmin,0);pr=Point(xmax,0);lix=Line(pl,pr);lix.setFill('green')
po=Point(0,ymin);pb=Point(0,ymax);liy=Line(po,pb);liy.setFill('Green')
lix.setArrow('last');liy.setArrow('last');lix.draw(win);liy.draw(win)
# rooster
for x in range(int(xmin),int(xmax+1)):
    for y in range(int(ymin),int(ymax+1)):
        pt=Point(x,y);pt.setFill('Red');pt.draw(win)
# grafiek
stap=0.02;x=xmin
while x<xmax:
    x=x+stap
    y=f(x);pt=Point(x,y);pt.setFill('Blue');pt.draw(win)
    y=df(x);pt=Point(x,y);pt.setFill('Cyan');pt.draw(win)
    y=d2f(x);pt=Point(x,y);pt.setFill('Yellow');pt.draw(win)
>>>
Geef een functie:sin(2*x)
Assenstelsel:
xmin,xmax,ymin,ymax:-6,8,-5,6
>>>
```



30. Grafieken van de verschillende transformaties van een functie

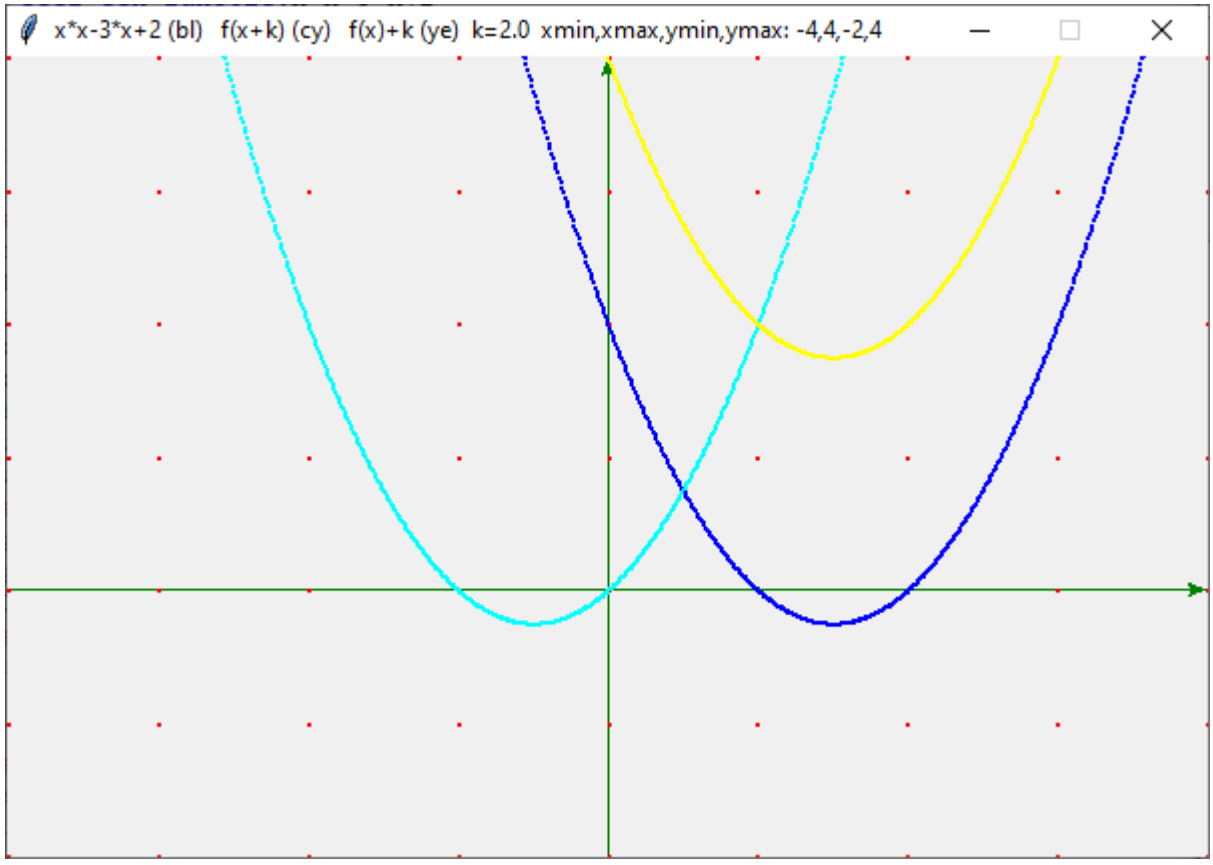
grafiek_transformaties

Terug uitgaande van het eerste grafisch programma kunnen we de invloed van een parameter k op de grafiek van een functie onderzoeken.
Wat gebeurt er met de grafiek als we van f(x) naar f(x+k) overgaan, of van f(x) naar f(x)+k ?
Een paar kleine wijzigingen en we hebben een nieuw programma.

Programma:

```
# grafieken van f(x), f(x+k), f(x)+k
from graphics import *
from math import *
# meervoudige invoer
def mult(s):
    l=s.split(',')
    co=[]
    for i in l:co.append(float(i))
    return co
def f(x):
    return eval(fs)
# hoofdprogramma
fs=input("Geef een functie:")
k=float(input('Geef k:'))
print("Assenstelsel: ")
xy='xmin,xmax,ymin,ymax:';assen=input(xy)
xmin,xmax,ymin,ymax=mult(assen)
```

```
hoofding=fs+' (bl)   f(x+k) (cy)   f(x)+k (ye) '+' k='+str(k)+'  '+xy+' '+assen
win = GraphWin(hoofding,600,400)
win.setCoords(xmin,ymin,xmax,ymax)
#assen
pl=Point(xmin,0);pr=Point(xmax,0);lix=Line(pl,pr);lix.setFill('green')
po=Point(0,ymin);pb=Point(0,ymax);liy=Line(po,pb);liy.setFill('Green')
lix.setArrow('last');liy.setArrow('last');lix.draw(win);liy.draw(win)
# rooster
for x in range(int(xmin),int(xmax+1)):
    for y in range(int(ymin),int(ymax+1)):
        pt=Point(x,y);pt.setFill('Red');pt.draw(win)
# grafiek
stap=0.01;x=xmin
while x<xmax:
    x=x+stap;y=f(x);pt=Point(x,y);pt.setFill('Blue');pt.draw(win)
    y=f(x+k);pt=Point(x,y);pt.setFill('Cyan');pt.draw(win)
    y=f(x)+k;pt=Point(x,y);pt.setFill('Yellow');pt.draw(win)
>>>
Geef een functie:x*x-3*x+2
Geef k:2
Assenstelsel:
xmin,xmax,ymin,ymax:-4,4,-2,4
>>>
```



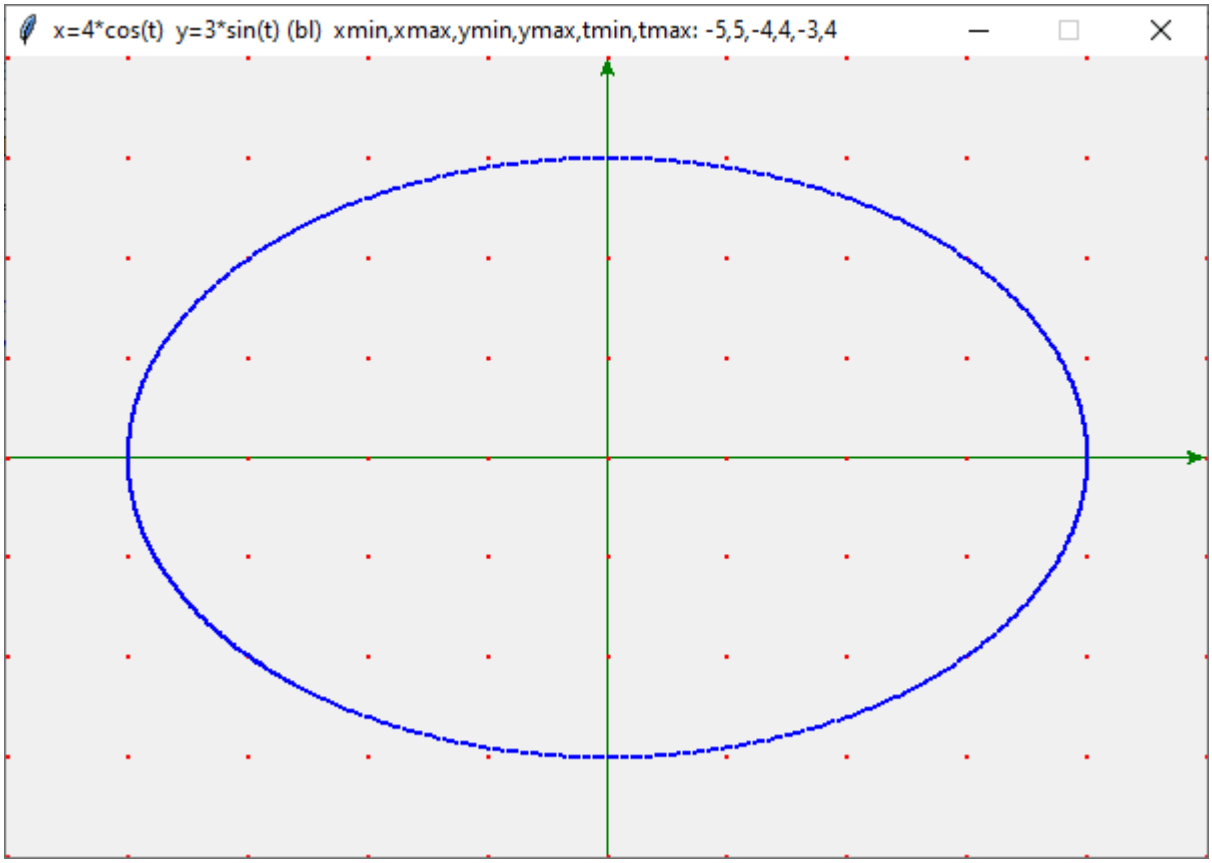
31. Grafiek van parametervergelijkingen

grafiek_parameter_vergl

Ook parametervergelijkingen kunnen gemakkelijk getekend worden. Je moet maar enkele lijntjes veranderen aan het programma van een functie.
De gebruiker moet nu 2 functies invoeren : x en y in functie van de parameter t.
Bovendien moet hij tmin en tmax invoeren
Om de grafiek te tekenen ook nu een while loop (while t<tmax:)

Programma:

```
# grafieken van parametervergelijkingen
from graphics import *
from math import *
# meervoudige invoer
def mult(s):
    l=s.split(',')
    co=[]
    for i in l:co.append(float(i))
    return co
def xp(t):
    return eval(fx)
def yp(t):
    return eval(fy)
# hoofdprogramma
fx,fy=input("Geef x,y als een functie van t:").split(',')
print("Assenstelsel: ")
xyt='xmin,xmax,ymin,ymax,tmin,tmax:';assen=input(xyt)
xmin,xmax,ymin,ymax,tmin,tmax=mult(assen)
hoofding='x='+fx+'  y='+fy+' (bl)  '+xyt+' '+assen
win = GraphWin(hoofding,600,400)
win.setCoords(xmin,ymin,xmax,ymax)
#assen
pl=Point(xmin,0);pr=Point(xmax,0);lix=Line(pl,pr);lix.setFill('green')
po=Point(0,ymin);pb=Point(0,ymax);liy=Line(po,pb);liy.setFill('Green')
lix.setArrow('last');liy.setArrow('last');lix.draw(win);liy.draw(win)
# rooster
for x in range(int(xmin),int(xmax+1)):
    for y in range(int(ymin),int(ymax+1)):
        pt=Point(x,y);pt.setFill('Red');pt.draw(win)
# grafiek
stap=0.01;t=tmin
while t<tmax:
    t=t+stap;pt=Point(xp(t),yp(t));pt.setFill('Blue');pt.draw(win)
>>>
Geef x,y als een functie van t:4*cos(t),3*sin(t)
Assenstelsel:
xmin,xmax,ymin,ymax,tmin,tmax:-5,5,-4,4,-3,4
```



32. Grafiek van poolvergelijkingen

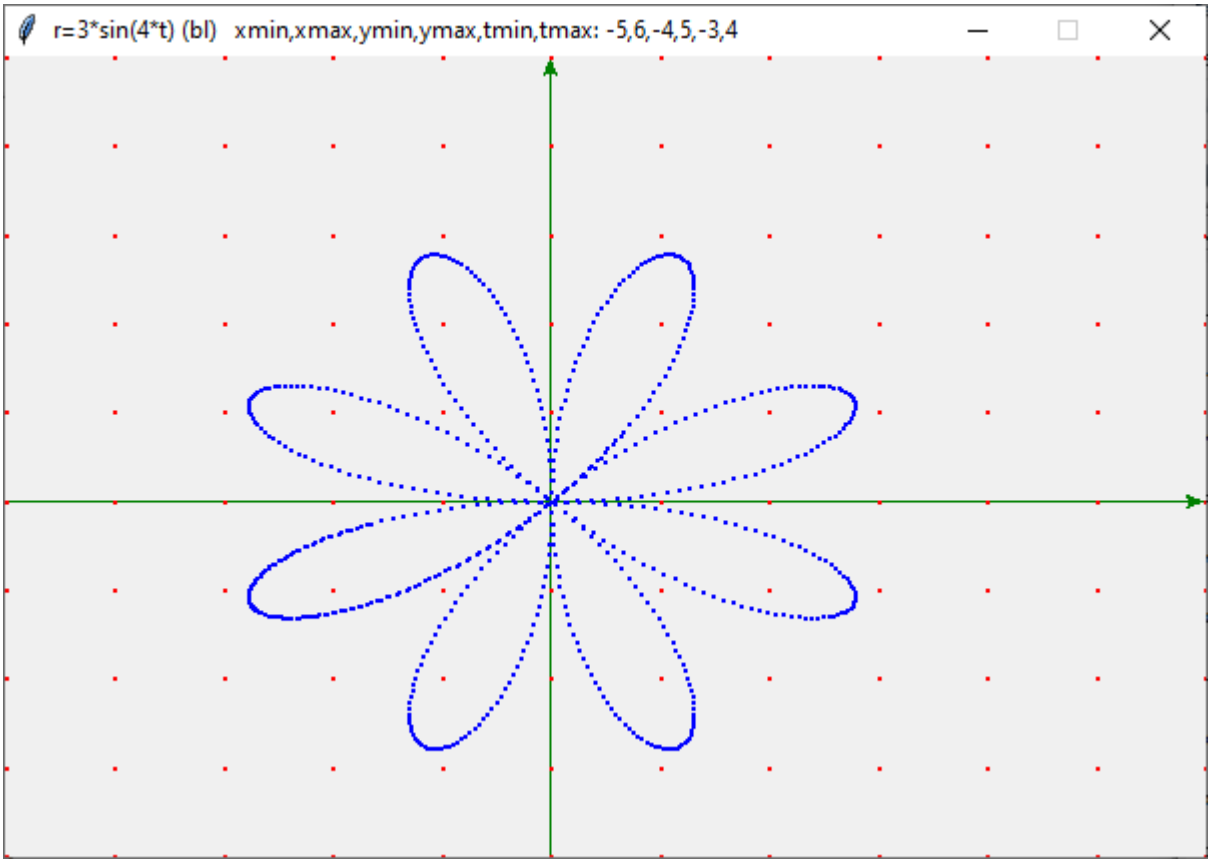
grafiek_poolvergl

Het verband tussen de poolvergelijking r=f(t) en (x,y) wordt gegeven door x=r.cos(t) en y=r.sin(t). De gebruiker moet nu 1 functie r=f(t) invoeren. Ook hier moet hij tmin en tmax invoeren .

Programma:

```
# grafieken van poolvergelijkingen
from graphics import *
from math import *
# meervoudige invoer
def mult(s):
    l=s.split(',')
    co=[]
    for i in l:co.append(float(i))
    return co
def r(t):
    return eval(fs)
# hoofdprogramma
fs=input("Geef een functie van t:")
print("Assenstelsel: ")
xyt='xmin,xmax,ymin,ymax,tmin,tmax:';assen=input(xyt)
xmin,xmax,ymin,ymax,tmin,tmax=mult(assen)
hoofding='r='+fs+' (bl)   '+xyt+' '+assen
```

```
win = GraphWin(hoofding,600,400)
win.setCoords(xmin,ymin,xmax,ymax)
#assen
pl=Point(xmin,0);pr=Point(xmax,0);lix=Line(pl,pr);lix.setFill('green')
po=Point(0,ymin);pb=Point(0,ymax);liy=Line(po,pb);liy.setFill('Green')
lix.setArrow('last');liy.setArrow('last');lix.draw(win);liy.draw(win)
# rooster
for x in range(int(xmin),int(xmax+1)):
    for y in range(int(ymin),int(ymax+1)):
        pt=Point(x,y);pt.setFill('Red');pt.draw(win)
# grafiek
stap=0.01;t=tmin
while t<tmax:
    t=t+stap;x=r(t)*cos(t);y=r(t)*sin(t);pt=Point(x,y)
    pt.setFill('Blue');pt.draw(win)

>>>
Geef een functie van t:3*sin(4*t)
Assenstelsel:
xmin,xmax,ymin,ymax,tmin,tmax:-5,6,-4,4,-3,4
>>>
```



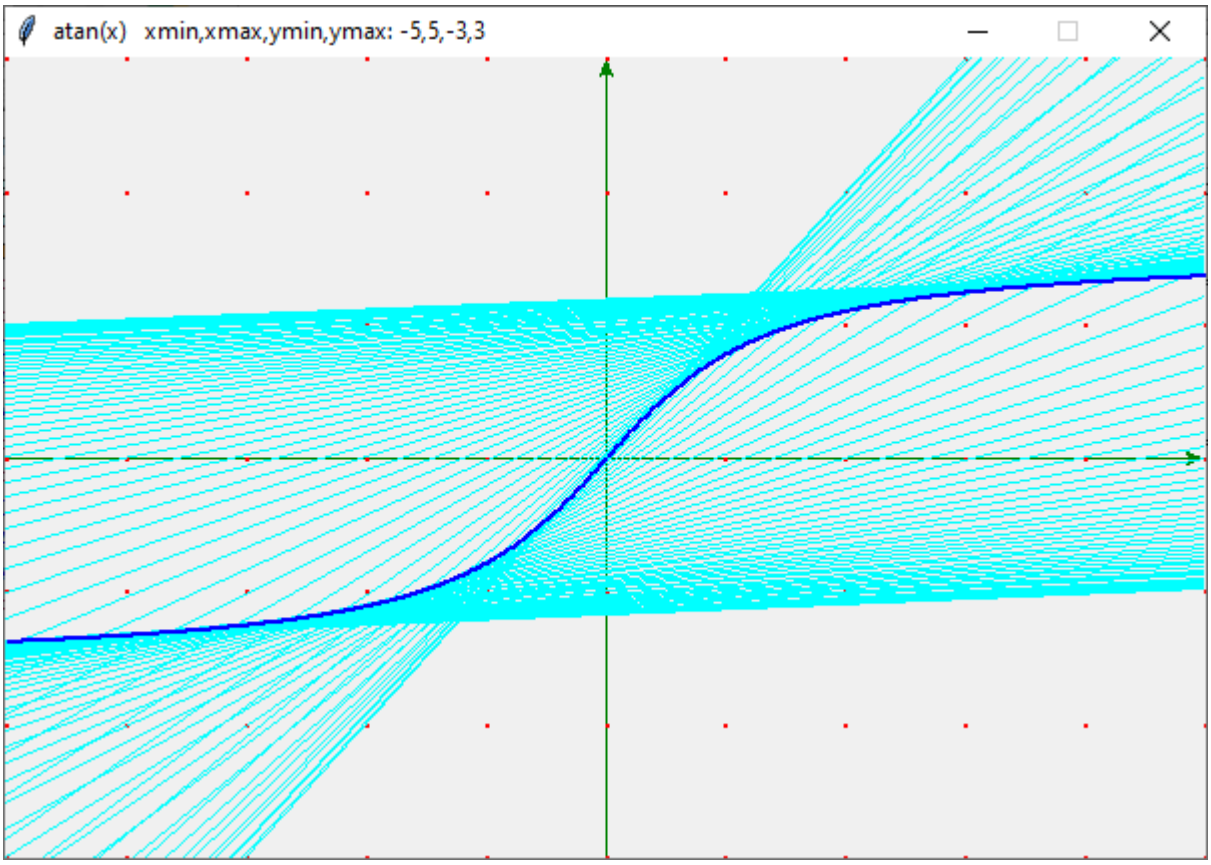
33. Een programma om raaklijnen te tekenen raaklijn

We tekenen eerst een aantal raaklijnen. Pas nadien tekenen we de bijhorende kromme.
Stel (x1,f(x1)) = punt van de kromme ==> vergelijking van de raaklijn is:

y-f(x1)=f'(x1)(x-x1) waaruit volgt y = r(x)= f'(x1).(x-x1)+f(x1))
Voor een aantal punten (x1,f(x1) van de kromme tekenen we de raaklijnen door het punt links pl (xmin,r(xmin))en het punt rechts pr(xmax,r(xmax))
Dan is pl = Point(xmin,r(xmin)) en pr = Point(xmax,r(xmax))

Programma:

```
# grafiek van een functie en haar raaklijnen
from graphics import *
from math import *
# meervoudige invoer
def mult(s):
   l=s.split(',')
   co=[]
   for i in l:co.append(float(i))
   return co
def f(x):
   return eval(fs)
def df(x):
   dx=0.00001
   return (f(x+dx)-f(x))/dx
# hoofdprogramma
fs=input("Geef een functie:")
print("Assenstelsel: ")
xy='xmin,xmax,ymin,ymax:';assen=input(xy)
xmin,xmax,ymin,ymax=mult(assen)
hoofding=fs+'   '+xy+' '+assen
win = GraphWin(hoofding,600,400)
win.setCoords(xmin,ymin,xmax,ymax)
#assen
pl=Point(xmin,0);pr=Point(xmax,0);lix=Line(pl,pr);lix.setFill('green')
po=Point(0,ymin);pb=Point(0,ymax);liy=Line(po,pb);liy.setFill('Green')
lix.setArrow('last');liy.setArrow('last');lix.draw(win);liy.draw(win)
# rooster
for x in range(int(xmin),int(xmax+1)):
   for y in range(int(ymin),int(ymax+1)):
      pt=Point(x,y);pt.setFill('Red');pt.draw(win)
#raaklijnen
stap1=0.1;x1=xmin-stap1
while x1<xmax:
   x1=x1+stap1
   x=xmin;y=df(x1)*(x-x1)+f(x1);pl=Point(x,y)
   x=xmax;y=df(x1)*(x-x1)+f(x1);pr=Point(x,y)
   rkl=Line(pl,pr);rkl.setFill('cyan');rkl.draw(win)
# grafiek
stap=0.02;x=xmin
while x<xmax:
   x=x+stap;y=f(x);pt=Point(x,y);pt.setFill('Blue');pt.draw(win)
>>>
Geef een functie:atan(x)
Assenstelsel:
xmin,xmax,ymin,ymax:-5,5,-3,3
>>>
```



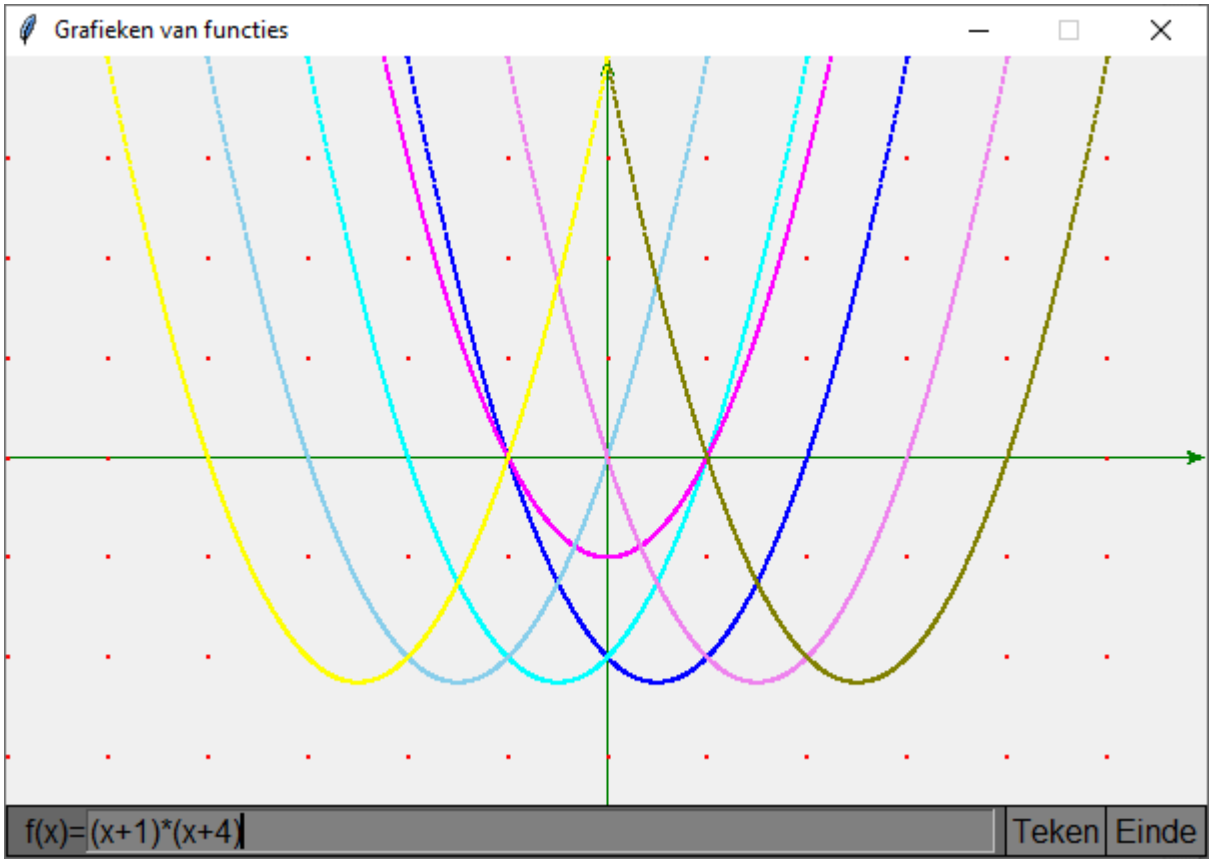
34. Grafieken van functies met entries en buttons grafiekbuttons_functie

Een andere manier om grafieken te tekenen is om een deel van het grafisch venster te gebruiken als invoervenster. In het volgende programma wordt hiervoor onderaan een rechthoek voorbehouden. In deze rechthoek komt de tekst 'f(x)=', gevolgd door een invoervenster (entry) waarin je strings (in dit geval functies) kunt invoeren. Aan het einde van de lijn heb je 2 rechthoekjes met de tekst 'Teken' en 'Einde'. Je kunt ze beschouwen als de opdracht-buttons uit bvb. Visual Basic.
puntklik=win.getMouse() ; xklik=puntklik.getX() ; yklik=puntklik.getY()
Met deze 3 instructies bevatten xklik en yklik de coördinaten van het punt waar met de muis geklikt is.
Is dit in de rechthoek 'Teken', dan wordt de grafiek getekend,waarvan het voorschrift ingevoerd is. Je kan de functie vervangen door een andere functie en dan zal ook deze bovenop de andere worden getekend in de volgende kleur (zie lijst kleur=[...]). Klik je in de rechthoek 'Einde' dan sluit het grafisch scherm.

Programma:

```
# grafieken van functies met entries en buttons
from graphics import *
from math import *
def f(x):
    return eval(fs)
#init
kleur=['blue','cyan','magenta','skyblue','violet','olive','yellow','pink']
breed=600;hoog=400
xmin=-6;xmax=6;ymin=-4;ymax=4
win = GraphWin('Grafieken van functies',breed,hoog)
```

```
win.setCoords(xmin,ymin,xmax,ymax)
# afmetingen onderste rechthoek
rhhoog=0.5;ymid=ymin+rhhoog/2;ytop=ymin+rhhoog
#assen
pl=Point(xmin,0);pr=Point(xmax,0);lix=Line(pl,pr);lix.setFill('green')
po=Point(0,ymin);pb=Point(0,ymax);liy=Line(po,pb);liy.setFill('Green')
lix.setArrow('last');liy.setArrow('last');lix.draw(win);liy.draw(win)
# rooster
for x in range(xmin,xmax):
   for y in range(ymin,ymax):
      pt=Point(x,y);pt.setFill('Red');pt.draw(win)
#teken rechthoek onderaan
prlo=Point(xmin,ymin);prrb=Point(xmax,ytop)
rh1=Rectangle(prlo,prrb);rh1.setFill('Grey40');rh1.draw(win)
t0=Text(Point(xmin+0.5,ymid),'f(x)=');t0.draw(win)
#teken entry blok
fblok=Entry(Point(-0.65,ymid),50);fblok.draw(win)
# teken buttonTeken
prlo2=Point(xmax-2,ymin);prrb2=Point(xmax-1,ytop)
rh2=Rectangle(prlo2,prrb2);rh2.setFill('Grey');rh2.draw(win)
t1=Text(Point(xmax-1.5,ymid),'Teken');t1.draw(win)
# teken buttonEinde
prlo3=Point(xmax-1,ymin);prrb3=Point(xmax,ytop)
rh3=Rectangle(prlo3,prrb3);rh3.setFill('Grey');rh3.draw(win)
t1=Text(Point(xmax-0.5,ymid),'Einde');t1.draw(win)
# als er op button 'Teken' geklikt wordt, grafiek tekenen
xklik=0;yklik=0;loop=1;kl=0
while loop==1:
   while xklik<xmax-2 or yklik>ytop:
      puntklik=win.getMouse()
      xklik=puntklik.getX();yklik=puntklik.getY()
   if xklik<xmax-1:
      fs=fblok.getText()
      stap=0.01;x=xmin
      while x<xmax:
         x =x+stap;y=f(x);pt=Point(x,y);pt.setFill(kleur[kl])
                  pt.draw(win);xklik=0;yklik=0;kl=kl+1
      if kl==len(kleur):kl=0
   else:
      loop=0
win.close()
```



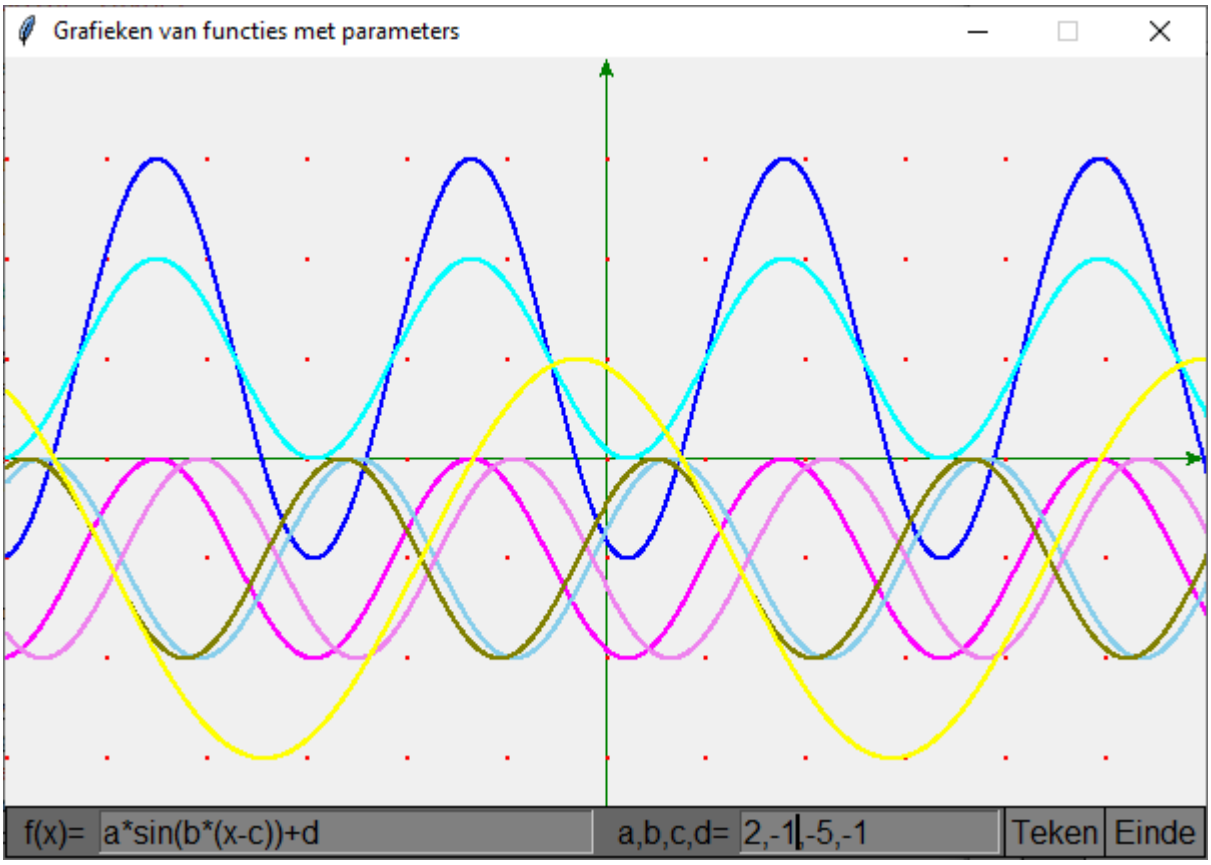
35. Variante met parameters grafiekbuttons_functie_param

Als we enkele lijntjes toevoegen aan het vorige programma, kunnen we de functiedefinitie voorzien van enkele parameters a,b,c,d. In een 2de invoervenster vullen we waarden voor a,b,c,d in. Door deze getallen te veranderen zien we hoe de functie getransformeerd wordt.
Dit kunnen we gebruiken bij de studie van de algemene sinusfunctie.

Programma:

```
# grafieken van functies (met parameters) met entries en buttons
from graphics import *
from math import *
# meervoudige invoer
def mult(s):
    l=s.split(',')
    co=[]
    for i in l:co.append(float(i))
    return co
def f(x):
    return eval(fs)
#init
kleur=['blue','cyan','magenta','skyblue','violet','olive','yellow','pink']
breed=600;hoog=400
xmin=-6;xmax=6;ymin=-4;ymax=4
win = GraphWin('Grafieken van functies met parameters',breed,hoog)
```

```
win.setCoords(xmin,ymin,xmax,ymax)
#afmetingen onderste rechthoek
rhhoog=0.5;ymid=ymin+rhhoog/2;ytop=ymin+rhhoog
#assen
pl=Point(xmin,0);pr=Point(xmax,0);lix=Line(pl,pr);lix.setFill('green')
po=Point(0,ymin);pb=Point(0,ymax);liy=Line(po,pb);liy.setFill('Green')
lix.setArrow('last');liy.setArrow('last');lix.draw(win);liy.draw(win)
#rooster
for x in range(xmin,xmax):
   for y in range(ymin,ymax):
      pt=Point(x,y);pt.setFill('Red');pt.draw(win)
#teken rechthoek onderaan
prlo=Point(xmin,ymin);prrb=Point(xmax,ytop);rh1=Rectangle(prlo,prrb)
rh1.setFill('Grey40');rh1.draw(win)
t0=Text(Point(xmin+0.5,ymid),'f(x)=');t1=Text(Point(xmin+6.7,ymid),'a,b,c,d=')
t0.draw(win);t1.draw(win)
#teken entry blok parameters
pblok=Entry(Point(2.65,ymid),14);pblok.draw(win)
#teken entry blok functie
fblok=Entry(Point(-2.6,ymid),27);fblok.draw(win)
#teken buttonTeken
prlo2=Point(xmax-2,ymin);prrb2=Point(xmax-1,ytop);rh2=Rectangle(prlo2,prrb2)
rh2.setFill('Grey');rh2.draw(win);t1=Text(Point(xmax-1.5,ymid),'Teken')
t1.draw(win)
#teken buttonEinde
prlo3=Point(xmax-1,ymin);prrb3=Point(xmax,ytop);rh3=Rectangle(prlo3,prrb3)
rh3.setFill('Grey');rh3.draw(win);t1=Text(Point(xmax-0.5,ymid),'Einde')
t1.draw(win)
#als er op button 'Teken' geklikt wordt, grafiek tekenen
xklik=0;yklik=0;loop=1;kl=0
while loop==1:
   while xklik<xmax-2 or yklik>ytop:
      puntklik=win.getMouse()
      xklik=puntklik.getX();yklik=puntklik.getY()
   if xklik<xmax-1:
      fs=fblok.getText()
      a,b,c,d=mult(pblok.getText())
      stap=0.01;x=xmin
      while x<xmax:
        x=x+stap;y=f(x);pt=Point(x,y)
                  pt.setFill(kleur[kl]);pt.draw(win)
      xklik=0;yklik=0;kl=kl+1
      if kl==len(kleur):kl=0
   else:
      loop=0
win.close()
```



36. Variante met kolom van voorschriften grafiekbuttons_functie_param_info

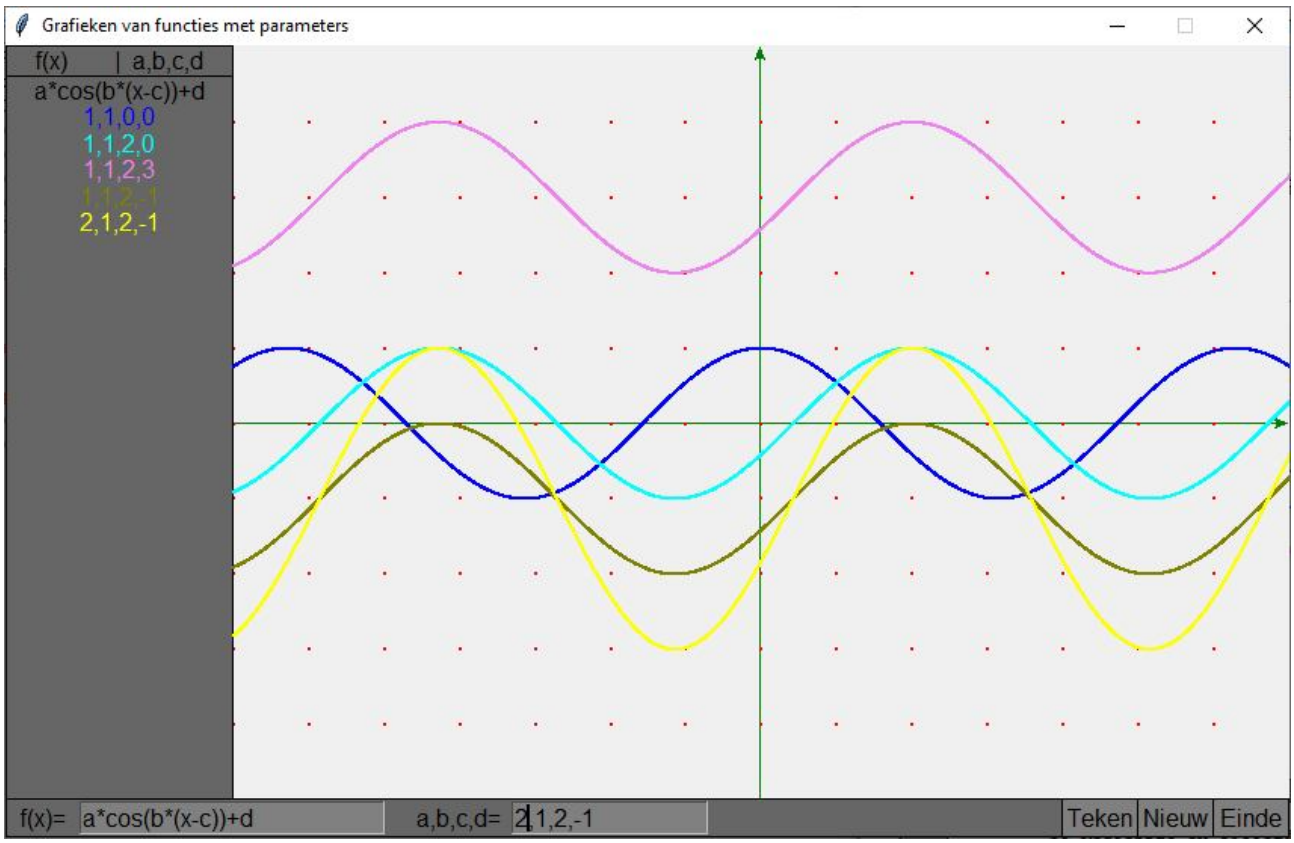
Het is beter natuurlijk als je van elke grafiek het voorschrift kunt aflezen op het scherm. In het volgende programma wordt er links een kolom voorzien, waarin je zowel het voorschrift als de waarde van de parameters kunt aflezen

Programma:

```
# grafieken van functies (met parameters) met entries en buttons
from graphics import *
from math import *
# meervoudige invoer
def mult(s):
    l=s.split(',');co=[]
    for i in l:co.append(float(i))
    return co
def f(x):
    return eval(fs)
def rechthoek(x1,y1,x2,y2):
    prlo=Point(x1,y1);prrb=Point(x2,y2);rh1=Rectangle(prlo,prrb)
    rh1.setFill('Grey40');rh1.draw(win)
    return
def tekenbutton(i,tek):
    prlo2=Point(xmax-i,ymin-0.5);prrb2=Point(xmax-i+1,ytop)
    rh2=Rectangle(prlo2,prrb2);rh2.setFill('Grey')
    rh2.draw(win);t1=Text(Point(xmax-i+0.5,ymid),tek);t1.draw(win)
    return
```

```
#hoofdprogramma
kleur=['blue','cyan','violet','olive','yellow','pink','magenta','skyblue']
breed=850;hoog=525
xmin=-7;xmax=7;ymin=-5;ymax=5;nieuw=1
while nieuw==1:
   nieuw=0
   win = GraphWin('Grafieken van functies met parameters',breed,hoog)
   win.setCoords(xmin-3,ymin-0.5,xmax,ymax)
   #afmetingen onderste rechthoek
   rhhoog=0.5;ymid=ymin-rhhoog/2;ytop=ymin
   #assen
   pl=Point(xmin,0);pr=Point(xmax,0);lix=Line(pl,pr); lix.setFill('green')
   po=Point(0,ymin);pb=Point(0,ymax);liy=Line(po,pb); liy.setFill('Green')
   lix.setArrow('last');liy.setArrow('last');lix.draw(win);liy.draw(win)
   #rooster
   for x in range(xmin,xmax):
      for y in range(ymin,ymax):
         pt=Point(x,y);pt.setFill('Red');pt.draw(win)
   #teken rechthoeken links inf, onderaan invoer
   rechthoek(xmin-3,ymin,xmin,ymax)
   rechthoek(xmin-3,ymax-0.4,xmin,ymax)
   rechthoek(xmin-3,ymin-0.5,xmax,ytop)
   #tekst invoeren
   t2=Text(Point(xmin-1.5,ymax-0.20),'f(x)       | a,b,c,d')
   t0=Text(Point(xmin-2.5,ymid),'f(x)=')
   t1=Text(Point(xmin+3,ymid),'a,b,c,d=')
   t0.draw(win);t1.draw(win);t2.draw(win)
   #teken entry blok parameters
   pblok=Entry(Point(-2,ymid),14);pblok.setText('1,0,0,0');
pblok.draw(win)
   #teken entry blok functie
   fblok=Entry(Point(-7,ymid),22);fblok.draw(win)
   #teken buttons
   tekenbutton(2,'Nieuw');tekenbutton(3,'Teken');tekenbutton(1,'Einde')
   #als er op button 'Teken' geklikt wordt, grafiek tekenen
   xklik=0;yklik=0;loop=1;kl=0;y_info=ymax-0.25;ofs="";aantal=0
   while loop==1:
      while xklik<xmax-3 or yklik>ytop:
         puntklik=win.getMouse()
         xklik=puntklik.getX();yklik=puntklik.getY()
      if xklik<xmax-2 and aantal<=20:
         fs=fblok.getText();fst=fs.replace('math.','')
         ps=pblok.getText();a,b,c,d=mult(ps)
         if fst!=ofs:
            y_info=y_info-0.35;t=Text(Point(xmin-1.5,y_info),fst)
            t.draw(win);ofs=fst;aantal=aantal+1
         y_info=y_info-0.35;t=Text(Point(xmin-1.5,y_info),ps)
         t.setFill(kleur[kl]);t.draw(win);aantal=aantal+1
         stap=0.01;x=xmin
         while x<xmax:
            x=x+stap;y=f(x);pt=Point(x,y);pt.setFill(kleur[kl])
            if y>ymin:pt.draw(win)
         xklik=0;yklik=0;kl=kl+1
         if kl==len(kleur):kl=0
      elif xklik<xmax-1 or aantal>20:
```

```
        win.close();loop=0;nieuw=1
    else:
        loop=0
win.close()
```



Regressie en correlatie

37. Lineair verband tussen puntenkoppels regressie_lineair

Als er een +/- lineair verband bestaat tussen 2 reeksen getallen (x- en y-waarden) , dan kunnen we proberen om dit te beschrijven door het functievoorschrift van een rechte, dwz y=ax+b. Wij moeten zoeken naar de beste rechte, d.i. de rechte waarvoor de som van de afwijkingen t.o.v. de gegeven punten minimaal zijn.
De afwijkingen zijn de verschillen tussen de y-waarden van de reeks getallen en de y-waarden die we vinden door het functievoorschrift y=ax+b toe te passen op de x-waarden van de reeks getallen.
Stel dat de reeks getallen zijn: (x_1,y_1), (x_2,y_2), (x_3,y_3),…… (x_n,y_n).
Met de afwijkingen bedoelen we dus: $y_1-(ax_1+b)$,$y_2-(ax_2+b)$,$y_3-(ax_3+b)$..
Vanaf nu zullen we om de berekeningen te vereenvoudigen ons beperken tot 3 koppels.
Sommige van deze afwijkingen zijn negatief, andere positief (sommige punten zullen resp. onder, boven de berekende rechte liggen).
Daarom zullen we de som van de kwadraten van de afwijkingen minimaal maken.
Dwz. we zoeken het minimum van $[y_1-(ax_1+b)]^2+[\ y_2-(ax_2+b)]^2+[y_3-(ax_3+b)]^2$ in functie van a en b.
Het minimum (of maximum) van een functie krijgen we als de afgeleide 0 wordt. Hebben we een functie van 2 veranderlijken a en b, dan moeten we de partiële afgeleiden van de functie nemen naar a en b.
We krijgen dan een stelsel van 2 vergelijkingen in a en b:
$f'(a,b)_b=0\ \ 2[y_1-(ax_1+b)].(-1)+2[\ y_2-(ax_2+b)].(-1)+2[y_3-(ax_3+b)].(-1)=0$
$f'(a,b)_a=0\ \ 2[y_1-(ax_1+b)].(-x_1)+2[\ y_2-(ax_2+b)].(-x_2)+2[y_3-(ax_3+b)].(-x_3)=0$
We delen beide vergelijkingen door 2 en herschikken:
$-y_1-y_2-y_3+3b+a(x_1+x_2+x_3)=0$

$-y_1x_1- y_2x_2- y_3x_3 +b(x_1+ x_2+ x_3)+a(x_1^2+ x_2^2+ x_3^2) = 0$
of
$3b +a(x_1+ x_2+ x_3) = y_1+ y_2+ y_3$
$b(x_1+ x_2+ x_3)+ a(x_1^2+ x_2^2+ x_3^2) = y_1x_1+ y_2x_2+ y_3x_3$

Meer algemeen als we n puntenkoppels hebben:
$nb + a(x_1+ x_2+ x_3+\ldots+x_n) = y_1+ y_2+ y_3+\ldots+ y_n$ (1)
$b(x_1+ x_2+ x_3+\ldots+ x_n)+ a(x_1^2+ x_2^2+ x_3^2+\ldots+x_n^2) = y_1x_1+ y_2x_2+\ldots+ x_n y_n$

Voor de regressierechte y=ax+b zijn a en b de oplossingen van het stelsel:

$$\begin{cases} n \cdot b + S_x \cdot a = S_y \\ S_x \cdot b + S_{x2} \cdot a = S_{xy} \end{cases} \text{ waarbij } S_x = \sum_{j=1}^{N} x_j \quad S_y = \sum_{j=1}^{N} y_j \quad S_{xy} = \sum_{j=1}^{N} x_j \cdot y_j \quad S_{x2} = \sum_{j=1}^{N} x_j^2$$

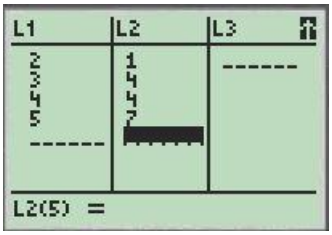
Dit is een stelsel waaruit we a en b kunnen berekenen.

Getallenvoorbeeld

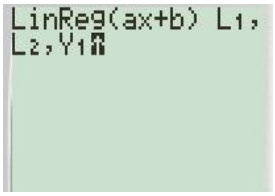
Neem 4 puntenkoppels: (2,1) (3,4) (4,4) (5,7)
$x_1^2+ x_2^2+ x_3^2 +x_4^2=4+9+16+25=54$
$x_1+ x_2+ x_3 +x_4 = 2+3+4+5=14$
$y_1x_1+ y_2x_2+ y_3x_3+ y_4x_4 = 2+12+16+35 = 65$
$y_1+ y_2+ y_3+ y_4 = 1+4+4+7 = 16$
We passen de formule (1) toe

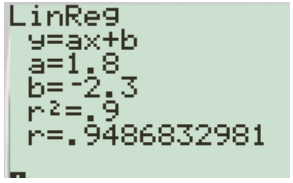
4b+ 14a=16	27	-7	108b – 98b = 432-455 =-23	b = -2.3
14b+ 54a=65	-7	2	-98a + 108a = 130-112 = 18	a = 1.8

De best passende rechte is dus y= 1.8 x-2.3
Het grafisch rekentoestel biedt de mogelijkheid om voor een reeks puntenkoppels automatisch de best passende rechte te berekenen.
Neem het getallenvoorbeeld dat we manueel berekend hebben.
Voer eerst de getallen 2,3,4,5 in lijst 1 en 1,4,4,7 in lijst 2 in. Dit gaat als volgt:
Kies **STAT EDIT** en vul de lijsten L1 en L2 met de getallen



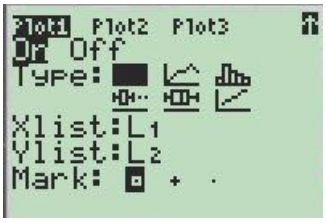
Kies dan STAT CALC en neem optie 8:
LinReg(ax+b) L1,L2,Y1

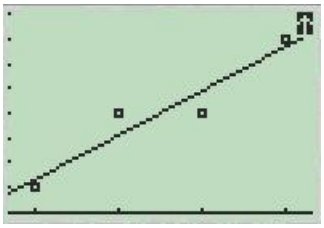




Om de punten te plotten alsook de best passende rechte kiezen we
2ND STAT PLOT
We selecteren de eerste functie en kiezen het eerste type,

XLIST=L1, YLIST=L2
Kies dan ten slotte **ZOOM ZoomStat**





Correlatie

Hoe kleiner de afwijkingen tussen de regressielijn en de verschillende punten in de puntenwolk, hoe groter het verband is tussen de x- en de y-waarden. De correlatie is een getal dat aangeeft in welke mate dit verband aanwezig is.
r=1 : volkomen positieve correlatie: de punten liggen precies op een stijgende rechte
r=-1 : volkomen negatieve correlatie: de punten liggen precies op een dalende rechte
r ~ 0: geen correlatie (puntenwolk)
Deze correlatie wordt bij de GRM enkel getoond indien **DiagnosticOn** geselecteerd is.
(dit vind je terug in de **CATALOG**)

$\bar{x} = \frac{2+3+4+5}{4} = 3.5$ en $\bar{y} = \frac{1+4+4+7}{4} = 4$

In het vorige voorbeeld is de berekening ook manueel uit te voeren:

$$\sum (x_i - \bar{x})(y_i - \bar{y}) = (2-3.5)(1-4)+(3-3.5)(4-4)+(4-3.5)(4-4)+(5-3.5)(7-4)=9$$

$$\sum \left(x_i - \bar{x}\right)^2 = (2-3.5)^2+(3-3.5)^2+(4-3.5)^2+(5-3.5)^2=5$$

$$\sum \left(y_i - \bar{y}\right)^2 = (1-4)^2+(4-4)^2+(4-4)^2+(7-4)^2=18$$

Invullen in de formule van de correlatie geeft ons:

$$r = \frac{\sum \left(x_i - \bar{x}\right)\cdot\left(y_i - \bar{y}\right)}{\sqrt{\sum (x_i - \bar{x})^2 \cdot \sum (y_i - \bar{y})^2}} = \frac{9}{\sqrt{5\cdot 18}} = 0.9487.. = 94.87\ \%$$

Conclusie: r ligt dicht bij 1, de puntenkoppels liggen dus vrij dicht bij de regressierechte.

38. Berekening van regressie en correlatie met python

Programma:

```
# lineaire regressie
import math
from graphics import *
def oploss(u1,u2,u3,v1,v2,v3):
    d=u1*v2-u2*v1;da=u3*v2-u2*v3;db=u1*v3-u3*v1
    return [da/d,db/d]
def correlatie(xco,yco,xsom,ysom):
    teller=0;x2noemer=0;y2noemer=0
    for j in range(0,n):
        teller=teller+(xco[j]-xsom/n)*(yco[j]-ysom/n)
        x2noemer=x2noemer+(xco[j]-xsom/n)*(xco[j]-xsom/n)
        y2noemer=y2noemer+(yco[j]-ysom/n)*(yco[j]-ysom/n)
```

```
    noemer=math.sqrt(x2noemer*y2noemer)
    correl=teller/noemer
    print("Correlatie=",round(correl,4))
#hoofdprogramma
xt=[];yt=[]
n=int(input("Geef het aantal koppels coördinaten:"))
for j in range(0,n):
    print("Punt",j+1, end=" :")
    coo=input("x,y=").split(',')
    x=float(coo[0]);y=float(coo[1])
    xt.append(x);yt.append(y)
# de noodzakelijke sommaties berekenen
sx=0;sy=0;sxy=0;sx2=0
for j in range(0,n):
    sx=sx+xt[j];sy=sy+yt[j];sxy=sxy+xt[j]*yt[j];sx2=sx2+xt[j]*xt[j]
a,b=oploss(sx,n,sy,sx2,sx,sxy)
# lineaire regressie
if b>=0:
    sg="+"
else:
    sg="-"
print("Regressierechte y=",round(a,4),"x"+sg,round(abs(b),4))
# lineaire correlatie
correlatie(xt,yt,sx,sy)

>>>
Geef het aantal koppels coördinaten:4
Punt 1 :x,y=2,3
Punt 2 :x,y=4,7
Punt 3 :x,y=5,9
Punt 4 :x,y=7,11
Regressierechte y= 1.6154 x+ 0.2308
Correlatie= 0.9845
>>>
```

39. Corona (exponentiële regressie)

Kunnen we met deze kennis iets berekenen over het aantal besmettingen met de Corona crisis? Zoals vele processen in de natuur, stijgt het aantal besmettingen niet lineair maar exponentieel. We zoeken m.a.w. een exponentiële functie die deze exponentiële stijging het beste benadert. Hierna volgt een tabelletje van het aantal besmettingen vanaf 12 maart 2020 = dag 1 (België)

12/03/20	1	399
13/03/20	2	559
14/03/20	3	724
15/03/20	4	886
16/03/20	5	1058
17/03/20	6	1243
18/03/20	7	1486
19/03/20	8	1795
20/03/20	9	2257

We zoeken nu een a en b zodat de exponentiële functie $y=b.a^x$
de gegeven getallen zo goed mogelijk benadert.
Eerst «lineariseren» we de vergelijking:
LOG(y)=LOG(b)+x.LOG(a), stel nu LOG(y)=Y , LOG(b) =B , x=X , LOG(a)=A ,

dan hebben we een lineaire vergelijking Y=B+A.X
Omdat X=x mogen we de x-gegevens laten: 1....9
Omdat Y=LOG(y) moeten we van de y-gegevens de LOG nemen:
LOG(399)=2.601 LOG(559)=2.7474 ....

We berekenen met Excel de noodzakelijke sommen om regressierechte en correlatie te berekenen:

	x=X	y	Y	X²	XY	X-μ X	Y-μY	(X-μ X)(Y-μY)	(X-μ X)²	(Y-μY)²
12/03/20	1	399	2,6010	1	2,6010	-4	-0,4050	1,6202	16	0,1641
13/03/20	2	559	2,7474	4	5,4948	-3	-0,2586	0,7758	9	0,0669
14/03/20	3	724	2,8597	9	8,5792	-2	-0,1463	0,2926	4	0,0214
15/03/20	4	886	2,9474	16	11,7897	-1	-0,0586	0,0586	1	0,0034
16/03/20	5	1058	3,0245	25	15,1224	0	0,0185	0,0000	0	0,0003
17/03/20	6	1243	3,0945	36	18,5668	1	0,0885	0,0885	1	0,0078
18/03/20	7	1486	3,1720	49	22,2041	2	0,1660	0,3320	4	0,0276
19/03/20	8	1795	3,2541	64	26,0325	3	0,2481	0,7442	9	0,0615
20/03/20	9	2257	3,3535	81	30,1818	4	0,3475	1,3901	16	0,1208
21/03/20										
Σ	45		27,0541	285	140,5724			5,3018	60	0,4738
μ	5		3,0060							

$$\begin{cases}\sum X_i \cdot A + n \cdot B = \sum Y_i \\ \sum X_i^2 \cdot A + \sum X_i \cdot B = \sum X_i . Y_i\end{cases} \qquad \begin{cases}45A + 9B = 27.0541 \\ 285A + 45B = 140.5724\end{cases} \qquad \begin{cases}5A + B = 3.006 \\ 19A + 3B = 9.3715\end{cases}$$

Hieruit berekenen we A=0.088375 en B=2,564125
De exponentiële functie wordt :

$y=366.54*1.2257^{x}$

$$r = \frac{\sum \left(X_i - \overline{X}\right) \cdot \left(Y_i - \overline{Y}\right)}{\sqrt{\sum (X_i - \overline{X})^2 \cdot \sum (Y_i - \overline{Y})^2}} = \frac{5.3018}{\sqrt{60 \cdot 04738}} = 0.9944.. = 99.44$$

De correlatie= 99,44%. Dit is zeer dicht bij 1: dat betekent dat het aantal besmettingen in de periode van 12 tot 20 maart bijna zuiver exponentieel stijgt.
Stel dat we geen quarantainemaatregelen zouden toepassen, hoeveel besmettingen zijn er dan ongeveer op 31 Maart? f(20)=21452

En nu een beetje sterrenkunde.

40. Derde wet van Kepler (machtsregressie)

We nemen de gemiddelde afstand en omlooptijd van enkele planeten in ons zonnestelsel. We zoeken dan de best passende rechte of kromme bij deze koppels.

planeet	**Afstand tot de zon**	**omlooptijd**
Mercurius	58	0.241
Saturnus	1427	29.46
Uranus	2870	84
Mars	228	1.881

Aarde	149.6	1

De afstand is uitgedrukt in miljoenen km, de omlooptijd in jaren.
We zullen nu proberen om een machtsfunctie $y=b.x^a$ te fitten met deze punten.
x= afstand tot de zon, y=omlooptijd in jaren

Om deze berekeningen manueel uit te voeren, moeten we de machtsfunctie «lineariseren».
Van $y=b.x^a$ nemen we de LOGaritme: LOG y =LOG b +a.LOG x
Stel Y=LOG y , A=a , B=LOG b, X=LOG x, dan krijgen we Y=B+AX
Dit is een lineaire vergelijking zoals in voorbeeld 1.
Omdat y wordt vervangen door LOG y en x wordt vervangen door LOG x, moeten we ook de gegevens lineariseren.
Met Excel kunnen we terug alle noodzakelijke sommen berekenen:

PLANEET	afs x	olt y	X	Y	XY	X²	Y²	X-μ X	Y-μY	(X-μ X)(Y-μY)	(X-μ X)²	(Y-μY)²
Mercurius	58	0,241	1,7634	-0,6180	-1,0898	3,1097	0,3819	-0,8183	-1,2280	1,0048	0,6696	1,5079
Saturnus	1427	29,46	3,1544	1,4692	4,6346	9,9504	2,1586	0,5727	0,8592	0,4921	0,3280	0,7383
Uranus	2870	84	3,4579	1,9243	6,6539	11,9569	3,7029	0,8762	1,3143	1,1515	0,7677	1,7274
Mars	228	1,881	2,3579	0,2744	0,6470	5,5599	0,0753	-0,2238	-0,3356	0,0751	0,0501	0,1126
Aarde	149,6	1	2,1749	0,0000	0,0000	4,7303	0,0000	-0,4068	-0,6100	0,2481	0,1655	0,3721
Σ			12,909	3,0499	10,8457	35,3072	6,3187			2,9717	1,9808	4,4583
μ			2,5817	0,6100								

$$\begin{cases} \sum X_i \cdot A + n \cdot B = \sum Y_i \\ \sum X_i^2 \cdot A + \sum X_i \cdot B = \sum X_i . Y_i \end{cases} \qquad \begin{cases} 12.909A + 5B = 3.0499 \\ 35.3072A + 12.909B = 10.8457 \end{cases}$$

Oplossing: A = 1.50169 B = -3.2671 waaruit volgt: a=1.50169

$b = 10^{B} = 10^{-3.2671} = 5.4064*10^{-4}$

$y = b.x^a = 5.4064*10^{-4} * x^{1.50169}$

De correlatie =bijna 1

$$r = \frac{\sum (X_i - \overline{X}) \cdot (Y_i - \overline{Y})}{\sqrt{\sum (X_i - \overline{X})^2 \cdot \sum (Y_i - \overline{Y})^2}} = \frac{2.9717}{\sqrt{1.9808 * 4.4583}} \cong 1$$

Als we beide leden van de machtsfunctie kwadrateren:

$y^2 = 29.23*10^{-8} * x^{3.003}$

Hieruit kunnen we een fysische wet afleiden: het kwadraat van de omloopstijd is recht evenredig met de derde macht van de gemiddelde afstand

41. Curvefitting met lineaire, exponentiële en machtsregressie

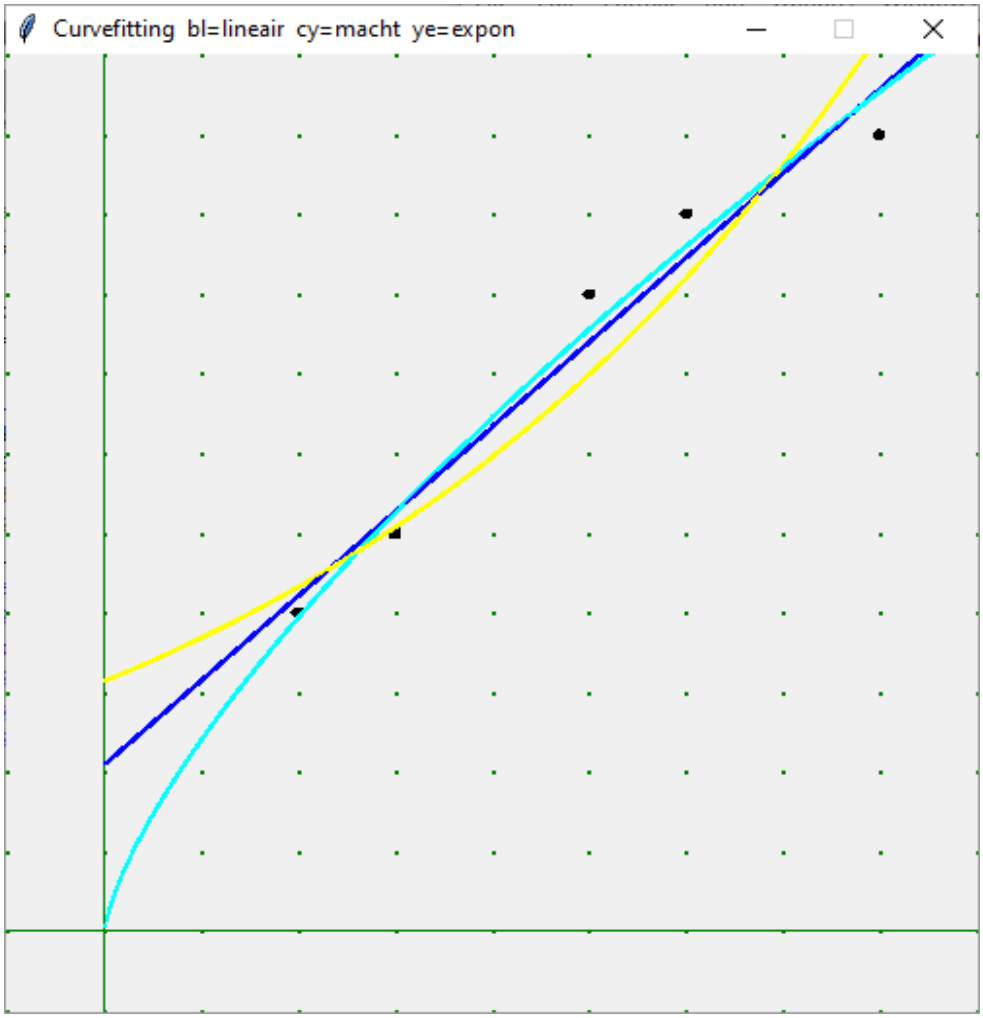
regressie_lin_expon_macht_grafie

Het volgende programma berekent voor een aantal puntenkoppels, lineaire, exponentiële en machtsregressie. Er wordt ook een grafiek getekend. Dit programma neemt als basis het vorige regressieprogramma, maar nu worden van de ingevoerde koppels de logaritmen opgeslagen in lijsten lxt en lyt.
Een meerwaarde is een grafiek van de 3 functies en een plot van alle ingegeven puntenkoppels.
(Enkel interessant als de x- en y-waarden van dezelfde orde grootte zijn)

Programma:

```
# lineaire, exponentiële en machtsregressie
import math
from graphics import *
def oploss(u1,u2,u3,v1,v2,v3):
    d=u1*v2-u2*v1;da=u3*v2-u2*v3;db=u1*v3-u3*v1
    return [da/d,db/d]
def correlatie(xco,yco,xsom,ysom):
    teller=0;x2noemer=0;y2noemer=0
    for j in range(0,n):
        teller=teller+(xco[j]-xsom/n)*(yco[j]-ysom/n)
        x2noemer=x2noemer+(xco[j]-xsom/n)*(xco[j]-xsom/n)
        y2noemer=y2noemer+(yco[j]-ysom/n)*(yco[j]-ysom/n)
    noemer=math.sqrt(x2noemer*y2noemer)
    correl=teller/noemer
    print("Correlatie=",round(correl,4))
#hoofdprogramma
xt=[];yt=[];lxt=[];lyt=[]
n=int(input("Geef het aantal koppels coördinaten:"))
for j in range(0,n):
    print("Punt",j+1, end=" :")
    coo=input("x,y=").split(',')
    x=float(coo[0]);y=float(coo[1])
    xt.append(x);yt.append(y)
    lxt.append(math.log(x));lyt.append(math.log(y))
# de noodzakelijke sommaties berekenen
sx=0;sy=0;sxy=0;sx2=0;lsx=0;lsy=0;lsxy=0;lsx2=0;esxy=0;esx2=0
for j in range(0,n):
     sx=sx+xt[j];sy=sy+yt[j];sxy=sxy+xt[j]*yt[j]
     sx2=sx2+xt[j]*xt[j];esxy=esxy+xt[j]*lyt[j]
     lsx=lsx+lxt[j];lsy=lsy+lyt[j];lsxy=lsxy+lxt[j]*lyt[j]
     lsx2=lsx2+lxt[j]*lxt[j]
a,b=oploss(sx,n,sy,sx2,sx,sxy);ao,bo=a,b
ea,eb=oploss(sx,n,lsy,sx2,sx,esxy)
ma,mb=oploss(lsx,n,lsy,lsx2,lsx,lsxy)
# lineaire regressie
if b>=0:sg="+"
else:sg="-"
print("Regressierechte y=",round(a,4),"x"+sg,round(abs(b),4))
# lineaire correlatie
correlatie(xt,yt,sx,sy)
# exponentiële regressie
a=math.e**ea;b=math.e**eb
print("Exponentiele functie y=",round(b,4),"*(",round(a,4),")**x")
# exponentiële correlatie
correlatie(xt,lyt,sx,lsy)
# machtsfunctie regressie
a=ma;b=math.e**mb
print("Machtsfunctie y=",round(b,4),"*x**(",round(a,4),')')
# machtsfunctie correlatie
```

```
correlatie(lxt,lyt,lsx,lsy)
# grafiek
xmin=min(xt);xmax=int(max(xt)+1);ymin=min(yt);ymax=int(max(yt)+1)
if xmin>=0:xmin=-1
if ymin>=0:ymin=-1
grafbr=xmax-xmin;grafho=ymax-ymin;ratio=grafho/grafbr;afmet=500
win=GraphWin('Curvefitting bl=lineair cy=macht ye=expon',afmet,afmet)
win.setCoords(xmin,ymin,xmax,ymax)
#assen
pl=Point(xmin,0);pr=Point(xmax,0);lix=Line(pl,pr);lix.setFill('Green')
po=Point(0,ymin);pb=Point(0,ymax);liy=Line(po,pb);liy.setFill('Green')
lix.draw(win);liy.draw(win)
#rooster
for x in range(int(xmin),int(xmax+1)):
   for y in range(int(ymin),int(ymax+1)):
      pt=Point(x,y);pt.setFill('Green');pt.draw(win)
for j in range(0,n):
   ci=Circle(Point(xt[j],yt[j]),0.025);ci.setFill('Red')
   ci.setWidth(5);ci.draw(win)
stap=0.01;x=0
while x<xmax:
   x=x+stap;y=ao*x+bo;pt=Point(x,y);pt.setFill('Blue');pt.draw(win)
   y=(math.e**mb)*x**a;pt=Point(x,y);pt.setFill('Cyan');pt.draw(win)
   y=(math.e**eb)*(math.e**ea)**x;pt=Point(x,y)
   pt.setFill('Yellow');pt.draw(win)
>>>
Geef het aantal koppels coördinaten:5
Punt 1 :x,y=2,4
Punt 2 :x,y=3,5
Punt 3 :x,y=5,8
Punt 4 :x,y=6,9
Punt 5 :x,y=8,10
Regressierechte y= 1.0614 x+ 2.1053
Correlatie= 0.979
Exponentiele functie y= 3.1526 *( 1.1732 )**x
Correlatie= 0.9612
Machtsfunctie y= 2.4234 *x**( 0.708 )
Correlatie= 0.9913
>>>
```



Er zijn ook nog andere functies die op één of andere manier kunnen 'gelineariseerd' worden.

42. De saturatie-groei functie regressie_saturatiegroei

Het gaat hier om de functie: $y = a.\frac{x}{b+x}$

Lineariseren kan hier door van beide leden het omgekeerde te nemen:

$$\frac{1}{y} = \frac{1}{a}.\frac{b+x}{x} = \frac{b}{a}.\frac{1}{x} + \frac{1}{a}$$

We stellen nu $\frac{1}{y} = Y \quad \frac{b}{a} = B \quad \frac{1}{x} = X \quad \frac{1}{a} = A$,

dan hebben we de lineaire vergelijking Y=B.X+A

We kunnen bijna volledig de listing van 'lineaire regressie' nemen en slechts een paar elementen wijzigen: we vervangen de ingevoerde getallen door de omgekeerden

x=float(coo[0]); y=float(coo[1]); X=1/x; Y=1/y

We bepalen dan A en B, tot slot is $a = \frac{1}{A}$ en $b = B.a$

Programma:

```
#  regressie saturatie groei
import math
def oploss(u1,u2,u3,v1,v2,v3):
```

```
    d=u1*v2-u2*v1;da=u3*v2-u2*v3;db=u1*v3-u3*v1
    return [da/d,db/d]
def correlatie(xco,yco,xsom,ysom):
    teller=0;x2noemer=0;y2noemer=0
    for j in range(0,n):
        teller=teller+(xco[j]-xsom/n)*(yco[j]-ysom/n)
        x2noemer=x2noemer+(xco[j]-xsom/n)*(xco[j]-xsom/n)
        y2noemer=y2noemer+(yco[j]-ysom/n)*(yco[j]-ysom/n)
    noemer=math.sqrt(x2noemer*y2noemer)
    correl=teller/noemer
    print("Correlatie=",round(correl,4))
#hoofdprogramma
xt=[];yt=[]
n=int(input("Geef het aantal koppels getallen:"))
for j in range(0,n):
    print("Punt",j+1, end=" :")
    coo=input("x,y=").split(',')
    x=float(coo[0]);y=float(coo[1]);X=1/x;Y=1/y
    xt.append(X);yt.append(Y)
sx=0;sy=0;sxy=0;sx2=0
for j in range(0,n):
    sx=sx+xt[j];sy=sy+yt[j];sxy=sxy+xt[j]*yt[j];sx2=sx2+xt[j]*xt[j]
B,A=oploss(sx,n,sy,sx2,sx,sxy);a=1/A;b=a*B
# regressie
print('Saturatie-groeikromme y=',round(a,4),'x/(',round(abs(b),4),'+x)')
# correlatie
correlatie(xt,yt,sx,sy)
>>>
Geef het aantal getallen:5
Punt 1 :x,y=1,0.5
Punt 2 :x,y=2,0.75
Punt 3 :x,y=3,0.9
Punt 4 :x,y=4,1
Punt 5 :x,y=6,1.1
Saturatie-groeikromme y= 1.478 x/( 1.9514 +x)
Correlatie= 0.9998
>>>
```

43. Curvefitting met y= b.f(x)+a regressie_a+bfx

Wat natuurlijk ook kan, is dat we in de lineaire functie x vervangen door een functie van x,bvb. sin(x):
De functie is dan van de vorm y= b.sin(x)+a:
het enige dat er dan moet gebeuren is, na invoer, de x-waarden vervangen door f(x)-waarden.
Neem de listing van lineaire regressie en vul aan of vervang met:

```
#  regressie y= a+b.f(x)
from math import *
def f(x):return(eval(fs))
.........
x=float(coo[0]);y=float(coo[1]);X=f(x);Y=y
.........
print('Functie y=',round(A,4),'+',round(abs(B),4),fs)

>>>
Geef een functie:sin(x)
Geef het aantal getallen:4
```

```
Punt 1 :x,y=1,4.7
Punt 2 :x,y=2,4.8
Punt 3 :x,y=3,3.3
Punt 4 :x,y=4,1.5
Functie y= 3.0104 + 1.9896 sin(x)
Correlatie= 0.9999
>>>
```

44. Ontbinden van een veelterm van de 3de graad ontbinden_3degraadsfunctie

Het volgende programma neemt een paar van de vorige programma's (welke ?) samen om een 3de graadsfunctie ax^3+bx^2+cx+d te ontbinden in factoren.
Als je denkt aan de grafiek van een derdegraadsfunctie, dan weet je dat er steeds minstens één reëel nulpunt is.
Dit punt x_3 benaderen we met Newton.
(Dit zal enkel mislukken als het buigpunt op de x-as ligt, bij een volkomen derde macht)
We delen door $x-x_3$ met de regel van Horner. Het quotiënt van deze opgaande deling is een kwadratische functie ax^2+bx+c. Merk op dat hiervan b=(oude) b+a*x_3 en c=(oude) c+(nieuwe) b*x_3.
Van deze vierkantswortels berekenen we de (eventueel complexe) wortels x_1 en x_2.
Tot slot krijgen we steeds een ontbinding a(x-x1)(x-x2)(x-x3)

Programma:

```
# ontbinden van een veelterm van de 3de graad
import math
# meervoudige invoer
def mult(s):
    l=s.split(',')
    co=[]
    for i in l:co.append(float(i))
    return co
def f(x):
    return a*x**3+b*x*x+c*x+d
def dif(x):
    eps=0.000001
    return (f(x+eps)-f(x))/eps
def newton():
# eerst zoeken naar geheel getal dicht bij nulpunt, dan Newton
    x=1
    while f(x)*f(-x)>0:x=x+1
    if abs(f(x))>abs(f(-x)):x=-x
    ox=x+1
    while math.fabs(x-ox)>0.00001:
        ox=x;x=x-f(x)/dif(x)
    return x
def v(x):
    if x>0:
        sgn='+'
    else:
        sgn=''
    return sgn+format(x,'.4f')
```

```
def vc(x):
    return "-("+format(x,'.4f')+")"
# hoofdprogramma
print('Ontbinden van een derdegraadsveelterm ax³+bx²+cx+d')
a,b,c,d=mult(input('a,b,c,d='))
x3=newton()
b=b+a*x3;c=c+b*x3 # horner
d=b*b-4*a*c
if d>0:
    root=math.sqrt(d)
    x1,x2=(-b-root)/2/a,(-b+root)/2/a
elif d==0:
    x1=x2=-b/2/a
else:
    root=math.sqrt(-d)
    x1,x2=(-b-root*(1j))/2/a,(-b+root*(1j))/2/a
print('Ontbinding =',format(a)+'(x'+v(-x3)+')',end='')
if d>0:
    print('(x'+v(-x2)+')(x'+v(-x1)+')')
else:
    print('[x'+vc(x2)+'][x'+vc(x1)+']')

>>>
Ontbinden van een derdegraadsveelterm ax³+bx²+cx+d
a,b,c,d=7,14,-45,11
Ontbinding = 7.0(x-1.5301)(x-0.2702)(x+3.8004)
>>>
```

Matrices en stelsels

45. Methode van Gauss stelsels_oplossen

In het eerste programma wordt een matrix gediagonaliseerd, dwz. met elementaire rijoperaties (verwisselen van 2 rijen, een rij vervangen door een lineaire combinatie van de rij met een andere rij, delen van een rij door de leider) een matrix bekomen waarbij de leiders trapsgewijs van linksboven naar rechtsonder voorkomen, elke leider =1 en onder en boven elke leider enkel 0.
In dit programma is de matrix de verhoogde matrix van een stelsel, dwz. inclusief het rechterlid, dan kunnen we uit de gediagonaliseerde matrix de oplossing van het stelsel aflezen.

Programma:

```
# Oplossen van stelsels met Gauss
from tabulate import tabulate
# matrix invoeren
def mult(s):
    l=s.split(',')
    co=[]
    for i in l:co.append(float(i))
    return co
def druk(a): print(tabulate(a))
a=[];i=0;inp=' '
print('Oplossen van een stelsel van n vergelijkingen')
print('Voor de matrix geldt:#kolommen m = #rijen n+1')
```

```
print('Geef de coëfficiënten van de matrix gescheiden door komma\'s.')
print('Nadat je de laatste rij hebt ingevoerd, tik <Enter>')
while inp!='':
    print('rij',i+1,end=' ')
    inp=input(':')
    if inp!='':
        rijcoef=mult(inp);a.append(rijcoef)
    m=len(rijcoef);n=len(a);i=i+1
druk(a)
if m!=n+1:
    print('Geen stelsel')
else:
    print('Na pivotage:')
    for i in range(0,n):
        for k in range(i+1,n):
            if abs(a[i][i])<abs(a[k][i]):
                for j in range(0,n+1):
                    wissel=a[i][j];a[i][j]=a[k][j];a[k][j]=wissel
    druk(a)
    print('Na Gauss-eliminatie:')
    for i in range(0,n):
        for k in range(0,n):
            if k!=i and a[i][i]!=0:
                t=a[k][i]/a[i][i]
                for j in range(0,n+1):
                    a[k][j]=a[k][j]-t*a[i][j]
    druk(a)
    print('Na deling door leiders:')
    for i in range(0,n):
        t=a[i][i]
        for j in range(0,n+1):
            a[i][j]=round(a[i][j]/t,6)
    druk(a)
>>>
Oplossen van een stelsel van n vergelijkingen
Voor de matrix geldt:#kolommen m = #rijen n+1
Geef de coëfficiënten van de matrix gescheiden door komma's.
Nadat je de laatste rij hebt ingevoerd, tik <Enter>
rij 1 :2,1,4,53
rij 2 :1,-2,1,4
rij 3 :3,-4,-2,1
rij 4 :
-  --  --  --
2   1   4  53
1  -2   1   4
3  -4  -2   1
-  --  --  --
Na pivotage:
-  --  --  --
3  -4  -2   1
1  -2   1   4
2   1   4  53
-  --  --  --
Na Gauss-eliminatie:
-  ---------  ----  --------
```

```
3   0         0    39
0  -0.666667   0    -4.66667
0   0        14.5  72.5
-  ---------  ----  --------
Na deling door leiders:
--  -  --  --
 1  0   0  13
-0  1  -0   7
 0  0   1   5
--  -  --  --
>>>
```

46. Inverse matrix inverse_matrix

Je kan een gelijkaardig programma maken om de inverse van een gegeven matrix te bepalen: als je tegelijkertijd op de overeenkomstige eenheidsmatrix dezelfde elementaire rijoperaties uitvoert, zal de oorspronkelijke matrix omgezet worden naar de eenheidsmatrix terwijl de eenheidsmatrix wordt omgezet naar de inverse matrix. Dit lukt enkel als we een reguliere matrix hebben.

Programma:

```
# bepalen van de inverse matrix
from tabulate import tabulate
# matrix invoeren
def mult(s):
    l=s.split(',')
    co=[]
    for i in l:co.append(float(i))
    return co
def druk(a,im):
    aim=[]
    for i in range(0,n):
        r=[]
        for j in range(0,n):r.append(a[i][j])
        for j in range(0,n):r.append(im[i][j])
        aim.append(r)
    print(tabulate(aim))
a=[];i=0;inp=' '
print('Bepalen van de inverse matrix')
print('Voor de matrix geldt:#kolommen = #rijen ')
print('Geef de coëfficiënten van de matrix gescheiden door komma\'s.')
print('Nadat je de laatste rij hebt ingevoerd, tik <Enter>')
while inp!='':
    print('rij',i+1,end=' ')
    inp=input(':')
    if inp!='':
        rijcoef=mult(inp);a.append(rijcoef)
    m=len(rijcoef);n=len(a);i=i+1
#initialiseren inverse matrix (n x n-eenheidsmatrix)
im=[]
for i in range(0,n):
    ir=[]
    for j in range(0,n):
        if i==j:
            ir.append(1)
```

```
        else:
            ir.append(0)
    im.append(ir)
druk(a,im)
if m!=n:
    print('Geen vierkante matrix')
else:
    print('Na pivotage:')
    for i in range(0,n):
        for k in range(i+1,n):
            if abs(a[i][i])<abs(a[k][i]):
                for j in range(0,n):
                    wissel=a[i][j];a[i][j]=a[k][j];a[k][j]=wissel
                    wissel=im[i][j];im[i][j]=im[k][j];im[k][j]=wissel
    druk(a,im)
    print('Na Gauss-eliminatie:')
    for i in range(0,n):
        for k in range(0,n):
            if k!=i and a[i][i]!=0:
                t=a[k][i]/a[i][i]
                for j in range(0,n):
                    a[k][j]=a[k][j]-t*a[i][j]
                    im[k][j]=im[k][j]-t*im[i][j]
    druk(a,im)
    print('Na deling door leiders:')
    for i in range(0,n):
        t=a[i][i]
        for j in range(0,n):
            a[i][j]=round(a[i][j]/t,6)
            im[i][j]=round(im[i][j]/t,6)
    druk(a,im)

>>>
Bepalen van de inverse matrix
Voor de matrix geldt:#kolommen = #rijen
Geef de coëfficiënten van de matrix gescheiden door komma's.
Nadat je de laatste rij hebt ingevoerd, tik <Enter>
rij 1 :1,2,4
rij 2 :3,5,7
rij 3 :2,3,2
rij 4 :
-  -  -  -  -  -
1  2  4  1  0  0
3  5  7  0  1  0
2  3  2  0  0  1
-  -  -  -  -  -
Na pivotage:
-  -  -  -  -  -
3  5  7  0  1  0
2  3  2  0  0  1
1  2  4  1  0  0
-  -  -  -  -  -

Na Gauss-eliminatie:
-  ---------  --  ---------  --  ---------
```

```
3   0         0  -33       24  -18
0  -0.333333   0   -2.66667   2   -1.66667
0   0        -1    1       -1    1
-  ---------  --  ---------  --  ---------
Na deling door leiders:
--  --  --  ---  --  --
 1   0   0  -11   8  -6
-0   1  -0    8  -6   5
-0  -0   1   -1   1  -1
--  --  --  ---  --  --
>>>
```

47. Berekenen van de determinant van een nxn-matrix determinanten

Een determinant wordt berekend als de som van de elementen van een rij of kolom, elk vermenigvuldigd met de minor van dat element. De minor M_{ij} van een element m_{ij} is het product van $(-1)^{i+j}$ en de determinant die we bekomen door de i-de rij en de j-de kolom te schrappen.
In de functie **det** van het volgende programma wordt er steeds ontwikkeld volgens de eerste kolom. De variabele mat van deze functie moet een vierkante matrix zijn. Als de lengte 2 is, hebben we een 2x2-matrix waarvan de determinant $= mat_{00} * mat_{11} - mat_{01} * mat_{10}$
Is de lengte le>2, dan wordt voor elk element van de eerste kolom, de minor berekend:
We construeren voor het i-de element een vierkante matrix s:

```
s=[]
for k in range(0,le):
if i!=k:s.append(mat[k][1:])
```

Dit is de matrix die overblijft als we in de volledige matrix de 0-de kolom en de i—de rij schrappen.
Daarna wordt de determinant berekend, door het recursief berekenen van det(s), en deze dan vermenigvuldigen met $(-1)^{i+j} = (-1)^{i}$ en het element $mat_{i0} = mat[i][0]$
Het resultaat van het product van deze 3 factoren wordt opgeteld bij de te berekenen determinant

Programma:

```
# berekenen van de determinant van een nxn-matrix
def det(mat):
    le=len(mat)
    if le==2:dt=mat[0][0]*mat[1][1]-mat[0][1]*mat[1][0]
    else:
        dt=0
        for i in range(0,le):
            s=[]
            for k in range(0,le):
                if i!=k:s.append(mat[k][1:])
            dt=dt+mat[i][0]*(-1)**i*det(s)
    return dt
def mult(s):
    l=s.split(',')
    co=[]
    for i in l:co.append(float(i))
    return co
a=[];i=0;inp=' '
print('Geef de coëfficiënten van de matrix gescheiden door kommas.')
print('Nadat je de laatste rij hebt ingevoerd, tik <Enter>')
```

```
while inp!='':
   print(i+1,'ste rij co',end=' ')
   inp=input(':')
   if inp!='':
      rijcoef=mult(inp);a.append(rijcoef)
   m=len(rijcoef);n=len(a);i=i+1
if m!=n:
   print('Niet vierkant')
else:
   print('Determinant=',det(a))
>>>
```

Geef de coëfficiënten van de matrix gescheiden door kommas.
Nadat je de laatste rij hebt ingevoerd, tik <Enter>
1 ste rij co :2,1,4
2 ste rij co :3,4,5
3 ste rij co :7,4,11
4 ste rij co :
Determinant= -14.0

48. Oplossen van stelsels van n vergelijkingen met n onbekenden (Cramer)

stelsels_determinant

We kunnen de functie **det** gebruiken om stelsels op te lossen.

$a_{00}x_0+a_{01}x_1+a_{02}x_2+...+ a_{0n}x_n=b_0$
$a_{10}x_0+a_{11}x_1+a_{12}x_2+...+ a_{1n}x_n=b_1$
...
$a_{n0}x_0+a_{n1}x_1+a_{n2}x_2+...+ a_{nn}x_n=b_n$

De coëfficiënten (inclusief het rechterlid) van het stelsel worden ingevoerd in matrix a. Nadien wordt acop = deepcopy(a). Deze functie deepcopy is noodzakelijk als je wenst dat de wijzigingen van acop niet toegepast worden op a.
We willen bij acop de laatste kolom verwijderen zodat acop enkel de coëfficiënten van het linkerlid bevat.
Dit kan je bijvoorbeeld doen met: for i in range(0,n):acop[i].pop()
We vullen de kolommatrix b op met de coëfficiënten van het rechterlid,
dit is de laatste kolom van a: for i in range(0,n):b.append(a[i][n])
Om $x_{0....}x_{k....}x_n$ te berekenen hebben we nog alle vierkante matrices
$m_{0...}m_{k...}m_n$ nodig die we bekomen door in acop de k-de kolom te vervangen door het rechterlid b. Dit bekomen we door elk van de m_k eerst gelijk te stellen aan acop (met deepcopy), het k-de element van elke i-rij weg te nemen en te vervangen door het rechterlid b_i van die rij.

```
for k in range(0,n):
    m[k]=deepcopy(acop)
    for i in range(0,n):
        m[k][i].pop(k)
        m[k][i].insert(k,b[i])
```

Elke onbekende x_k is dan gelijk aan det(m_k)/det(acop)

Programma:

```
# oplossen van stelsels van n vergelijkingen met n onbekenden
from copy import deepcopy
def det(mat):
   le=len(mat)
   if le==2:d=mat[0][0]*mat[1][1]-mat[0][1]*mat[1][0]
   else:
      d=0
      for i in range(0,le):
         s=[]
         for k in range(0,le):
            if i!=k:s.append(mat[k][1:])
         d=d+mat[i][0]*(-1)**i*det(s)
   return d
def mult(s):
   l=s.split(',')
   co=[]
   for i in l:co.append(float(i))
   return co
a=[];i=0;inp=' '
print('Geef de coëfficiënten van de matrix gescheiden door kommas.')
print('Nadat je de laatste rij hebt ingevoerd, tik <Enter>')
while inp!='':
   print(i+1,'ste rij co',end=' ')
   inp=input(':')
   if inp!='':
      rijcoef=mult(inp);a.append(rijcoef)
   n=len(a);i=i+1
if len(rijcoef)!=n+1:
   print('Geen stelsel van n vergelijkingen met n onbekenden')
else:
   acop=deepcopy(a);m=[];b=[]
   for i in range(0,n):b.append(a[i][n])
   for i in range(0,n):acop[i].pop()
   for k in range(0,n):m.append([])
   for k in range(0,n):
      m[k]=deepcopy(acop)
      for i in range(0,n):
         m[k][i].pop(k)
         m[k][i].insert(k,b[i])
   for k in range(0,n):
      print('x(',k,')=',det(m[k])/det(acop),sep='',end='\n')
>>>
Geef de coëfficiënten van de matrix gescheiden door kommas.
Nadat je de laatste rij hebt ingevoerd, tik <Enter>
1 ste rij co :4,3,5,1
2 ste rij co :2,2,3,0
3 ste rij co :8,-1,4,10
4 ste rij co :
x(0)=0.5
```

```
x(1)=-2.0
x(2)=1.0
>>>
```

49. Berekenen van de inverse matrix met adjunct en determinant

inverse_matrix_met_determinanten

De inverse matrix kunnen we bepalen door elk element van de adjunct te delen door de determinant. De adjunct matrix kan berekend worden door elk element van de getransponeerde matrix te vervangen door zijn minor. Het volgende programma volgt deze methode.

Programma:

```
# inverse matrix met determinanten
from copy import deepcopy
from tabulate import tabulate
def det(mat):
    le=len(mat)
    if le==1:d=mat[0][0]
    elif le==2:d=mat[0][0]*mat[1][1]-mat[0][1]*mat[1][0]
    else:
        d=0
        for i in range(0,le):
            s=[]
            for k in range(0,le):
                if i!=k:s.append(mat[k][1:])
            d=d+mat[i][0]*(-1)**i*det(s)
    return d
def mult(s):
    l=s.split(',')
    co=[]
    for i in l:co.append(float(i))
    return co
# hoofdprogramma
a=[];i=0;inp=' '
print('Geef de coëfficiënten van de matrix gescheiden door kommas.')
print('Nadat je de laatste rij hebt ingevoerd, tik <Enter>')
while inp!='':
    print(i+1,'ste rij co',end=' ')
    inp=input(':')
    if inp!='':
        rijcoef=mult(inp);a.append(rijcoef)
    n=len(a);i=i+1
if len(rijcoef)!=n:
    print('Geen vierkante matrix')
else:
    print('Matrix=');print(tabulate(a))
    acop=deepcopy(a);D=det(a)
    print('Determinant=',D)
    if D==0:
        print('Geen inverse matrix')
    else:
        adjunct=deepcopy(a)
        for i in range(0,n):
            for j in range(0,n):
                minor=deepcopy(a)
```

```
            minor.pop(j)
            minor2=deepcopy(minor)
            for m in range(0,n-1):
               minor2[m].pop(i)
            adjunct[i][j]=(-1)**(i+j)*det(minor2)
      print('Adjunct=');print(tabulate(adjunct))
      invers=deepcopy(adjunct)
      for i in range(0,n):
         for j in range(0,n):
            invers[i][j]=adjunct[i][j]/D
      print('Invers=');print(tabulate(invers))
```

```
>>>
Geef de coëfficiënten van de matrix gescheiden door kommas.
Nadat je de laatste rij hebt ingevoerd, tik <Enter>
1 ste rij co :4,1,5
2 ste rij co :2,6,1
3 ste rij co :5,8,3
4 ste rij co :
Matrix=
-  -  -
4  1  5
2  6  1
5  8  3
-  -  -
Determinant= -31.0
Adjunct=
---  ---  ---
 10   37  -29
 -1  -13    6
-14  -27   22
---  ---  ---
Invers=
----------  ---------  ---------
-0.322581   -1.19355    0.935484
 0.0322581   0.419355  -0.193548
 0.451613    0.870968  -0.709677
----------  ---------  ---------
>>>
```

50. Een veeltermfunctie bepalen die door n gegeven punten gaat.

veelterm_n_punten

Door 2 punten van π_0 gaat de grafiek van een eerstegraadsfunctie (een rechte)

Door 3 punten van π_0 gaat de grafiek van een tweedegraadsfunctie (een parabool)

Door n punten van π_0 gaat de grafiek van een (n-1)-de graadsfunctie

Stel gegeven n-punten (x_0,y_0) , (x_1,y_1) ... (x_{n-1},y_{n-1}). Een veeltermfunctie $y= a_0x^{n-1}+a_1x^{n-2}+...+ a_{n-1}x^0$ zal door deze n punten gaan als elk koppel voldoet aan de vergelijking van de functie.

Door dit uit te drukken krijgen we een stelsel van n vergelijkingen met n onbekenden: opgepast, de onbekenden zijn hier $a_0,a_1,...a_{n-1}$

$a_0x_0^{n-1}+a_1x_0^{n-2}+... \quad +a_{n-1}x_0^0 = y_0$

$a_0x_1^{n-1}+a_1x_1^{n-2}+... \quad +a_{n-1}x_1^0 = y_1$

..

$a_0x_{n-1}^{n-1}+a_1x_{n-1}^{n-2}+... +a_{n-1}x_{n-1}^0 = y_{n-1}$

Na het invoeren van de n-punten $(x_0,y_0),(x_1,y_1)...(x_{n-1},y_{n-1})$, de x_i-waarden in de lijst xt[i],de y_i-waarden in de lijst yt[i], wordt de lijst a lijn voor lijn opgevuld met rijen xm van de opeenvolgende machten van xt[i]. Omdat met xm.append(xt[i]**j) de machten van laag naar hoog gaan, moet de rij na invulling omgedraaid worden met xm.reverse()

```
a=[]
for i in range(0,n):
        xm=[]
        for j in range(0,n):
                xm.append(xt[i]**j)
        xm.reverse()
        a.append(xm)
```

a is dan de matrix van de coëfficiënten van het linkerlid. We stellen ook **b = yt**, de getallen in het rechterlid. Het stelsel wordt dan opgelost met de methode van Cramer. Op het einde wordt de vergelijking van de veeltermfunctie uitgeschreven met de gevonden coëfficiënten.

Programma:

```
# een veeltermfunctie bepalen die door n gegeven punten gaat.
from copy import *
def det(mat):
   le=len(mat)
   if le==2:d=mat[0][0]*mat[1][1]-mat[0][1]*mat[1][0]
   else:
      d=0
      for i in range(0,le):
         s=[]
         for k in range(0,le):
            if i!=k:s.append(mat[k][1:])
         d=d+mat[i][0]*(-1)**i*det(s)
   return d
def mult(s):
   l=s.split(',')
   co=[]
   for i in l:co.append(float(i))
   return co
xt=[];yt=[]
n=int(input("Geef het aantal coördinatenkoppels:"))
for j in range(0,n):
   print("Punt",j+1, end=" :")
   coo=input("x,y=").split(',')
   x=float(coo[0]);y=float(coo[1])
   xt.append(x);yt.append(y)
a=[]
for i in range(0,n):
        xm=[]
        for j in range(0,n):
                xm.append(xt[i]**j)
        xm.reverse()
```

```
            a.append(xm)
acop=deepcopy(a);m=[];b=yt
for k in range(0,n):m.append([])
for k in range(0,n):
       m[k]=deepcopy(acop)
       for i in range(0,n):
          m[k][i].pop(k)
          m[k][i].insert(k,b[i])
print('Veelterm:\ny=',end=' ')
for k in range(0,n):
            co=det(m[k])/det(acop)
            cost=format(co,'5f')
            if co>=0:cost='+'+cost
            print(cost,'x^',n-k-1,sep='',end='')
>>>
Geef het aantal coördinatenkoppels:4
Punt 1 :x,y=1,5
Punt 2 :x,y=3,8
Punt 3 :x,y=4,21
Punt 4 :x,y=6,11
Veelterm:
y= -1.966667x^3+19.566667x^2-51.200000x^1+38.600000x^0
>>>
```

51. Oefening: periodiek sparen

Stel dat we een overeenkomst sluiten om maandelijks een vast bedrag a te sparen aan een maandelijkse rentevoet r gedurende een periode van n maanden. Stel ook nog u=1+r/100.
Op tijdstip 0 sparen we een eerste maal het bedrag a.
Na 1 maand is dit spaarbedrag aangegroeid tot a.(1+r/100)=a.u
Dus totaal na 1 maand = a+a.u
Het volgende tabelletje toont telkens het gespaard bedrag:

t	Spaarbedrag
0	a
1	a+a.u
2	$a+a.u+a.u^2$
3	...
.	...
n-1	$a+a.u+a.u^2+..+..+a.u^{n-1}$

Het gespaarde kapitaal k na n maanden
=u.(kapitaal na n-1 maanden)
$k = u(a+a.u+a.u^2+..+..+a.u^{n-1})$
$= u.a(1+u+u^2+..+..+u^{n-1})$
$k.u= u.a(u+u^2+..+..+u^{n-1}+u^n)$
$k.u-k=u.a(u^n-1)$

Samengevat hebben we:

$u=1+r/100$ $k= u.a.(u^n-1)/(u-1)$

Bepaal, uitgaande van deze formule, formules voor:
a in functie van k,r,n n in functie van k,r,a

Het is niet mogelijk om een formule op te stellen voor r in functie van k,a,n. Wat we wel kunnen, is u benaderen als nulpunt van de verschilfunctie k- u.a.(u^n-1)/(u-1) met de methode van Newton.
En met deze u-waarde berekenen we r.
Maak een programma met een keuzemenu tussen 4 opties:
Bij elke keuze, moet je één van de 4 parameters laten berekenen, als de andere 3 ingevoerd worden.
Je kan eventueel ook met een formulier van tkinter (zie verder) werken.

52. Een mogelijk alternatieve module voor de module math. wis wistest

Een module zoals we er reeds een aantal gebruikt hebben (math, tabulate, graphwin, tkinter,.....) is eigenlijk gewoon een *.py python programma dat uitsluitend bestaat uit definities van functies en constanten.
Het is dus perfect mogelijk om zelf een module te maken.

Het volgende programma is een module die de meest gangbare wiskundige functies bevat.

Sommige functiewaarden kunnen gemakkelijk benaderd worden met een reeksontwikkeling, andere met een integraal, nog andere met de methode van Newton.

Eerst een lijstje van reeksontwikkelingen:

$$\sin(x) = x - \frac{x^3}{3!} + \frac{x^5}{5!} - \frac{x^7}{7!} + \ldots$$

$$\cos(x) = 1 - \frac{x^2}{2!} + \frac{x^4}{4!} - \frac{x^6}{6!} + \ldots$$

$$\text{atn}(x) = \frac{\pi}{2} - \frac{1}{x} + \frac{1}{3x^3} - \frac{1}{5x^5} + \frac{1}{7x^7} - \ldots$$

$$\text{asn}(x) = x + \frac{1}{2}.\frac{x^3}{3} + \frac{1}{2}.\frac{3}{4}.\frac{x^5}{5} + \frac{1}{2}.\frac{3}{4}.\frac{5}{6}.\frac{x^7}{7} + \ldots.$$

$$e^x = 1 + x + \frac{x^2}{2!} + \frac{x^3}{3!} + \frac{x^4}{4!} + \ldots.$$

Welke van deze reeksontwikkelingen convergeren aanvaardbaar snel ?

sin(x) en cos(x) convergeren snel op voorwaarde dat we het argument x eerst herleid hebben tot een waarde in het interval $[0,2\pi]$ met de formule $x \pm k.2\pi \in [0,2\pi]$

asn(x) convergeert trager, maar omdat $x \in [-1,1]$ (domein) is er zeker convergentie.

Een alternatief is simpsonintegratie toe passen op $\frac{1}{\sqrt{1-x^2}}$ tussen 0 en x.

e^x convergeert traag maar ook als x groter wordt, is er nog convergentie: x kan niet echt grote waarden aannemen zonder overflow, en de faculteit in de noemer wordt snel zeer groot.

atn(x) convergeert voor x > 1. Maar het domein van atn = IR

Hier moeten we bij kleine waarden van x, bvb. <3 kiezen voor simpson-integratie $\int_0^x \frac{1}{1+t^2}.dt$

Is x>3 dan nemen we de reeksontwikkeling.

Voor ln(x) maken we steeds gebruik van simpson-integratie van de functie 1/x. Hier kunnen we wel een slechte benadering hebben als x zeer groot of zeer klein is. Dit zouden we als volgt kunnen oplossen:

Stel bvb. x= 254789000=2,54789x10^8
Dan is ln(x)=ln(2,54789)+ln(10^8)=ln(2,54789)+8ln(10)=ln(2,54789)+8/log(e)
stel w=2,54789 m=8 log(e)=ll
ln(x)=ln(w)+m/ll = simpson('1/x',1,w)+m/ll

log(x)=log(e).ln(x)=ll*simpson('1/x',1,w)+m

Voor sqrt(x) is de methode van Heroon zeer geschikt die eigenlijk een bijzonder geval van de methode van Newton is.

nroot(x,n) is een functie die de n-de wortel trekt uit een gegeven getal x: Ook hier wordt de methode van Newton gebruikt: we zoeken een nulpunt van de functie t^n-x .
De benaderingsfunctie is dan $t-\frac{f(t)}{f'(t)}=t-\frac{t^n-x}{n.t^{n-1}}=\frac{(n-1).t^n+x}{n.t^{n-1}}$

of in Pythontaal: t=((n-1)*t**n+x)/t**(n-1)/n

Programma:

```
# module wis
pi=3.141592653589793;e=2.718281828459045;eps=0.00000001
ll=0.43429448190325187
#ll=log(e)
def f(x):
    return eval(fs)
def simpson(fs,a,b):
    n=500;h=(b-a)/n
    x=a
    som=0
    for j in range(1,n):
        x=x+h
        if j%2==0:
          t=2
        else:
          t=4
        som=som+t*f(x)
    som=som+f(a)+f(b)
    integ=som*h/3
    return integ
def mod2p(x):
    q=x/pi/2
    qi=int(q)
    if qi<0:qi=qi-1
    xmo=x-qi*pi*2
    return xmo
def sqrt(x):
    t=0
    while t*t<x:t=t+1
```

```
        ot=t+1
        while abs(t-ot)>eps:
                ot=t;t=(t+x/t)/2
        return (t)
def sin(x):
        x=mod2p(x);t=x;s=0;n=1
        while abs(t)>eps:
                s=s+t; t=-t*x*x/(n+1)/(n+2)
                n=n+2
        return(s)
def cos(x):
        x=mod2p(x);t=1;s=0;n=0
        while abs(t)>eps:
                s=s+t;t=-t*x*x/(n+1)/(n+2)
                n=n+2
        return(s)
def tan(x):return(sin(x)/cos(x))
def atan(x):return (sin(x)/cos(x))
def nroot(x,n):
        t=0
        while t**n<x:t=t+1
        ot=t+1
        while abs(ot-t)>eps:
                ot=t;t=((n-1)*t**n+x)/t**(n-1)/n
        return(t)
def wm(x):
    w=0;m=x
    while m>10:
        w=w+1;m=m/10
    while m<1:
        w=w-1;m=m*10
    return([m,w])
def ln(x):
        global fs;fs='1/x'
        m=wm(x)[0];w=wm(x)[1]
        return(simpson(fs,1,m)+w/ll)
def log(x):
    global fs;fs='1/x'
    m=wm(x)[0];w=wm(x)[1]
    return(ll*simpson(fs,1,m)+w)
def atn(x):
    global fs;fs='1/(1+x*x)'
    if x*x<=10:
        res=simpson(fs,0,x)
    else:
        t=-1/x;res=pi/2+t;n=1
        while abs(t/n)>eps:
            t=-t/x/x;n=n+2;res=res+t/n
    return(res)
def asn(x):
        s=x;n=1;t=x
```

```
        while abs(t)>eps:
                t=t*x*x*(2*n-1)**2/n/2/(2*n+1)
                n=n+1
                s=s+t
        return(s)
def acs(x):return(pi/2-asn(x))
def exp(x):
    t=x;res=1+t;n=1;f=1
    while abs(t)>eps:
        n=n+1;t=t*x/n;res=res+t
    return(res)
def rad(x):return pi*x/180
def deg(x):return 180*x/pi
```

Om deze module uit te testen, het volgende programmaatje:

Programma:

```
# test wis
from wis import *
print(sqrt(5),nroot(32,5))
print(sin(rad(30)),cos(rad(135)),tan(rad(135)))
print(ln(exp(3)),log(1000))
print(deg(asn(sqrt(2)/2)),deg(acs(1/2)),(deg(atn(sqrt(3)/3))))

>>>
2.23606797749979 2.0
0.4999999918690232 -0.7071067804153877 -0.9999999927725302
2.999999999917366 3.000000001518363
44.99999952370583 60.000000064116186 30.000000000001677
>>>
```

De benadering van de juiste waarden is een stuk minder goed dan met de echte module math. Niettemin is de module niet slecht als oefening op reeksontwikkelingen, numerieke integraalberekening en methode van Newton.

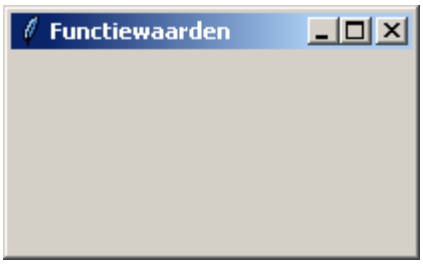
53. Werken met de module tkinter

demo_functie_tk

Alle vorige programma's werden uitgevoerd in de 'Shell' of in de 'graphics' module.
Maar je kan de programma's ook in een Windowsformulier laten uitvoeren
Met de module **tkinter** hebben we gemakkelijk toegang tot de GUI: dit is de Graphic Unit Interface; het laat ons toe om onze programma's een Windows- uiterlijk te geven.
tkinter is een module die zowel toegankelijk is vanuit python als vanuit pydroid.
Stapsgewijze maak je nu kennis met enkele mogelijkheden die deze module geeft.

1. Een venster creëren

```
from tkinter import *
form=Tk();form.geometry('200x100');form.title('Functiewaarden')
form.mainloop()
```



Met form = Tk() wordt een formulier gedefinieerd, die met form.geometry('200x100') afmetingen '200x100' en met form.title('Functiewaarden') als titel 'Functiewaarden' krijgt.

In de meeste tutorials neemt men als variabele tk, dus tk=Tk()
Ik gebruik de variabele form (formulier) naar analogie met Visual Basic.
Maar je kiest natuurlijk zelf je eigen variabelen.

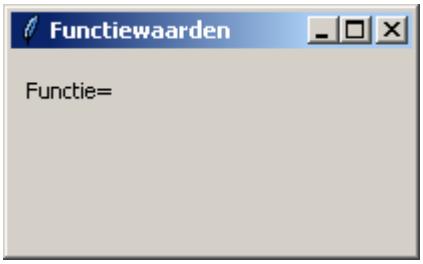
form.mainloop() zorgt voor een eventloop (die dus wacht op een gebeurtenis) en waaruit je alleen kan ontsnappen als je klikt op ✕. Gebeurt er ergens een fout, bvb. omdat je iets verkeerd, te veel of te weinig hebt ingevoerd, dan krijg je een foutmelding in de shell 'op de achtergrond' en kan je meestal het programma gewoon verder zetten.

2. Een label toevoegen (label = een stukje tekst)

Voeg voor de laatste lijn de instructie toe:
L1=Label(form,text='Functie=',);L1.place(x=5,y=10)

Opmerking: Je mag hier ook kortweg tikken:
Label(form,text='Functie=',).place(x=5,y=10)

Deze instructie zegt dus: vul in het formulier form de tekst 'Functie=' in op positie (x=5,y=10) van linksboven gerekend.



3. Een Entry toevoegen

Voeg toe: E1=Entry(form);E1.place(x=60,y=10)

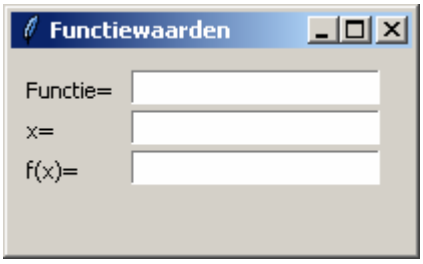
Het programma ziet er nu zo uit:

```
from tkinter import *
form=Tk();form.geometry('200x130');form.title('Titel')
L1=Label(form,text='Functie=',);L1.place(x=5,y=10)
E1=Entry(form);E1.place(x=60,y=10)
form.mainloop()
```

De bedoeling van zo een entry- venster is om de gebruiker iets te laten invoeren, in ons geval zal dat een wiskundige functie zijn.

We voegen nu nog een 2de en 3de label en entry toe:

```
from tkinter import *
form=Tk();form.geometry('200x130');form.title('Functiewaarden')
L1=Label(form,text='Functie=',);L1.place(x=5,y=10)
E1=Entry(form);E1.place(x=60,y=10)
L2=Label(form,text='x=',);L2.place(x=5,y=30)
E2=Entry(form);E2.place(x=60,y=30)
L3=Label(form,text='f(x)=',);L3.place(x=5,y=50)
E3=Entry(form);E3.place(x=60,y=50)
form.mainloop()
```



4. Opdrachtknoppen toevoegen

De bedoeling zal reeds duidelijk zijn.
Stel dat we een functie invoeren in de 1ste entry en een getal in de 2de, dan zouden we een knop (een ' Button') moeten hebben die ons de functiewaarde levert in de 3de entry

```
B1=Button(form,text='Bereken',command=bereken);B1.place(x=25,y=75)
```

Als je op de knop 'Bereken' klikt wordt de functie bereken opgeroepen,

In de definitie van deze functie wordt daartoe eerst de veranderlijke x gelijkgesteld aan de float van de tekst in E2: `x=float(E2.get())`
Dan wordt y = f(x) berekend met: `y= eval(E1.get())`
En de string van deze waarde wordt ingevuld in E3: `E3.insert(0,str(y))`
Opmerking: vooraf is het aangeraden om E3 leeg te maken: `E3.delete(0,len(E3.get()))`

Een 2de opdrachtknop 'Nieuw' dient om alle entries leeg te maken

```
B2=Button(form,text='Nieuw',command=nieuw);B2.place(x=125,y=75)
```

Deze knop roept de functie **nieuw** op:

```
for ent in [E1,E2,E3]:ent.delete(0,len(ent.get()))
```

Opmerkingen:

- functies worden voor het hoofdprogramma geplaatst.
- om alle wiskundige functies te mogen invoeren, importeren we ook de module 'math'

Progamma:

```
# functiewaarden berekenen
from math import *
from tkinter import *
def bereken():
    x=float(E2.get());y= eval(E1.get())
    E3.delete(0,len(E3.get()));E3.insert(0,str(y))
def nieuw():
    for ent in [E1,E2,E3]:ent.delete(0,len(ent.get()))
form=Tk();form.geometry('200x110');form.title('Functiewaarden')
L1=Label(form,text='Functie=',);L1.place(x=5,y=10)
E1=Entry(form);E1.place(x=60,y=10)
L2=Label(form,text='x=',);L2.place(x=5,y=30)
E2=Entry(form);E2.place(x=60,y=30)
L3=Label(form,text='f(x)=',);L3.place(x=5,y=50)
E3=Entry(form);E3.place(x=60,y=50)
B1=Button(form,text='Bereken',command=bereken);B1.place(x=25,y=75)
B2=Button(form,text='Nieuw',command=nieuw);B2.place(x=125,y=75)
form.mainloop()
```

54. methode van Heroon met tkinter heroonTk

Programma:

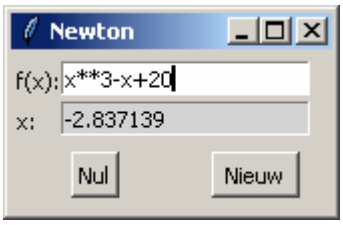
```
# heroon met tkinter
from tkinter import *
def bereken():
    a=float(E1.get());x=0
    while x*x<a:x=x+1
    while abs(x*x-a)>eps:x=(x+a/x)/2
    wis(E2);E2.insert(0,format(x,'.6f'))
def wis(E):E.delete(0,len(E.get()))
def nieuw():wis(E1);wis(E2)
# main
form=Tk();form.geometry('160x80');form.title('Vierkantswortel')
Label(form,text='a:').place(x=0,y=5)
E1=Entry(form);E1.place(x=25,y=5)
Label(form,text='Va:').place(x=0,y=25)
E2=Entry(form,bg='lightgrey');E2.place(x=25,y=25)
```

```
Button(form,text='Wortel',command=bereken).place(x=30,y=50)
Button(form,text=' Nieuw ',command=nieuw).place(x=100,y=50)
eps=1E-6;form.mainloop()
```

55. Methode van Newton met tkinter newtonTk

Programma:

```
# newton met tkinter: berekenen kleinste nulpunt f(x)
from tkinter import *;from math import *
def f(x):return eval(E1.get())
def df(x):return(f(x+dx)-f(x))/dx
def bereken():
    x=-100
    while f(x)*f(x+1)>0:x=x+1
    while abs(f(x))>1E-6:x=x-f(x)/df(x)
    wis(E2);E2.insert(0,format(x,'.6f'))
def wis(E):E.delete(0,len(E.get()))
def nieuw():wis(E1);wis(E2)
# main
form=Tk();form.geometry('160x80');form.title('Newton')
Label(form,text='f(x):').place(x=0,y=5)
E1=Entry(form);E1.place(x=25,y=5)
Label(form,text='x:').place(x=0,y=25)
E2=Entry(form,bg='lightgrey');E2.place(x=25,y=25)
Button(form,text='Nul',command=bereken).place(x=30,y=50)
Button(form,text=' Nieuw ',command=nieuw).place(x=100,y=50)
dx=1E-6;form.mainloop()
```



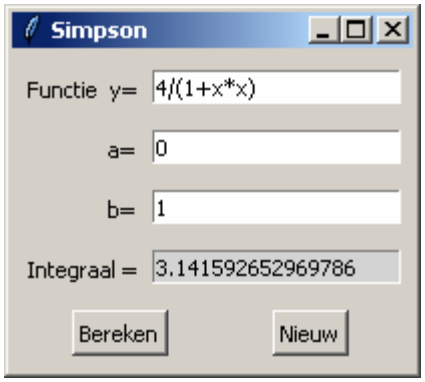
56. Een numerieke integraalberekening in een windows- formulier simpson_tk

Maak een formulier waarbij je in de invoervensters een functie, en de integratiegrenzen ingeeft. Na het drukken op een knop (Button) wordt de integraal berekend. In de functie bereken() vul je het algoritme in voor de numerieke berekening van de integraal met de methode van Simpson.
Een mogelijke oplossing zie je hieronder:

Programma:

```
#integraalberekening
from math import *
```

```
from tkinter import *
def f(x):return(eval(E1.get()))
def nieuw():
   for ent in [E1,E2,E3,E4]:ent.delete(0,len(ent.get()))
def bereken():
        a=float(E2.get());b=float(E3.get());n=20
        h=(b-a)/n;x=a;som=0
        for j in range(1,n):
                x=x+h
                if j%2==0:t=2
                else:t=4
                som=som+t*f(x)
        som=som+f(a)+f(b)
        integ=som*h/3
        E4.insert(0,str(integ))
#hoofdprogramma
form=Tk();form.geometry('200x160');form.title('Simpson')
Label(form,text='Functie  y=').place(x=5,y=10)
E1=Entry(form);E1.place(x=70,y=10) #invoer functie
Label(form,text='a=').place(x=45,y=40)
E2=Entry(form);E2.place(x=70,y=40)
Label(form,text='b=').place(x=45,y=70)
E3=Entry(form);E3.place(x=70,y=70)
Label(form,text='Integraal =').place(x=5,y=100)
E4=Entry(form,bg='lightgrey');E4.place(x=70,y=100)
B1=Button(form,text='Bereken',command=bereken);B1.place(x=30,y=130)
B2=Button(form,text='Nieuw',command=nieuw);B2.place(x=130,y=130)
form.mainloop()
```



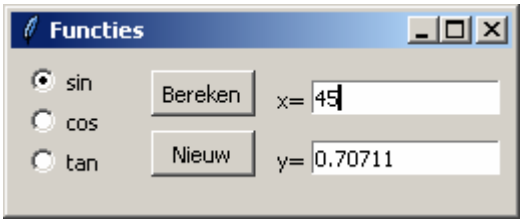
Bereken analytisch $\int_0^1 \frac{4}{1+x^2}\,dx$. En, klopt het met het antwoord van jouw programma ?

57. Radiobuttons demo_radiobuttons

Bij programma's die enkel uitgevoerd worden in de shell, kunnen we met een while-loop de gebruiker laten kiezen tussen verschillende mogelijkheden.
Bij gebruik van een tkinter-formulier gebruiken we daarvoor beter een rijtje van radiobuttons
Het volgende programma laat toe om een goniometrisch getal van een hoek, gegeven in decimale graden te berekenen.

Programma:

```
#demo radiobuttons
from math import *
from tkinter import *
def nieuw():
    for ent in [xE,yE]:ent.delete(0,len(ent.get()))
def bereken():
    x=radians(float(xE.get()));y=eval(keuzelijst[keuze]+'(x)')
    yE.delete(0,len(yE.get()));yE.insert(0,format(y,'.5f'))
def selec():global keuze;keuze=var.get()
def entr(lab,xp,yp,w):
    Label(form,text=lab+'=').place(x=xp,y=yp)
    ent=Entry(form,width=w);ent.place(x=xp+20,y=yp)
    return ent
form=Tk();form.geometry('250x80');form.title('Functies')
xE=entr('x',130,15,15);yE=entr('y',130,45,15)
Button(form,text='Bereken',command=bereken,width=7).place(x=70,y=10)
Button(form,text='Nieuw',command=nieuw,width=7).place(x=70,y=40)
kl=['sin','cos','tan'];var=IntVar();keuze=0
for i in range(0,3):
    rb=Radiobutton(form,text=kl[i],variable=var,value=i,command=selec)
    rb.place(x=5,y=i*20+5)
form.mainloop()
```



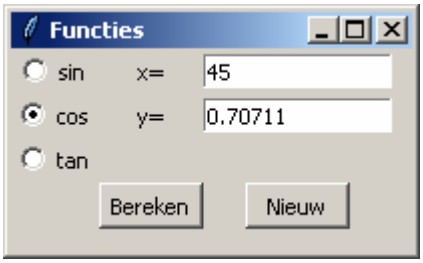
Opmerking (variante)

Je kan ook de plaatsing van de labels, entries en knoppen overlaten aan tkinter (met **grid**)

Programma:

```
#demo radiobuttons 2 (met grid)
from math import *
from tkinter import *
def nieuw():
    for ent in [xE,yE]:ent.delete(0,len(ent.get()))
def bereken():
    x=radians(float(xE.get()));y=eval(keuzelijst[keuze]+'(x)')
    yE.delete(0,len(yE.get()));yE.insert(0,format(y,'.5f'))
def selec():global keuze;keuze=var.get()
def entr(lab,r,w):
    Label(form,text=lab+'=').grid(column=2,row=r)
    ent=Entry(form,width=w);ent.grid(column=3,row=r)
    return ent
```

```
form=Tk();form.geometry('210x110');form.title('Functies')
xE=entr('x',0,15);yE=entr('y',1,15)
Button(form,text='Bereken',command=bereken,width=7).grid(row=3,column=2)
Button(form,text='Nieuw',command=nieuw,width=7).grid(row=3,column=3)
kl=['sin','cos','tan'];var=IntVar();keuze=0
for i in range(0,3):
   rb=Radiobutton(form,text=kl[i],variable=var,value=i,command=selec)
   rb.grid(row=i,column=1)
form.mainloop()
```



Het programma wordt er niet korter door maar we moeten zelf voor de plaatsing van de widgets geen berekeningen maken.
Ik heb de gewoonte om zelf de x,y-coordinaten van de widgets te berekenen. D.w.z. berekenen, uitproberen, aanpassen, tot als ze naar mijn mening goed staan.

Nog een woordje over de werking van radiobuttons: neem het stukje programma:

```
def selec():global keuze;keuze=var.get()
.........
kl=['sin','cos','tan']; var=IntVar(); keuze=0
for i in range(0,3):
   rb=Radiobutton(form,text=kl[i],variable=var,value=i,command=selec)
   rb.grid(row=i,column=1)
```

Voor de definitie van de radiobuttons wordt vooraf een integer- variabele var gedeclareerd
Als bvb. i=1 hebben we de Radiobutton 'cos' met 'value' 1.
Als je een keuze doet, een radiobutton selecteert , dan krijgt var de 'waarde' van de geselecteerde button en wordt de functie selec (command=...) uitgevoerd. In dit geval wordt met de functie selec enkel keuze gelijkgesteld aan var.get() . Met deze keuze kan de gepaste berekening gemaakt worden als we nadien op de knop bereken klikken.
Nog iets, als je vooraf keuze geen initiële waarde geeft, en je wil onmiddellijk 'bereken'-en , dan wordt de functie selec niet opgeroepen en kent hij de variabele 'keuze' niet.
Vandaar ook : keuze=0

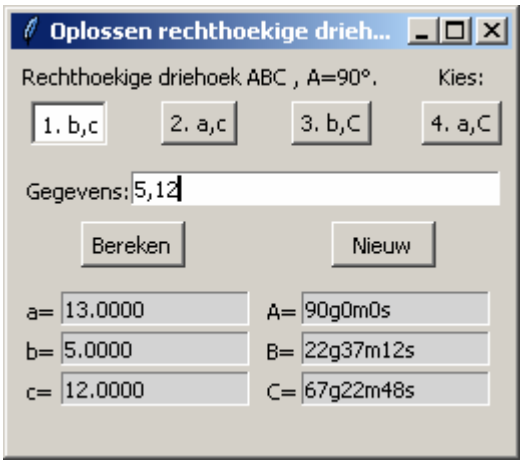
De volgende 2 programmas zijn variantes op het oplossen van rechthoekige en willekeurige driehoeken met tkinter. Beide programmas maken gebruik van radiobuttons: in het eerste geval is indicatoron= 0 gesteld. I.p.v. radiobuttons krijg je dan buttons waarvan de geselecteerde keuze verzonken is.

58. Oplossen van rechthoekige driehoeken met tkinter rech_drieh_tk

Programma:

```
# oplossen van rechthoekige driehoeken
from math import *;from tkinter import *
# dms ==>dd
def dg(hoek):
    gpos=hoek.index('g');gr=float(hoek[0:gpos]);hoek=hoek[gpos+1:]
    mpos=hoek.index('m');mi=float(hoek[0:mpos]);hoek=hoek[mpos+1:]
    spos=hoek.index('s');se=float(hoek[0:spos])
    dechoek=gr+mi/60+se/3600
    return dechoek
# dd ==>dms
def gms(hoekdd):
    hoekstr=''
    gr=int(hoekdd);hoekstr=hoekstr+str(gr)+'g';decmi=60*(hoekdd-gr)
    mi=int(decmi);hoekstr=hoekstr+str(mi)+'m';decs=60*(decmi-mi)
    se=round(decs);hoekstr=hoekstr+str(se)+'s'
    return hoekstr
def mult(s):
    l=s.get().split(',');co=[]
    for i in l:
        if 'g' in i:co.append(dg(i))
        else:co.append(float(i))
    return co
def nieuw():
    for ent in [E1,aE,bE,cE,AE,BE,CE]:ent.delete(0,len(ent.get()))
def selec():global keuze;keuze=var.get()
def entr(lab,xp,yp):
    Label(form,text=lab+'=').place(x=xp,y=yp)
    ent=Entry(form,bg='lightgrey',width=15);ent.place(x=xp+20,y=yp)
    return ent
def r(z):return format(z,'.4f')+'  '
def oplossing():
    nieuw()
    aE.insert(0,r(a));bE.insert(0,r(b));cE.insert(0,r(c))
    AE.insert(0,gms(A));BE.insert(0,gms(B));CE.insert(0,gms(C))
def bereken():
    global uitleg,a,b,c,A,B,C;A=90.0
    if keuze==0:b,c=mult(E1);a=sqrt(b*b+c*c);B=degrees(asin(b/a));C=90-B;oplossing()
    if keuze==1:a,c=mult(E1);b=sqrt(a*a-c*c);B=degrees(asin(b/a));C=90-B;oplossing()
    if keuze==2:b,C=mult(E1);B=90-C;c=b*tan(radians(C));a=sqrt(b*b+c*c);oplossing()
    if keuze==3:a,C=mult(E1);B=90-C;c=a*sin(radians(C));b=sqrt(a*a-c*c);oplossing()
#hoofdprogramma
form=Tk();form.geometry('250x200');form.title('Oplossen rechthoekige driehoeken')
Label(form,text='Rechthoekige driehoek ABC , A=90°.       Kies:').place(x=3,y=3)
Label(form,text='Gegevens:').place(x=5,y=60);E1=Entry(form,width=30);
E1.place(x=60,y=60)
aE=entr('a',5,120);bE=entr('b',5,140);cE=entr('c',5,160)
AE=entr('A',125,120);BE=entr('B',125,140);CE=entr('C',125,160)
```

```
Button(form,text='Bereken',command=bereken,width=7).place(x=35,y=85)
Button(form,text='Nieuw',command=nieuw,width=7).place(x=160,y=85)
k1='1. b,c';k2='2. a,c';k3='3. b,C';k4='4. a,C'
kl=[k1,k2,k3,k4]
var=IntVar();keuze=0
for i in range(0,4):
  rb=Radiobutton(form,text=kl[i],variable=var,value=i,command=selec, indicatoron=0)
    rb.place(x=65*i+10,y=25)
form.mainloop()
```



59. Oplossen van willekeurige driehoeken met tkinter will_drieh_tk

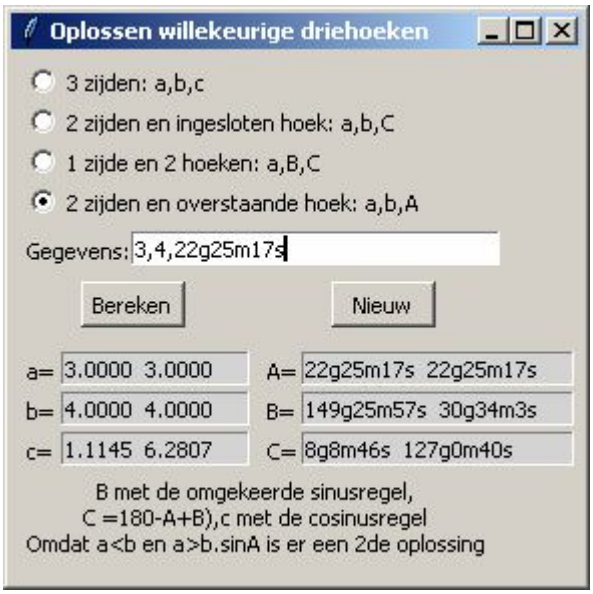
Ook 'oplossen van willekeurige driehoeken' kunnen we een formulier-uiterlijk geven. De listing is grotendeels overgenomen van het gelijkaardig programma in de 'shell'
Niet elke verkeerde invoer is beveiligd: voert men bvb. 3 zijden in waarvan één groter is dan de som van de andere, dan krijgen we een foutmelding in de shell. Omdat het programma in de form.mainloop zit kan je het programma (meestal) gewoon verder zetten. Je kan natuurlijk zelf het programma aanpassen dat het beveiligd is tegen alle mogelijke 'verkeerde' invoer.

Programma:

```
from math import *
from tkinter import *
# dms ==>dd
def dg(hoek):
    gpos=hoek.index("g");gr=float(hoek[0:gpos]);hoek=hoek[gpos+1:]
    mpos=hoek.index("m");mi=float(hoek[0:mpos]);hoek=hoek[mpos+1:]
    spos=hoek.index("s");se=float(hoek[0:spos])
    dechoek=gr+mi/60+se/3600
    return dechoek
# dd ==>dms
def gms(hoekdd):
    hoekstr=''
    gr=int(hoekdd);hoekstr=hoekstr+str(gr)+'g';decmi=60*(hoekdd-gr)
    mi=int(decmi);hoekstr=hoekstr+str(mi)+'m';decs=60*(decmi-mi)
    se=round(decs);hoekstr=hoekstr+str(se)+'s  '
    return hoekstr
# goniometrische functies
```

```
def invcos(p,q,r):return degrees(acos((q*q+r*r-p*p)/2/q/r))
def sinreg(p,Q,R):return p*sin(radians(Q))/sin(radians(R))
def cosreg(p,q,R):return sqrt(p*p+q*q-2*p*q*cos(radians(R)))
def invsin(p,q,P):return degrees(asin(q*sin(radians(P))/p))
def r(z):return format(z,'.4f')+'  '
def oplossing():
   if sol2==0:nieuw()
   Label(form,text=uitleg).place(x=5,y=210)
   aE.insert(0,r(a));bE.insert(0,r(b));cE.insert(0,r(c))
   AE.insert(0,gms(A));BE.insert(0,gms(B));CE.insert(0,gms(C))
def mult(s):
   l=s.get().split(',')
   co=[]
   for i in l:
      if 'm' in i:co.append(dg(i))
      else:co.append(float(i))
   return co
def bereken():
   global uitleg,a,b,c,A,B,C,sol2;sol2=0
   if keuze==0:
      a,b,c=mult(E1);A=invcos(a,b,c);B=invcos(b,c,a);C=invcos(c,a,b)
      uitleg='De 3 hoeken worden berekend\n met de omgekeerde cosinusregel';oplossing()
   if keuze==1:
      a,b,C=mult(E1);c=cosreg(a,b,C);A=invcos(a,b,c);B=invcos(b,c,a)
      uitleg='De 3de zijde c met de cosinusregel,\nA en B met de omgekeerde cosinusregel';oplossing()
   if keuze==2:
      a,B,C=mult(E1);A=180-B-C;c=sinreg(a,C,A);b=sinreg(a,B,A)
      uitleg='A = 180-(B+C),\nb en c met de sinusregel';oplossing()
   if keuze==3:
      a,b,A=mult(E1)
      if a<b*sin(radians(A)):
         uitleg='Omdat a<b.sinA is er geen oplossing\n';Label(form,text=uitleg).place(x=5,y=210)
      else:
         B=invsin(a,b,A);C=180-A-B;c=cosreg(a,b,C)
         uitleg='B met de omgekeerde sinusregel,\nC =180-A+B),c met de cosinusregel';oplossing()
         if a<b:
            B=180-B;C=180-A-B;c=cosreg(a,b,C)
            uitleg=uitleg+'\nOmdat a<b en a>b.sinA is er een 2de oplossing';sol2=1;oplossing()
def nieuw():
   for ent in [E1,aE,bE,cE,AE,BE,CE]:ent.delete(0,len(ent.get()))
   bl=(' ').ljust(80);wis=bl+'\n'+bl+'\n'+bl+'\n'
   Label(form,text=wis).place(x=5,y=210)
def selec():global keuze;keuze=var.get()
def entr(lab,xp,yp,w):
   Label(form,text=lab+'=').place(x=xp,y=yp)
   ent=Entry(form,bg='lightgrey',width=w);ent.place(x=xp+20,y=yp)
   return ent
#hoofdprogramma
form=Tk();form.geometry('285x265');form.title('Oplossen willekeurige driehoeken')
Label(form,text='Gegevens:').place(x=5,y=90);E1=Entry(form,width=30);E1.place(x=60,y=90)
aE=entr('a',5,150,15);bE=entr('b',5,170,15);cE=entr('c',5,190,15)
```

```
AE=entr('A',125,150,22);BE=entr('B',125,170,22);CE=entr('C',125,190,22)
Button(form,text='Bereken',command=bereken,width=7).place(x=35,y=115)
Button(form,text='Nieuw',command=nieuw,width=7).place(x=160,y=115)
k1="3 zijden: a,b,c"
k2="2 zijden en ingesloten hoek: a,b,C "
k3="1 zijde en 2 hoeken: a,B,C"
k4="2 zijden en overstaande hoek: a,b,A"
kl=[k1,k2,k3,k4]
var=IntVar();keuze=0
for i in range(0,4):
   rb=Radiobutton(form,text=kl[i],variable=var,value=i,command=selec)
   rb.place(x=5,y=i*20+5)
form.mainloop()
```



Opmerking: beide programma's verwachten dat als de gebruiker een hoek moet ingeven, dit van de vorm ..g..m..s is: wens je dit niet, dan moet je def dg(hoek):..... aanpassen, d.w.z. beveiligen tegen een verkeerde invoer.

60. Numerieke afgeleiden

Alhoewel het gemakkelijk is om analytisch het voorschrift van de afgeleide functie f'(x) van een willekeurige functie te berekenen, is het op een PC eenvoudiger om een numerieke waarde f'(a) van de afgeleide te bepalen: we hebben dit reeds een paar keer gedaan met het differentiequotiënt $\frac{f(x+dx)-f(x)}{dx}$ voor een kleine dx.
Bij de methode van Newton alsook bij de grafiek van de afgeleide functie steekt het eigenlijk niet zo nauw hoe klein we dx nemen. Hebben we daarentegen een booglengte of de zijdelingse oppervlakte van een omwentelingslichaam te berekenen met Simpsonintegratie:
(met de formules: $\int_a^b \sqrt{1+y'^2}\,dx$ en $\int_a^b 2\pi\cdot|y|\cdot\sqrt{1+y'^2}\,dx$)
dan moeten we de best mogelijke dx nemen: heel klein maar toch ook niet te klein
Een goede waarde van dx vinden we met een testprogramma: neem bvb. $f(x)=x^5$,
Dan is $f'(x)=5x^4$ en $f'(1)=5$.

We laten het differentiequötiënt df(x) berekenen voor dx variërend van 10^{-5} tot 10^{-9}.

```
# test dx
def f(x):return x**5
def df(x):return (f(x+dx)-f(x))/dx
for t in range(5,10):
    dx=10**(-t)
    print(t,df(1))
>>>
5 5.000100001040231
6 5.000009999589494
7 5.000001002120058
8 5.000000080634948
9 5.000000413701855
>>>
```

df(1) ligt het dichtst bij 5, voor dx=10^{-8} . Dit is dan ook onze keuze.

Toepassing: de numerieke afgeleide berekenen van een willekeurige functie in een willekeurig punt

Voorbeeld: de afgeleide functie van tan(x) = $\frac{1}{\cos^2 x}$

Programma:

```
# numerieke afgeleide
from math import *
def f(x):return eval(fs)
def df(x):return (f(x+dx)-f(x))/dx
dx=10**-8
fs=input('Geef een functie:')
a=float((input('Geef een getal:')))
print('Afgeleide=',df(a))

>>>
Geef een functie:tan(x)
Geef een getal:2
Afgeleide= 5.774399047808743
>>> print(1/cos(2)**2)
5.774399204041917
>>>
```

Om de benadering te vergelijken met de correcte waarde, zie je dat $\frac{1}{\cos^2(2)}$

pas bij het 7de cijfer na de komma een afwijking vertoond.

Opmerking: het kan ook nog anders.

Als we vanuit x een stap dx vooruitgaan, krijgen we voor df(x)= $\frac{f(x+dx)-f(x)}{dx}$.

Doen we vanuit x een stap dx achteruit, dan hebben we df(x)= $\frac{f(x)-f(x-dx)}{dx}$.

Een betere df(x) vinden we door van deze 2 het rekenkundig gemiddelde te nemen:

df(x)= $\frac{f(x+dx)-f(x-dx)}{2\cdot dx}$. Een zeer goede benadering vinden we al voor dx=10^{-6}

Indien we ook de 2de afgeleide nodig hebben, kunnen we dezelfde formule toepassen:

Omdat $df(x) \cong \frac{f(x+dx)-f(x-dx)}{2\cdot dx}$ is

$$d2f(x) \cong \frac{df(x+dx)-df(x-dx)}{2\cdot dx} = \frac{\frac{f(x+2\cdot dx)-f(x)}{2\cdot dx} - \frac{f(x)-f(x-2\cdot dx)}{2.dx}}{2.dx}$$

of als we 2·dx =dx stellen: $d2f(x) \cong \frac{f(x+dx)+f(x-dx)-2\cdot f(x)}{dx^2}$

In python:
def d2f(x):return (f(x+dx)+f(x-dx)-2*f(x))/dx**2
Voor de 2de afgeleide krijgen we het beste resultaat als we dx=1E-4 stellen.

61. Tkinter programma om oppervlaktes, inhouden, booglengtes en zijdelingse oppervlaktes te berekenen simpsonTk_inhoud_booglengte_zijdopp

Je kent de formules voor de inhoud van een omwentelingslichaam: $\int_a^b \pi.f(x)^2.dx$

De booglengte: $\int_a^b \sqrt{1+f'(x)^2}.dx$

De zijdelingse oppervlakte van een omwentelingslichaam: $\int_a^b 2\pi\cdot|f(x)|.\sqrt{1+f'(x)^2}.dx$

Slechts voor weinig functies f(x) kunnen de primitieve functies berekend worden met deze formules. Vandaar dat een numerieke methode hier een oplossing kan bieden. We maken wijzigingen aan het vorige Simpson- programma om elk van deze bepaalde integralen te berekenen.We brengen op het formulier radiobuttons aan, waarmee de gebruiker kan kiezen tussen oppervlakte, inhoud, booglengte en zijdelingse oppervlakte.

Programma:

```
#integraalberekening
from math import *
from tkinter import *
def f(x):return(eval(E1.get()))
# f= ingevoerde functie df=afgeleide
# inh=inhoud bl=booglengte zo=zijdelingse oppervlakte
def df(x):return(f(x+dx)-f(x))/dx
def inh(x):return pi*f(x)*f(x)
def bl(x):return sqrt(1+df(x)**2)
def zo(x):return 2*pi*abs(f(x))*sqrt(1+df(x)**2)
def selec():global k;k=var.get()
def wis(ent):ent.delete(0,len(ent.get()))
def nieuw():wis(E1);wis(E2);wis(E3);wis(E4)
def bereken():
    wis(E4);a=float(E2.get());b=float(E3.get());n=100
    h=(b-a)/n;x=a;som=0
    for j in range(1,n):
        x=x+h
```

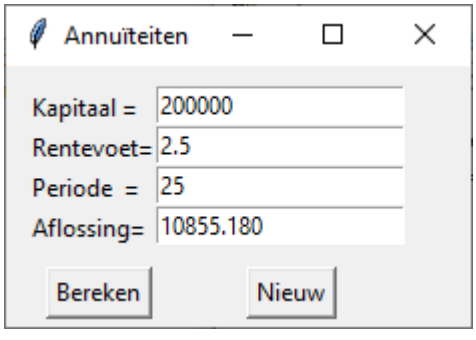
```
                if j%2==0:t=2
                else:t=4
                if k==0:som=som+t*f(x)
                if k==1:som=som+t*inh(x)
                if k==2:som=som+t*bl(x)
                if k==3:som=som+t*zo(x)
        if k==0:som=som+f(a)+f(b)
        if k==1:som=som+inh(a)+inh(b)
        if k==2:som=som+bl(a)+bl(b)
        if k==3:som=som+zo(a)+zo(b)
        integ=som*h/3
        E4.insert(0,str(integ))
#hoofdprogramma
form=Tk();form.geometry('200x220');form.title('Simpson 2')
Label(form,text='Functie  y=').place(x=5,y=10)
E1=Entry(form);E1.place(x=70,y=10) #invoer functie
Label(form,text='a=').place(x=45,y=40)
E2=Entry(form);E2.place(x=70,y=40)
Label(form,text='b=').place(x=45,y=70)
E3=Entry(form);E3.place(x=70,y=70)
Label(form,text='Integraal =').place(x=5,y=100)
E4=Entry(form,bg='lightgrey');E4.place(x=70,y=100)
#enkel uitvoer integraal
B1=Button(form,text='Bereken',command=bereken,width=7);B1.place(x=130,y=140)
B2=Button(form,text='Nieuw',command=nieuw,width=7);B2.place(x=130,y=170)
keuze=['Oppervlakte','Inhoud','Booglengte','Zijdel.opp.']
var=IntVar();k=0;dx=10**-8
for i in range(0,4):
   rb=Radiobutton(form,text=keuze[i],variable=var,value=i,command=selec)
   rb.place(x=5,y=i*20+130)
form.mainloop()
```

62. Annuïteitsmenu met tkinter annuiteiten_tkinter

Het volgende programma is een variante van het annuïteitsmenu dat reeds eerder besproken werd. Hier wordt ook gebruik gemaakt van tkinter. De 4 invoervelden (entries) staan ingesteld op 0.
Bedoeling is om 3 willekeurige velden in te vullen. Met de button 'Bereken' wordt het 4de veld ingevuld.

```
# annuïteiten
from math import *
```

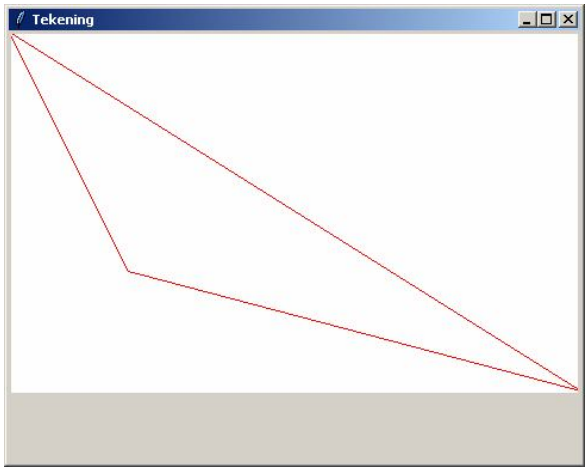
```
from tkinter import *
def mult(s):
    l=s.split(',')
    co=[]
    for i in l:co.append(float(i))
    return co
def nul(E):
    E.delete(0,len(E.get()));E.insert(0,'0')
def inv(ts,yco):
    Label(form,text=ts).place(x=10,y=yco)
    k=Entry(form);k.place(x=75,y=yco)
    return k
def nieuw():
    nul(kap);nul(ren);nul(per);nul(los)
def f(k,r,n,a):return a-k*r/100*(1+(1/((1+r/100)**n-1)))
def df(k,r,n,a):
    dr=0.00001
    return (f(k,r+dr,n,a)-f(k,r,n,a))/dr
form=Tk();form.geometry('210x130');form.title('Annuïteiten')
kap=inv('Kapitaal =',10);ren=inv('Rentevoet=',30)
per=inv('Periode  =',50);los=inv('Aflossing=',70)
nieuw()
def bereken():
    k=float(kap.get());r=float(ren.get())
    n=float(per.get());a=float(los.get())
    if k==0:
        k=100*a/r/(1+(1/((1+r/100)**n-1)))
        kap.insert(0,format(k,'.2f'))
    if r==0:
        r=-0.9;t=0
        while f(k,r,n,a)*f(k,r+1,n,a)>0 and t<100:
            r=r+1;t=t+1
        oudr=r+1
        while abs(oudr-r)>0.0001 and t<100:
            oudr=r;r=r-f(k,r,n,a)/df(k,r,n,a)
        if t<100 and r>0:
            ren.insert(0,format(r,'.2f'))
        else:
            messagebox.showinfo("Foutmelding", "Geen oplossing")
    if a==0:
        a=k*r/100*(1+(1/((1+r/100)**n-1)))
        los.insert(0,format(a,'.2f'))
    if n==0:
        n=log(1/(100*a/k/r-1)+1)/log(1+r/100)
        per.insert(0,format(n,'.2f'))
B1=Button(form,text='Bereken',command=bereken)
B1.place(x=20,y=100)
B2=Button(form,text='Nieuw',command=nieuw)
B2.place(x=120,y=100)
form.mainloop()
```



63. Grafieken van wiskundige functies tekenen met tkinter grafiek_tkinter

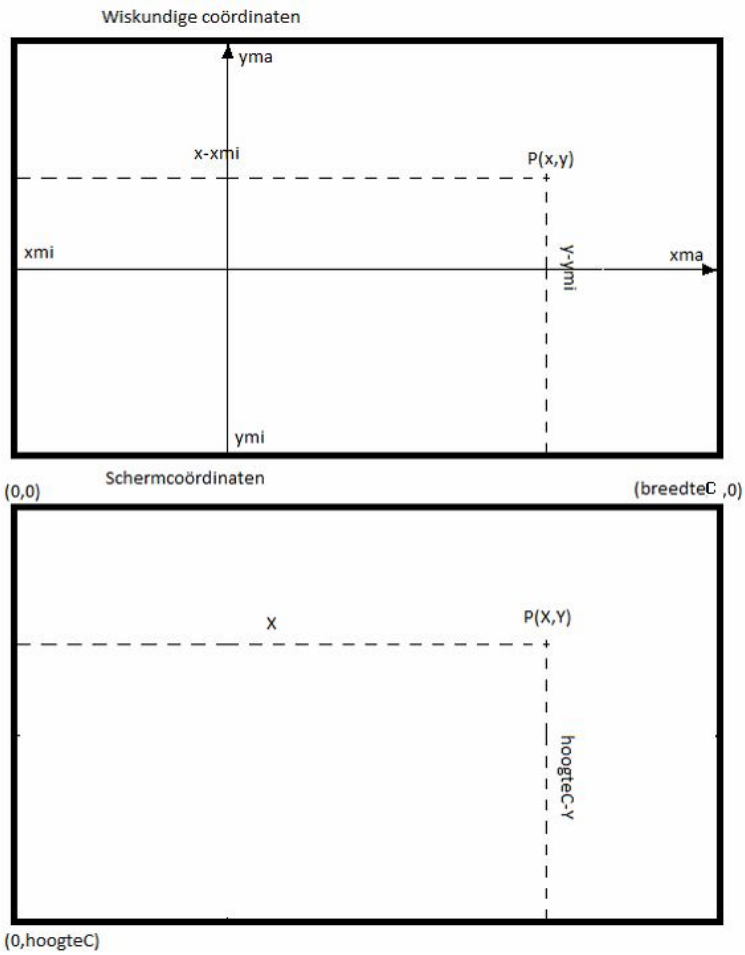
Inleidend voorbeeld: start met een formulier van 480 x 360 waarin je een rechthoekig venster definieert met breedte 480 en hoogte 300. Hierin maak je een Canvas, dat is de widget waarin je grafieken kunt tekenen.
Je maakt ook een lijst van 4 puntenkoppels (0,0), (100,200) , (480,300) en terug (0,0).
De instructie lijn = C.create_line(lijstpuntenkoppels,fill='red') zorgt er voor dat deze punten worden verbonden door lijnstukken, dat je m.a.w. een driehoek tekent.

Programma:




```
from tkinter import *
form=Tk();form.geometry('480x360');form.title('Tekening')
C = Canvas(form, bg="white",
width=480,height=300);C.pack()
lijstpuntenkoppels=[0,0,100,200,480,300,0,0]
C.create_line(lijstpuntenkoppels,fill='red')
```

Het punt (0,0) is het hoekpunt linksboven en het punt (480,300) is het punt rechtsonder.
Dit komt niet overeen met de wiskundige coördinaten. Om een mooie wiskundige grafiek te kunnen tekenen, moeten we wiskundige coördinaten omzetten naar schermcoördinaten.
Noem de breedte van het canvas 'breedteC' en de hoogte 'hoogteC'.
In ons geval is breedteC =480 en hoogteC=300



We zullen nu formules opstellen om de wiskundige coördinaten (x,y) van een punt P om te zetten naar schermcoördinaten (X,Y) van dit punt.Om de grafiek van een functie correct te tekenen, moeten we het verband bepalen tussen x en X en tussen y en Y. Uit de tekeningen kan je afleiden dat:

$$\frac{x - xmi}{xma - xmi} = \frac{X - 0}{breedteC - 0}$$ waaruit volgt: X =breedteC *(x-xmi)/(xma-xmi)

$$\frac{y - ymi}{yma - ymi} = \frac{hoogteC - Y}{hoogteC}$$ waaruit volgt: Y =hoogteC-hoogteC *(y-ymi)/(yma-ymi)

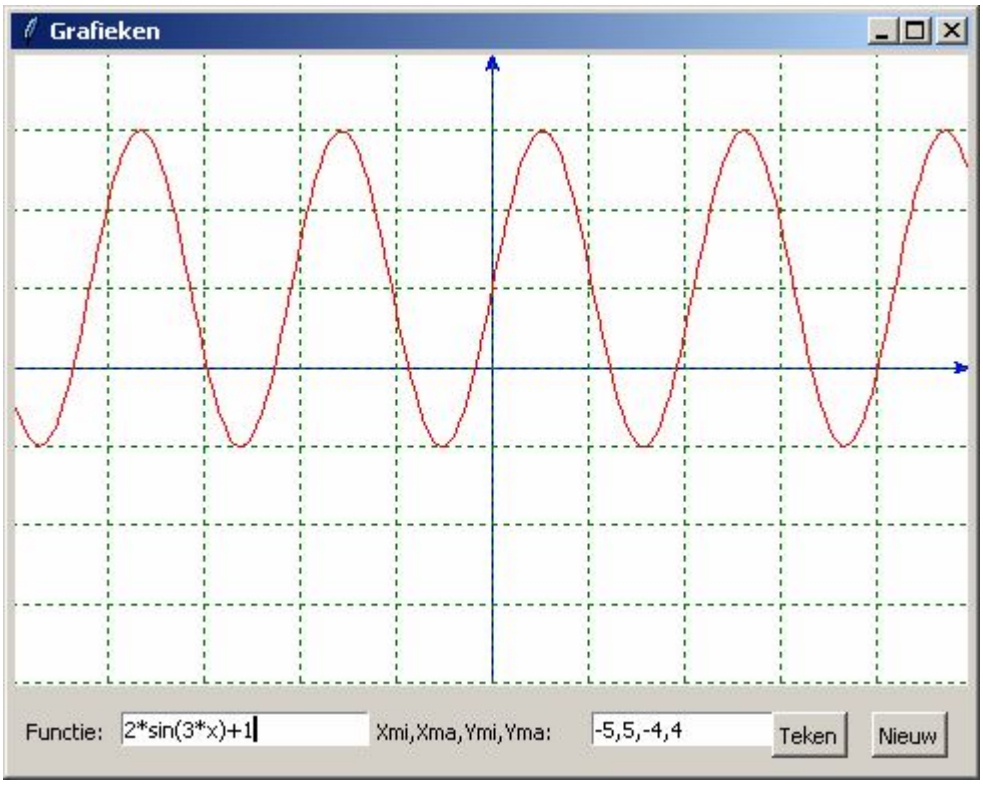
Vandaar de formules: transx(x)= breedteC*(x-xmi)/(xma-xmi)
en transy(y)= hoogteC-hoogteC*(y-ymi)/(yma-ymi)

Het volgende programma gebruikt deze omzetting om de grafiek van een willekeurig in te voeren functie te tekenen in een vrij te kiezen assenstelsel.

Programma:

```
# grafieken met tkinter
from math import *; from tkinter import *
def mult(s):
   l=s.split(',');co=[]
   for i in l:co.append(float(i))
   return co
def transx(x):return (x-xmi)*breedteC/(xma-xmi)
def transy(y):return hoogteC-hoogteC*(y-ymi)/(yma-ymi)
def f(x):return eval(E1.get())
def teken():
   global xmi,xma,ymi,yma
   xmi,xma,ymi,yma=mult(E2.get());C.delete(ALL)
#assen
   X=transx(0);Y=transy(0)
   lijn = C.create_line(X,0,X,hoogteC,fill='blue',arrow='first')
   lijn = C.create_line(0,Y,breedte,Y,fill='blue',arrow='last')
#rooster
   for x in range(int(xmi),int(xma+1)):
      X=transx(x)
      lijn = C.create_line(X,0,X,hoogteC,fill='green',dash=[1,2])
   for y in range(int(ymi),int(yma+1)):
      Y=transy(y)
      lijn = C.create_line(0,Y,breedte,Y,fill='green',dash=[1,2] )
#kromme
   lis=[];stap=0.05;x=xmi
   while x<xma:
      x=x+stap;y=f(x);X=transx(x);Y=transy(y)
      lis.append(X);lis.append(Y)
   lijn = C.create_line(lis,fill='red')
def nieuw():
    E2.delete(0,len(E2.get()));E1.delete(0,len(E1.get()))
    C.delete(ALL)
#hoofdprogramma
```

```
breedte=480;hoogte=360;breedteC=480;hoogteC=hoogte-45;hoInv=hoogte-30
form=Tk();form.geometry(str(breedte)+'x'+str(hoogte));form.title('Grafieken')
C = Canvas(form, bg="white", height=hoogteC, width=breedteC);C.pack()
Label(form,text='Functie:',).place(x=5,y=hoInv)
E1=Entry(form);E1.place(x=55,y=hoInv)
Label(form,text='Xmi,Xma,Ymi,Yma:',).place(x=180,y=hoInv)
E2=Entry(form);E2.place(x=290,y=hoInv);E2.insert(0,'-5,5,-4,4')
Button(form,text='Teken',command=teken).place(x=380,y=hoInv)
Button(form,text='Nieuw',command=nieuw).place(x=430,y=hoInv)
form.mainloop()
```



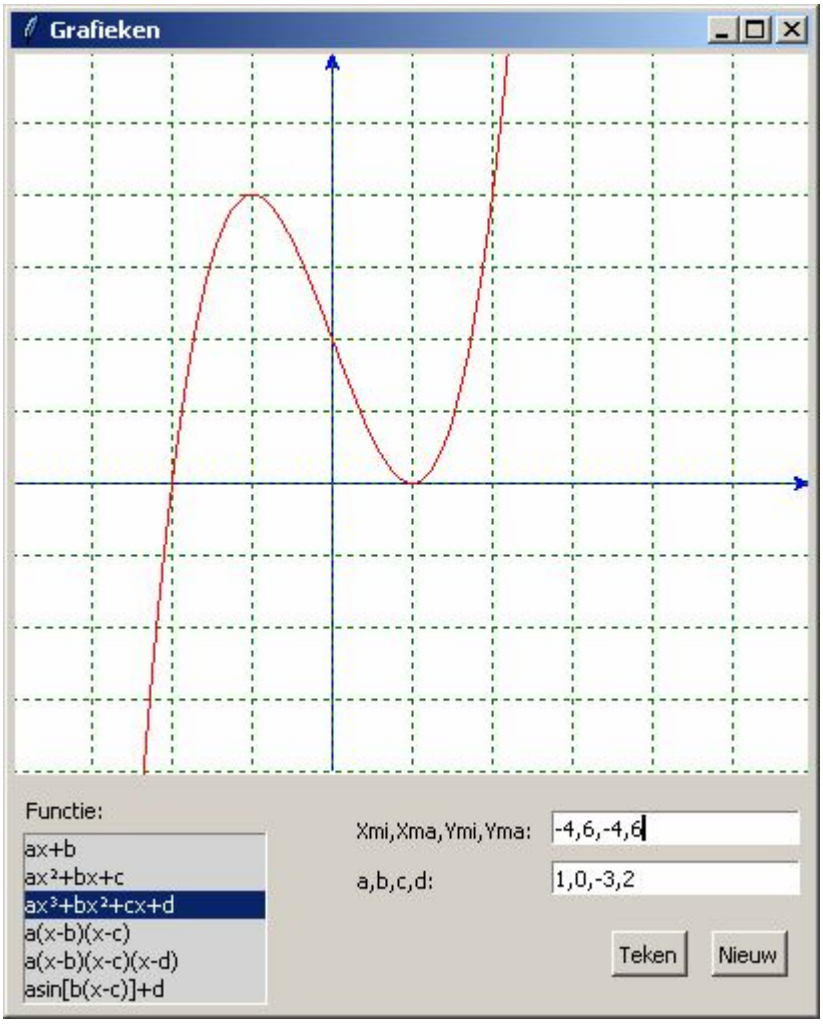
64. Grafieken van enkele elementaire functies met parameters grafiekTk_elementairefuncties

Je kan natuurlijk ook in een listbox enkele functievoorschriften tonen, in dit geval van 1ste, 2de en 3de graadsfuncties en de algemene sinusfunctie. De gebruiker 'markeert' een functie en stelt waarden in voor xmin,xmax,ymin,ymax en voor de parameters a,b,c,d.
Merk op, in lijst lis1 is de vorm van de functies zoals wij ze normaal opschrijven: het is deze vorm die in de listbox getoond wordt. Eerst moet deze lis1 omgezet worden naar een tuple tupfun. Deze tupfun wordt dan ingevuld in de listvariable (=StringVar(value=tupfun))
In lis2 is de vorm van de functies een vorm die het pythonprogramma kan evalueren
def f(x):return eval(lis2[Lb.curselection()[0]])
Voorbeeld: de string 'ax³+bx²+cx+d' in lis1 moet natuurlijk dezelfde index hebben als de string 'a*x**3+b*x**2+c*x+d' in lis2

Programma:

```
# grafieken van enkele elementaire functies met parameters (met tkinter)
```

```
from math import *; from tkinter import *
def mult(s):
   l=s.split(',');co=[]
   for i in l:co.append(float(i))
   return co
def transx(x):return (x-xmi)*breedte/(xma-xmi)
def transy(y):return hoogteC-hoogteC*(y-ymi)/(yma-ymi)
def f(x):return eval(lis2[Lb.curselection()[0]])
def teken():
   global xmi,xma,ymi,yma,a,b,c,d
   xmi,xma,ymi,yma=mult(E2.get());C.delete(ALL)
#assen
   X=transx(0);line = C.create_line(X,0,X,hoogteC,fill='blue',arrow='first')
   Y=transy(0);line = C.create_line(0,Y,breedte,Y,fill='blue',arrow='last')
# rooster
   for x in range(int(xmi),int(xma+1)):
      X=transx(x);line = C.create_line(X,0,X,hoogteC,fill='green',dash=[1,2])
   for y in range(int(ymi),int(yma+1)):
      Y=transy(y);line = C.create_line(0,Y,breedte,Y,fill='green',dash=[1,2] )
#kromme
   a,b,c,d=mult(E3.get());lis=[];stap=0.05;x=xmi
   while x<xma:
     x=x+stap;y=f(x);X=transx(x);Y=transy(y);lis.append(X);lis.append(Y)
   line = C.create_line(lis,fill='red')
def nieuw():
    E2.delete(0,len(E2.get()));E3.delete(0,len(E3.get()));C.delete(ALL)
# hoofdprogramma
breedte=410;hoogte=490;hoogteC=hoogte-130;hoInv=hoogte-120
form=Tk();form.geometry(str(breedte)+'x'+str(hoogte));form.title('Grafieken')
lis1=['ax+b','ax²+bx+c','ax³+bx²+cx+d','a(x-b)(x-c)','a(x-b)(x-c)(x-d)','asin[b(x-c)]+d'];tupfun=tuple(lis1)
lis2=['a*x+b','a*x*x+b*x+c','a*x**3+b*x**2+c*x+d','a*(x-b)*(x-c)','a*(x-b)*(x-c)*(x-d)','a*sin(b*(x-c))+d']
Lb=Listbox(form,listvariable=StringVar(value=tupfun),selectmode='Single',height=6,width=20,bg='lightgre
y')
C = Canvas(form, bg="white", height=hoogteC, width=breedte);C.pack()
L1=Label(form,text='Functie:',);L1.place(x=5,y=hoInv-10);Lb.place(x=5,y=hoInv+10)
Label(form,text='Xmi,Xma,Ymi,Yma:',).place(x=170,y=hoInv);E2=Entry(form);E2.place(x=280,y=hoInv)
Label(form,text='a,b,c,d:',).place(x=170,y=hoInv+25);E3=Entry(form);E3.place(x=280,y=hoInv+25)
E2.insert(0,'-4,4,-4,4');E3.insert(0,'1,0,0,0')
B1=Button(form,text='Teken',command=teken);B1.place(x=300,y=hoInv+60)
B2=Button(form,text='Nieuw',command=nieuw);B2.place(x=350,y=hoInv+60)
form.mainloop()
```



65. Trainers om bij de grafiek van een gegeven functie de juiste waarden van de parameters te bepalen.

De volgende 3 programma's hebben dezelfde basis:
Door enkele lijntjes te vervangen, zet je het één om naar het andere.

De lineaire functie trainer_lineaire_functie

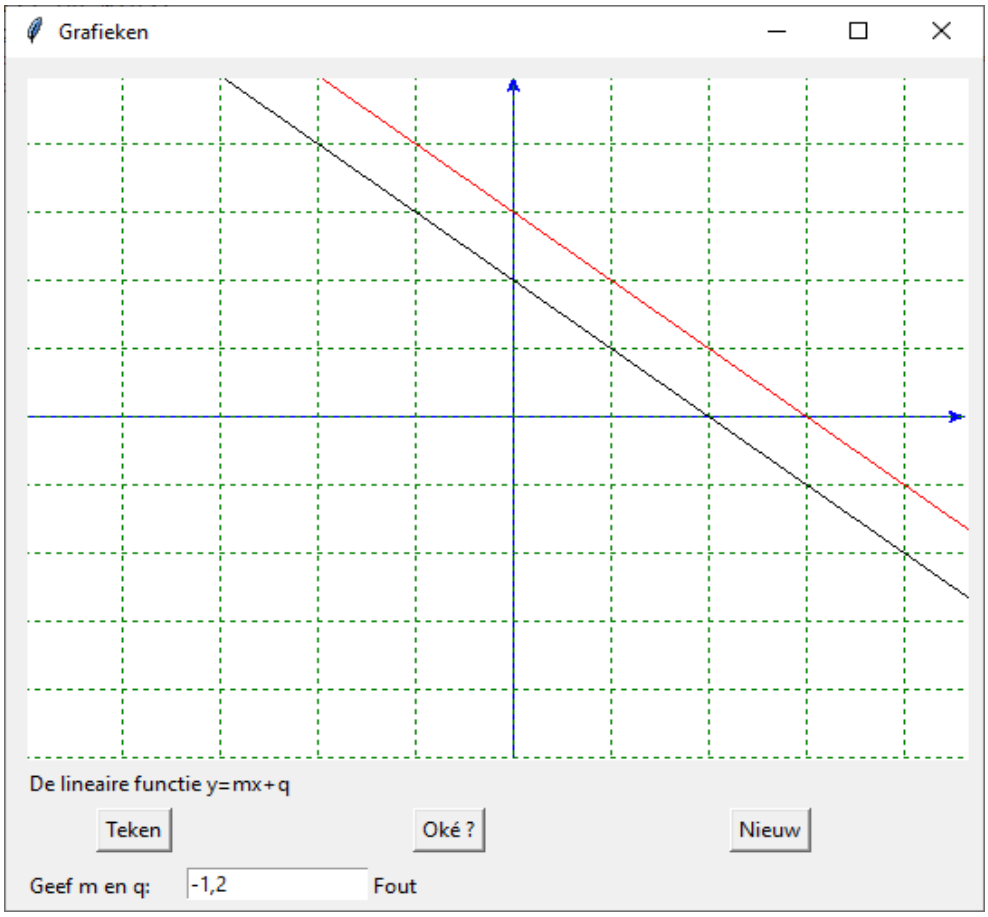
Dit programma toont, telkens we op **Teken** tikken, de grafiek van een rechte **y = mx+q** (in het rood) . De gebruiker moet waarden voor m en q invullen. De hiermee overeenstemmende functie wordt getekend in het zwart. Wordt de oorspronkelijke rechte niet overtekend, dan heeft de gebruiker foutieve waarden ingevuld en mag hij nieuwe waarden invullen tot hij de juiste waarden gevonden heeft.

Programma:

```
# trainer lineaire functie: bepaling van m en q
from random import *; from math import *; from tkinter import *
def mult(s):
        l=s.split(',')
        co=[]
        for i in l:co.append(float(i))
        return co
def transx(x):return (x-xmi)*breedte/(xma-xmi)
```

```
def transy(y):return hoogteC-hoogteC*(y-ymi)/(yma-ymi)
def lijn(m,q,kl):
	x1=transx(xmi);y1=transy(m*xmi+q)
	x2=transx(xma);y2=transy(m*xma+q)
	line = C.create_line(x1,y1,x2,y2,fill=kl)
def teken():
	global m,q,xmi,xma,ymi,yma
	wis()
	m=randint(-5,5);q=randint(-8,8)
	xmi=-5;xma=5;ymi=-5;yma=5
	if m!=0:
		snx=-q/m
		if snx<xmi:
			xmi=int(snx-2)
		if snx>xma:
			xma=int(snx+2)
	sny=q
	if sny<ymi:ymi=int(sny-2)
	if sny>yma:yma=int(sny+2)
#assen
	X=transx(0)
	line=C.create_line(X,0,X,hoogteC,fill='blue',arrow='first')
	Y=transy(0)
	line=C.create_line(0,Y,breedte-20,Y,fill='blue',arrow='last')
# rooster
	for x in range(int(xmi),int(xma+1)):
		X=transx(x)
		line=C.create_line(X,0,X,hoogteC,fill='green',dash=[1,2])
	for y in range(int(ymi),int(yma+1)):
		Y=transy(y)
		line=C.create_line(0,Y,breedte,Y,fill='green',dash=[1,2] )
#rechte
	lijn(m,q,'red')
def invoer():
	mi,qi=mult(E1.get())
	lijn(mi,qi,'black')
	if (mi==m and qi==q):
		sco='Goed'
	else:
		sco='Fout  '
	L2=Label(form,text=sco)
	L2.place(x=200,y=honv+60)
def wis():
	E1.delete(0,len(E1.get()));C.delete(ALL)
	L2=Label(form,text=(' ').ljust(40))
	L2.place(x=200,y=honv+60)
def nieuw():
	i=0;s=0
	wis()
#hoofdprogramma
breedte=540;hoogte=500;hoogteC=hoogte-100;honv=hoogte-85
form=Tk();form.geometry(str(breedte)+'x'+str(hoogte));form.title('Grafieken')
C = Canvas(form, bg="white",height=hoogteC, width=breedte-20);C.place(x=10,y=10)
L0=Label(form,text='De lineaire functie y=mx+q');L0.place(x=10,y=honv)
L1=Label(form,text='Geef m en q:',);L1.place(x=10,y=honv+60)
```

```
E1=Entry(form);E1.place(x=100,y=honv+60)
B1=Button(form,text='Teken',command=teken);B1.place(x=50,y=honv+25)
B2=Button(form,text='Nieuw',command=nieuw);B2.place(x=400,y=honv+25)
B3=Button(form,text='Oké ?',command=invoer);B3.place(x=225,y=honv+25)
form.mainloop()
```



66. Trainer om bij de grafiek van een gegeven algemene sinusfunctie y=a.sin[b(x-c)]+d de juiste waarden van a, b, c en d te bepalen trainer_algemene_sinus

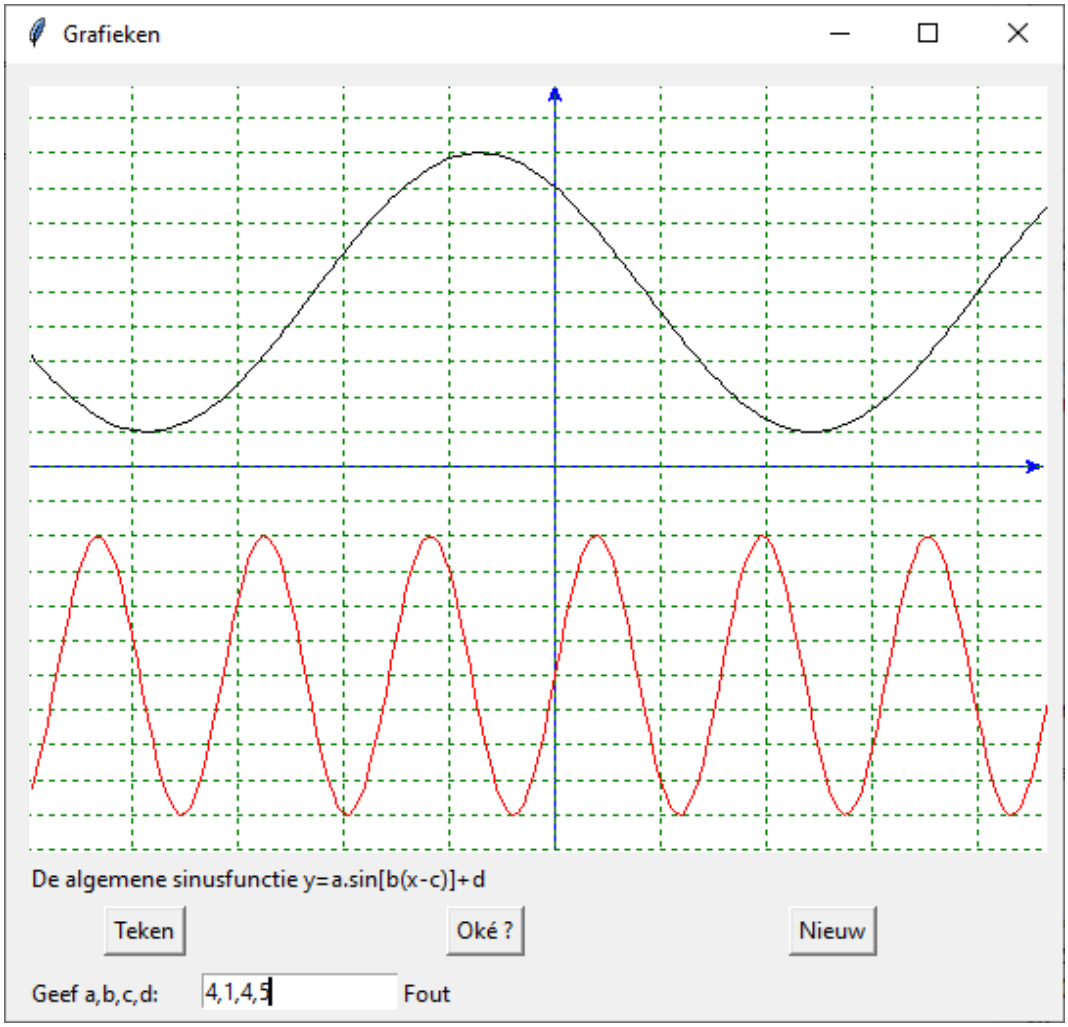
Het volgende programma bekomen we door een kleine aanpassing van het vorige: telkens we op **Teken** tikken, wordt een grafiek van de algemene sinusfunctie getekend(in het rood). De gebruiker moet waarden voor a,b,c en d invullen. De hiermee overeenstemmende sinusfunctie wordt getekend in het zwart. Ook hier mag de gebruiker getallen invullen tot hij de juiste waarden gevonden heeft.

Programma:

```
# trainer algemene sinusfunctie: bepaling van a,b,c en d
from random import *; from math import *; from tkinter import *
def mult(s):
        l=s.split(',')
        co=[]
        for i in l:co.append(float(i))
        return co
def transx(x):return (x-xmi)*breedte/(xma-xmi)
def transy(y):return hoogteC-hoogteC*(y-ymi)/(yma-ymi)
def kromme(a,b,c,d,kl):
```

```
        lis=[]
        stap=0.05;x=xmi
        while x<xma:
                x=x+stap;y=a*sin(b*(x-c))+d
                X=transx(x);Y=transy(y)
                lis.append(X);lis.append(Y)
        line = C.create_line(lis,fill=kl)
def teken():
        global a,b,c,d,xmi,xma,ymi,yma
        wis();a=0;b=0
        while a==0:a=randint(-5,5)
        while b==0:b=randint(-8,8)
        c=randint(-5,5);d=randint(-8,8)
        xmi=min(-5,c-1);xma=max(5,c+1)
        yma=max(5,abs(a+d)+1,abs(a-d)+1)
        ymi=min(-5,-abs(a+d)-1,-abs(a-d)-1)
#assen
        X=transx(0)
        line=C.create_line(X,0,X,hoogteC,fill='blue',arrow='first')
        Y=transy(0)
        line=C.create_line(0,Y,breedte-20,Y,fill='blue',arrow='last')
# rooster
        for x in range(int(xmi),int(xma+1)):
                X=transx(x)
                line=C.create_line(X,0,X,hoogteC,fill='green',dash=[1,2])
        for y in range(int(ymi),int(yma+1)):
                Y=transy(y)
                line=C.create_line(0,Y,breedte,Y,fill='green',dash=[1,2] )
#kromme
        kromme(a,b,c,d,'red')
def invoer():
        ai,bi,ci,di=mult(E1.get())
        kromme(ai,bi,ci,di,'black')
        if (((ai==a and bi==b) or (ai==-a and bi==-b)) and ci==c and di==d):
                sco='Goed'
        else:
                sco='Fout  '
        L2=Label(form,text=sco)
        L2.place(x=200,y=honv+60)
def wis():
        E1.delete(0,len(E1.get()));C.delete(ALL)
        L2=Label(form,text=(' ').ljust(40))
        L2.place(x=200,y=honv+60)
def nieuw():wis()
#hoofdprogramma
breedte=540;hoogte=500;hoogteC=hoogte-100;honv=hoogte-85
form=Tk();form.geometry(str(breedte)+'x'+str(hoogte));form.title('Grafieken')
C = Canvas(form, bg="white",height=hoogteC, width=breedte-20);C.place(x=10,y=10)
L0=Label(form,text='De algemene sinusfunctie y=a.sin[b(x-c)]+d');L0.place(x=10,y=honv)
L1=Label(form,text='Geef a,b,c,d:',);L1.place(x=10,y=honv+60)
E1=Entry(form);E1.place(x=100,y=honv+60)
B1=Button(form,text='Teken',command=teken);B1.place(x=50,y=honv+25)
B2=Button(form,text='Nieuw',command=nieuw);B2.place(x=400,y=honv+25)
B3=Button(form,text='Oké ?',command=invoer);B3.place(x=225,y=honv+25)
```

form.mainloop()



67. Trainer om bij de grafiek van de kwadratische functie $y=ax^2+bx+c$ de juiste waarden van a, b en c te bepalen. trainer_kwadratische_functie

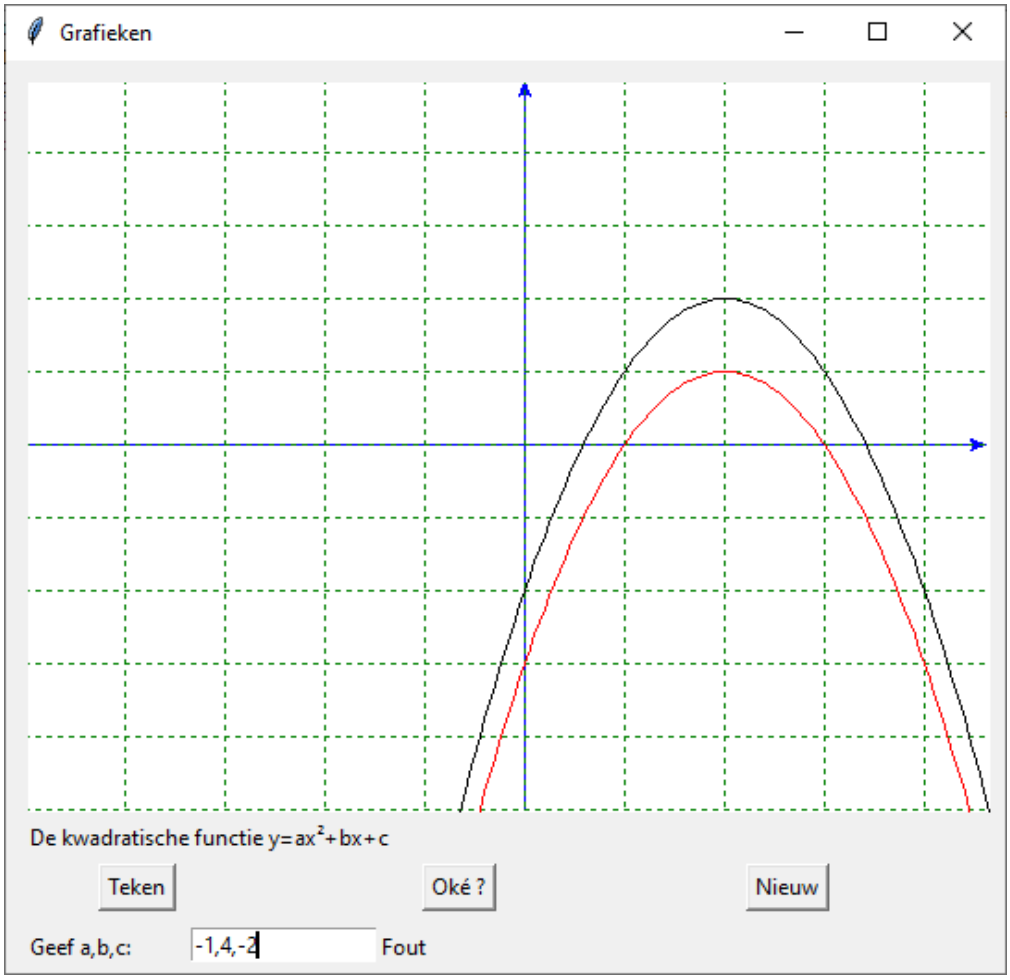
Om a, b en c te berekenen, denk aan de formules voor som en product van de wortels:
som= x1+x2= -b/a product= x1.x2= c/a

Programma:

```
# trainer kwadratische functie: bepaling van a,b en c
from random import *;from math import *;from tkinter import *
def mult(s):
        l=s.split(',')
        co=[]
        for i in l:co.append(float(i))
        return co
def transx(x):return (x-xmi)*breedte/(xma-xmi)
def transy(y):return hoogteC-hoogteC*(y-ymi)/(yma-ymi)
def kromme(a,b,c,kl):
        lis=[]
        stap=0.05;x=xmi
        while x<xma:
                x=x+stap;y=a*x*x+b*x+c
```

```
                X=transx(x);Y=transy(y)
                lis.append(X);lis.append(Y)
        line = C.create_line(lis,fill=kl)
def teken():
        global a,b,c,xmi,xma,ymi,yma
        wis()
        x1=randint(-3,3);x2=randint(-3,3);a=0
        while a==0:a=randint(-3,3)
        c=a*x1*x2;b=-a*(x1+x2)
        xmi=min(-5,x1-1,x2-1);xma=max(5,x1+1,x2+1)
        ymi=min(-5,c-1,(4*a*c-b*b)/4/a-1);yma=max(5,c+1,(4*a*c-b*b)/4/a+1)
#assen
        X=transx(0)
        line=C.create_line(X,0,X,hoogteC,fill='blue',arrow='first')
        Y=transy(0)
        line=C.create_line(0,Y,breedte-20,Y,fill='blue',arrow='last')
# rooster
        for x in range(int(xmi),int(xma+1)):
                X=transx(x)
                line=C.create_line(X,0,X,hoogteC,fill='green',dash=[1,2])
        for y in range(int(ymi),int(yma+1)):
                Y=transy(y)
                line=C.create_line(0,Y,breedte,Y,fill='green',dash=[1,2] )
#kromme
        kromme(a,b,c,'red')
def invoer():
        ai,bi,ci=mult(E1.get())
        kromme(ai,bi,ci,'black')
        if (ai==a and bi==b and ci==c):
                sco='Goed'
        else:
                sco='Fout  '
        L2=Label(form,text=sco)
        L2.place(x=200,y=honv+60)
def wis():
        E1.delete(0,len(E1.get()));C.delete(ALL)
        L2=Label(form,text=(' ').ljust(40))
        L2.place(x=200,y=honv+60)
def nieuw():
      wis()
#hoofdprogramma
breedte=540;hoogte=500;hoogteC=hoogte-100;honv=hoogte-85
form=Tk();form.geometry(str(breedte)+'x'+str(hoogte));form.title('Grafieken')
C = Canvas(form, bg="white",height=hoogteC, width=breedte-20);C.place(x=10,y=10)
L0=Label(form,text='De kwadratische functie y=ax²+bx+c');L0.place(x=10,y=honv)
L1=Label(form,text='Geef a,b,c:',);L1.place(x=10,y=honv+60)
E1=Entry(form);E1.place(x=100,y=honv+60)
B1=Button(form,text='Teken',command=teken);B1.place(x=50,y=honv+25)
B2=Button(form,text='Nieuw',command=nieuw);B2.place(x=400,y=honv+25)
B3=Button(form,text='Oké ?',command=invoer);B3.place(x=225,y=honv+25)
```

form.mainloop()



68. Grafiek van een functie, raaklijn in een punt (met tkinter) grafiek_raaklijn

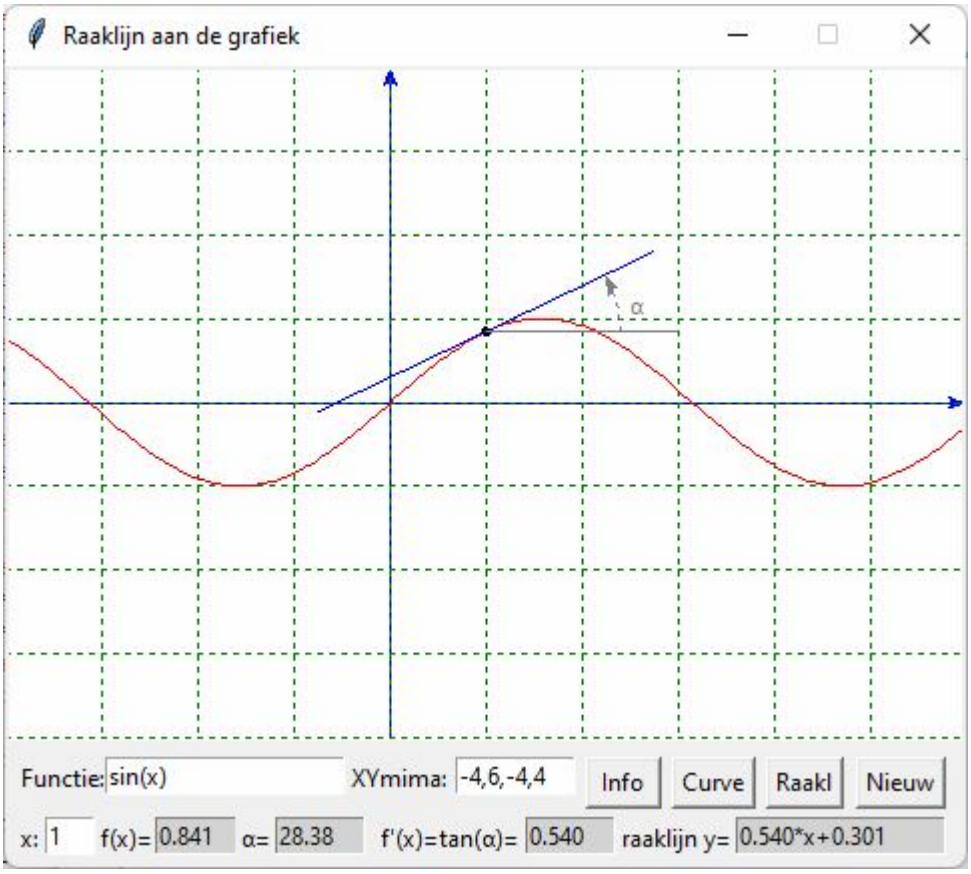
Het volgende programma is vrij uitgebreid gedocumenteerd maar ook gebaseerd op het eerste grafisch programma in tkinter. Daarom is het, denk ik vrij vlot leesbaar.
Met 'Curve' wordt de grafiek van de ingevoerde functie getekend. Je kan in een willekeurig punt x de raaklijn tekenen en de vergelijking berekenen maar ook het verband zien tussen de hoek α en f'(x)

Programma:

```
# grafiek van een functie, raaklijn in een punt (met tkinter)
from math import *;from tkinter import *;from tkinter import messagebox
# meervoudige invoer
def mult(s):
    l=s.split(','); co=[]
    for i in l:co.append(float(i))
    return co
# omzetten wiskundige coördinaten x,y => schermcoördinaten X,Y
def transx(x):return (x-xmi)*breedte/(xma-xmi)
def transy(y):return hoogteC-hoogteC*(y-ymi)/(yma-ymi)
# berekenen functie en afgeleide functie
def f(x):return eval(E1.get())
def df(x):return (f(x+dx)-f(x-dx))/dx/2
```

```
# tekenen van de grafiek
def teken():
    global xmi,xma,ymi,yma
    C.delete(ALL);xmi,xma,ymi,yma=mult(E2.get())
# assen
    X=transx(0);C.create_line(X,0,X,hoogteC,fill='blue',arrow='first')
    Y=transy(0);C.create_line(0,Y,breedte,Y,fill='blue',arrow='last')
# rooster
    for x in range(int(xmi),int(xma+1)):
        X=transx(x);C.create_line(X,0,X,hoogteC,fill='green',dash=[1,2])
    for y in range(int(ymi),int(yma+1)):
        Y=transy(y);C.create_line(0,Y,breedte,Y,fill='green',dash=[1,2] )
# kromme
    lis=[];stap=0.05;x=xmi
    while x<xma:
        x=x+stap;y=f(x);ap(lis,x,y)
    C.create_line(lis,fill='red')
# toevoegen punt aan lijst [x1,y1,x2,y2,...] van een kromme (of rechte)
def ap(li,x,y):X=transx(x);Y=transy(y);li.append(X);li.append(Y)
#tekenen van de raaklijn
def raakl():
    # opstellen van de vergelijking
    x1=float(E3.get())
    rico=df(x1);alf=degrees(atan(rico))
    q=f(x1)-x1*rico
    # afronding
    y1R=format(f(x1),'.3f');alfR=format(alf,'.2f')
    ricoR=format(rico,'.3f');qR=format(q,'.3f')
    if q>=0:qR='+'+qR
    vgl=ricoR+'*x'+qR
    #invullen hoek,richtingscoëfficiënt,vergelijking
    for E in (E4,E5,E6,E7):wisentr(E)
    E4.insert(0,y1R);E5.insert(0,alfR);E6.insert(0,ricoR);E7.insert(0,vgl)
    # tekenen raaklijn
    teken();r=0.05
    lis=[];ap(lis,x1-r,f(x1)-r);ap(lis,x1+r,f(x1)+r)
    C.create_oval(lis,fill='black') # tekenen raakpunt
    # raaklijn met lengte 2*2: x1-lgr => x1+lgr
    lgr=2*cos(atan(rico));lis=[];x=x1-lgr;y=rico*x+q;ap(lis,x,y)
    x=x1+lgr;y=rico*x+q;ap(lis,x,y)
    C.create_line(lis,fill='blue')
    # tekenen horizontale stippellijn vanuit raakpunt
    lis=[];x=x1;y1=rico*x+q;ap(lis,x,y1);x=x1+2;ap(lis,x,y1)
    C.create_line(lis,fill='grey')
    t=-1;lis=[];hk=atan(rico)/10 # boog bvb.in 10 stukjes verdelen
    # tekenen stippelboog met pijl
    while t< 10:
        t=t+1
        x=x1+0.7*2*cos(t*hk);y=y1+0.7*2*sin(t*hk);ap(lis,x,y)
    C.create_line(lis,fill='grey',dash=[1,2],arrow='last')
    # alfa toevoegen
```

```
    X=transx(x1+0.8*2*cos(hk*4));Y=transy(y1+0.8*2*sin(hk*4))
    C.create_text(X,Y,text=alfa,fill='grey')
def info():messagebox.showinfo(tit+': info',infstr)
def wisentr(E):E.delete(0,len(E.get()))
def nieuw():
    for E in (E1,E2,E3,E4,E5,E6,E7):wisentr(E)
    C.delete(ALL)
# hoofdprogramma
dx=1E-6;alfa=chr(945);col='lightgrey';tit='Raaklijn aan de grafiek'
breedte=480;hoogte=400;hoogteC=hoogte-65;hoInv=hoogte-55;hoInv2=hoogte-25
form=Tk();form.geometry(str(breedte)+'x'+str(hoogte));form.title(tit);form.resizable(False,False)
C = Canvas(form, bg="white", height=hoogteC, width=breedte);C.pack()
Label(form,text='Functie:').place(x=5,y=hoInv)
E1=Entry(form);E1.place(x=50,y=hoInv,width=120);E1.insert(0,'sin(x)')
Label(form,text='XYmima:').place(x=170,y=hoInv)
E2=Entry(form);E2.place(x=225,y=hoInv,width=60);E2.insert(0,'-4,6,-4,4')
Label(form,text='x:').place(x=5,y=hoInv2)
E3=Entry(form);E3.place(x=20,y=hoInv2,width=25);E3.insert(0,'1')
Label(form,text='f(x)=').place(x=45,y=hoInv2)
E4=Entry(form,bg=col);E4.place(x=75,y=hoInv2,width=45)
Label(form,text=alfa+'=').place(x=115,y=hoInv2)
E5=Entry(form,bg=col);E5.place(x=135,y=hoInv2,width=45)
Label(form,text="f'(x)=tan("+alfa+')=').place(x=185,y=hoInv2)
E6=Entry(form,bg=col);E6.place(x=260,y=hoInv2,width=45)
Label(form,text='raaklijn y= ').place(x=305,y=hoInv2)
E7=Entry(form,bg=col);E7.place(x=365,y=hoInv2,width=105)
B1=Button(form,text=' Info ',command=info);B1.place(x=290,y=hoInv)
B2=Button(form,text='Curve',command=teken);B2.place(x=333,y=hoInv)
B3=Button(form,text='Raakl',command=raakl);B3.place(x=380,y=hoInv)
B4=Button(form,text='Nieuw',command=nieuw);B4.place(x=425,y=hoInv)
infstr='Grafiek van een functie en raaklijn\nVul in: functie,xmin,ymin,xmax,ymax\nen een getalwaarde voor
x\n'
infstr=infstr+'Curve : de grafiek van de functie\nRaakl : vergelijking en grafiek raaklijn\n'
form.mainloop()
```



69. Goniometrische getallen goniometrie

Dit programma kan zeker gebruikt worden in alle afdelingen bij de eerste lessen goniometrie.
Je kan er de definitie van de goniometrische getallen mee bespreken, de meetkundige betekenis, de goniometrische functies, verwante hoeken, lijstje met formules die je kan aanpassen (txt-bestand).

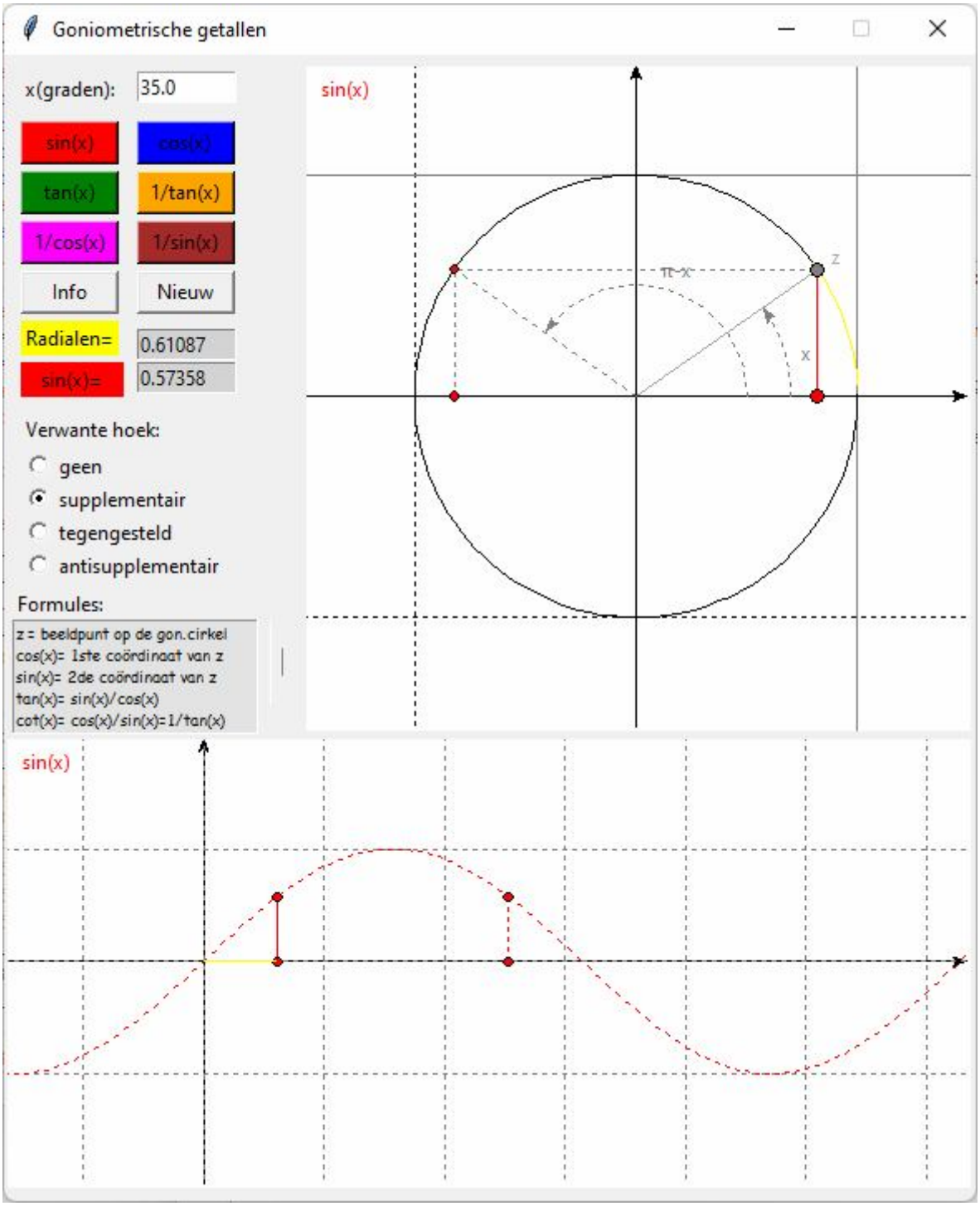
Programma:

```
# goniometrische getallen (met tkinter)
from math import *;from tkinter import *;from tkinter import messagebox
# omzetten wiskundige coördinaten x,y => schermcoördinaten X,Y voor C en C2
def transx(x):return (x-xmi)*breedteC/(xma-xmi)
def transy(y):return hoogteC-hoogteC*(y-ymi)/(yma-ymi)
def transx2(x):return (x-xmi2)*breedteC2/(xma2-xmi2)
def transy2(y):return hoogteC2-hoogteC2*(y-ymi2)/(yma2-ymi2)
# tekenen van de grafiek
def teken(i):
    global xmi,xma,ymi,yma,xmi2,xma2,ymi2,yma2
    C.delete(ALL);C2.delete(ALL);xmi,xma,ymi,yma=-1.5,1.5,-1.5,1.5
# assen
    X=transx(0);C.create_line(X,0,X,hoogteC,fill='black',arrow='first')
    Y=transy(0);C.create_line(0,Y,breedteC,Y,fill='black',arrow='last')
# rooster
    for x in range(int(xmi),int(xma+1)):
        X=transx(x);C.create_line(X,0,X,hoogteC,fill='black',dash=[1,2])
    for y in range(int(ymi),int(yma+1)):
```

```
      Y=transy(y);C.create_line(0,Y,breedteC,Y,fill='black',dash=[1,2] )
# tekenen horizontale en verticale raaklijn
   lis=[];ap(lis,-2,1);ap(lis,2,1);C.create_line(lis,fill='grey')
   lis=[];ap(lis,1,-2);ap(lis,1,2);C.create_line(lis,fill='grey')
# gonio cirkel
   lis=[];stap=0.05;tmi=0;tma=2*pi;t=tmi;r=0.03
   while t<tma:
      t=t+stap;x=cos(t);y=sin(t);ap(lis,x,y)
   C.create_line(lis,fill='black')
# invoer hoek
   alf=float(E1.get());
   while alf>=360:alf=alf-360
   while alf<=-90:alf=alf+360
   if alf==0:alf=0.001
   if alf==90:alf==90.001
   if alf==180:alf=180.001
   if alf==270:alf=270.001
   wisentr(E1);E1.insert(0,str(alf))
   xR=radians(alf);ad(E3,xR)
   lis=[];ap(lis,cos(xR),0);ap(lis,cos(xR),sin(xR));ap(lis,0,sin(xR))
   C.create_line(lis,fill='grey',dash=[1,2])
# boog
   if abs(xR)>stap:
      if xR>0:tmi=0;tma=xR
      else:tma=0;tmi=xR
      lis=[];t=tmi
      while t<tma:
         t=t+stap;x=cos(t);y=sin(t);ap(lis,x,y)
      C.create_line(lis,fill=fc[6])
   Label(form,text=funlis[i]+'=',width=8,bg=fc[i]).place(x=10,y=185)
   x=xR;ad(E2,eval(funlis[i]))
   if i==0:a,b,c,d=cos(xR),0,cos(xR),sin(xR)
   if i==1:a,b,c,d=0,sin(xR),cos(xR),sin(xR)
   if i==2:a,b,c,d=1,0,1,tan(xR)
   if i==3:a,b,c,d=0,1,1/tan(xR),1
   if i==4:a,b,c,d=0,0,1,tan(xR)
   if i==5:a,b,c,d=0,0,1/tan(xR),1
   lis=[];ap(lis,a,b);ap(lis,c,d);C.create_line(lis,fill=fc[i])
   rondje(a,b,r,i);rondje(c,d,r,i)
   lis=[]
   if i==2:ap(lis,0,0);ap(lis,1,tan(xR))
   if i==3:ap(lis,0,0);ap(lis,1/tan(xR),1)
   if i==2 or i==3:C.create_line(lis,fill='darkgrey',dash=[1,2])
   lis=[];ap(lis,0,0);ap(lis,cos(xR),sin(xR))
   C.create_line(lis,fill='darkgrey')
   rondje(cos(xR),sin(xR),r,7)
   lis=[];ap(lis,1.1*cos(xR),1.1*sin(xR));
   C.create_text(lis,text='z',fill='darkgrey')
   t=-1;lis=[];hk=radians(alf)/50 # boog bvb.in 50 stukjes verdelen
   # tekenen stippelboog met pijl
   while t< 50:
```

```
        t=t+1
        x=0.7*cos(t*hk);y=0.7*sin(t*hk);ap(lis,x,y)
    C.create_line(lis,fill='grey',dash=[1,2],arrow='last')
    # 'x' toevoegen
    X=transx(0.8*cos(hk*20));Y=transy(0.8*sin(hk*20))
    C.create_text(X,Y,text='x',fill='grey')
    C.create_text(25,15,text=funlis[i],fill=fc[i])
# 2de grafiek
    xmi2,xma2,ymi2,yma2=-1.65,6.35,-1.99,2;r=0.04
    # assen
    X=transx2(0);C2.create_line(X,0,X,hoogteC2,fill='black',arrow='first')
    Y=transy2(0);C2.create_line(0,Y,breedteC2,Y,fill='black',arrow='last')
# rooster
    for x in range(int(xmi2),int(xma2+1)):
        X=transx2(x);C2.create_line(X,0,X,hoogteC2,fill='grey',dash=[1,2])
    for y in range(int(ymi2),int(yma2+1)):
        Y=transy2(y);C2.create_line(0,Y,breedteC2,Y,fill='grey',dash=[1,2] )
# krommen
    lis=[];stap=0.05;x=xmi2-stap-0.001
    while x <xma2:
        x=x+stap;ap2(lis,x,eval(funlis[i]))
    C2.create_line(lis,fill=fc[i],dash=[1,2])
    lis=[];x=xR;ap2(lis,x,0);ap2(lis,x,eval(funlis[i]))
    C2.create_line(lis,fill=fc[i])
    rondje2(x,0,r,i)
    rondje2(x,eval(funlis[i]),r,i)
    C2.create_text(25,15,text=funlis[i],fill=fc[i])
    lis=[];ap2(lis,0,0);ap2(lis,xR,0)
    C2.create_line(lis,fill=fc[6])
# verwante hoek
    if keuze==1:x=pi-xR;hs='π-x'
    if keuze==2:x=-xR;hs='-x'
    if keuze==3:x=pi+xR;hs='π+x'
    if keuze!=0:
        lis=[];ap(lis,cos(x),0);ap(lis,cos(x),sin(x));ap(lis,0,sin(x))
        C.create_line(lis,fill='grey',dash=[1,2])
        lis=[];ap2(lis,x,0);ap2(lis,x,eval(funlis[i]))
        C2.create_line(lis,fill=fc[i],dash=[1,2])
        lis=[];ap(lis,0,0);ap(lis,cos(x),sin(x))
        C.create_line(lis,fill=fc[7],dash=[1,2])
        rondje(cos(x),0,r/2,i);rondje(cos(x),sin(x),r/2,i)
        rondje2(x,0,r,i);rondje2(x,eval(funlis[i]),r,i)
        t=-1;lis=[];hk=x/50
        # tekenen stippelboog met pijl
        while t< 50:
            t=t+1
            x=0.5*cos(t*hk);y=0.5*sin(t*hk);ap(lis,x,y)
        C.create_line(lis,fill='grey',dash=[1,2],arrow='last')
        X,Y=transx(0.6*cos(hk*25)),transy(0.6*sin(hk*25))
        C.create_text(X,Y,text=hs,fill='grey')
def rondje(a,b,rf,i):lis=[];ap(lis,a-rf,b-rf);ap(lis,a+rf,b+rf);C.create_oval(lis,fill=fc[i])
```

```
def rondje2(a,b,rf,i):lis=[];ap2(lis,a-rf,b-rf);ap2(lis,a+rf,b+rf);C2.create_oval(lis,fill=fc[i])
def ad(E,f):wisentr(E);E.insert(0,format(f,'.5f'))
def ap(li,x,y):X=transx(x);Y=transy(y);li.append(X);li.append(Y)
def ap2(li,x,y):X=transx2(x);Y=transy2(y);li.append(X);li.append(Y)
def info():
   tit='Grafieken van goniometrische functies\n'
   infstr='Enkel een hoek in graden invullen\n'
   infstr=infstr+'en een goniometrisch getal kiezen'
   h=messagebox.showinfo(tit,infstr)
def selec():global keuze;keuze=var.get()
def wisentr(E):E.delete(0,len(E.get()))
def nieuw():
   for E in (E1,E2,E3):wisentr(E)
   C.delete(ALL);C2.delete(ALL)
   Label(form,text='Gon.get.=',width=8,bg='darkgrey').place(x=10,y=185)
def but(i):Button(form,text=funlis[i],bg=fc[i],width=7,
       command=lambda:teken(i)).place(x=10+70*(i%2),y=40+30*(i//2))
# hoofdprogramma
dx=0.00001;col='lightgrey'
funlis=['sin(x)','cos(x)','tan(x)','1/tan(x)','1/cos(x)','1/sin(x)']
fc=['red','blue','green','orange','magenta','brown','yellow','grey']
breedte=585;hoogte=690;hoogteC=400;breedteC=400;hoogteC2=270;breedteC2=breedte-5
form=Tk();form.geometry(str(breedte)+'x'+str(hoogte))
form.title('Goniometrische getallen');form.resizable(False,False)
C = Canvas(form, bg="white", height=hoogteC, width=breedteC);C.place(x=180,y=5)
C2= Canvas(form, bg="white", height=hoogteC2, width=breedteC2);C2.place(x=0,y=410)
fk=open('gonioform.txt');inhoud=fk.read();fk.close()
inhoud2=inhoud.replace('pi','π')
Label(form,text='Formules:').place(x=5,y=320)
tex=Text(form,bg='grey89',height=5,width=24,wrap=WORD,font=('Comic Sans
MS','7'));tex.place(x=5,y=340)
vsb=Scrollbar(form,orient=VERTICAL,command=tex.yview,width=15)
tex['yscrollcommand']=vsb.set;vsb.place(x=160,y=340);tex.insert(INSERT,inhoud2)
Label(form,text='x(graden):').place(x=10,y=10)
Label(form,text='Radialen=',bg=fc[6]).place(x=10,y=160)
Label(form,text='Gon.get.=',width=8,bg='darkgrey').place(x=10,y=185)
E1=Entry(form);E1.place(x=80,y=10,width=60);E1.insert(0,'35')
E2=Entry(form,bg=col);E2.place(x=80,y=185,width=60)
E3=Entry(form,bg=col);E3.place(x=80,y=165,width=60)
keuzelijst=['geen','supplementair','tegengesteld','antisupplementair'];var=IntVar();keuze=0
Label(form,text='Verwante hoek:').place(x=10,y=215)
for i in range(0,4):Radiobutton(form,text=keuzelijst[i],variable=var,
       value=i,command=selec).place(x=10,y=235+i*20)
but(0);but(1);but(2);but(3);but(4);but(5)
Button(form,text=' Info ',command=info,width=7).place(x=10,y=130)
Button(form,text='Nieuw',command=nieuw,width=7).place(x=80,y=130)
form.mainloop()
```



70. Trainer afgeleide functies trainer_afgeleide_functies

Met dit programma kunnen leerlingen oefenen in het bepalen van afgeleide functies. De leerkracht maakt met kladblok een tekstbestandje, vb. oef_afg2.txt

```
sin(x)
cos(x)
(x-2)**3/9
3*sin(x)-2*cos(x)
sin(x)**2
```

Het bestand kan ingeladen worden en als de leerling op f(x) tikt, krijgt hij de functies 1 voor 1 te zien:

vb. sin(x)+ de grafiek. Als hij tikt op 'Raaklijn' worden in elk geheel getal van de x-as de raaklijnen met lengte 1 getekend. De overstaande zijde (in -magenta-kleur) is de rico, dus de afgeleide.
Deze lengtes worden overgebracht naar de figuur rechts. De leerling moet bij f'(x) een functie invullen die door de toppen van deze lijnstukken gaat: de afgeleide functie. Hij drukt dan op de knop f'(x)?. Is hij verkeerd,dan krijgt hij een 'fout'-melding en wordt de grafiek in een rode stippellijn gezet. Bij een correct antwoord een 'goed'-melding, een verhoging van de score en een groene grafiek door de 'magenta'-punten. Hij tikt dan terug op f(x) voor de volgende functie, enz...

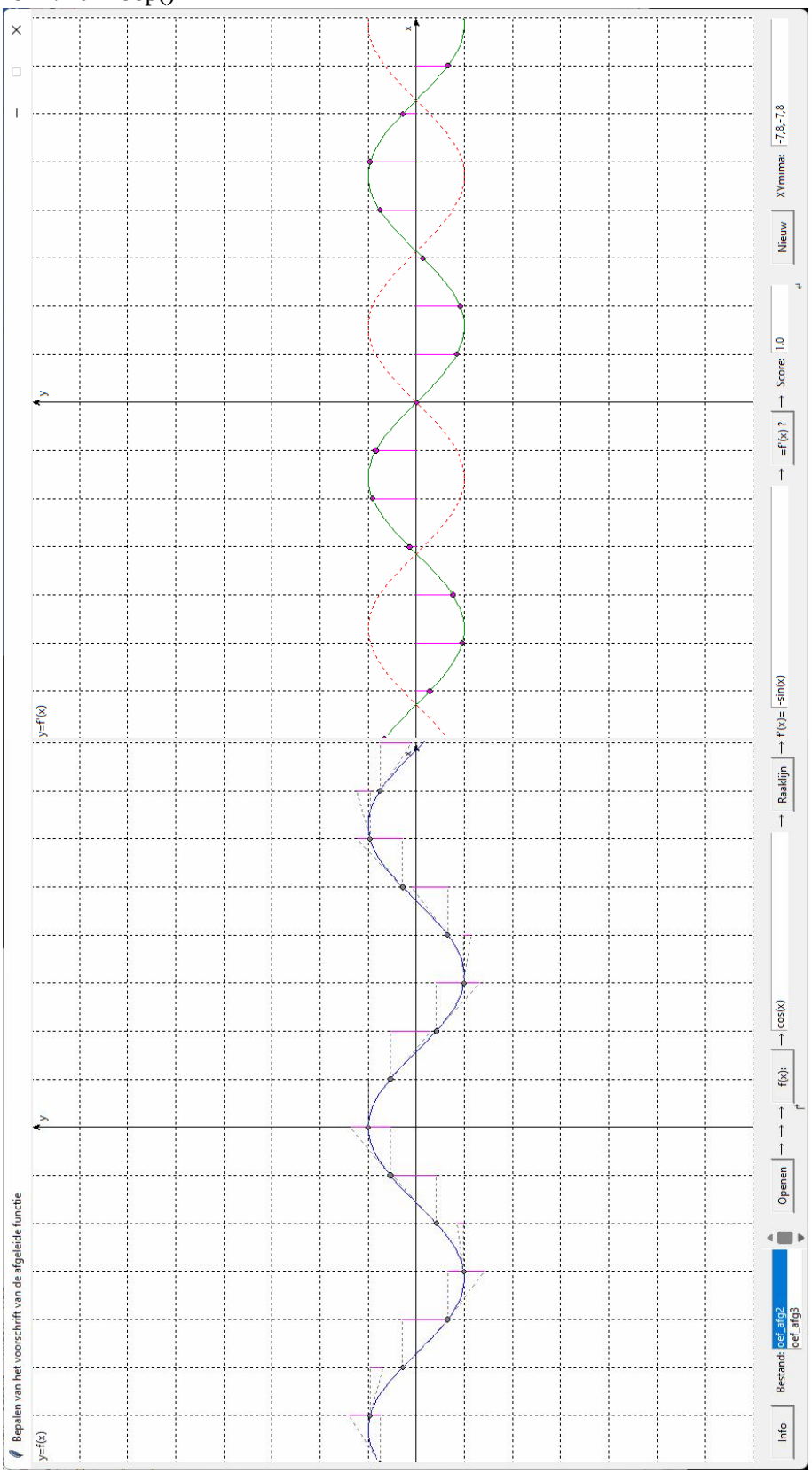
Programma:

```
# trainer afgeleide functie (met tkinter)
from math import *;from tkinter import *;from tkinter import messagebox;import os
# meervoudige invoer
def mult(s):
    l=s.split(','); co=[]
    for i in l:co.append(float(i))
    return co
# omzetten wiskundige coördinaten x,y => schermcoördinaten X,Y
def transx(x):return (x-xmi)*breedteC/(xma-xmi)
def transy(y):return hoogteC-hoogteC*(y-ymi)/(yma-ymi)
# berekenen functie en afgeleide functie
def f(x):return eval(E1.get())
def df(x):return (f(x+dx)-f(x-dx))/dx/2
def g(x):return eval(E3.get())
# tekenen van de grafiek
def init(Cv):
    global kl;Cv.delete(ALL);kl='black'
# assen
    X=transx(0);Cv.create_line(X,0,X,hoogteC,fill=kl,arrow='first')
    Y=transy(0);Cv.create_line(0,Y,breedteC,Y,fill=kl,arrow='last')
# rooster
    for x in range(int(xmi),int(xma+1)):
        X=transx(x);Cv.create_line(X,0,X,hoogteC,fill=kl,dash=[1,2])
    for y in range(int(ymi),int(yma+1)):
        Y=transy(y);Cv.create_line(0,Y,breedteC,Y,fill=kl,dash=[1,2] )
# x,y
    d=0.2;lis=[];ap(lis,xma-d,d);Cv.create_text(lis,text='x',fill=kl)
    lis=[];ap(lis,d,yma-d);Cv.create_text(lis,text='y',fill=kl)
def teken():
    global xmi,xma,ymi,yma,i,vraag;xmi,xma,ymi,yma=mult(E2.get())
    for Cv in (C,C2):init(Cv)
# kromme
    C.create_text(20,10,text='y=f(x)',fill=kl)
    C2.create_text(20,10,text="y=f'(x)",fill=kl)
    aantfun=len(func)
    if i<aantfun-1:
        vraag=0;i=i+1;wis(E1);E1.insert(0,func[i]);wis(E3)
        lis=[];stap=0.1;x=xmi
        while x<xma:
            x=x+stap;y=f(x);ap(lis,x,y)
        line = C.create_line(lis,fill='blue')
```

```
# toevoegen punt aan lijst [x1,y1,x2,y2,...] van een kromme (of rechte)
def ap(li,x,y):X=transx(x);Y=transy(y);li.append(X);li.append(Y)
# tekenen van de raaklijn
def raakl():
    global lisafg,adsc;adsc=2
    # opstellen van de vergelijking
    x1=xmi+1E-6;r=0.05
    while x1<xma:
        rico=df(x1);alf=degrees(atan(rico))
        q=f(x1)-x1*rico
    # raakpunt
        lis=[];ap(lis,x1-r,f(x1)-r);ap(lis,x1+r,f(x1)+r)
        C.create_oval(lis,fill='grey') # tekenen raakpunt
    # raaklijn: x1 => x1+1
        lgr=cos(atan(rico));lis=[];x=x1;y=rico*x+q;ap(lis,x,y)
        x=x1+1;y=rico*x+q;ap(lis,x,y)
        line = C.create_line(lis,fill='grey',dash=[1,2])
    # horizontaal
        lis=[];x=x1;y1=rico*x+q;ap(lis,x,y1);x=x1+1;ap(lis,x,y1)
        C.create_line(lis,fill='grey',dash=[1,2])
    # verticaal  (afgeleide, ook in C2 )
        lisafg=[];kl='magenta'
        lis=[];ap(lis,x,y1);y1=rico*x+q;ap(lis,x,y1)
        C.create_line(lis,fill=kl)
        lis=[];ap(lis,x1,0);ap(lis,x1,rico)
        C2.create_line(lis,fill=kl)
        lis=[];ap(lis,x1-r,rico-r);ap(lis,x1+r,rico+r)
        C2.create_oval(lis,fill=kl)
        ap(lisafg,x1,rico)
        x1=x1+1
        messagebox.showinfo("f'(x)","Vul nu een voorschrift in\nbij f'(x)= en tik op '=f'(x) ?'")
def test():
    global adsc,score,vraag;x1=xmi+1E-6;dif=0
    if vraag==0:
        while x1<xma:
            a=df(x1);b=g(x1);d=abs(a-b)
            if d<1000:dif=dif+d
            x1=x1+1
        if dif>0.01:messagebox.showinfo("f'(x)=",'fout');ds=[1,4]   ;kl='red';adsc=adsc/2
        else:messagebox.showinfo("f'(x)=",'goed');ds=[];kl='green';vraag=1
        if adsc<0.5:adsc=0
        if dif<0.01:score=score+adsc
        wis(E4);E4.insert(0,score)
        lis=[];stap=0.1;x=xmi+1E-6
        while x<xma:
            x=x+stap;y=g(x);ap(lis,x,y)
        C2.create_line(lis,fill=kl,dash=ds)
    else:messagebox.showinfo('!!!','Je mag op een vraag slechts\n1keer juist antwoorden')
def openen():
    global func,i
    nmb=Lb.get(Lb.curselection())+'.txt'
```

```
    fk=open(nmb);inhoud=fk.read();fk.close()
    func=inhoud.split('\n');i=-1
    return
def help():messagebox.showinfo(tit+': info',helstr)
def wis(E):E.delete(0,len(E.get()))
def nieuw():
    global i,score;i=-1;score=0
    for E in (E1,E3,E4):wis(E)
    for Cv in (C,C2):init(Cv)
# hoofdprogramma
tit='Bepalen van het voorschrift van de afgeleide functie'
dx=1E-6;alfa=chr(945);col='lightgrey';score=0;adsc=2;vraag=0
breedte=1600;hoogte=850;hoogteC=hoogte-60;hoInv=hoogte-40;breedteC=breedte/2-10
form=Tk();form.geometry(str(breedte)+'x'+str(hoogte));form.title(tit);form.resizable(False,False)
C = Canvas(form, bg="white", height=hoogteC, width=breedteC);C.place(x=5,y=0)
C2=Canvas(form, bg="white", height=hoogteC, width=breedteC);C2.place(x=800,y=0)
cwd=os.getcwd();ld=os.listdir();mapcsv=str(cwd);lijstsrt=[]
for fun in ld:
    fs=fun.split('.')
    if len(fs)>1 and fs[1]=='txt':lijstsrt.append(fs[0])
Lb=Listbox(form,listvariable=StringVar(value=tuple(lijstsrt)),selectmode='SINGLE',height=2,width=20,bg='
white')
Lb.place(x=130,y=hoInv)
vsb=Scrollbar(form,orient=VERTICAL,command=Lb.yview,width=25);Lb['yscrollcommand']=vsb.set;vsb.p
lace(x=240,y=hoInv-10)
fk=open('oef_afg.txt');inhoud=fk.read();fk.close()
func=inhoud.split('\n');i,j=-1,-1;aantfun=len(func)
B0=Button(form,text=' Info ',command=help);B0.place(x=10,y=hoInv,width=60)
Label(form,text='Bestand:').place(x=80,y=hoInv)
B2=Button(form,text='Openen',command=openen);B2.place(x=280,y=hoInv,width=60)
Label(form,text='→ '*48).place(x=340,y=hoInv)
Label(form,text='↱').place(x=390,y=hoInv+20);Label(form,text='↵').place(x=1290,y=hoInv+20)
B1=Button(form,text='f(x):',command=teken);B1.place(x=400,y=hoInv,width=60)
E1=Entry(form);E1.place(x=480,y=hoInv,width=218)
B3=Button(form,text='Raaklijn',command=raakl);B3.place(x=720,y=hoInv,width=60)
Label(form,text="f'(x)=").place(x=800,y=hoInv)
B5=Button(form,text="=f'(x) ?",command=test);B5.place(x=1100,y=hoInv,width=60)
E3=Entry(form);E3.place(x=835,y=hoInv,width=243)
Label(form,text='Score:').place(x=1183,y=hoInv)
E4=Entry(form);E4.place(x=1223,y=hoInv,width=75)
B4=Button(form,text='Nieuw',command=nieuw);B4.place(x=1320,y=hoInv,width=60)
Label(form,text='XYmima:').place(x=1390,y=hoInv)
E2=Entry(form);E2.place(x=1450,y=hoInv,width=140);E2.insert(0,'-7,8,-7,8')
helstr="Eerst een bestand 'oef_afg..' markeren en openen\n"
helstr=helstr+"Als je 'f(x)' tikt, krijg je een voorschrift van f(x)\n"
helstr=helstr+'en bovendien links de grafiek van y=f(x)\n'
helstr=helstr+"Met 'Raaklijn' krijg je alle raaklijnen van xmin tot xmax\n"
helstr=helstr+"De verticale lijnstukken van de driehoekjes zijn de rico's\n"
helstr=helstr+'Deze worden overgebracht naar de grafiek rechts\n'
helstr=helstr+'Probeer een functie te vinden die door de punten rechts gaat\n'
helstr=helstr+"Dit is de afgeleide functie: invullen en tekenen met f'(x)\n"
```

```
helstr=helstr+"Als je het goed hebt, komt de info: 'goed'\n"
helstr=helstr+"Zoniet 'fout'. Nadien wordt de grafiek getekend\n"
helstr=helstr+'Was je fout, dan krijg je de kans om te verbeteren\n'
helstr=helstr+'SUCCES!'
form.mainloop()
```



71. Middelwaardestelling van Lagrange (afgeleiden) middelwaardestelling_afgeleiden

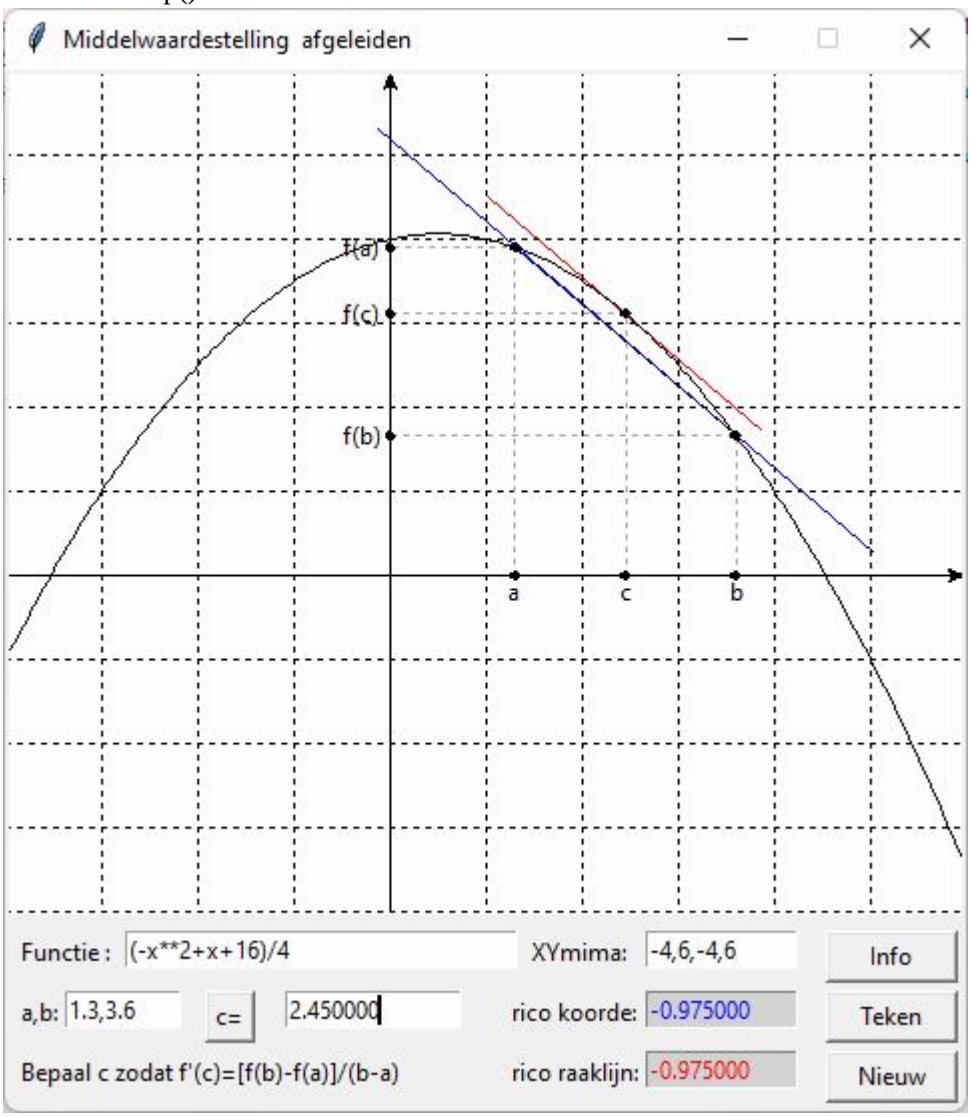
Als f(x) een continue functie is op [a,b] , dan is de rico van de koorde door [a,f(a)] en [b,f(b)] = $\frac{f(b)-f(a)}{b-a}$. Dan $\exists\ c \in\ ]a , b[$ zodat $f'(c) = \frac{f(b)-f(a)}{b-a}$, m.a.w. zodat de raaklijn in [c,f(c)] evenwijdig is met de koorde. In het programma kies je een willekeurige functie en 2 getallen a<b.
Met 'Teken' toont hij de grafiek van de functie, de koorde en de raaklijn. Probeer eerst zelf om een c in te vullen waarvoor de raaklijn // is met de koorde. Nadien bepaal je met de knop 'c=' de exacte waarde.
Als ricoK= $\frac{f(b)-f(a)}{b-a}$, dan wordt deze c-waarde in het programma berekend door met de methode van Newton een nulpunt te berekenen van de functie f '(x)-ricoK

Programma:

```
# middelwaardestelling afgeleiden
from math import *;from tkinter import *;from tkinter import messagebox
def mult(s):
    l=s.split(',');co=[]
    for i in l:co.append(float(i))
    return co
def transx(x):return (x-xmi)*breedte/(xma-xmi)
def transy(y):return hoogteC-hoogteC*(y-ymi)/(yma-ymi)
def f(x):return eval(E1.get())
def df(x):return (f(x+dx)-f(x-dx))/dx/2
def d2f(x):return(f(x+dx)+f(x-dx)-2*f(x))/dx**2
def bereken():
    global xmi,xma,ymi,yma,a,b,c
    xmi,xma,ymi,yma=mult(E2.get())
    a,b=mult(E3.get());c=float(E4.get())
    if a>=b:C.delete(ALL);messagebox.showinfo('Fout','Herstel: a<b');return
    C.delete(ALL);raakl();koord();kromme()
def raakl():
    ricoR=df(c);wis(E6);E6.insert(0,format(ricoR,nwk))
    q=f(c)-c*ricoR
    lgr=2*cos(atan(ricoR));lis=[];x=c-lgr;y=ricoR*x+q;ap(lis,x,y)
    x=c+lgr;y=ricoR*x+q;ap(lis,x,y)
    C.create_line(lis,fill='red')
    lis=[];ap(lis,c,0);ap(lis,c,f(c));C.create_line(lis,dash=[1,2])
def koord():
    ricoK=(f(b)-f(a))/(b-a);wis(E5);E5.insert(0,format(ricoK,nwk))
    q=f(a)-a*ricoK
    lgr=2*cos(atan(ricoK));lis=[];x=a-lgr;y=ricoK*x+q;ap(lis,x,y)
    x=b+lgr;y=ricoK*x+q;ap(lis,x,y)
    C.create_line(lis,fill='blue')
    lis=[];ap(lis,a,f(a));ap(lis,b,f(b));C.create_line(lis,fill='darkblue')
def newton():
    x=(a+b)/2;ricoK=(f(b)-f(a))/(b-a)
    xo=x+1
    while abs(x-xo)>eps:xo=x;x=x-(df(x)-ricoK)/d2f(x)
    wis(E4);E4.insert(0,format(x,nwk))
def kromme():
```

```
#assen
   X=transx(0);C.create_line(X,0,X,hoogteC,fill='black',arrow='first')
   Y=transy(0);C.create_line(0,Y,breedte,Y,fill='black',arrow='last')
# rooster
   for x in range(int(xmi),int(xma+1)):
      X=transx(x);C.create_line(X,0,X,hoogteC,fill='black',dash=[1,2])
   for y in range(int(ymi),int(yma+1)):
      Y=transy(y);C.create_line(0,Y,breedte,Y,fill='black',dash=[1,2] )
#kromme
   lis=[];stap=0.05;x=xmi
   while x<xma:
      x=x+stap;y=f(x);X=transx(x);Y=transy(y)
      lis.append(X);lis.append(Y)
   C.create_line(lis,fill='black')
#stippels
   for p in (a,b,c):
      lis=[];ap(lis,p,0);ap(lis,p,f(p));ap(lis,0,f(p));C.create_line(lis,dash=[1,2],fill='darkgrey')
#punten
   for p in (a,b,c):rondje(p,f(p));rondje(p,0);rondje(0,f(p))
#tekst
   for ps in('a','b','c'):naamX(eval(ps),ps);psy='f('+ps+')';naamY(eval(psy),psy)
def naamX(p,pn):lis=[];ap(lis,p,-0.2);C.create_text(lis,text=pn)
def naamY(p,pn):lis=[];ap(lis,-0.3,p);C.create_text(lis,text=pn)
def rondje(a,b):lis=[];r=0.05;ap(lis,a-r,b-r);ap(lis,a+r,b+r);C.create_oval(lis,fill='black')
def info():h=messagebox.showinfo(tit,infstr)
def ap(li,x,y):X=transx(x);Y=transy(y);li.append(X);li.append(Y)
def wis(E):E.delete(0,len(E.get()))
def nieuw():
   for E in (E1,E2,E3,E4,E5,E6):wis(E)
   C.delete(ALL)
#Hoofdprogramma
tit='Middelwaardestelling  afgeleiden';gr='lightgrey';eps=1E-7;dx=1E-6;nwk='.6f'
breedte=480;hoogte=520;hoogteC=hoogte-100;hoInv1=hoogte-90;hoInv2=hoogte-60;hoInv3=hoogte-30
form=Tk();form.geometry(str(breedte)+'x'+str(hoogte));form.title(tit);form.resizable(False,False)
C = Canvas(form, bg="white", height=hoogteC, width=breedte);C.pack()
Label(form,text='Functie :').place(x=5,y=hoInv1)
E1=Entry(form,width=32);E1.place(x=60,y=hoInv1);E1.insert(0,'(-x**2+x+16)/4')
Label(form,text='XYmima:').place(x=260,y=hoInv1)
E2=Entry(form,width=12);E2.place(x=320,y=hoInv1);E2.insert(0,'-4,6,-4,6')
Label(form,text='a,b:').place(x=0,y=hoInv2)
E3=Entry(form,width=9);E3.place(x=30,y=hoInv2);E3.insert(0,'1.3,3.6')
Button(form,text='c=',command=newton).place(x=100,y=hoInv2)
E4=Entry(form,width=14);E4.place(x=140,y=hoInv2);E4.insert(0,'3.2')
Label(form,text='rico raaklijn:').place(x=250,y=hoInv3)
E6=Entry(form,bg=gr,fg='red',width=12);E6.place(x=320,y=hoInv3)
Label(form,text='rico koorde:').place(x=250,y=hoInv2)
Label(form,text="Bepaal c zodat f'(c) = [f(b)-f(a)]/(b-a)").place(x=5,y=hoInv3)
E5=Entry(form,width=12,bg=gr,fg='blue');E5.place(x=320,y=hoInv2)
Button(form,text='Teken',command=bereken,width=8).place(x=410,y=hoInv2)
Button(form,text='Info',command=info,width=8).place(x=410,y=hoInv1)
Button(form,text='Nieuw',command=nieuw,width=8).place(x=410,y=hoInv3)
```

```
infstr='[f(b)-f(a)]/(b-a) is rico koorde\n'
infstr=infstr+'(rechte tussen (a,f(a)) en (b,f(b))\n'
infstr=infstr+'Probeer dan om zelf een c\n'
infstr=infstr+'in te vullen waarvoor geldt\n'
infstr=infstr+"f'(c)=[f(b)-f(a)]/(b-a). Zodat rico.\n"
infstr=infstr+'van de raaklijn in (c,f(c))=rico koorde.\n'
infstr=infstr+"Met de knop 'c=' kan je uiteindelijk\n"
infstr=infstr+"de exacte waarde van c berekenen."
form.mainloop()
```



72. Meetkundige betekenis van 1ste en 2de afgeleide meetk_betek_f_df_d2f

Voor elke functie en voor elke x kan je inzien wat f(x) , f '(x) , f ''(x) meetkundig betekent.
Uit het teken van deze functiewaarden, kan je afleiden: de ligging van de kromme t.o.v. de x-as , stijgen of dalen, holle kant ↑ of holle kant ↓

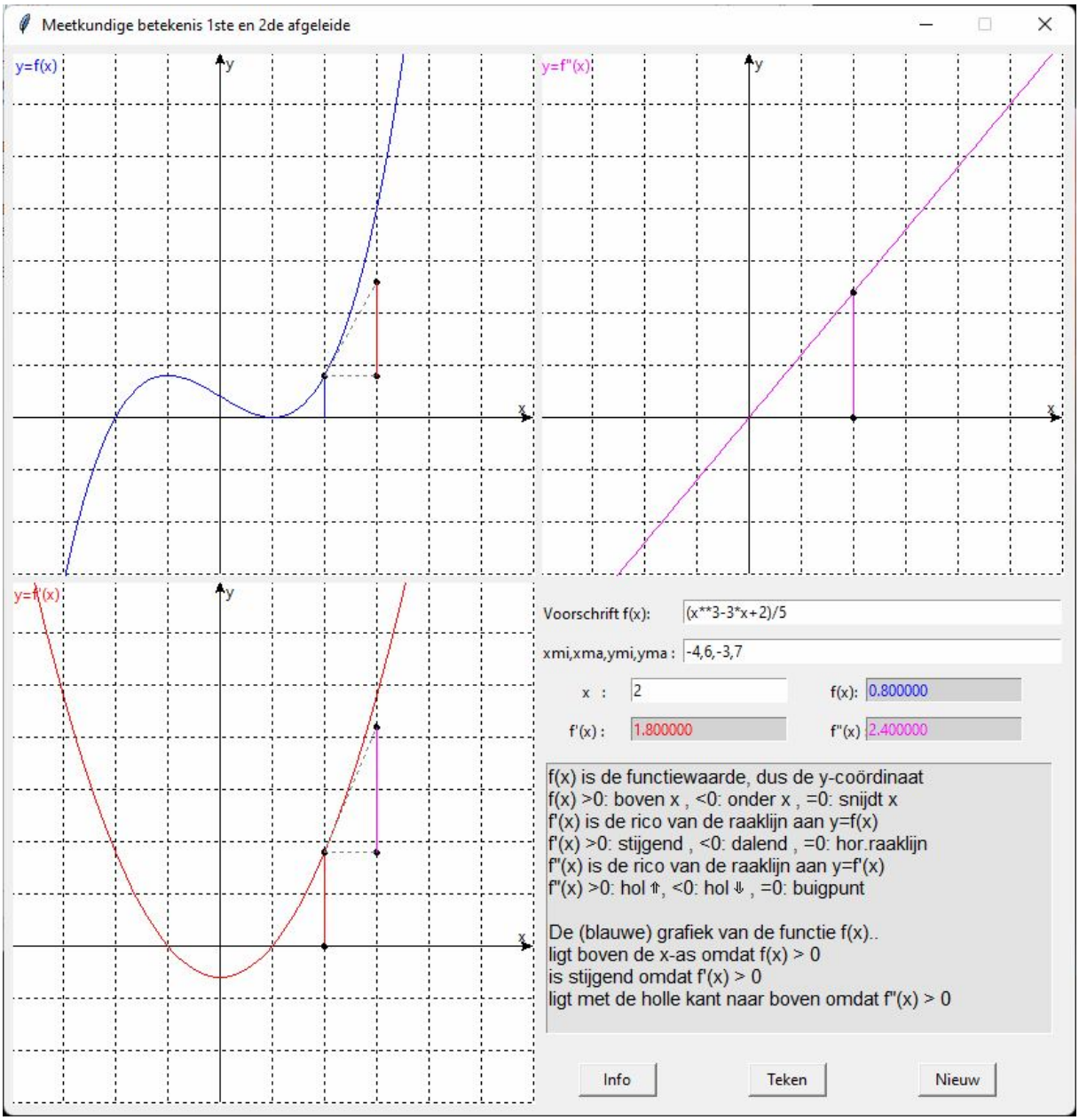
Programma:

```
# grafiek van een functie, 1ste en 2de afgeleide (met tkinter)
from math import *;from tkinter import *;from tkinter import messagebox
# meervoudige invoer
```

```
def mult(s):
   l=s.split(','); co=[]
   for i in l:co.append(float(i))
   return co
# omzetten wiskundige coördinaten x,y => schermcoördinaten X,Y
def transx(x):return (x-xmi)*breedteC/(xma-xmi)
def transy(y):return hoogteC-hoogteC*(y-ymi)/(yma-ymi)
# berekenen functie en afgeleide functie
def f(x):return eval(E1.get())
def df(x):return (f(x+dx)-f(x-dx))/dx/2
def d2f(x):return (f(x+d2x)+f(x-d2x)-2*f(x))/d2x**2
def init(Cv):
   global kl;Cv.delete(ALL);kl='black'
# assen
   X=transx(0);Cv.create_line(X,0,X,hoogteC,fill=kl,arrow='first')
   Y=transy(0);Cv.create_line(0,Y,breedteC,Y,fill=kl,arrow='last')
# rooster
   for x in range(int(xmi),int(xma+1)):
      X=transx(x);Cv.create_line(X,0,X,hoogteC,fill=kl,dash=[1,2])
   for y in range(int(ymi),int(yma+1)):
      Y=transy(y);Cv.create_line(0,Y,breedteC,Y,fill=kl,dash=[1,2] )
# x,y
   d=0.2;lis=[];ap(lis,xma-d,d);Cv.create_text(lis,text='x',fill=kl)
   lis=[];ap(lis,d,yma-d);Cv.create_text(lis,text='y',fill=kl)
# toevoegen punt aan lijst [x1,y1,x2,y2,...] van een kromme (of rechte)

def ap(li,x,y):X=transx(x);Y=transy(y);li.append(X);li.append(Y)
def teken(C,lis,tek,col):C.create_line(lis,fill=col);C.create_text(20,10,text=tek,fill=col)
def bereken():
   #krommen
   global xmi,xma,ymi,yma,i,vraag;xmi,xma,ymi,yma=mult(E2.get())
   tex.delete('1.0',END)
   ad('f(x) is de functiewaarde, dus de y-coördinaat')
   ad('f(x) >0: boven x , <0: onder x , =0: snijdt x')
   ad("f'(x) is de rico van de raaklijn aan y=f(x)")
   ad("f'(x) >0: stijgend , <0: dalend , =0: hor.raaklijn")
   ad('f''(x) is de rico van de raaklijn aan '+"y=f'(x)")
   ad('f''(x) >0: hol ⇑, <0: hol ⇓ , =0: buigpunt\n')
   ad('De (blauwe) grafiek van de functie f(x)..')
   for Cv in (C1,C2,C3):init(Cv)
   lis1,lis2,lis3=[],[],[];stap=0.1;x=xmi
   while x<xma:
      x=x+stap;ap(lis1,x,f(x));ap(lis2,x,df(x));ap(lis3,x,d2f(x))
   teken(C1,lis1,'y=f(x)','blue');teken(C2,lis2,"y=f'(x)",'red');teken(C3,lis3,'y=f''(x)','magenta')
   #overeenkomst
   x=float(E3.get())
   wis(E4);E4.insert(0,format(f(x),'.6f'))
   wis(E5);E5.insert(0,format(df(x),'.6f'))
   wis(E6);E6.insert(0,format(d2f(x),'.6f'))
   lis=[];ap(lis,x,0);ap(lis,x,f(x));C1.create_line(lis,fill='blue')
#stap 1
```

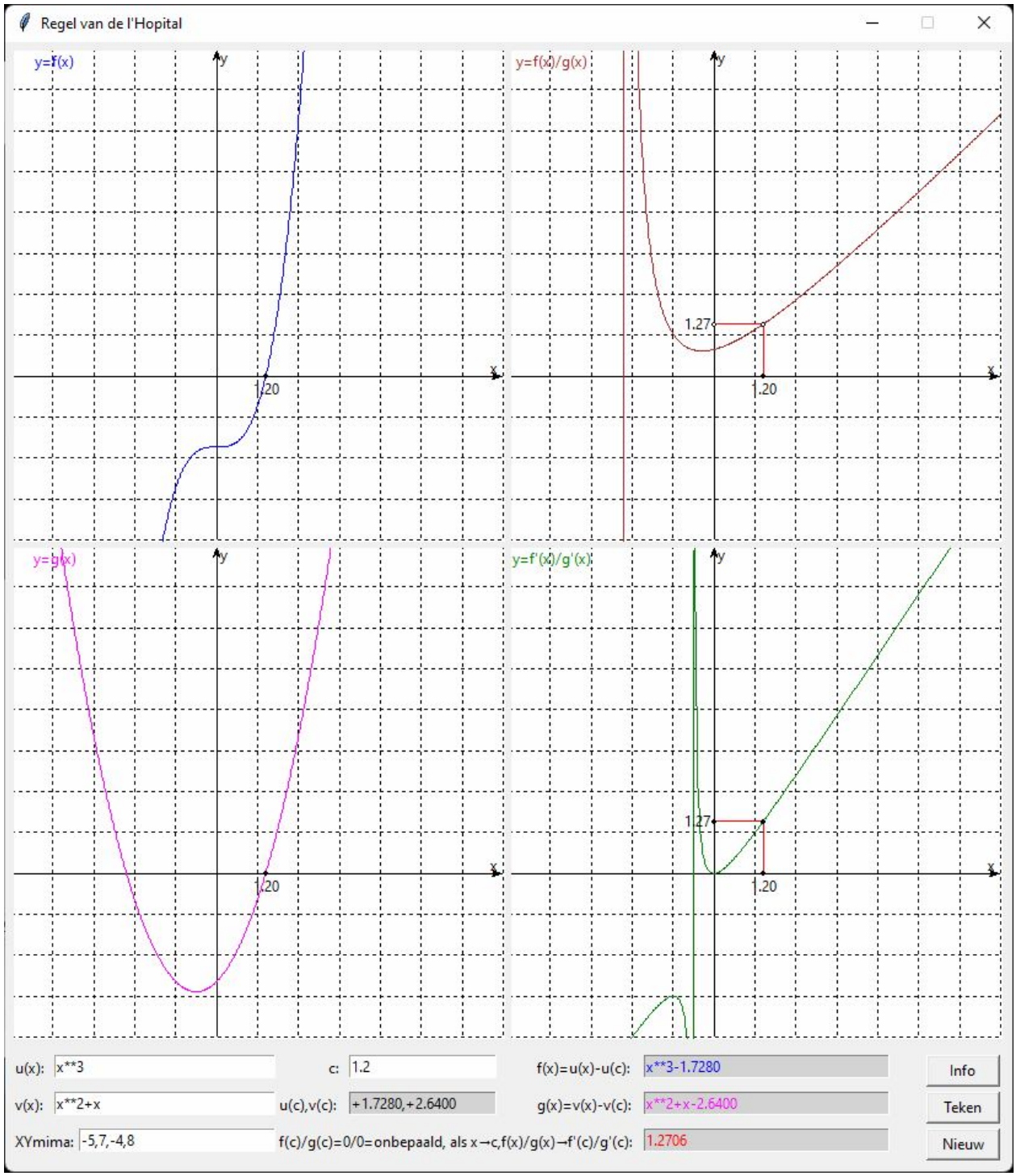
```
    rico1=df(x);q1=f(x)-x*rico1
    rondje(C1,x,f(x));rondje(C1,x+1,f(x));rondje(C1,x+1,f(x)+rico1) # raakpunt
# raaklijn: x => x+1
    lis=[];y=rico1*x+q1;ap(lis,x,y)
    y=y+rico1;ap(lis,x+1,y)
    C1.create_line(lis,fill='grey',dash=[1,2])
# horizontaal
    lis=[];y=rico1*x+q1;ap(lis,x,y);ap(lis,x+1,y)
    C1.create_line(lis,fill='grey',dash=[1,2])
# verticaal  (afgeleide)
    lisafg=[];kl='red'
    lis=[];ap(lis,x+1,y);y=y+rico1;ap(lis,x+1,y)
    C1.create_line(lis,fill=kl)
    lis=[];ap(lis,x,0);ap(lis,x,rico1)
    C2.create_line(lis,fill=kl)
    rondje(C2,x,0);rondje(C2,x,rico1)
#stap 2
    rico2=d2f(x);q2=df(x)-x*rico2
    rondje(C2,x,df(x));rondje(C2,x+1,df(x));rondje(C2,x+1,df(x)+rico2) # raakpunt
# raaklijn: x => x+1
    lis=[];y=rico2*x+q2;ap(lis,x,y)
    y=y+rico2;ap(lis,x+1,y)
    line = C2.create_line(lis,fill='grey',dash=[1,2])
# horizontaal
    lis=[];y=rico2*x+q2;ap(lis,x,y);ap(lis,x+1,y)
    C2.create_line(lis,fill='grey',dash=[1,2])
# verticaal  (afgeleide)
    lisafg=[];kl='magenta'
    lis=[];ap(lis,x+1,y);y=y+rico2;ap(lis,x+1,y)
    C2.create_line(lis,fill=kl)
    lis=[];ap(lis,x,0);ap(lis,x,rico2)
    C3.create_line(lis,fill=kl)
    rondje(C3,x,0);rondje(C3,x,rico2)
    if f(x)>0:ad('ligt boven de x-as omdat f(x) > 0')
    elif f(x)<0:ad('ligt onder de x-as omdat f(x) < 0')
    else:ad('snijdt de x-as omdat f(x) = 0')
    if df(x)>0:ad("is stijgend omdat f'(x) > 0")
    elif df(x)<0:ad("is dalend omdat f'(x) < 0")
    else:
       ad("heeft een horizontale raaklijn omdat f'(x) = 0")
    if d2f(x)>0:ad('ligt met de holle kant naar boven omdat f''(x) > 0')
    elif d2f(x)<0:ad('ligt met de holle kant naar beneden omdat f''(x) < 0')
    else:ad('heeft bijna steeds een buigpunt omdat f''(x)= 0')
def rondje(C,a,b):lis=[];r=0.05;ap(lis,a-r,b-r);ap(lis,a+r,b+r);C.create_oval(lis,fill='black')
def wis(E):E.delete(0,len(E.get()))
def ad(lin):tex.insert(INSERT,lin+'\n')
def nieuw():
    for E in (E1,E2,E3,E4,E5,E6):wis(E)
    for C in (C1,C2,C3):C.delete(ALL)
    tex.delete('1.0',END)
def info():messagebox.showinfo(tit+': info',infstr)
```

```
# hoofdprogramma
tit='Meetkundige betekenis 1ste en 2de afgeleide'
dx=1E-6;d2x=1E-4
breedte,hoogte=820,820;breedteC,hoogteC=400,400;lg='lightgrey'
form=Tk();form.geometry(str(breedte)+'x'+str(hoogte));form.title(tit);form.resizable(False,False)
C1=Canvas(form, bg="white", height=hoogteC, width=breedteC);C1.place(x=5,y=5)
C2=Canvas(form, bg="white", height=hoogteC, width=breedteC);C2.place(x=5,y=410)
C3=Canvas(form, bg="white", height=hoogteC, width=breedteC);C3.place(x=410,y=5)
Label(form,text='Voorschrift f(x):').place(x=410,y=425)
E1=Entry(form);E1.place(x=520,y=425,width=290);E1.insert(0,'(x**3-3*x+2)/5')
Label(form,text='xmi,xma,ymi,yma :').place(x=410,y=455)
E2=Entry(form);E2.place(x=520,y=455,width=290);E2.insert(0,'-4,6,-3,7')
Label(form,text='x   :').place(x=440,y=485)
E3=Entry(form);E3.place(x=480,y=485,width=120);E3.insert(0,'2')
Label(form,text='f(x):').place(x=630,y=485)
E4=Entry(form,bg=lg,fg='blue');E4.place(x=660,y=485,width=120)
Label(form,text="f'(x) :").place(x=430,y=515)
E5=Entry(form,bg=lg,fg='red');E5.place(x=480,y=515,width=120)
Label(form,text="f''(x) :').place(x=630,y=515)
E6=Entry(form,bg=lg,fg='magenta');E6.place(x=660,y=515,width=120)
tex=Text(form,bg='grey89',height=12,width=48,wrap=WORD,font=('"TkDefaultFont"','11'));tex.place(x=41
5,y=550)
Button(form,text="Info",command=info).place(x=440,y=780,width=60)
Button(form,text="Teken",command=bereken).place(x=570,y=780,width=60)
Button(form,text="Nieuw",command=nieuw).place(x=700,y=780,width=60)
infstr='Wijzig eventueel f(x) en xmi,xma,ymi,yma\n'
infstr=infstr+"Neem ≠ waarden voor x en klik 'Teken'"
form.mainloop()
```



73. Regel van de l'Hopital hopital

Dit programma illustreert deze regel. Je moet 2 willekeurige functies u(x) en v(x) en een getal c invoeren. Hiermee worden 2 functies f(x)=u(x)-u(c) en g(x)=v(x)-v(c) gemaakt. Er geldt steeds f(c)=g(c)=0. Alhoewel f(c)/g(c) niet bestaat: (zie wit 'rondje' in de bruine grafiek) , nadert de functie f(x)/g(x) tot f '(c)/g '(c) (vergelijk bruine en groene grafiek)



Programma:

```
# regel van de l'Hopital (met tkinter)
from math import *;from tkinter import *;from tkinter import messagebox
# meervoudige invoer
def mult(s):
    l=s.split(','); co=[]
    for i in l:co.append(float(i))
    return co
# omzetten wiskundige coördinaten x,y => schermcoördinaten X,Y
def transx(x):return (x-xmi)*breedteC/(xma-xmi)
def transy(y):return hoogteC-hoogteC*(y-ymi)/(yma-ymi)
```

```
# berekenen functie en afgeleide functie
def u(x):return eval(E0.get())
def v(x):return eval(E1.get())
def f(x):return eval(E5.get())
def g(x):return eval(E6.get())
def df(x):return (f(x+dx)-f(x-dx))/dx/2
def dg(x):return (g(x+dx)-g(x-dx))/dx/2
def st(x):
    if x==int(x):xs=str(int(x))
    else:xs=format(x,'.4f')
    if x>=0:return '+'+xs
    else:return xs
def init(Cv):
    global kl;Cv.delete(ALL);kl='black'
# assen
    X=transx(0);Cv.create_line(X,0,X,hoogteC,fill=kl,arrow='first')
    Y=transy(0);Cv.create_line(0,Y,breedteC,Y,fill=kl,arrow='last')
# rooster
    for x in range(int(xmi),int(xma+1)):
        X=transx(x);Cv.create_line(X,0,X,hoogteC,fill=kl,dash=[1,2])
    for y in range(int(ymi),int(yma+1)):
        Y=transy(y);Cv.create_line(0,Y,breedteC,Y,fill=kl,dash=[1,2] )
# x,y
    d=0.2;lis=[];ap(lis,xma-d,d);Cv.create_text(lis,text='x',fill=kl)
    lis=[];ap(lis,d,yma-d);Cv.create_text(lis,text='y',fill=kl)
# toevoegen punt aan lijst [x1,y1,x2,y2,...] van een kromme (of rechte)
def ap(li,x,y):X=transx(x);Y=transy(y);li.append(X);li.append(Y)
def teken(C,lis,tek,col):C.create_line(lis,fill=col);C.create_text(35,10,text=tek,fill=col)
def bereken():
    #krommen
    global xmi,xma,ymi,yma,c;xmi,xma,ymi,yma=mult(E2.get())
    us=E0.get();vs=E1.get()
    c=float(E3.get());uc=u(c);vc=v(c)
    wis(E4);E4.insert(0,st(uc)+','+st(vc))
    wis(E5);E5.insert(0,us+st(-uc))
    wis(E6);E6.insert(0,vs+st(-vc))
    for Cv in (C1,C2,C3,C4):init(Cv)
    lis1,lis2,lis3,lis4=[],[],[],[];stap=0.02;x=xmi
    while x<xma:
        x=x+stap;ap(lis1,x,f(x));ap(lis2,x,g(x))
        if abs(g(x))>0.001:ap(lis3,x,f(x)/g(x))
        if abs(dg(x))>0.001:ap(lis4,x,df(x)/dg(x))
    yc=df(c)/dg(c)
    li=[];ap(li,0,yc);ap(li,c,yc);ap(li,c,0)
    for C in(C3,C4):C.create_line(li,fill='red')
    teken(C1,lis1,'y=f(x)','blue');teken(C2,lis2,"y=g(x)",'magenta')
    teken(C3,lis3,'y=f(x)/g(x)','brown');teken(C4,lis4,"y=f'(x)/g'(x)",'green')
    for C in (C1,C2,C3,C4):rondje(C,c,0,'black');prinX(C,c)
    prinY(C3,yc);prinY(C4,yc)
    rondje(C3,c,yc,'white');rondje(C4,c,yc,'black')
    rondje(C3,0,yc,'white'); rondje(C4,0,yc,'black');
```

```
    wis(E7);E7.insert(0,format(yc,'.4f'))
    return
def prinX(C,a):lis=[];ap(lis,a,-0.3);C.create_text(lis,text=format(a,'.2f'))
def prinY(C,a):lis=[];ap(lis,-0.4,a);C.create_text(lis,text=format(a,'.2f'))
def rondje(C,a,b,col):lis=[];r=0.05;ap(lis,a-r,b-r);ap(lis,a+r,b+r);C.create_oval(lis,fill=col)
def wis(E):E.delete(0,len(E.get()))
def nieuw():
    for E in (E0,E1,E2,E3,E4,E5,E6,E7):wis(E)
    for C in (C1,C2,C3,C4):C.delete(ALL)
def info():messagebox.showinfo(tit+': info',infstr)
# hoofdprogramma
tit="Regel van de l'Hopital"
dx=1E-6;breedte,hoogte=820,920;breedteC,hoogteC=400,400;lg='lightgrey'
form=Tk();form.geometry(str(breedte)+'x'+str(hoogte));form.title(tit);form.resizable(False,False)
C1=Canvas(form, bg="white", height=hoogteC, width=breedteC);C1.place(x=5,y=5)
C2=Canvas(form, bg="white", height=hoogteC, width=breedteC);C2.place(x=5,y=410)
C3=Canvas(form, bg="white", height=hoogteC, width=breedteC);C3.place(x=410,y=5)
C4=Canvas(form, bg="white", height=hoogteC, width=breedteC);C4.place(x=410,y=410)
Label(form,text='u(x):').place(x=5,y=825)
E0=Entry(form);E0.place(x=40,y=825,width=180);E0.insert(0,'x**3')
Label(form,text='v(x):').place(x=5,y=855)
E1=Entry(form);E1.place(x=40,y=855,width=180);E1.insert(0,'x**2+x')
Label(form,text='XYmima:').place(x=5,y=885)
E2=Entry(form);E2.place(x=60,y=885,width=160);E2.insert(0,'-5,7,-4,8')
Label(form,text='c:').place(x=260,y=825)
E3=Entry(form);E3.place(x=280,y=825,width=120);E3.insert(0,'1.2')
Label(form,text='u(c),v(c):').place(x=220,y=855)
E4=Entry(form,bg=lg,fg='black');E4.place(x=280,y=855,width=120)
Label(form,text="f(x)=u(x)-u(c):").place(x=430,y=825)
E5=Entry(form,bg=lg,fg='blue');E5.place(x=520,y=825,width=200)
Label(form,text='g(x)=v(x)-v(c):').place(x=430,y=855)
E6=Entry(form,bg=lg,fg='magenta');E6.place(x=520,y=855,width=200)
Label(form,text="f(c)/g(c)=0/0=onbepaald, als x→c,f(x)/g(x)→f'(c)/g'(c):").place(x=220,y=885)
E7=Entry(form,bg=lg,fg='red');E7.place(x=520,y=885,width=200)
Button(form,text="Info",command=info).place(x=750,y=825,width=60)
Button(form,text="Teken",command=bereken).place(x=750,y=855,width=60)
Button(form,text="Nieuw",command=nieuw).place(x=750,y=885,width=60)
infstr='Vul in: u(x), v(x), c en xmi,xma,ymi,yma\n'
infstr=infstr+'Als f(x)= u(x)-u(c) en g(x)= v(x)-v(c),\n'
infstr=infstr+'dan is f(c)=g(c)=0. Alhoewel f(c)/g(c)\n'
infstr=infstr+"onbepaald is, nadert f(x)/g(x) tot f'(c)/g'(c)"
form.mainloop()
```

74. Constructie ellips met de hulpcirkels ellips_hulpcirkels

In tkinter kunnen we ook werken met 'sliders'
In het hoofdprogramma definieer je bijvoorbeeld als volgt een slider:

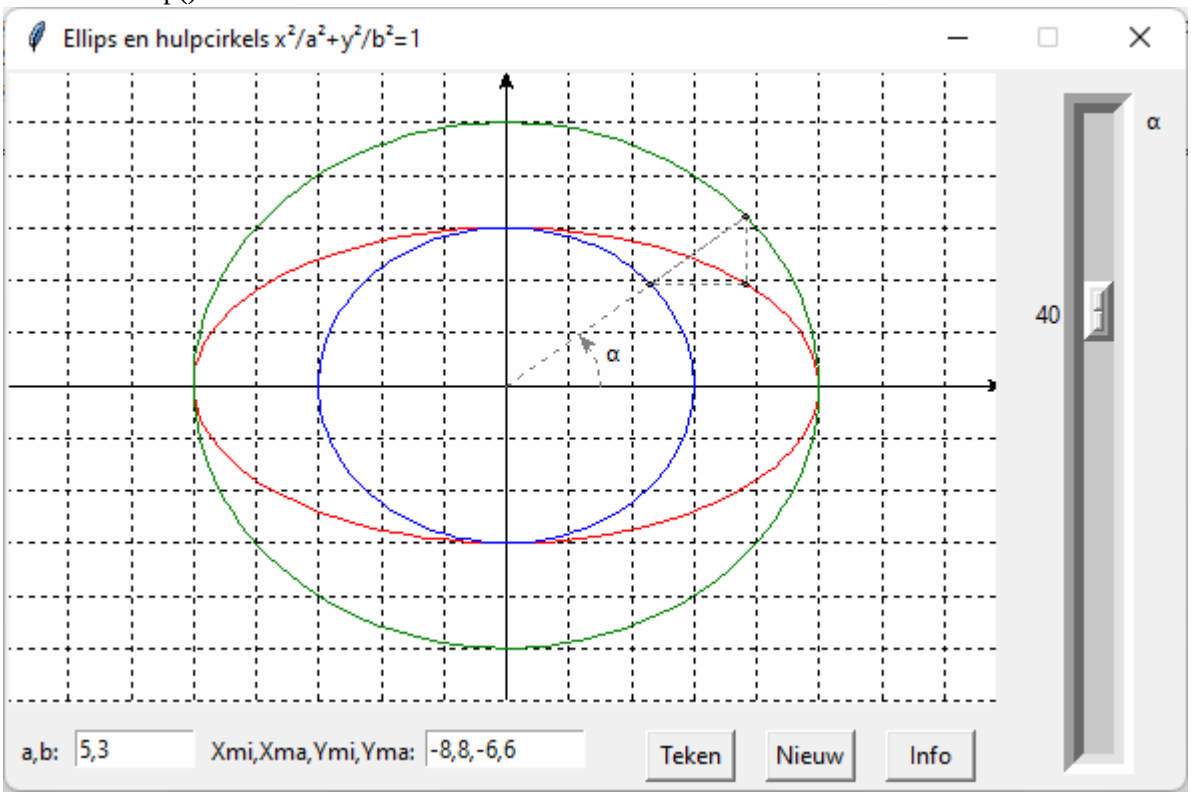
```
var=DoubleVar()
w=Scale(form,from_=..,to=..,length=..,bd=..,orient='vertical' ,resolution=..,label='..',variable=var)
w.set(40);w.place(x=..,y=..)
```

Gelijk waar nodig kan je de waarde van de slider uitlezen met: a=float(var.get())

Programma:

```
# ellips, hulpcirkels
from math import *;from tkinter import *;from tkinter import messagebox
def mult(s):
   l=s.split(',');co=[]
   for i in l:co.append(float(i))
   return co
def transx(x):return (x-xmi)*breedteC/(xma-xmi)
def transy(y):return hoogteC-hoogteC*(y-ymi)/(yma-ymi)
def teken():
   global xmi,xma,ymi,yma,r
   xmi,xma,ymi,yma=mult(E2.get());C.delete(ALL);bl='black'
#assen
   X=transx(0);C.create_line(X,0,X,hoogteC,fill=bl,arrow='first')
   Y=transy(0);C.create_line(0,Y,breedteC,Y,fill=bl,arrow='last')
#rooster
   for x in range(int(xmi),int(xma+1)):
      X=transx(x);C.create_line(X,0,X,hoogteC,fill=bl,dash=[1,2])
   for y in range(int(ymi),int(yma+1)):
      Y=transy(y);C.create_line(0,Y,breedte,Y,fill=bl,dash=[1,2] )
#ellips en hulpcirkels
   lisE,lisKC,lisGC=[],[],[];stap=0.05;t=-0.05;r=0.05
   a,b=mult(E1.get())
   while t<2*pi:
      t=t+stap;ap(lisE,a*cos(t),b*sin(t))
      ap(lisKC,b*cos(t),b*sin(t));ap(lisGC,a*cos(t),a*sin(t))
   C.create_line(lisE,fill='red');C.create_line(lisKC,fill='blue');
   C.create_line(lisGC,fill='green')
#hulplijnen
   alfa=radians(float(var.get()));lis=[];ap(lis,0,0);
   ap(lis,a*cos(alfa),a*sin(alfa))
   ap(lis,a*cos(alfa),b*sin(alfa));ap(lis,b*cos(alfa),b*sin(alfa));
   ap(lis,a*cos(alfa),a*sin(alfa))
   C.create_line(lis,fill='grey',dash=[1,2])
#punten
   rondje(a*cos(alfa),a*sin(alfa));rondje(a*cos(alfa),b*sin(alfa));
   rondje(b*cos(alfa),b*sin(alfa))
#boogje
   aantst=20;t=0;stap=alfa/aantst;i=-1;t=-stap;lis=[]
   while i<aantst:
      i=i+1;t=t+stap;ap(lis,b*cos(t)/2,b*sin(t)/2)
   C.create_line(lis,fill='grey',dash=[1,2],arrow='last')
   lis=[];ap(lis,0.6*b*cos(alfa/2),0.6*b*sin(alfa/2));
   C.create_text(lis,text='α')
def rondje(a,b):lis=[];ap(lis,a-r,b-r);ap(lis,a+r,b+r); C.create_oval(lis,fill='grey')
def ap(li,x,y):X=transx(x);Y=transy(y);li.append(X);li.append(Y)
def info():messagebox.showinfo(tit+': info',infstr)
```

```
def nieuw():
    E2.delete(0,len(E2.get()));E1.delete(0,len(E1.get()));C.delete(ALL)
# hoofdprogramma
tit='Ellips en hulpcirkels x²/a²+y²/b²=1'
breedte=590;hoogte=360;hoogteC=hoogte-45;breedteC=breedte-90;hoInv=hoogte-30
form=Tk();form.geometry(str(breedte)+'x'+str(hoogte));form.title(tit);form.resizable(False,False)
C = Canvas(form, bg="white", height=hoogteC, width=breedteC)
C.place(x=0,y=0);var=DoubleVar()
w=Scale(form,from_=-90,to=360,length=340,bd=10,orient='vertical',
resolution=5,label='α',variable=var)
w.set(40);w.place(x=breedteC,y=0)
Label(form,text='a,b:').place(x=5,y=hoInv)
E1=Entry(form);E1.place(x=35,y=hoInv,width=60);E1.insert(0,'5,3')
Label(form,text='Xmi,Xma,Ymi,Yma:').place(x=100,y=hoInv)
E2=Entry(form);E2.place(x=210,y=hoInv,width=80);E2.insert(0,'-8,8,-6,6')
Button(form,text='Teken',command=teken,width=5).place(x=320,y=hoInv)
Button(form,text='Nieuw',command=nieuw,width=5).place(x=380,y=hoInv)
Button(form,text='Info',command=info,width=5).place(x=440,y=hoInv)
infstr='Wijzig eventueel a,b en xmi,xma,ymi,yma\nMet de slider α instellen. '
infstr=infstr+'Met behulp van de kleine \nen de grote hulpcirkel kan telkens een punt van de\n'
infstr=infstr+'ellips getekend worden. De coördinaten van dit \npunt voldoen aan x=a.cosα , y=b.sinα'
form.mainloop()
```



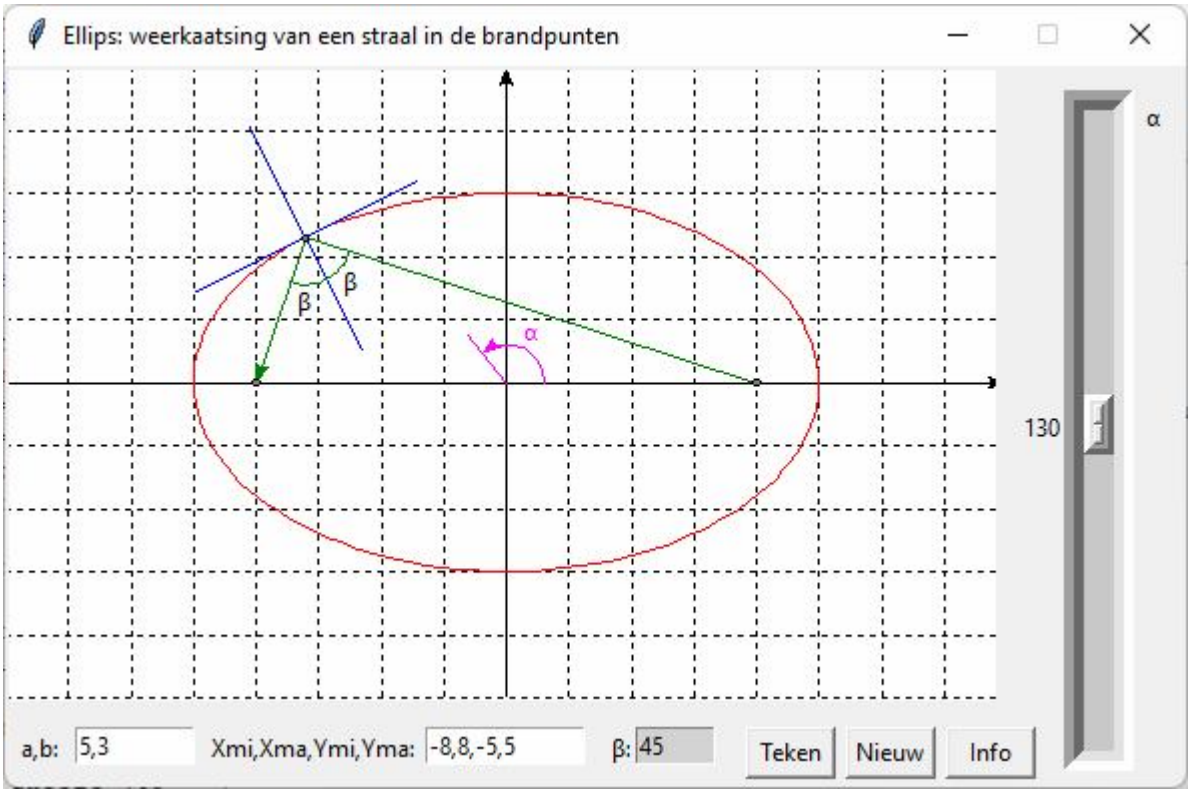
75. Ellips:weerkaatsing straal in de brandpunten ellips_brandpunt

Wentelen we een ellips rond de x-as, dan krijgen we een omwentelingsoppervlak: een ellipsoïde.
Alle stralen die vertrekken in een brandpunt, zullen door het oppervlak van de ellipsoïde weerkaatst worden in het andere brandpunt. Dit kan wiskundig bewezen worden.
Een illustratie hiervan wordt gegeven door het volgende programma:

Programma:

```
#ellips:weerkaatsing straal in de brandpunten
from math import *;from tkinter import *;from tkinter import messagebox
def mult(s):
   l=s.split(',');co=[]
   for i in l:co.append(float(i))
   return co
def transx(x):return (x-xmi)*breedteC/(xma-xmi)
def transy(y):return hoogteC-hoogteC*(y-ymi)/(yma-ymi)
def teken():
   global xmi,xma,ymi,yma,r
   xmi,xma,ymi,yma=mult(E2.get());C.delete(ALL);bl='black'
#assen
   X=transx(0);C.create_line(X,0,X,hoogteC,fill=bl,arrow='first')
   Y=transy(0);C.create_line(0,Y,breedteC,Y,fill=bl,arrow='last')
#rooster
   for x in range(int(xmi),int(xma+1)):
      X=transx(x);C.create_line(X,0,X,hoogteC,fill=bl,dash=[1,2])
   for y in range(int(ymi),int(yma+1)):
      Y=transy(y);C.create_line(0,Y,breedte,Y,fill=bl,dash=[1,2] )
#ellips
   lisE=[];stap=0.05;t=-0.05;r=0.05;a,b=mult(E1.get())
   if a<=b: messagebox.showinfo('Fout','Kies a>b');return
   f=sqrt(a**2-b**2)
   while t<2*pi:
      t=t+stap;ap(lisE,a*cos(t),b*sin(t))
   C.create_line(lisE,fill='red')
#hulplijnen
   alfa=radians(float(var.get()));lis=[];ap(lis,f,0)
   ap(lis,a*cos(alfa),b*sin(alfa));ap(lis,-f,0)
   C.create_line(lis,fill='green',arrow='last')
   lis=[];ap(lis,0,0);ap(lis,cos(alfa),sin(alfa))
   C.create_line(lis,fill='magenta')
#raaklijn en normaal
   x1,y1=a*cos(alfa),b*sin(alfa);
   if abs(y1)>=1E-6:
      ricoR=-b*b*x1/a/a/y1;le=2*cos(atan(ricoR))
      lis=[];ap(lis,x1+le,y1+le*ricoR);ap(lis,x1-le,y1-le*ricoR)
      C.create_line(lis,fill='blue')
   if x1!=0:
      ricoN=a*a*y1/b/b/x1;le=2*cos(atan(ricoN))
      lis=[];ap(lis,x1+le,y1+le*ricoN);ap(lis,x1-le,y1-le*ricoN)
      C.create_line(lis,fill='blue')
   if abs(y1)<1E-6:
      le=2;lis=[];ap(lis,x1,-le);ap(lis,x1,le);C.create_line(lis,fill='blue')
   if x1==0:
      le=2;lis=[];ap(lis,-le,y1);ap(lis,le,y1);
      C.create_line(lis,fill='blue')
#punten
   rondje(x1,y1);rondje(-f,0);rondje(f,0);betarad=atan(f*y1/b/b)
```

```
    beta=int(abs(degrees(betarad)));wis(E3);E3.insert(0,beta)
#boogjes
    aantst=20;t=0;stap=alfa/aantst;i=-1;t=-stap;lis=[]
    while i<aantst:
        i=i+1;t=t+stap;ap(lis,b*cos(t)/5,b*sin(t)/5)
    C.create_line(lis,fill='magenta',arrow='last')
    lis=[];ap(lis,0.3*b*cos(alfa/2),0.3*b*sin(alfa/2))
    C.create_text(lis,text='α',fill='magenta')
    gamma=atan(y1/(x1+f))
    if x1+f<0:gamma=pi+gamma
    stap=2*betarad/aantst;t=gamma-stap;i=-1;lis=[]
    while i<aantst:
        i=i+1;t=t+stap;ap(lis,x1-b*cos(t)/4,y1-b*sin(t)/4)
    C.create_line(lis,fill='green')
#beta
    for i in (1,3):
        t=(gamma-stap)+i*aantst*stap/4;lis=[];
        ap(lis,x1-b*cos(t)/3,y1-b*sin(t)/3);C.create_text(lis,text='β')
def rondje(a,b):lis=[];ap(lis,a-r,b-r);ap(lis,a+r,b+r);C.create_oval(lis,fill='grey')
def ap(li,x,y):X=transx(x);Y=transy(y);li.append(X);li.append(Y)
def info():messagebox.showinfo(tit+': info',infstr)
def wis(E):E.delete(0,len(E.get()))
def nieuw():
    for E in (E1,E2,E3):wis(E)
    C.delete(ALL)
# hoofdprogramma
tit='Ellips: weerkaatsing van een straal in de brandpunten'
breedte=590;hoogte=360;hoogteC=hoogte-45;breedteC=breedte-90;hoInv=hoogte-30
form=Tk();form.geometry(str(breedte)+'x'+str(hoogte))
form.title(tit);form.resizable(False,False)
C = Canvas(form, bg="white", height=hoogteC, width=breedteC)
C.place(x=0,y=0);var=DoubleVar()
w=Scale(form,from_=-90,to=360,length=340,bd=10,orient='vertical',
resolution=5,label='α',variable=var)
w.set(35);w.place(x=breedteC-5,y=0)
Label(form,text='a,b:',).place(x=5,y=hoInv)
E1=Entry(form);E1.place(x=35,y=hoInv,width=60);E1.insert(0,'5,3')
Label(form,text='Xmi,Xma,Ymi,Yma:').place(x=100,y=hoInv)
E2=Entry(form);E2.place(x=210,y=hoInv,width=80);E2.insert(0,'-8,8,-5,5')
Label(form,text='β:').place(x=300,y=hoInv)
E3=Entry(form,bg='lightgrey');E3.place(x=315,y=hoInv,width=40)
Button(form,text='Teken',command=teken,width=5).place(x=370,y=hoInv)
Button(form,text='Nieuw',command=nieuw,width=5).place(x=420,y=hoInv)
Button(form,text='Info',command=info,width=5).place(x=470,y=hoInv)
infstr='Wijzig eventueel a,b en xmi,xma,ymi,yma\n'
infstr=infstr+'Met de slider α instellen \n'
infstr=infstr+'Elke straal die vertrekt uit een brandpunt wordt\n'
infstr=infstr+'door de ellips weerkaatst in het andere brandpunt\n'
infstr=infstr+'Tussen stralen en normaal is de hoek β steeds gelijk'
form.mainloop()
```



76. Parabool:weerkaatsing straal in het brandpunt parabool_brandpunt

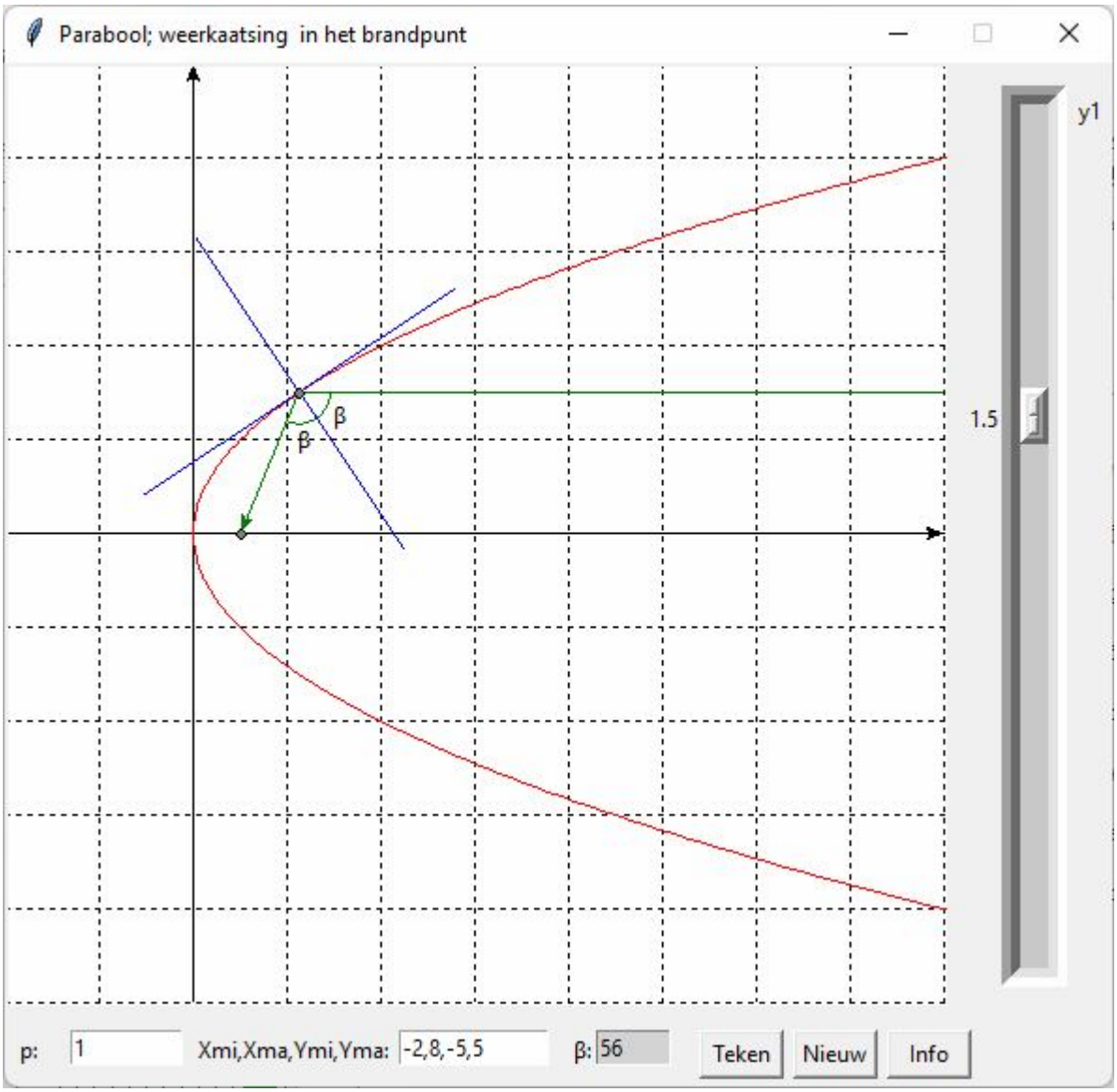
Wentelen we een parabool rond de x-as, dan krijgen we een paraboloïde.
Alle stralen die evenwijdig met de as lopen, zullen door het oppervlak van de paraboloïde weerkaatst worden in het brandpunt. Dit wordt getoond met het volgende programma:

Programma:

```
#parabool:weerkaatsing straal in het brandpunt
from math import *;from tkinter import *;from tkinter import messagebox
def mult(s):
    l=s.split(',');co=[]
    for i in l:co.append(float(i))
    return co
def transx(x):return (x-xmi)*breedteC/(xma-xmi)
def transy(y):return hoogteC-hoogteC*(y-ymi)/(yma-ymi)
def teken():
    global xmi,xma,ymi,yma,r
    xmi,xma,ymi,yma=mult(E2.get());C.delete(ALL);bl='black'
#assen
    X=transx(0);C.create_line(X,0,X,hoogteC,fill=bl,arrow='first')
    Y=transy(0);C.create_line(0,Y,breedteC,Y,fill=bl,arrow='last')
#rooster
    for x in range(int(xmi),int(xma+1)):
        X=transx(x);C.create_line(X,0,X,hoogteC,fill=bl,dash=[1,2])
    for y in range(int(ymi),int(yma+1)):
        Y=transy(y);C.create_line(0,Y,breedte,Y,fill=bl,dash=[1,2] )
#parabool
    lisE=[];stap=0.05;y=ymi-0.05;r=0.05
```

```
    p=float(E1.get());
    while y<yma:
        y=y+stap;ap(lisE,y*y/2/p,y)
    C.create_line(lisE,fill='red')
#hulplijnen
    y1=float(var.get());x1=y1*y1/2/p;lis=[];ap(lis,p/2,0);ap(lis,x1,y1)
    ap(lis,xma,y1)
    C.create_line(lis,fill='green',arrow='first')
#raaklijn en normaal
    if abs(y1)>=1E-6:
        ricoR=p/y1;ricoN=-y1/p;le=2*cos(atan(ricoR))
        lis=[];ap(lis,x1+le,y1+le*ricoR);ap(lis,x1-le,y1-le*ricoR)
        C.create_line(lis,fill='blue')
        le=2*cos(atan(ricoN))
        lis=[];ap(lis,x1+le,y1+le*ricoN);ap(lis,x1-le,y1-le*ricoN)
        C.create_line(lis,fill='blue')
    if abs(y1)<1E-6:
        le=2;lis=[];ap(lis,x1,-le);ap(lis,x1,le);C.create_line(lis,fill='blue')
        lis=[];ap(lis,-le,y1);ap(lis,le,y1);C.create_line(lis,fill='blue')
#punten
    rondje(x1,y1);rondje(p/2,0)
#boogjes
    betarad=atan(-y1/p);gamma=pi+2*betarad;aantst=20
    stap=-2*betarad/aantst;t=gamma-stap;i=-1;lis=[]
    while i<aantst:i=i+1;t=t+stap;ap(lis,x1-p*cos(t)/3,y1-p*sin(t)/3)
    C.create_line(lis,fill='green')
#beta
    beta=int(abs(degrees(betarad)));wis(E3);E3.insert(0,str(beta))
    for h in (0.5,1.5):lis=[];ap(lis,x1-0.5*cos(gamma-h*betarad),y1-0.5*sin(gamma-
h*betarad));C.create_text(lis,text='β')
def rondje(a,b):lis=[];ap(lis,a-r,b-r);ap(lis,a+r,b+r);C.create_oval(lis,fill='grey')
def ap(li,x,y):X=transx(x);Y=transy(y);li.append(X);li.append(Y)
def info():messagebox.showinfo(tit+': info',infstr)
def wis(E):E.delete(0,len(E.get()))
def nieuw():
    for E in (E1,E2,E3):wis(E)
    C.delete(ALL)
# hoofdprogramma
tit='Parabool; weerkaatsing  in het brandpunt'
breedte=590;hoogte=545;hoogteC=hoogte-45;breedteC=breedte-90;hoInv=hoogte-30
form=Tk();form.geometry(str(breedte)+'x'+str(hoogte));form.title(tit);form.resizable(False,False)
C = Canvas(form, bg="white", height=hoogteC, width=breedteC);C.place(x=0,y=0)
var=DoubleVar()
w=Scale(form,from_=5,to=-5,length=480,bd=10,orient='vertical', resolution=0.25,label='y1',variable=var)
w.set(2);w.place(x=breedteC-5,y=0)
C = Canvas(form, bg="white", height=hoogteC, width=breedteC);C.place(x=0,y=0)
L1=Label(form,text='p:',);L1.place(x=5,y=hoInv)
E1=Entry(form);E1.place(x=35,y=hoInv,width=60);E1.insert(0,'1')
L2=Label(form,text='Xmi,Xma,Ymi,Yma:',);L2.place(x=100,y=hoInv)
E2=Entry(form);E2.place(x=210,y=hoInv,width=80);E2.insert(0,'-2,8,-5,5')
L3=Label(form,text='β:',);L3.place(x=300,y=hoInv)
```

```
E3=Entry(form,bg='lightgrey');E3.place(x=315,y=hoInv,width=40)
B1=Button(form,text='Teken',command=teken,width=5);B1.place(x=370,y=hoInv)
B2=Button(form,text='Nieuw',command=nieuw,width=5);B2.place(x=420,y=hoInv)
B3=Button(form,text='Info',command=info,width=5);B3.place(x=470,y=hoInv)
infstr='Wijzig eventueel p en xmi,xma,ymi,yma\n'
infstr=infstr+'Met de slider y instellen \n'
infstr=infstr+'Elke straal die evenwijdig met de symmetrieas invalt\n'
infstr=infstr+'wordt weerkaatst door de parabool in het brandpunt.\n'
infstr=infstr+'Tussen stralen en normaal is de hoek β steeds gelijk'
form.mainloop()
```



77. Grafieken van kegelsneden-raaklijn en normaal kegelsneden

$ax^2+by^2+cx+dy+e=0$ (*) is (meestal) de vergelijking van een kegelsnede waarvan de symmetrieas(-sen) evenwijdig is (zijn) met X of/en Y-as. We bespreken alle mogelijkheden.

Cartesische vergelijkingen		Parametervergelijkingen			
a=b=0 d≠0	: cx+dy+e=0 :	x=t	$y=-\frac{ct+e}{d}$	rechte	(1)
a=b=d=0 c≠0	: cx+e=0 :	x=-e/c	y=t	verticale rechte	(2)
a=c=0	: by^2+dy+e=0 :	x=t	$y=y_{1,2}$	2,1,of 0 horizontalen	(3)
b=d=0	: ax^2+cx+e=0 :	$x=x_{1,2}$	y=t	2,1,of 0 verticalen	(4)
a=b=c=d=0	: Π_0 of φ				(5)

a· b>0: herleidbaar naar ofwel:

$\frac{(x-x_1)^2}{A^2}+\frac{(y-y_1)^2}{B^2}=1$:	$x=x_1+A.\cos(t)$	$y=y_1+B.\sin(t)$	ellips	(6)
$\frac{(x-x_1)^2}{A^2}+\frac{(y-y_1)^2}{B^2}=0$:			puntcirkel (x_1, y_1)	(7)
$\frac{(x-x_1)^2}{A^2}+\frac{(y-y_1)^2}{B^2}<0$:			geen reële kegelsnede	(8)

a.b<0: herleidbaar naar ofwel:

$\frac{(x-x_1)^2}{A^2}-\frac{(y-y_1)^2}{B^2}=1$:	$x=x_1+A/\cos(t)$	$y=y_1+B.\tan(t)$	liggende hyperbool	(9)
$\frac{(x-x_1)^2}{A^2}-\frac{(y-y_1)^2}{B^2}=0$:	$x=x_1+t$	$y=y_1\pm\frac{B}{A}\cdot t$	2 schuine rechten	(10)
$\frac{(x-x_1)^2}{A^2}+\frac{(y-y_1)^2}{B^2}=-1$:	$x=x_1+A.\tan(t)$	$y=y_1+B/\cos(t)$	staande hyperbool	(11)

a=0 b≠0 : by^2+cx+dy+e=0

$(y-y_1)^2=P.(x-x_1)$:	$x=x_1+\frac{t^2}{P}$	$y=y_1+t$	liggende parabool	(12)

a≠0 b=0 : ax^2+cx+dy+e=0

$(x-x_1)^2=P.(y-y_1)$:	$x=x_1+t$	$y=y_1+\frac{t^2}{P}$	staande parabool	(13)

Om A, B ,P ,x1 ,y1 te berekenen uit a, b, c, d, e werken we de C.V. uit en vergelijken met (*)
We doen de berekeningen voor **6,9,11,12,13**

6.
$(x-x_1)^2.B^2+(y-y_1)^2.A^2-A^2B^2=0$
$x^2.B^2+y^2.A^2-2x_1.B^2.x-2.y_1.A^2.y+x_1^2.B^2+y_1^2.A^2-A^2.B^2=0$ (vgl met *)
$B^2=a.k$ $A^2=b.k$ $-2x_1.B^2=c.k$ $-2y_1.A^2=d.k$ $x_1^2.B^2+y_1^2.A^2-A^2.B^2=e.k$
$-2x_1.a=c$ $-2y_1.b=d$ $x_1^2.a.k+y_1^2.b.k-a.b.k^2=e.k$
$abk=x_1^2.a+y_1^2.b-e$ $k=x_1^2/b+y_1^2/a-e/ab$
Zodat in volgorde: $x_1=-c/(2a)$ $y_1=-d/(2b)$ $k=x_1^2/b+y_1^2/a-e/ab$ $B=\text{sqrt}(a.k)$ $A=\text{sqrt}(b.k)$
Omdat a.b>0 hebben we enkel een ellips als ook a.k >0 , anders hebben we geen reële kegelsnede

9.
$(x-x_1)^2.B^2-(y-y_1)^2.A^2-A^2.B^2=0$
$x^2.B^2-y^2.A^2-2x_1.B^2.x+2.y_1.A^2.y+x_1^2.B^2-y_1^2.A^2-A^2.B^2=0$ (vgl met *)
$B^2=a.k$ $A^2=-b.k$ $-2x_1.B^2=c.k$ $2y_1.A^2=d.k$ $x_1^2.B^2-y_1^2.A^2-A^2.B^2=e.k$

$-2x_1.a=c$ $2y_1.b=-d$ $x_1^2.a.k+y_1^2.b.k+a.b.k^2=e.k$

$a.b.k=-x_1^2.a-y_1^2.b+e$ $k=-x_1^2/b-y_1^2/a + e/ab$

Zodat in volgorde: $x_1=-c/(2a)$ $y_1=-d/(2b)$ $k=-x_1^2/b-y_1^2/a + e/ab$ $B=sqrt(a.k)$ $A=sqrt(-b.k)$

11.

$(x-x_1)^2.B^2-(y-y_1)^2.A^2 +A^2.B^2=0$

$x^2.B^2-y^2.A^2-2x_1.B^2.x+2.y_1.A^2.y+x_1^2.B^2-y_1^2.A^2+A^2.B^2=0$ (vgl met *)

$B^2=a.k$ $A^2=-b.k$ $-2x_1.B^2=c.k$ $2y_1.A^2=d.k$ $x_1^2.B^2-y_1^2.A^2+A^2.B^2=e.k$

$-2x_1.a=c$ $2y_1.b=-d$ $x_1^2.a.k+y_1^2.b.k-a.b.k^2=e.k$

$a.b.k=x1^2.a+y1^2.b-e$ $k=x1^2/b+y1^2/a - e/ab$

Zodat in volgorde: $x_1=-c/(2a)$ $y_1=-d/(2b)$ $k=x_1^2/b+y_1^2/a - e/ab$ $B=sqrt(a.k)$ $A=sqrt(-b.k)$

Merk op dat bij de vergelijkingen van (9) en (11) enkel k tegengesteld is,de rest blijft gelijk

Bij (9) is $x_1^2.B^2-y_1^2.A^2-e.k=A^2\cdot B^2$, bij (11) is $x_1^2.B^2-y_1^2.A^2-e.k=-A^2\cdot B^2$

Dus als $x_1^2.B^2-y_1^2.A^2-e.k<0$ dan moeten we k=-k nemen en hebben we geval 11

12:

$y^2-2y_1.y+y_1^2-P.x+P.x1=0$ of $b.y^2-2y_1.b.y-P.b.x+y_1^2.b+P.b.x_1=0$ (vgl.met *)

$-P.b=c$ $-2y_1.b=d$ $y_1^2.b+P.b.x_1=e$

$P=-c/b$ $y_1=-d/(2b)$ $d^2/(4b)-c.x_1=e$ of $x_1=(d^2-4eb)/(4bc)$

13:

$x^2-2x_1.x+x_1^2-P.y+P.y_1=0$ of $a.x^2-2x_1.a.x -P.a.y+x_1^2.a +P.ay_1=0$ (vgl.met *)

$-P.a=d$ $-2x_1.a=c$ $x_1^2.a +P.a.y_1=e$

$P=-d/a$ $x_1=-c/(2a)$ $c^2/(4a)-d.y_1=e$ of $y_1=(c^2-4ea)/(4ad)$

Raaklijnen en normalen:

Eerst bepalen we het raakpunt (x_r , y_r):

x_r moet gegeven worden: hieruit wordt yr berekend door de vergelijking

$ax_r^2+by^2+cx_r+dy+e=0$ op te lossen naar y:

Als b=0, dan krijgen we een lineaire vergelijking waaruit 1 y_r kan berekend worden:

$$y_r = -\frac{ax_r^2 + cx_r + e}{d}$$

Is $b\neq 0$ dan hebben we een vierkantsvergelijking : $by^2+dy+ ax_r^2+ cx_{r+} e=0$

$D=B^2-4AC = d^2-4b.(ax_r^2+ cx_{r+} e)$

D>0: $y_{r_{1,2}} = \frac{-d \pm \sqrt{D}}{2 \cdot b}$ D=0: $y_r = \frac{-d}{2 \cdot b}$ D<0: geen

En nu de vergelijkingen:

We nemen de afgeleide van (*): $2ax+2byy'+c+dy'=0$ waaruit volgt: $y' = -\frac{2ax + c}{2by + d}$

Is (x_r , y_r) het raakpunt, dan is

de rico van de raaklijn: $\text{ricoR} = -\frac{2ax_r + c}{2by_r + d}$ en van de normaal: $\text{ricoN} = \frac{2by_r + d}{2ax_r + c}$

Voor de raaklijn stellen we nm= $2by_r + d$, voor de normaal nm= $2ax_r + c$ (nm =noemer)

Als nm≠0, dan wordt vergelijking van raaklijn of normaal gegeven door :

$y = y_r + \text{rico} \cdot (x - x_r) = \text{rico} \cdot x - \text{rico} \cdot x_r + y_r = \text{rico} \cdot x + q$

Is nm=0 , dan hebben we een verticale rechte met vergelijking x= x_r

Een raaklijn of normaal tekenen met lengte 2 kan met:

```
lis=[];x=xr-cos(atan(rico));y=rico*x+q;ap(lis,x,y)
x=xr+cos(atan(rico));y=rico*x+q;ap(lis,x,y)
```

De ap-functie vult lis aan (append) met de schermcoördinaten van 2 punten van raaklijn of normaal die op afstand 1 liggen van (x_r , y_r)
Omdat $x-x_r=\pm \cos(\text{atan}(\text{rico}))$ en $y-y_r = \text{rico}.(x-x_r)$
zodat $(x-x_r)^2+(y-y_r)^2 = (x-x_r)^2\cdot(1+\text{rico}^2)= \cos^2(\text{atan}(\text{rico}))[1+\tan(\text{atan}(\text{rico}))^2] =\cos^2\alpha[1+\tan^2\alpha]=1$

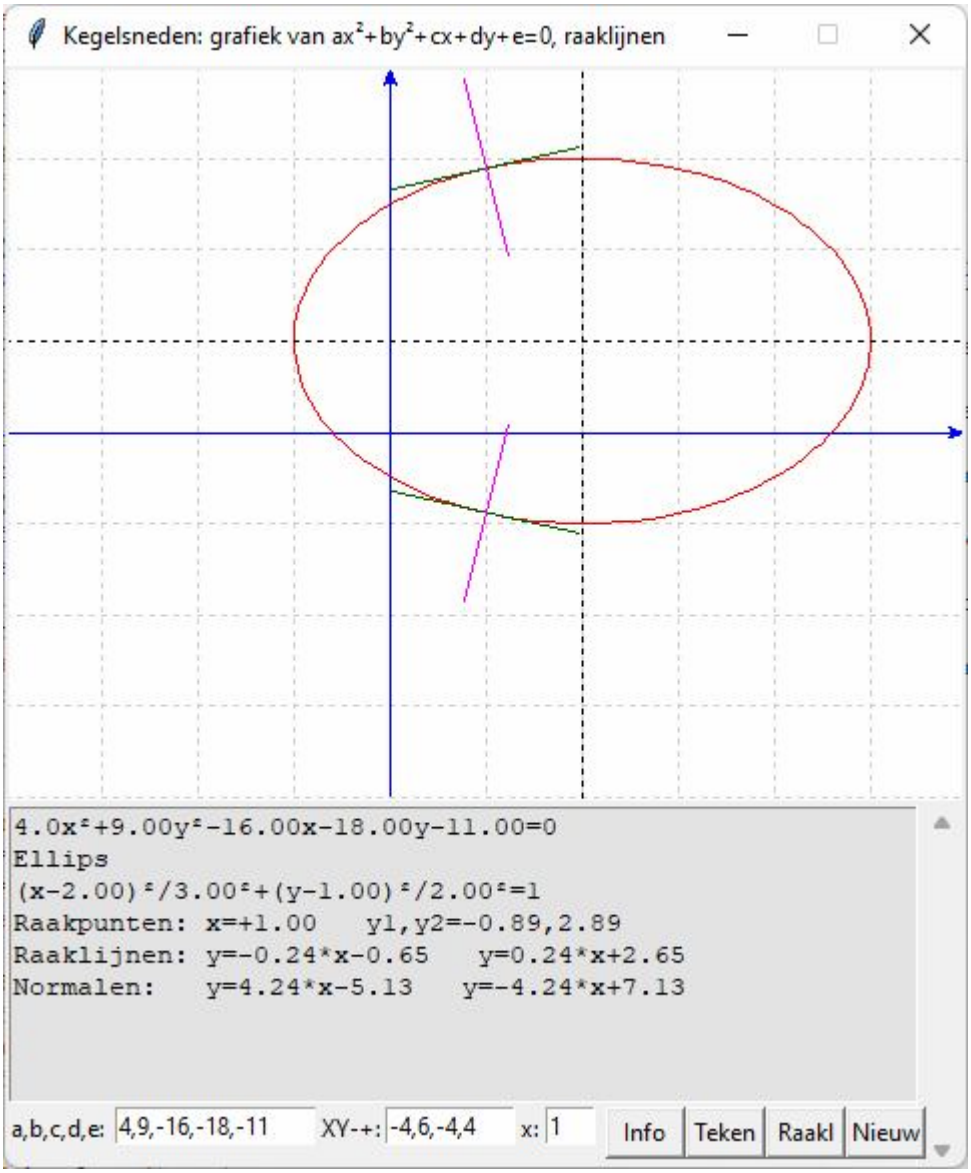
Programma:

```
# grafieken kegelsneden
from math import *;from tkinter import *;dx=0.00001;nw='.2f'
from tkinter import messagebox
def vc(g):
   sg=format(g,nw)
   if g>=0:return '+'+sg
   else:return(sg)
def v(g):return format(g,nw)
def mult(s):
   l=s.split(',')
   co=[]
   for i in l:co.append(float(i))
   return co
def transx(x):return (x-xmi)*breedte/(xma-xmi)
def transy(y):return hoogteC-hoogteC*(y-ymi)/(yma-ymi)
def f(x):return eval(E1.get())
def df(x):return (f(x+dx)-f(x))/dx
def teken():
   global xmi,xma,ymi,yma,k
   xmi,xma,ymi,yma=mult(E2.get());C.delete(ALL);tex.delete('1.0',END)
# rooster
   for x in range(int(xmi),int(xma+1)):
      X=transx(x);C.create_line(X,0,X,hoogteC,fill='lightgrey',dash=[1,2])
   for y in range(int(ymi),int(yma+1)):
      Y=transy(y);C.create_line(0,Y,breedte,Y,fill='lightgrey',dash=[1,2] )
#assen
   X=transx(0);C.create_line(X,0,X,hoogteC,fill='blue',arrow='first')
   Y=transy(0);C.create_line(0,Y,breedte,Y,fill='blue',arrow='last')
# berekening params kegelsneden
# rechten
   a,b,c,d,e=mult(E1.get());k=1
   ad(str(a)+'x²'+vc(b)+'y²'+vc(c)+'x'+vc(d)+'y'+vc(e)+'=0\n')
   if (a,b,c,d)==(0,0,0,0):ad('Geen kegelsnede\n');return
   if (a,b)==(0,0):ad('Rechte\n')
   if (a,c)==(0,0):ad('Horizontale(n)\n')
   if (b,d)==(0,0):ad('Verticale(n)\n')
# parabolen
   if a==0 and b!=0 and c!=0:
      P=-c/b;y1=-d/2/b;x1=(d*d-4*e*b)/4/b/c;ad('Liggende parabool\n')
      ad('(y'+vc(-y1)+')²='+vc(P)+'(x'+vc(-x1)+')\n')
```

```
    if b==0 and a!=0 and d!=0:
        P=-d/a;x1=-c/2/a;y1=(c*c-4*e*a)/4/a/d;ad('Staande parabool\n')
        ad('y'+vc(-y1)+'='+vc(1/P)+'(x'+vc(-x1)+')²\n')
# ellipsen
    if a*b>0:
        x1=-c/2/a;y1=-d/2/b;k=x1**2/b+y1**2/a-e/a/b
        if a*k>0:
            B=sqrt(a*k);A=sqrt(b*k);ad('Ellips\n')
            ad('(x'+vc(-x1)+')²/'+v(A)+'²+(y'+vc(-y1)+')²/'+v(B)+'²=1\n')
        elif k==0:ad('Puntcirkel('+vc(x1)+','+vc(y1)+')\n')
        else:ad('Geen reële kegelsnede\n')
# hyperbolen
    if a*b<0:
        x1=-c/2/a;y1=-d/2/b;k=x1**2/b+y1**2/a-e/a/b
        if b*k>0:k=-k
        if k!=0:
            B=sqrt(a*k);A=sqrt(-b*k)
            l=(x1*B)**2-(y1*A)**2-k*e
            ad('Hyperbool\n')
            ad('(x'+vc(-x1)+')²/'+v(A)+'²-(y'+vc(-y1)+')²/'+v(B)+'²=')
            if l>0:ad('1\n')
            else:ad('-1\n')
        else:ad(str(a)+'(x'+vc(-x1)+')²'+vc(b)+'(y'+vc(-y1)+')²=0')
#kromme
    if a*b==0:tmi=-50;tma=50
    else: tmi=-pi;tma=pi+0.1
    lis=[];stap=0.1;t=tmi
    if a==0 and b==0 and d!=0: #rechte
        ap(lis,xmi,(c*xmi+e)/-d);ap(lis,xma,(c*xma+e)/-d)
        C.create_line(lis,fill='red')
    elif a==0 and c==0 and b!=0:
        D=d*d-4*b*e
        if D>=0:
            lis=[];ap(lis,xmi,(-d-sqrt(D))/2/b);ap(lis,xma,(-d-sqrt(D))/2/b);C.create_line(lis,fill='red')
            lis=[];ap(lis,xmi,(-d+sqrt(D))/2/b);ap(lis,xma,(-d+sqrt(D))/2/b);C.create_line(lis,fill='red')
    elif b==0 and d==0 and a!=0:
        D=c*c-4*a*e
        if D>=0:
            lis=[];ap(lis,(-c-sqrt(D))/2/a,ymi);ap(lis,(-c-sqrt(D))/2/a,yma);C.create_line(lis,fill='red')
            lis=[];ap(lis,(-c+sqrt(D))/2/a,ymi);ap(lis,(-c+sqrt(D))/2/a,yma);C.create_line(lis,fill='red')
    elif a==0 and b==0 and d==0 and c!=0:
        lis=[];ap(lis,-e/c,ymi);ap(lis,-e/c,yma);C.create_line(lis,fill='red')
    elif a==0 and b==0 and c==0 and d!=0:
        lis=[];ap(lis,xmi,-e/d);ap(lis,xma,-e/d);C.create_line(lis,fill='red')
    elif a*b>0 and k==0:lis=[];rf=0.05;ap(lis,x1-rf,y1-rf);ap(lis,x1+rf,y1+rf);C.create_oval(lis,fill='red')
    elif a*b<0 and k==0:
            ric=sqrt(abs(a/b));lis=[];ap(lis,xmi,ric*(xmi-x1)+y1)
             ap(lis,xma,ric*(xma-x1)+y1);C.create_line(lis,fill='red')
            lis=[];ap(lis,xmi,-ric*(xmi-x1)+y1);ap(lis,xma,-ric*(xma-x1)+y1);C.create_line(lis,fill='red')
            ad('\nOntaarde hyperbool\n')
    else:
```

```
        while t<tma:
            t=t+stap
            if a==0 and b!=0:x=x1+t*t/P;y=y1+t
            if b==0 and a!=0:x=x1+t;y=y1+t*t/P
            if a*b>0 and a*k>0:x=x1+A*cos(t);y=y1+B*sin(t)
            if a*b<0 and a*k>0 and l>0:x=x1+A/cos(t);y=y1+B*tan(t)
            if a*b<0 and a*k>0 and l<0:x=x1+A*tan(t);y=y1+B/cos(t)
            ap(lis,x,y)
        if a*k>=0:C.create_line(lis,fill='red')
    # symmetrieassen
        lis=[];ap(lis,xmi,y1);ap(lis,xma,y1);C.create_line(lis,fill='black',dash=[1,2])
        lis=[];ap(lis,x1,ymi);ap(lis,x1,yma);C.create_line(lis,fill='black',dash=[1,2])
def raaknorm():
    global a,b,c,d,e,xr,vglR,vglN;vglR='';vglN='';teken()
    a,b,c,d,e=mult(E1.get());x=float(E3.get());xr=x
    if (b,d)==(0,0) or (a*b>0 and (k==0 or a*k<0)):ad('Geen raaklijnen');return
# staande parabool
    if b==0 and d!=0:
        y=-(a*x**2+c*x+e)/d;raakl(y);norml(y);ywaarden=format(y,nw)
# ellips of hyperbool of liggende parabool
    D=d**2-4*b*(a*x**2+c*x+e)
    if D>0 and b!=0:
        y=(-d-sqrt(D))/2/b;raakl(y);norml(y);ywaarden=format(y,nw)
        y=(-d+sqrt(D))/2/b;raakl(y);norml(y);ywaarden=ywaarden+','+format(y,nw)
    if D==0 and b!=0:
       y=-d/2/b;raakl(y);norml(y);ywaarden=format(y,nw)
    if D<0 and b!=0:ad('Geen raaklijnen');return
    ad('Raakpunten: x='+vc(xr)+'  y1='+ywaarden)
    ad('\nRaaklijnen: '+vglR+'\nNormalen:   '+vglN)
def norml(y):
    global vglN
    nm=2*a*xr+c;lis=[]
    if nm!=0:
        rico=(2*b*y+d)/nm
        ricoR=format(rico,nw)
        q=-rico*xr+y
        qR=format(q,nw)
        if q>0:qR='+'+qR
        vgl=ricoR+'*x'+qR
        x=xr-cos(atan(rico));y=rico*x+q;ap(lis,x,y)
        x=xr+cos(atan(rico));y=rico*x+q;ap(lis,x,y)
        vglN=vglN+'y='+vgl+'   '
    else:
        vglN=vglN+'x='+vc(xr)+'   '
        x=xr;ap(lis,x,y-1);ap(lis,x,y+1)
    C.create_line(lis,fill='magenta')
def raakl(y):
    global vglR
    nm=2*b*y+d;lis=[]
    if nm!=0:
        rico=(-2*a*xr-c)/nm
```

```
        ricoR=format(rico,nw)
        q=-rico*xr+y
        qR=format(q,nw)
        if q>0:qR='+'+qR
        vgl=ricoR+'*x'+qR
        x=xr-cos(atan(rico));y=rico*x+q;ap(lis,x,y)
        x=xr+cos(atan(rico));y=rico*x+q;ap(lis,x,y)
        vglR=vglR+'y='+vgl+'   '
    else:
        vglR=vglR+'x='+vc(xr)
        x=xr;y=-d/2/b-1;ap(lis,x,y);y=-d/2/b+1;ap(lis,x,y)
    C.create_line(lis,fill='darkgreen')
def ap(li,x,y):X=transx(x);Y=transy(y);li.append(X);li.append(Y)
def ad(lin):tex.insert(INSERT,lin+'')
def info():messagebox.showinfo('Kegelsneden, info:',helstr)
def wisentr(E):E.delete(0,len(E.get()))
def nieuw():
    for E in (E1,E2,E3):wisentr(E)
    C.delete(ALL);tex.delete('1.0',END)
# hoofdprogramma
alfa=chr(945);eps=chr(949);col='lightgrey'
breedte=480;hoogte=550;hoogteC=hoogte-185;hoInv=hoogte-30
form=Tk();form.geometry(str(breedte)+'x'+str(hoogte));form.resizable(False,False)
form.title('Kegelsneden: grafiek van ax²+by²+cx+dy+e=0, raaklijnen')
C = Canvas(form, bg="white", height=hoogteC, width=breedte);C.pack()
Label(form,text='a,b,c,d,e:').place(x=0,y=hoInv)
E1=Entry(form);E1.place(x=55,y=hoInv,width=105);E1.insert(0,'4,9,-16,-18,-11')
Label(form,text='XY-+:').place(x=155,y=hoInv)
E2=Entry(form);E2.place(x=190,y=hoInv,width=65);E2.insert(0,'-4,6,-4,4')
Label(form,text='x:').place(x=255,y=hoInv)
E3=Entry(form);E3.place(x=270,y=hoInv,width=25);E3.insert(0,'1')
tex=Text(form,bg='grey89',height=9,width=70,wrap=WORD);tex.place(x=2,y=hoogteC+5)
vsb=Scrollbar(form,orient=VERTICAL,command=tex.yview,width=25)
tex['yscrollcommand']=vsb.set;vsb.pack(side=RIGHT,fill='y')
B1=Button(form,text=' Info ',command=info);B1.place(x=300,y=hoInv,width=40)
B2=Button(form,text='Teken',command=teken);B2.place(x=340,y=hoInv,width=40)
B3=Button(form,text='Raakl',command=raaknorm);B3.place(x=380,y=hoInv,width=40)
B4=Button(form,text='Nieuw',command=nieuw);B4.place(x=420,y=hoInv,width=40)
helstr='Vul in: a,b,c,d,e. De grafiek van ax²+by²+cx+dy+e=0\n'
helstr=helstr+'wordt getekend. Is a of b=0, dan hebben we een parabool\n'
helstr=helstr+'Is a*b>0, dan hebben we soms een ellips, soms niets\n'
helstr=helstr+'Is a*b<0, dan is de grafiek een hyperbool\n'
helstr=helstr+"Met 'raakl' worden eventueel de raaklijnen\n"
helstr=helstr+'berekend en getekend voor de ingevoerde x\n'
form.mainloop()
```



78. Translatie en rotatie van een kegelsnede F: $ax^2+by^2+c=0$ kegelsnedentrans

1. Als a en b ≠ 0 en a,b en c niet tegelijkertijd positief of negatief zijn, dan is de grafiek van F: $ax^2+by^2+c=0$ een ellips (als a*b>0) of een hyperbool (als a*b<0) met symmetrieassen de x-as en y-as.
2. T: we verschuiven (translatie) deze kegelsnede van (0,0) naar (x0,y0)
3. R: We draaien de kegelsnede T over een hoek α in tegenwijzerzin.

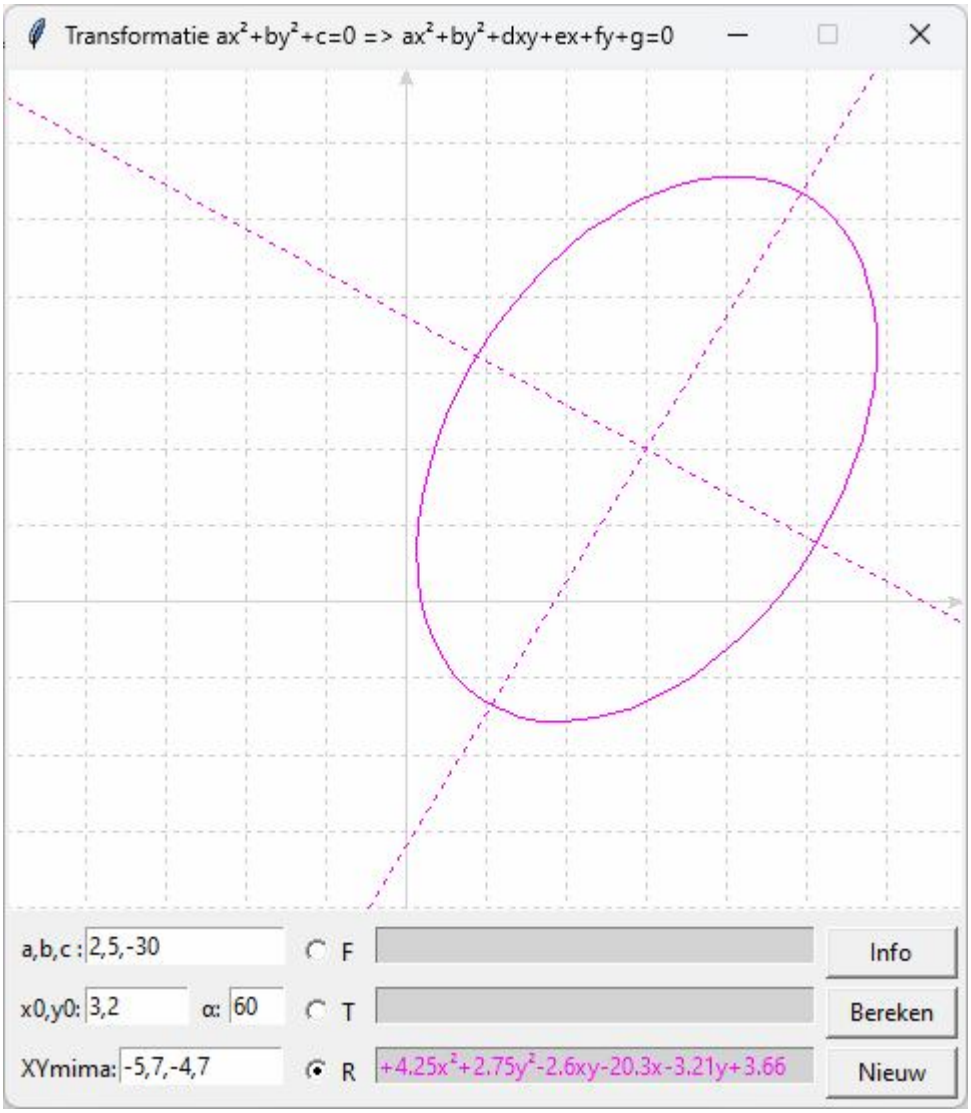
In dit programma worden de 3 kegelsneden getekend en de cartesische vergelijkingen berekend.
De vergelijking van T vinden we door $a(x-x0)^2+b(y-y0)^2+c=0$ uit te werken
De vergelijking van R vinden we door $a[(x-x0)\cdot\cos\alpha+(y-y0)\cdot\sin\alpha]^2+b[-(x-x0)\cdot\sin\alpha+(y-y0)\cdot\cos\alpha]^2+c=0$ uit te werken
Om een ellips te tekenen gebruiken we x1,y1=A*cos(t),B*sin(t)
Voor een hyperbool als b·c>0: x1,y1=A/cos(t),B*tan(t)
als b·c<0: y1,x1=A/cos(t),B*tan(t) (t∈[0,2π] , stap=0.1)
Op elk van deze punten moeten we voor T en R nog een transformatie toepassen
Voor T een translatie: x2,y2=x0+x1,y0+y1
Voor R een rotatie: x3,y3=x1*cos(α)-y1*sin(α),x1*sin(α)+y1*cos(α)
en een translatie: x4,y4=x0+x3,y0+y3

Programma

```
# transformatie ax²+by²+c=0 => ax²+by²+dxy+ex+fy+g=0
from math import *;from tkinter import *;from tkinter import messagebox
def mult(s):
   l=s.split(',');co=[]
   for i in l:co.append(float(i))
   return co
def transx(x):return (x-xmi)*breedte/(xma-xmi)
def transy(y):return hoogteC-hoogteC*(y-ymi)/(yma-ymi)
def bereken():
   global xmi,xma,ymi,yma,x0,y0,x1,y1,a,b,c,lis1,lis2,lis4,alfa
   C.delete(ALL);nwk='.7f'; xmi,xma,ymi,yma=mult(E2.get());lg='lightgrey'
   a,b,c=mult(E1.get());x0,y0=mult(E3.get());alfa=radians(float(E4.get()))
   for E in (E5,E6,E7):wis(E)
#assen
   X=transx(0);C.create_line(X,0,X,hoogteC,fill=gr,arrow='first')
   Y=transy(0);C.create_line(0,Y,breedte,Y,fill=gr,arrow='last')
# rooster
   for x in range(int(xmi),int(xma+1)):
      X=transx(x);C.create_line(X,0,X,hoogteC,fill=gr,dash=[1,2])
   for y in range(int(ymi),int(yma+1)):
      Y=transy(y);C.create_line(0,Y,breedte,Y,fill=gr,dash=[1,2] )
#krommen
   if a*b>0 and c*a<0:
      A=sqrt(-c/a);B=sqrt(-c/b)
      stap=0.1;t=-stap;lis1,lis2,lis4=[],[],[]
      while t<2*pi:
         t=t+stap;x1,y1=A*cos(t),B*sin(t)
         calcxy()
      teken()
   elif a*b<0 and b*c>0:
      A=sqrt(-c/a);B=sqrt(c/b)
      stap=0.1;t=-stap;lis1,lis2,lis4=[],[],[]
      while t<2*pi:
         t=t+stap;x1,y1=A/cos(t),B*tan(t)
         calcxy()
      teken()
   elif a*b<0 and b*c<0:
      A=sqrt(c/a);B=sqrt(-c/b)
      stap=0.1;t=-stap;lis1,lis2,lis4=[],[],[]
      while t<2*pi:
         t=t+stap;y1,x1=A/cos(t),B*tan(t)
         calcxy()
      teken()
   else:messagebox.showinfo('Uitzondering','Geen ellips of hyperbool')
def stipp1(x0,y0,col):
   lis=[];ap(lis,xmi,y0);ap(lis,xma,y0);C.create_line(lis,dash=[1,2],fill=col)
   lis=[];ap(lis,x0,ymi);ap(lis,x0,yma);C.create_line(lis,dash=[1,2],fill=col)
def teken():
# grafieken lis1,lis2,lis4 en vergelijkingen verg1, verg2 en verg4
```

```
    if k==0:
        C.create_line(lis1,fill=bl )
        verg1=vc(a)+'x²'+vc(b)+'y²'+vc(c);wis(E5);E5.insert(0,verg1)
        stipp1(0,0,bl)
    if k==1:
        C.create_line(lis2,fill=re )
        a2,b2,c2,d2,e2=a,b,-2*a*x0,-2*b*y0,a*x0**2+b*y0**2+c
        verg2=vc(a2)+'x²'+vc(b2)+'y²'+vc(c2)+'x'+vc(d2)+'y'+vc(e2);wis(E6);E6.insert(0,verg2)
        stipp1(x0,y0,re)
    if k==2:
        C.create_line(lis4,fill=ma )
        p=-x0*cos(alfa)-y0*sin(alfa);q=x0*sin(alfa)-y0*cos(alfa)
        a4=a*cos(alfa)**2+b*sin(alfa)**2;b4=b*cos(alfa)**2+a*sin(alfa)**2
        c4=2*(a-b)*cos(alfa)*sin(alfa);d4=2*(a*p*cos(alfa)-b*q*sin(alfa))
        e4=2*(a*p*sin(alfa)+b*q*cos(alfa));f4=a*p*p+b*q*q+c
        verg4=vc(a4)+'x²'+vc(b4)+'y²'+vc(c4)+'xy'+vc(d4)+'x'+vc(e4)+'y'+vc(f4);wis(E7);E7.insert(0,verg4)
        q=2*alfa/pi
        if q==int(q):stipp1(x0,y0,ma)
        else: #stipp2
            x1=xmi;y1=y0+(x1-x0)*tan(alfa);x2=xma;y2=y0+(x2-x0)*tan(alfa)
            lis=[];ap(lis,x1,y1);ap(lis,x2,y2);C.create_line(lis,dash=[1,2],fill=ma)
            x1=xmi;y1=y0+(x1-x0)*tan(alfa+pi/2);x2=xma;y2=y0+(x2-x0)*tan(alfa+pi/2)
            lis=[];ap(lis,x1,y1);ap(lis,x2,y2);C.create_line(lis,dash=[1,2],fill=ma)
def calcxy():
    global x2,y2,x4,y4
    x2,y2=x0+x1,y0+y1
    x3,y3=x1*cos(alfa)-y1*sin(alfa),x1*sin(alfa)+y1*cos(alfa)
    x4,y4=x0+x3,y0+y3
    ap(lis1,x1,y1);ap(lis2,x2,y2);ap(lis4,x4,y4)
def vc(x):
    if x==int(x):x=int(x)
    else: x=round(x,2)
    if x>=0:s='+'+str(x)
    else:s=str(x)
    return s
def selec():global k;k=var.get()
def info():messagebox.showinfo(tit,infstr)
def ap(li,x,y):X=transx(x);Y=transy(y);li.append(X);li.append(Y)
def wis(E):E.delete(0,len(E.get()))
def nieuw():
    for E in (E1,E2,E3,E4,E5,E6,E7):wis(E)
    C.delete(ALL)
#Hoofdprogramma
tit='Transformatie ax²+by²+c=0 => ax²+by²+dxy+ex+fy+g=0'
gr='lightgrey';bl='blue';re='red';ma='magenta';dx=1E-6
breedte=480;hoogte=520;hoogteC=hoogte-100;hoInv1=hoogte-90;hoInv2=hoogte-60;hoInv3=hoogte-30
form=Tk();form.geometry(str(breedte)+'x'+str(hoogte))
form.title(tit);form.resizable(False,False)
C = Canvas(form, bg="white", height=hoogteC, width=breedte);C.pack()
L1=Label(form,text='a,b,c :',);L1.place(x=5,y=hoInv1)
E1=Entry(form,width=16);E1.place(x=40,y=hoInv1);E1.insert(0,'2,5,-30')
```

```
L2=Label(form,text='XYmima:',);L2.place(x=5,y=hoInv3)
E2=Entry(form,width=13);E2.place(x=57,y=hoInv3);E2.insert(0,'-5,7,-4,7')
L3=Label(form,text='x0,y0:',);L3.place(x=5,y=hoInv2)
E3=Entry(form,width=8);E3.place(x=40,y=hoInv2);E3.insert(0,'3,2')
L4=Label(form,text='α:',);L4.place(x=95,y=hoInv2)
E4=Entry(form,width=4);E4.place(x=112,y=hoInv2);E4.insert(0,'60')
E5=Entry(form,bg=gr,fg=bl,width=36);E5.place(x=185,y=hoInv1)
E6=Entry(form,bg=gr,fg=re,width=36);E6.place(x=185,y=hoInv2)
E7=Entry(form,bg=gr,fg=ma,width=36);E7.place(x=185,y=hoInv3)
var=IntVar();k=0;keuze=['F','T','R']
for i in range(0,3):
   rb=Radiobutton(form,text=keuze[i],variable=var,value=i,command=selec)
   rb.place(x=145,y=hoogte-90+i*30)
Button(form,text='Info',command=info,width=8).place(x=410,y=hoInv1)
Button(form,text='Bereken',command=bereken,width=8).place(x=410,y=hoInv2)
Button(form,text='Nieuw',command=nieuw,width=8).place(x=410,y=hoInv3)
infstr='Vul in: a, b, c, x0, y0, α en kies F T R\n'
infstr=infstr+'F: de kegelsnede ax²+bx+c=0\n'
infstr=infstr+'T: translatie van F: (0,0) => (x0,y0)\n'
infstr=infstr+'R: rotatie van T over hoek α \n'
infstr=infstr+'De grafiek is een ellips of een hyperbool of niets\n'
infstr=infstr+'De kegelsnede wordt door middel van een translatie\n'
infstr=infstr+'naar (x0,y0) en een rotatie over een hoek α\n'
infstr=infstr+'getransformeerd naar ax²+by²+dxy+ex+fy+g=0\n'
form.mainloop()
```



79. Integraalberekening met Simpson + grafiek simpson_grafTK

Om het gebied tussen de x-as en een kromme y=f(x) in een interval [a,b] te arceren (bvb. in het geel) kunnen we van a tot b dicht bij elkaar verticale lijnstukjes tekenen die beginnen op de x-as (y=0) en eindigen op de kromme (y=f(x)). We verbinden dus (x,0) met (x,f(x)) voor $x \in [a,b]$. Dit kunnen we doen met het volgende stukje programma:

```
x=a;stap=0.01
while x<=b:
    lis=[];y=0;X=transx(x);Y=transy(y);lis.append(X);lis.append(Y);
    y=f(x);X=transx(x);Y=transy(y);lis.append(X);lis.append(Y);x=x+stap
    C.create_line(lis,fill='yellow')
```

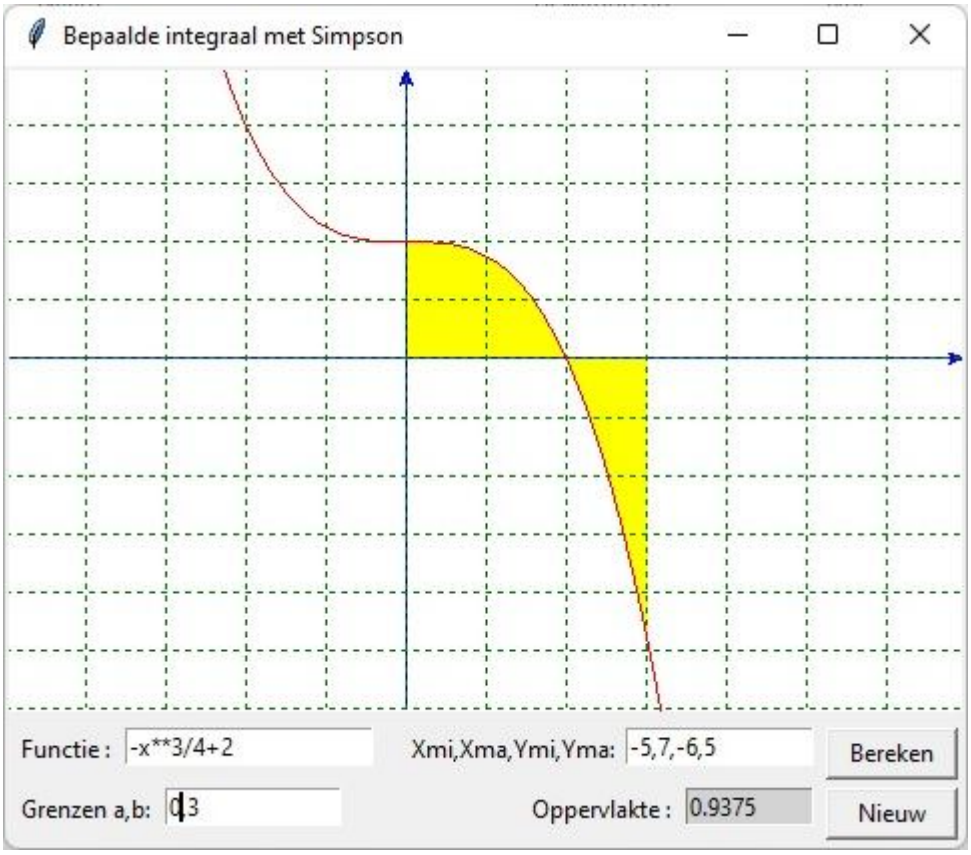
Met het volgende programma wordt er, naast de berekening van de oppervlakte ,ook een grafiek getekend van deze oppervlakte.

Programma:

```
# grafieken met tkinter + oppervlakteberekening met Simpson
from math import *;from tkinter import *
def mult(s):
```

```
    l=s.split(',');co=[]
    for i in l:co.append(float(i))
    return co
def transx(x):return (x-xmi)*breedte/(xma-xmi)
def transy(y):return hoogteC-hoogteC*(y-ymi)/(yma-ymi)
def f(x):return eval(E1.get())
def integr(a,b,n):
    h=(b-a)/n;x=a;som=0
    for j in range(1,n):
        x=x+h
        if j%2==0:t=2
        else:t=4
        som=som+t*f(x)
    som=som+f(a)+f(b)
    integ=som*h/3
    return(integ)
def bereken():
    global xmi,xma,ymi,yma,a,b,n
    xmi,xma,ymi,yma=mult(E2.get());C.delete(ALL)
    a,b=mult(E3.get());n=50
    opp=round(integr(a,b,n),5)
    wis(E4);E4.insert(0,str(opp))
# arcering
    x=a;stap=0.01
    while x<=b:
        lis=[];y=0;X=transx(x);Y=transy(y);lis.append(X);lis.append(Y);
        y=f(x);X=transx(x);Y=transy(y);lis.append(X);lis.append(Y);x=x+stap
        C.create_line(lis,fill='yellow' )
#assen
    X=transx(0);C.create_line(X,0,X,hoogteC,fill='blue',arrow='first')
    Y=transy(0);C.create_line(0,Y,breedte,Y,fill='blue',arrow='last')
# rooster
    for x in range(int(xmi),int(xma+1)):
        X=transx(x);C.create_line(X,0,X,hoogteC,fill='green',dash=[1,2])
    for y in range(int(ymi),int(yma+1)):
        Y=transy(y);C.create_line(0,Y,breedte,Y,fill='green',dash=[1,2])
#kromme
    lis=[]
    stap=0.05;x=xmi
    while x<xma:
        x=x+stap;y=f(x);X=transx(x);Y=transy(y)
        lis.append(X);lis.append(Y)
    C.create_line(lis,fill='red')
def wis(E):E.delete(0,len(E.get()))
def nieuw():wis(E1);wis(E2);wis(E3);wis(E4);C.delete(ALL)
#Hoofdprogramma
breedte=480;hoogte=390;hoogteC=hoogte-70;hoInv1=hoogte-60;hoInv2=hoogte-30
form=Tk();form.geometry(str(breedte)+'x'+str(hoogte));form.title('Bepaalde integraal met Simpson')
C = Canvas(form, bg="white", height=hoogteC, width=breedte);C.pack()
L1=Label(form,text='Functie :',);L1.place(x=5,y=hoInv1)
E1=Entry(form,width=20);E1.place(x=60,y=hoInv1);E1.insert(0,'-x**2+2*x+3')
L2=Label(form,text='Xmi,Xma,Ymi,Yma:',);L2.place(x=200,y=hoInv1)
E2=Entry(form,width=15);E2.place(x=310,y=hoInv1);E2.insert(0,'-5,7,-3,5')
L3=Label(form,text='Grenzen a,b:',);L3.place(x=5,y=hoInv2)
E3=Entry(form,width=14);E3.place(x=80,y=hoInv2);E3.insert(0,'1,3')
```

```
L4=Label(form,text='Oppervlakte :',);L4.place(x=260,y=hoInv2)
E4=Entry(form,bg='lightgrey',width=10);E4.place(x=340,y=hoInv2)
B1=Button(form,text='Bereken',command=bereken,width=8);B1.place(x=410,y=hoInv1)
B2=Button(form,text='Nieuw',command=nieuw,width=8);B2.place(x=410,y=hoInv2)
form.mainloop()
```



80. Trapezium integratie met grafiek trapezium_grafTK

Om in een interval [a,b], n rechthoekjes te tekenen met basis op de x-as, met breedte h en een hoogte die gegeven wordt als een functie van x , moeten we telkens 3 lijnstukjes tekenen.
Als die functie bvb. de gemiddelde functiewaarde is in de eindpunten van het interval, dan is de hoogtefunctie = ((f(x)+f(x+h))/2. De 3 lijnstukjes zijn dan
(x,0) -- (x, ((f(x)+f(x+h))/2) --(x+h, ((f(x)+f(x+h))/2) -- (x+h,0)
Hiervoor volstaat de code:

```
  x=a; h=(b-a)/n
    for i in range(0,n):
        lis=[]
        X=transx(x);Y=transy(0);lis.append(X);lis.append(Y)
        Y=transy((f(x)+f(x+h))/2);lis.append(X);lis.append(Y)
        x=x+h;X=transx(x);lis.append(X);lis.append(Y)
        Y=transy(0);lis.append(X);lis.append(Y)
        C.create_line(lis,fill='green' )
```

Opmerking: de vorige code laat ons toe om de trapeziumregel grafisch voor te stellen.
Willen we ondersommen, respectievelijk bovensommen grafisch voorstellen, dan nemen we als hoogtefunctie $\min(f(x))_{x,x+h}$ respect. $\max(f(x))_{x,x+h}$
We kunnen dezelfde code ook gebruiken om het histogram van een steekproef te tekenen.

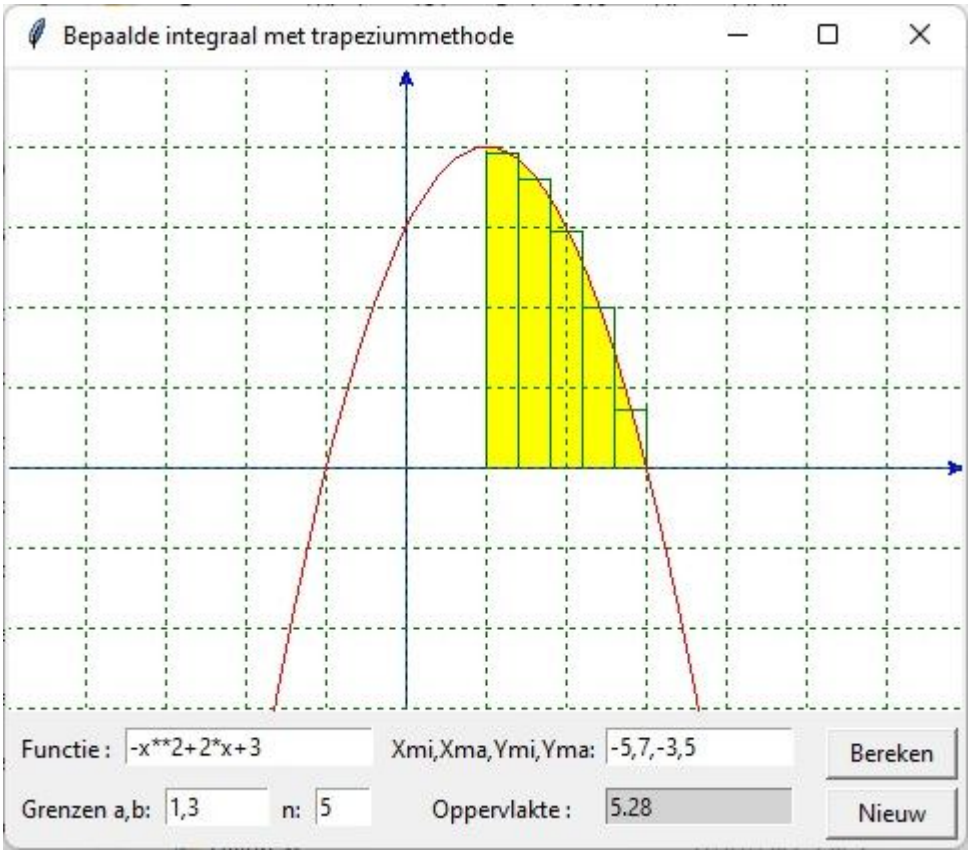
Maar dan is h de klassebreedte, en de hoogte=klassefrequentie

Behalve de berekening worden per interval geen trapeziums maar wel rechthoekjes getekend, waarvan de hoogte= gemiddelde van de functiewaarde in de eindpunten

Programma:

```
# grafieken met tkinter + oppervlakteberekening met trapeziummethode
from math import *;from tkinter import *
def mult(s):
   l=s.split(',');co=[]
   for i in l:co.append(float(i))
   return co
def transx(x):return (x-xmi)*breedte/(xma-xmi)
def transy(y):return hoogteC-hoogteC*(y-ymi)/(yma-ymi)
def f(x):return eval(E1.get())
def integr(a,b,n):
   #h=(b-a)/n
   x=a
   som=0
   for j in range(1,n):
      x=x+h
      som=som+f(x)
   som=som+(f(a)+f(b))/2
   integ=som*h
   return(integ)
def bereken():
   global xmi,xma,ymi,yma,a,b,n,h
   xmi,xma,ymi,yma=mult(E2.get());C.delete(ALL)
   a,b=mult(E3.get());n=int(E4.get());h=(b-a)/n
   opp=round(integr(a,b,n),5)
   wis(E5);E5.insert(0,str(opp))
# arcering
   x=a;stap=0.01
   while x<=b:
      lis=[];y=0;X=transx(x);Y=transy(y);lis.append(X);lis.append(Y);
      y=f(x);X=transx(x);Y=transy(y);lis.append(X);lis.append(Y);x=x+stap
      line = C.create_line(lis,fill='yellow' )
#assen
   X=transx(0);line = C.create_line(X,0,X,hoogteC,fill='blue',arrow='first')
   Y=transy(0);line = C.create_line(0,Y,breedte,Y,fill='blue',arrow='last')
# rooster
   for x in range(int(xmi),int(xma+1)):
      X=transx(x);line = C.create_line(X,0,X,hoogteC,fill='green',dash=[1,2])
   for y in range(int(ymi),int(yma+1)):
      Y=transy(y);line = C.create_line(0,Y,breedte,Y,fill='green',dash=[1,2] )
#kromme
   lis=[]
   stap=0.05;x=xmi
   while x<xma:
      x=x+stap;y=f(x);X=transx(x);Y=transy(y)
      lis.append(X);lis.append(Y)
   line = C.create_line(lis,fill='red')
#rechthoekjes
   x=a
```

```
    for i in range(0,n):
        lis=[]
        X=transx(x);Y=transy(0);lis.append(X);lis.append(Y)
        Y=transy((f(x)+f(x+h))/2);lis.append(X);lis.append(Y)
        x=x+h;X=transx(x);lis.append(X);lis.append(Y)
        Y=transy(0);lis.append(X);lis.append(Y)
        C.create_line(lis,fill='green' )
def wis(E):E.delete(0,len(E.get()))
def nieuw():wis(E1);wis(E2);wis(E3);wis(E4);wis(E5);C.delete(ALL)
#Hoofdprogramma
breedte=480;hoogte=390;hoogteC=hoogte-70;hoInv1=hoogte-60;hoInv2=hoogte-30
form=Tk();form.geometry(str(breedte)+'x'+str(hoogte))
form.title('Bepaalde integraal met trapeziummethode')
C = Canvas(form, bg="white", height=hoogteC, width=breedte);C.pack()
L1=Label(form,text='Functie :',);L1.place(x=5,y=hoInv1)
E1=Entry(form,width=20);E1.place(x=60,y=hoInv1);E1.insert(0,'-x**2+2*x+3')
L2=Label(form,text='Xmi,Xma,Ymi,Yma:',);L2.place(x=190,y=hoInv1)
E2=Entry(form,width=15);E2.place(x=300,y=hoInv1);E2.insert(0,'-5,7,-3,5')
L3=Label(form,text='Grenzen a,b:',);L3.place(x=5,y=hoInv2)
E3=Entry(form,width=8);E3.place(x=80,y=hoInv2);E3.insert(0,'1,3')
L4=Label(form,text='n:',);L4.place(x=135,y=hoInv2)
E4=Entry(form,width=4);E4.place(x=155,y=hoInv2);E4.insert(0,'5')
L5=Label(form,text='Oppervlakte :',);L5.place(x=210,y=hoInv2)
E5=Entry(form,bg='lightgrey',width=15);E5.place(x=300,y=hoInv2)
B1=Button(form,text='Bereken',command=bereken,width=8);B1.place(x=410,y=hoInv1)
B2=Button(form,text='Nieuw',command=nieuw,width=8);B2.place(x=410,y=hoInv2)
form.mainloop()
```



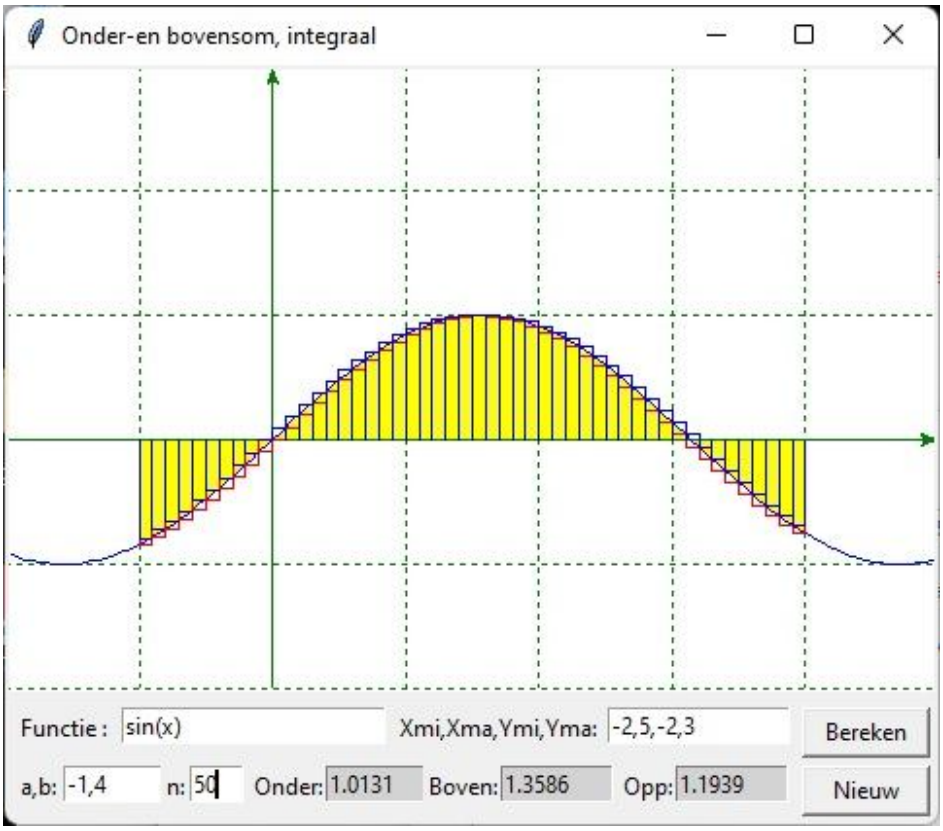
81. Onder- en bovensommen - grafiek onder_boven_integrTK

De bepaalde integraal van een functie f(x) in een interval [a,b] is de oppervlakte begrensd door de kromme y=f(x) en de x-as in dit interval. Verdelen we [a,b] in n intervallen met breedte h, dan is n.h=b-a. In elk van deze intervallen kunnen we de minimale en maximale waarde van f(x) bepalen. Tellen we alle minimale, respect. maximale waarden op en vermenigvuldigen we met h, dan krijgen we 2 oppervlaktes, die we onder- en bovensom noemen. De bepaalde integraal is begrepen tussen deze 2 oppervlaktes. In het programma moet je f(x), a,b en n invullen:
Onder en bovensom alsook de oppervlakte onder de kromme worden getekend en berekend.

Programma:

```
# grafieken met tkinter + oppervlakteberekening met Simpson + onder- en bovensommen
from math import *;from tkinter import *
def mult(s):
   l=s.split(',');co=[]
   for i in l:co.append(float(i))
   return co
def transx(x):return (x-xmi)*breedte/(xma-xmi)
def transy(y):return hoogteC-hoogteC*(y-ymi)/(yma-ymi)
def tr(x,y):return transx(x),transy(y)
def ap(l):l.append(X);l.append(Y)
def f(x):return eval(E1.get())
def mimaf(i):
   mi_i=1E8;ma_i=-1E8;stap=1E-2;xo=a+i*h;x=xo
   while x<=xo+h:
      if f(x)<mi_i:mi_i=f(x)
      if f(x)>ma_i:ma_i=f(x)
      x=x+stap
   return mi_i,ma_i
def maxf(x,h):
   ma=-1E8;stap=1E-3;xo=x
def ond_bov(a,b,n):
   som_ond=0;som_bov=0;h=(b-a)/n;x=a-h
   for j in range(0,n):
      x=x+h;som_ond=som_ond+mi[j];som_bov=som_bov+ma[j]
   return som_ond*h,som_bov*h
def integr(a,b,ni): # met Simpson
   h=(b-a)/nSim;x=a;som=0
   for j in range(1,nSim):
      x=x+h
      if j%2==0:t=2
      else:t=4
      som=som+t*f(x)
   som=som+f(a)+f(b);integ=som*h/3
   return(integ)
def bereken():
   global xmi,xma,ymi,yma,a,b,h,nSim,n,mi,ma,X,Y
   mi,ma=[],[];xmi,xma,ymi,yma=mult(E2.get());C.delete(ALL)
   a,b=mult(E3.get());nSim=200;n=int(E4.get());h=(b-a)/n
   for i in range(0,n):oi,bi=mimaf(i);mi.append(oi);ma.append(bi)
   ond,bov=ond_bov(a,b,n);ond=format(ond,'.4f');bov=format(bov,'.4f');opp=format(integr(a,b,nSim),'.4f')
   wis(E5);wis(E6);wis(E7);E5.insert(0,ond);E6.insert(0,bov);E7.insert(0,opp)
# arcering
   x=a;stap=0.01
```

```
    while x<=b:
        lis=[];y=0;X=transx(x);Y=transy(y);lis.append(X);lis.append(Y);
        y=f(x);X=transx(x);Y=transy(y);lis.append(X);lis.append(Y);x=x+stap
        line = C.create_line(lis,fill='yellow' )
# assen
    X=transx(0);C.create_line(X,0,X,hoogteC,fill='green',arrow='first')
    Y=transy(0);C.create_line(0,Y,breedte,Y,fill='green',arrow='last')
# rooster
    for x in range(int(xmi),int(xma+1)):
        X=transx(x);line = C.create_line(X,0,X,hoogteC,fill='green',dash=[1,2])
    for y in range(int(ymi),int(yma+1)):
        Y=transy(y);C.create_line(0,Y,breedte,Y,fill='green',dash=[1,2] )
# kromme
    lis=[];stap=0.05;x=xmi
    while x<xma:x=x+stap;y=f(x);X,Y=tr(x,y);ap(lis)
    C.create_line(lis,fill='blue')
# rechthoekjes
    x=a;h=(b-a)/n
    for i in range(0,n):
        lisx=[];lisy=[]
        X,Y=tr(x,0);ap(lisx);X,Y=tr(x,mi[i]);ap(lisx);X,Y=tr(x+h,mi[i]);ap(lisx);X,Y=tr(x+h,0);ap(lisx)
        X,Y=tr(x,0);ap(lisy);X,Y=tr(x,ma[i]);ap(lisy);X,Y=tr(x+h,ma[i]);ap(lisy);X,Y=tr(x+h,0);ap(lisy)
x=x+h;C.create_line(lisx,fill='red');C.create_line(lisy,fill='blue')
def wis(E):E.delete(0,len(E.get()))
def nieuw():
    for E in [E1,E2,E3,E4,E5]:wis(E)
    C.delete(ALL)
#Hoofdprogramma
breedte=480;hoogte=390;hoogteC=hoogte-70;hoInv1=hoogte-60;hoInv2=hoogte-30;lg='lightgrey'
form=Tk();form.geometry(str(breedte)+'x'+str(hoogte));form.title('Onder-en bovensom, integraal')
C = Canvas(form, bg="white", height=hoogteC, width=breedte);C.pack()
Label(form,text='Functie :').place(x=5,y=hoInv1)
E1=Entry(form,width=22);E1.place(x=60,y=hoInv1);E1.insert(0,'sin(x)')
Label(form,text='Xmi,Xma,Ymi,Yma:').place(x=200,y=hoInv1)
E2=Entry(form,width=15);E2.place(x=310,y=hoInv1);E2.insert(0,'-2,5,-2,3')
Label(form,text='a,b:').place(x=5,y=hoInv2)
E3=Entry(form,width=8);E3.place(x=30,y=hoInv2);E3.insert(0,'-1,4')
Label(form,text='n:',).place(x=80,y=hoInv2)
E4=Entry(form,width=4);E4.place(x=95,y=hoInv2);E4.insert(0,'5')
Label(form,text='Onder:').place(x=125,y=hoInv2)
E5=Entry(form,bg=lg,width=9);E5.place(x=165,y=hoInv2)
Label(form,text='Boven:').place(x=215,y=hoInv2)
E6=Entry(form,bg=lg,width=9);E6.place(x=255,y=hoInv2)
Label(form,text='Opp:').place(x=315,y=hoInv2)
E7=Entry(form,bg=lg,width=9);E7.place(x=345,y=hoInv2)
B1=Button(form,text='Bereken',command=bereken,width=8);B1.place(x=410,y=hoInv1)
B2=Button(form,text='Nieuw',command=nieuw,width=8);B2.place(x=410,y=hoInv2)
form.mainloop()
```



82. Middelwaardestelling van de integralen middelwaardestelling_integralen

Ook nu is f(x) een continue functie op [a,b].

De oppervlakte onder de kromme tussen a en b = $\int_a^b f(x) \cdot dx$

Dan $\exists$ c $\in$]a , b[zodat $\int_a^b f(x) \cdot dx = f(c) \cdot (b - a)$ = opp.rechthoek met breedte b-a en hoogte f(c).

Met ' $\int_a^b f(x) \cdot dx$ ' wordt de oppervlakte onder de kromme getekend en berekend.

Met ' f(c).(b-a) ' wordt de oppervlakte van de rechthoek getekend en berekend.
Ook nu kan je c wijzigen om te zien of de laatste oppervlakte de eerste benadert.
Wil je de exacte waarde van c kennen, tik dan op 'c='

Als opp_kr = $\int_a^b f(x)dx$, dan wordt deze c-waarde in het programma berekend door met de methode van

Newton een nulpunt te berekenen van de functie f (x) - $\frac{opp\_kr}{b - a}$

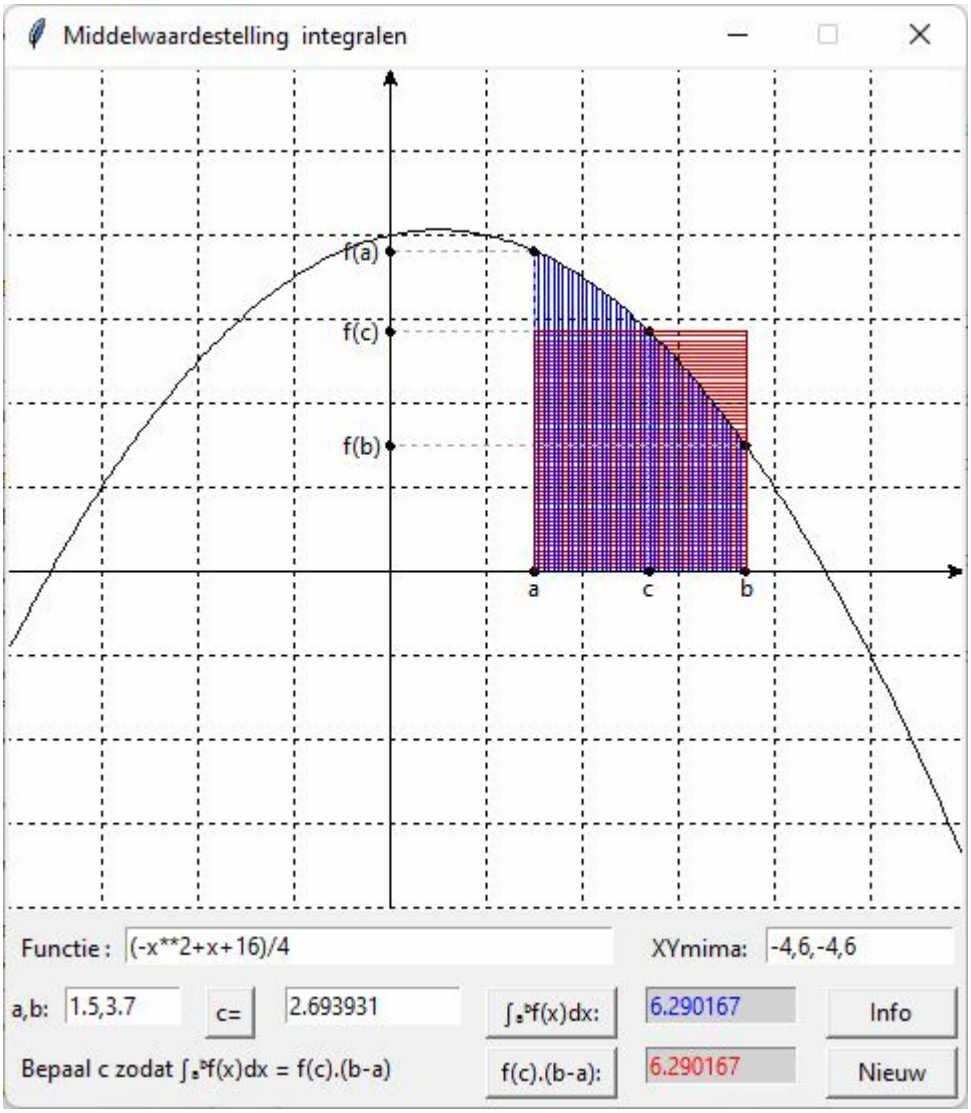
Programma:

```
# middelwaardestelling integralen
from math import *;from tkinter import *;from tkinter import messagebox
def mult(s):
   l=s.split(',');co=[]
   for i in l:co.append(float(i))
   return co
def transx(x):return (x-xmi)*breedte/(xma-xmi)
```

```
def transy(y):return hoogteC-hoogteC*(y-ymi)/(yma-ymi)
def f(x):return eval(E1.get())
def df(x):return (f(x+dx)-f(x-dx))/dx/2
def integr(a,b): # integraalberekening met simpson
    n=100;h=(b-a)/n;x=a;som=0
    for j in range(1,n):
        x=x+h
        if j%2==0:t=2
        else:t=4
        som=som+t*f(x)
    som=som+f(a)+f(b);integ=som*h/3
    return integ
def bereken():
    global xmi,xma,ymi,yma,a,b,c,n,h
    C.delete(ALL)
    xmi,xma,ymi,yma=mult(E2.get())
    a,b=mult(E3.get());c=float(E4.get());
    wis(E6);E6.insert(0,str(b-a))
    wis(E5);E5.insert(0,format(integr(a,b),nwk))
    func();kromme()
# arcering functie
def func():
    x=a;stap=0.05
    while x<=b:
        lis=[];ap(lis,x,0);ap(lis,x,f(x));x=x+stap
        C.create_line(lis,fill='blue' )
def opp1():
    global xmi,xma,ymi,yma,a,b,c,n,h
    xmi,xma,ymi,yma=mult(E2.get())
    a,b=mult(E3.get());c=float(E4.get());
    wis(E7);E7.insert(0,format((b-a)*f(c),nwk))
    C.delete(ALL);y=0;stap=0.05
    if f(c)>0:stap=0.05
    else:stap=-0.05
    while abs(y)<=abs(f(c)):
        lis=[];ap(lis,a,y);ap(lis,b,y);y=y+stap
        C.create_line(lis,fill='red' )
    func();kromme()
    lis=[];ap(lis,a,0);ap(lis,a,f(c));ap(lis,b,f(c));ap(lis,b,0)
    C.create_line(lis,fill='darkred')
def newton():
    x=(a+b)/2;opp_kr=float(E5.get())
    xo=x+1
    while abs(x-xo)>eps:xo=x;x=x+(opp_kr/(b-a)-f(x))/df(x)
    wis(E4);E4.insert(0,format(x,nwk))
    def kromme():
#assen
    X=transx(0);C.create_line(X,0,X,hoogteC,fill='black',arrow='first')
    Y=transy(0);C.create_line(0,Y,breedte,Y,fill='black',arrow='last')
# rooster
    for x in range(int(xmi),int(xma+1)):
```

```
        X=transx(x);C.create_line(X,0,X,hoogteC,fill='black',dash=[1,2])
    for y in range(int(ymi),int(yma+1)):
        Y=transy(y);C.create_line(0,Y,breedte,Y,fill='black',dash=[1,2] )
#kromme
    lis=[];stap=0.05;x=xmi
    while x<xma:
        x=x+stap;y=f(x);X=transx(x);Y=transy(y)
        lis.append(X);lis.append(Y)
    C.create_line(lis,fill='black')
#stippels
    for p in (a,b,c):
        lis=[];ap(lis,p,0);ap(lis,p,f(p));ap(lis,0,f(p))
        C.create_line(lis,dash=[1,2],fill='darkgrey')
#punten
    for p in (a,b,c):rondje(p,f(p));rondje(p,0);rondje(0,f(p))
#tekst
    for ps in('a','b','c'):naamX(eval(ps),ps);psy='f('+ps+')';naamY(eval(psy),psy)
def naamX(p,pn):lis=[];ap(lis,p,-0.2);C.create_text(lis,text=pn)
def naamY(p,pn):lis=[];ap(lis,-0.3,p);C.create_text(lis,text=pn)
def rondje(a,b):lis=[];r=0.05;ap(lis,a-r,b-r);ap(lis,a+r,b+r);C.create_oval(lis,fill='black')
def info():h=messagebox.showinfo(tit,infstr)
def ap(li,x,y):X=transx(x);Y=transy(y);li.append(X);li.append(Y)
def wis(E):E.delete(0,len(E.get()))
def nieuw():
    for E in (E1,E2,E3,E4,E5,E7):wis(E)
    C.delete(ALL)
#Hoofdprogramma
tit='Middelwaardestelling  integralen';gr='lightgrey';eps=1E-7;dx=1E-6;nwk='.6f'
breedte=480;hoogte=520;hoogteC=hoogte-100;hoInv1=hoogte-90;hoInv2=hoogte-60;hoInv3=hoogte-30
form=Tk();form.geometry(str(breedte)+'x'+str(hoogte));form.title(tit);form.resizable(False,False)
C = Canvas(form, bg="white", height=hoogteC, width=breedte);C.pack()
Label(form,text='Functie :').place(x=5,y=hoInv1)
E1=Entry(form,width=40);E1.place(x=60,y=hoInv1);E1.insert(0,'(-x**2+x+16)/4')
Label(form,text='XYmima:').place(x=320,y=hoInv1)
E2=Entry(form,width=15);E2.place(x=380,y=hoInv1);E2.insert(0,'-4,6,-4,6')
Label(form,text='a,b:').place(x=0,y=hoInv2)
E3=Entry(form,width=9);E3.place(x=30,y=hoInv2);E3.insert(0,'1.5,3.7')
Button(form,text='c=',command=newton).place(x=100,y=hoInv2)
E4=Entry(form,width=14);E4.place(x=140,y=hoInv2);E4.insert(0,'3')
Button(form,text='f(c).(b-a):',command=opp1,width=8).place(x=240,y=hoInv3)
Label(form,text="Bepaal c zodat ∫ab f(x)dx = f(c).(b-a)").place(x=5,y=hoInv3)
E5=Entry(form,bg=gr,fg='blue',width=12);E5.place(x=320,y=hoInv2)
E7=Entry(form,bg=gr,fg='red',width=12);E7.place(x=320,y=hoInv3)
Button(form,text='∫ab  f(x)dx:',command=bereken,width=8).place(x=240,y=hoInv2)
Button(form,text='Info',command=info,width=8).place(x=410,y=hoInv2)
Button(form,text='Nieuw',command=nieuw,width=8).place(x=410,y=hoInv3)
infstr='∫abf(x)dx = oppervlakte onder de \nkromme f(x) tussen a en b\n'
infstr=infstr+'Bepaal eerst deze oppervlakte\n'
infstr=infstr+'Probeer dan om zelf een c\n'
infstr=infstr+'in te vullen waarvoor geldt\n'
infstr=infstr+'∫ab  f(x)dx= f(c).(b-a). Zodat opp.\n'
```

```
infstr=infstr+'onder de kromme =opp. rechthoek.\n'
infstr=infstr+"Met de knop 'c=' kan je uiteindelijk\n"
infstr=infstr+"de exacte waarde van c berekenen."
form.mainloop()
```



83. Hoofdstelling van de integraalrekening hoofdstelling_integraal

Stel F(x) de oppervlakte onder de grafiek van y=f(x) is, tussen de grenzen [a,x] .
Dan is F(x+dx) - F(x) deze oppervlakte tussen [x,x+dx]. en f(x).dx de oppervlakte van de rechthoek tussen [x,x+dx] .
Met dit programma zie je dat [F(x+dx) - F(x)]/dx nadert tot f(x) als dx→0.

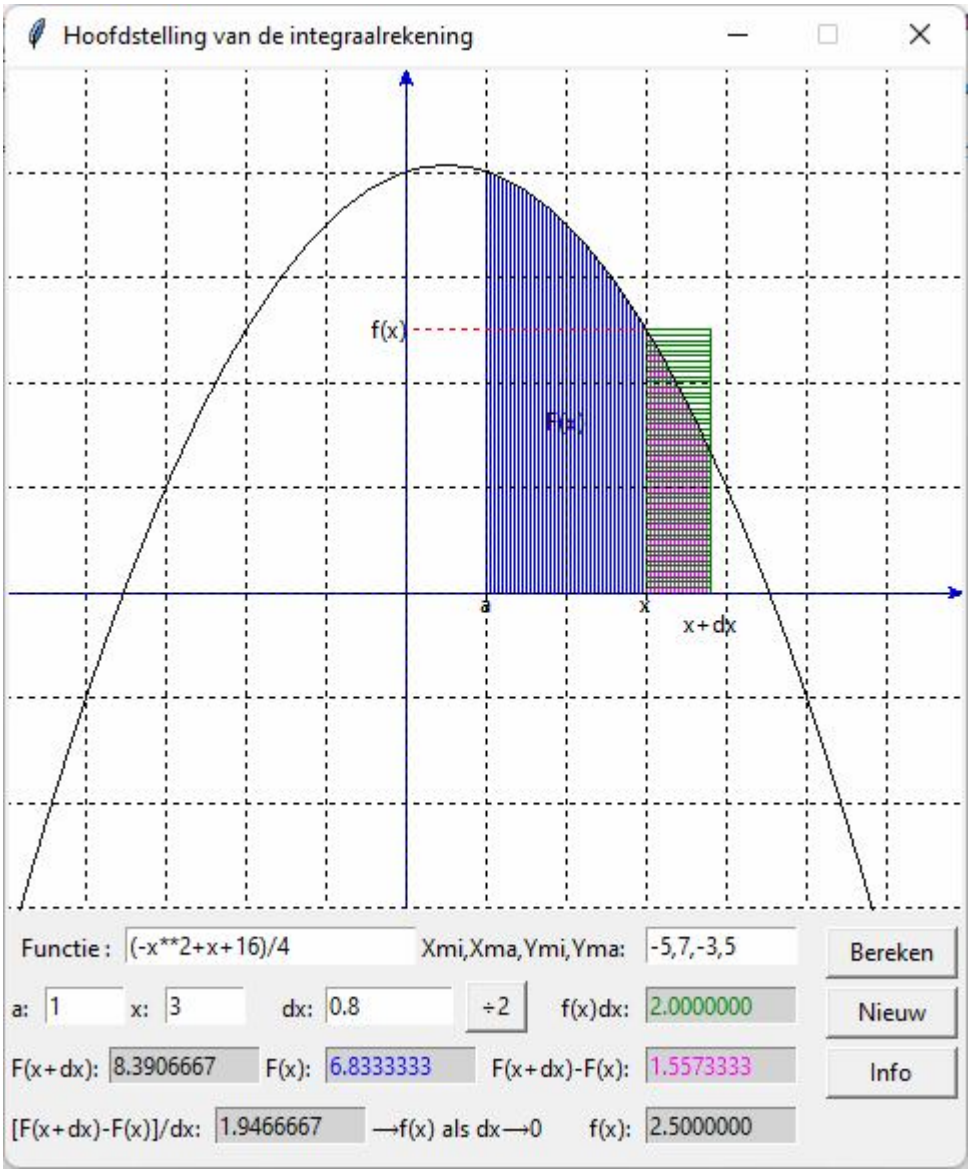
Zodat geldt: $\lim_{dx\to 0} \frac{F(x+dx)-F(x)}{dx} = f(x)$

Programma:

```
# hoofdstelling van de integraalrekening
from math import *;from tkinter import *;from tkinter import messagebox
def mult(s):
   l=s.split(',');co=[]
   for i in l:co.append(float(i))
   return co
def transx(x):return (x-xmi)*breedte/(xma-xmi)
def transy(y):return hoogteC-hoogteC*(y-ymi)/(yma-ymi)
def f(x):return eval(E1.get())
def integr(a,b): # integraalberekening met simpson
   n=100;h=(b-a)/n;x=a;som=0
   for j in range(1,n):
      x=x+h
      if j%2==0:t=2
      else:t=4
      som=som+t*f(x)
   som=som+f(a)+f(b);integ=som*h/3
   return integ
def bereken():
   global xmi,xma,ymi,yma,a,b,n,h;nwk='.7f'
   xmi,xma,ymi,yma=mult(E2.get());C.delete(ALL)
   a=float(E3.get());xg=float(E4.get());del_x=float(E6.get())
   if a>=xg:wisuitv();messagebox.showinfo('Fout','Kies a<x');return
   opprh=f(xg)*del_x;wis(E5);E5.insert(0,format(opprh,nwk))
   opp1=integr(a,xg);opp2=integr(a,xg+del_x);opp3=opp2-opp1
   wis(E7);E7.insert(0,format(opp1,nwk))
   wis(E8);E8.insert(0,format(opp2,nwk))
   wis(E9);E9.insert(0,format(opp3,nwk))
   wis(E10);E10.insert(0,format(f(xg),nwk))
   wis(E11);E11.insert(0,format(opp3/del_x,nwk))
# arcering
   x=a;stap=0.05
   while x<=xg:
      lis=[];ap(lis,x,0);ap(lis,x,f(x));x=x+stap
      line = C.create_line(lis,fill='blue' )
   x=xg;stap=0.05
   while x<=xg+del_x:
      lis=[];ap(lis,x,0);ap(lis,x,f(x));x=x+stap
      C.create_line(lis,fill='magenta' )
   y=0;stap=0.05
   while y<=f(xg):
      lis=[];ap(lis,xg,y);ap(lis,xg+del_x,y);y=y+stap
      C.create_line(lis,fill='green' )
#stippellijn
   lis=[];ap(lis,xg,f(xg));ap(lis,0,f(xg))
   C.create_line(lis,fill='red',dash=[1,2])
#rechthoek
   lis=[];ap(lis,xg,0);ap(lis,xg,f(xg))
   ap(lis,xg+del_x,f(xg));ap(lis,xg+del_x,0)
```

```
    C.create_line(lis,fill='green' )
#tekst
    x=xmi;stap=0.1
    while f(x)<ymi or f(x)>yma:x=x+stap
    X=transx(x);Y=transy(f(x));C.create_text(X,Y,text='y=f(x)',fill='black')
    X=transx(-0.2);Y=transy(f(xg));C.create_text(X,Y,text='f(x)',fill='black')
    X=transx((a+xg)/2);Y=transy((f(a)+f(xg))/4);C.create_text(X,Y,text='F(x)',fill='black')
    lis=[];ap(lis,a,-0.1);C.create_text(lis,text='a')
    lis=[];ap(lis,xg,-0.1);C.create_text(lis,text='x')
    lis=[];ap(lis,xg+del_x,-0.3);C.create_text(lis,text='x+dx')
#assen
    X=transx(0);C.create_line(X,0,X,hoogteC,fill='blue',arrow='first')
    Y=transy(0);C.create_line(0,Y,breedte,Y,fill='blue',arrow='last')
# rooster
    for x in range(int(xmi),int(xma+1)):
        X=transx(x);C.create_line(X,0,X,hoogteC,fill='black',dash=[1,2])
    for y in range(int(ymi),int(yma+1)):
        Y=transy(y);C.create_line(0,Y,breedte,Y,fill='black',dash=[1,2] )
#kromme
    lis=[];stap=0.05;x=xmi
    while x<xma:
        x=x+stap;y=f(x);X=transx(x);Y=transy(y)
        lis.append(X);lis.append(Y)
    C.create_line(lis,fill='black')
def info():h=messagebox.showinfo(tit,infstr)
def halveerdel_x():del_x=float(E6.get());del_x=del_x/2;wis(E6);E6.insert(0,str(del_x))
def ap(li,x,y):X=transx(x);Y=transy(y);li.append(X);li.append(Y)
def wis(E):E.delete(0,len(E.get()))
def wisuitv():
    for E in (E5,E7,E8,E9,E10,E11):wis(E)
def nieuw():
    for E in (E1,E2,E3,E4,E5,E6,E7,E8,E9,E10,E11):wis(E)
    C.delete(ALL)
#Hoofdprogramma
tit='Hoofdstelling van de integraalrekening';gr='lightgrey'
breedte=480;hoogte=550;hoogteC=hoogte-130;hoInv1=hoogte-120;hoInv2=hoogte-90;hoInv3=hoogte-
60;hoInv4=hoogte-30
form=Tk();form.geometry(str(breedte)+'x'+str(hoogte));form.title(tit);form.resizable(False,False)
C = Canvas(form, bg="white", height=hoogteC, width=breedte);C.pack()
L1=Label(form,text='Functie :',);L1.place(x=5,y=hoInv1)
E1=Entry(form,width=30);E1.place(x=60,y=hoInv1);E1.insert(0,'(-x**2+x+16)/4')
L2=Label(form,text='Xmi,Xma,Ymi,Yma:',);L2.place(x=205,y=hoInv1)
E2=Entry(form,width=12);E2.place(x=320,y=hoInv1);E2.insert(0,'-5,7,-3,5')
L3=Label(form,text='a:',);L3.place(x=0,y=hoInv2)
E3=Entry(form,width=6);E3.place(x=20,y=hoInv2);E3.insert(0,'1')
L4=Label(form,text='x:',);L4.place(x=60,y=hoInv2)
E4=Entry(form,width=6);E4.place(x=80,y=hoInv2);E4.insert(0,'3')
L5=Label(form,text='f(x)dx:',);L5.place(x=275,y=hoInv2)
E5=Entry(form,bg=gr,fg='green',width=12);E5.place(x=320,y=hoInv2)
L6=Label(form,text='dx:',);L6.place(x=135,y=hoInv2)
E6=Entry(form,width=10);E6.place(x=160,y=hoInv2);E6.insert(0,'0.8')
```

```
L7=Label(form,text='F(x):');L7.place(x=127,y=hoInv3)
E7=Entry(form,bg=gr,fg='blue',width=12);E7.place(x=160,y=hoInv3)
L8=Label(form,text='F(x+dx):');L8.place(x=0,y=hoInv3)
E8=Entry(form,bg=gr,width=12);E8.place(x=52,y=hoInv3)
L9=Label(form,text='F(x+dx)-F(x):');L9.place(x=240,y=hoInv3)
E9=Entry(form,bg=gr,fg='magenta',width=12);E9.place(x=320,y=hoInv3)
L10=Label(form,text='→f(x) als dx→0       f(x):',);L10.place(x=180,y=hoInv4)
E10=Entry(form,bg=gr,width=12);E10.place(x=320,y=hoInv4)
L11=Label(form,text='[F(x+dx)-F(x)]/dx:');L11.place(x=0,y=hoInv4)
E11=Entry(form,bg=gr,width=12);E11.place(x=105,y=hoInv4)
Button(form,text='Bereken',command=bereken,width=8).place(x=410,y=hoInv1)
Button(form,text='Nieuw',command=nieuw,width=8).place(x=410,y=hoInv2)
Button(form,text='Info',command=info,width=8).place(x=410,y=hoInv3)
Button(form,text='÷2',command=halveerdel_x,width=3).place(x=230,y=hoInv2-3)
infstr='F(x)=oppervlakte onder de kromme f(x) tussen a en x\n'
infstr=infstr+'F(x+dx)=oppervlakte onder f(x) tussen a en x+dx\n'
infstr=infstr+'F(x+dx)-F(x)= oppervlakte onder f(x) tussen x en x+dx\n'
infstr=infstr+'Vergelijk deze oppervlakte met opp.rechthoek f(x)dx\n'
infstr=infstr+'Neem voor dx achtereenvolgens 0.8,0.4,0.2,0.1...\n'
infstr=infstr+'Als dx → 0, dan nadert [F(x+dx)-F(x)]/dx tot f(x)\n'
infstr=infstr+"waaruit volgt F'(x)=f(x)"
form.mainloop()
```



84. Normale verdeling: grafiek P(a<x<b) grafiek_normalcdfTK

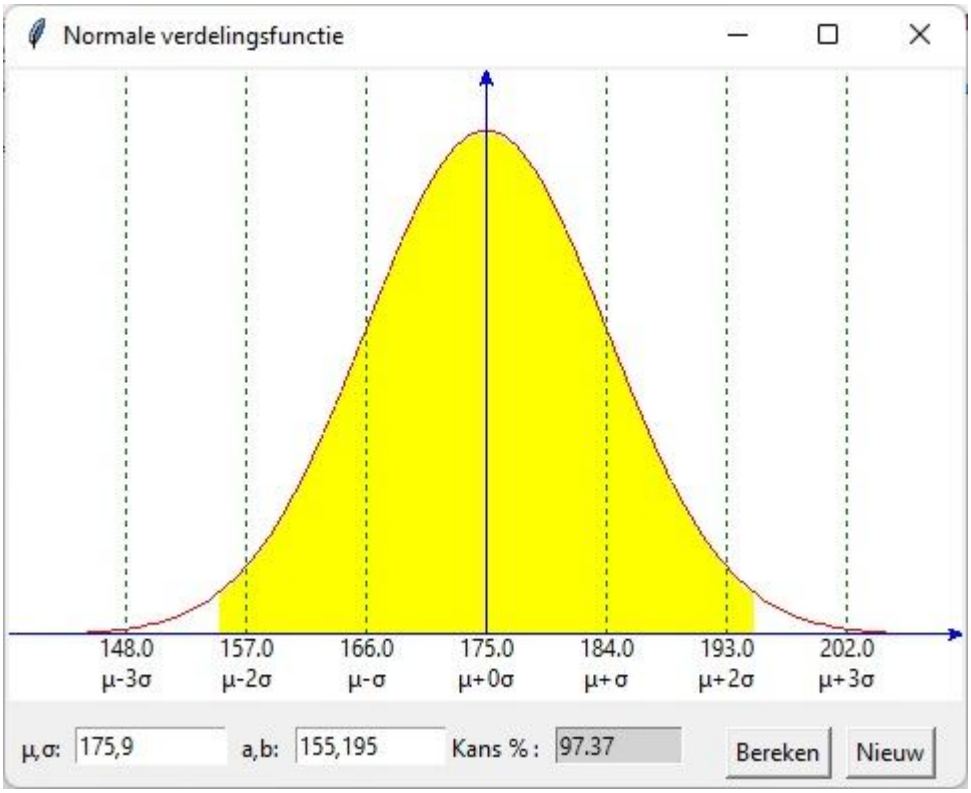
Het programma berekent de kans dat een gegeven in een interval [a,b] ligt, bij gegeven gemiddelde en standaardafwijking. Bovendien wordt deze kans grafisch voorgesteld als een oppervlakte onder de normale verdelingsfunctie.

Programma:

```
# normale verdeling -  P(a<x<b)
from math import *;from tkinter import *
def mult(s):
   l=s.split(',');co=[]
   for i in l:co.append(float(i))
   return co
def transx(x):return (x-xmi)*breedte/(xma-xmi)
def transy(y):return hoogteC-hoogteC*(y-ymi)/(yma-ymi)
def snd(x):return e**(-x**2/2)/sqrt(2*pi)
def nd(x):return e**(-(x-m)**2/2/s/s)/s/sqrt(2*pi)
```

```
def integr(a,b,n):
    h=(b-a)/n;x=a;som=0
    for j in range(1,n):
        x=x+h
        if j%2==0:t=2
        else:t=4
        som=som+t*nd(x)
    som=som+nd(a)+nd(b)
    integ=som*h/3
    return(integ)
def bereken():
    global xmi,xma,ymi,yma,m,s,n;C.delete(ALL)
    m,s=mult(E1.get());a,b=mult(E2.get());n=50
    percentgeg=round(100*integr(a,b,n),2)
    wis(E3);E3.insert(0,str(percentgeg))
    xmi,xma,ymi,yma=-4,4,-0.05,1/sqrt(2*pi)+0.05
# kromme
    lis=[];stap=0.05;x=xmi
    while x<xma:
        x=x+stap;y=snd(x);X=transx(x);Y=transy(y)
        lis.append(X);lis.append(Y)
    C.create_line(lis,fill='red')
# arcering
    sa=(a-m)/s;sb=(b-m)/s
    x=sa;stap=0.01
    while x<=sb:
        lis=[];y=0.002;X=transx(x);Y=transy(y);lis.append(X)
        lis.append(Y);y=snd(x)-0.002;X=transx(x);Y=transy(y)
        lis.append(X);lis.append(Y);x=x+stap
        C.create_line(lis,fill='yellow' )
# rooster
    for x in range(int(xmi),int(xma+1)):
        X=transx(x);Y=transy(0)
        C.create_line(X,Y,X,0,fill='green',dash=[1,2])
    for y in range(int(ymi),int(yma+1)):
        Y=transy(y)
        C.create_line(0,Y,breedte,Y,fill='green',dash=[1,2] )
# assen
    X=transx(0);Y=transy(0)
    C.create_line(X,Y,X,0,fill='blue',arrow='last')
    Y=transy(0);C.create_line(0,Y,breedte,Y,fill='blue',arrow='last')
# tekst
    for i in range(1,8):
        xpos=i*breedte/8;tek=format(m+(i-4)*s,'.1f')
        C.create_text(xpos,hoogteC-25,text=tek)
        C.create_text(xpos,hoogteC-10,text='\u03BC'+vc(i-4)+'\u03C3')
def vc(x):
    if x==1:s='+'
    elif x==-1:s='-'
    elif x>=0:s='+'+str(x)
    else:s=str(x)
    return s
def wis(E):E.delete(0,len(E.get()))
def nieuw():wis(E1);wis(E2);wis(E3);C.delete(ALL)
# hoofdprogramma
```

```
breedte=480;hoogte=360;hoogteC=hoogte-45;hoInv=hoogte-30
form=Tk();form.geometry(str(breedte)+'x'+str(hoogte))
form.title('Normale verdelingsfunctie')
C=Canvas(form, bg="white", height=hoogteC, width=breedte);C.pack()
L1=Label(form,text='\u03BC,\u03C3:',);L1.place(x=5,y=hoInv)
E1=Entry(form,width=12);E1.place(x=35,y=hoInv);E1.insert(0,'175,9')
L2=Label(form,text='a,b:',);L2.place(x=115,y=hoInv)
E2=Entry(form,width=12);E2.place(x=145,y=hoInv);E2.insert(0,'155,195')
L3=Label(form,text='Kans % :',);L3.place(x=220,y=hoInv)
E3=Entry(form,bg='lightgrey',width=10);E3.place(x=275,y=hoInv)
B1=Button(form,text='Bereken',command=bereken);B1.place(x=360,y=hoInv)
B2=Button(form,text='Nieuw',command=nieuw);B2.place(x=420,y=hoInv)
form.mainloop()
```



85. Steekproeven - frequentietabel - histogram steekproefTK

Het programma genereert automatisch steekproeven met een variabel aantal getallen. De steekproef wordt ingedeeld in klassen, met frequentietabel en berekening van gemiddelde en standaardafwijking. Bovendien een grafiek van het histogram en de overeenstemmende normale verdelingsfunctie.
De breedte en hoogte van het histogram worden aangepast zodat de oppervlakte van het histogram=opp. onder de standaardnormale kromme = 1.
De klassebreedte wordt herleid naar klassebreedte/standaardafwijking = klbr/staf
Is de klassefrequentie van een staafje =klfr(i), dan nemen we als hoogte $h(i)=\frac{klfr(i)*staf}{klbr*n}$

De totale oppervlakte van het histogram is dan $\sum h(i)*\frac{klbr}{staf}=\sum\frac{klfr(i)*staf}{klbr*n}.\frac{klbr}{staf}=\frac{\sum klfr(i)}{n}=1$

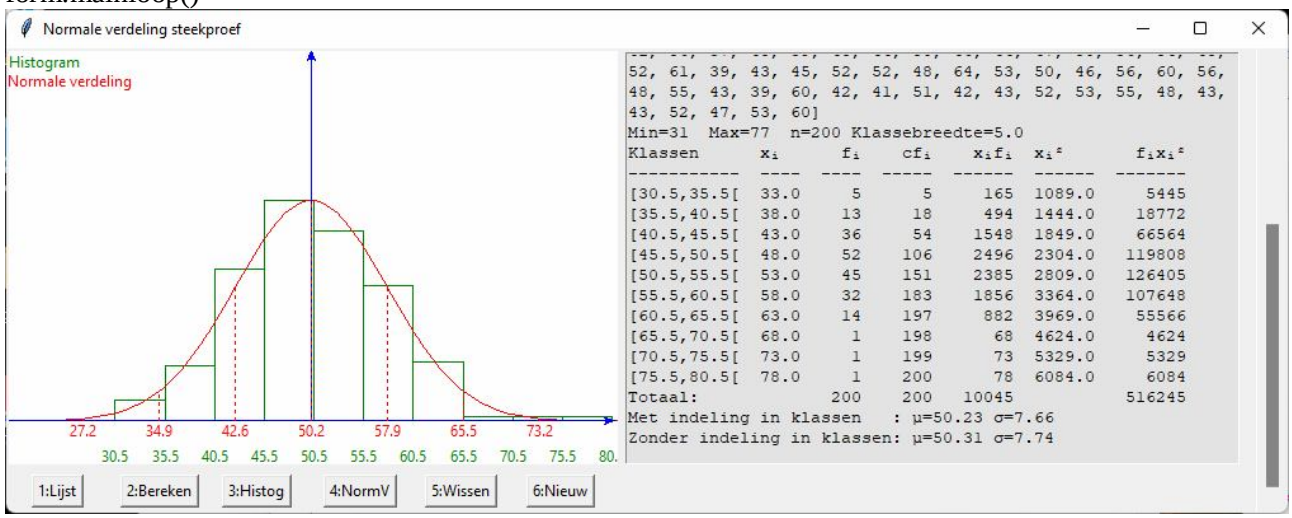
Programma:

```
# steekproef - normale verdelingsfunctie I
from random import randint as ri,normalvariate as nv;from math import sqrt,pi,e
```

```
from tkinter import *;from tabulate import tabulate
def transx(x):return (x-xmi)*breedte/(xma-xmi)
def transy(y):return hoogteC-hoogteC*(y-ymi)/(yma-ymi)
def snd(x):return e**(-x**2/2)/sqrt(2*pi)
def ad(lin):tex.insert(INSERT,lin+'')
def toonlijst():
    # genereren lijst en berekening; m1,staf1: gemiddelde en standaardafwijking
    global l,n,m1,staf1
    mu=ri(40,60);si=ri(5,10);C.delete(ALL);varbr=0
    while varbr<36:
        s=0;n=10*ri(10,20);l=[]
        for i in range(0,n):x=int(nv(mu,si)+0.5);l.append(x);s=s+x
        varbr=max(l)-min(l)+1
    m1=s/n;afw=0
    for i in range(0,n):afw=afw+(l[i]-m1)**2
    staf1=sqrt(afw/n)
    ad('Steekproef: lijst\n'+str(l)+'\n')
def bereken():
    # indelen in klassen; gem,staf:
    #gemiddelde en standaardafwijking met klassemiddens en frequenties
    global l,n,m,gem,staf,klmid,akl,klgr,klfr,klbr
    if l!=[]:
        l.sort();mal=max(l);mil=min(l);varbr=mal-mil+1;akl=10
        klbr=round(varbr/akl,0);klgr=[];klfr=[];klmid=[]
        akl=int((mal-mil)/klbr)+1
        for i in range(0,akl+1):
            klgr.append(min(l)+i*klbr-0.5)
            klmid.append(min(l)+(i+0.5)*klbr-0.5);klfr.append(0)
        for i in range (0,akl):
            for j in range(0,n):
                if l[j]>=klgr[i] and l[j]<klgr[i+1]:klfr[i]=klfr[i]+1
        klfr.pop()
        ad('Min='+str(min(l))+'  Max='+str(max(l))+'  n='+str(n)+' Klassebreedte='+str(klbr)+'\n')
        cf=[klfr[0]];tabel=[]
        for i in range(1,akl):cf.append(cf[i-1]+klfr[i])
        for i in range(0,akl):
            lijn=[]
            lijn.append('['+format(klgr[i],'.1f')+','+format(klgr[i+1],'.1f')+'[')
            lijn.append(format(klmid[i],'.1f'));lijn.append(str(klfr[i]));
            lijn.append(cf[i]);lijn.append(format(klmid[i]*klfr[i],'.1f'))
            lijn.append(format(klmid[i]**2,'.1f'))
            lijn.append(format(klmid[i]**2*klfr[i],'.1f'))
            tabel.append(lijn)
        somX=0;somX2=0
        for i in range(0,akl):
            somX=somX+klmid[i]*klfr[i]
            somX2=somX2+klmid[i]**2*klfr[i]
        gem=somX/n;staf=sqrt(somX2/n-gem**2)
        lijn=[];lijn.append('Totaal:');lijn.append('')
        lijn.append(str(n));lijn.append(str(n))
        lijn.append(format(somX,'.1f'));lijn.append('')
        lijn.append(format(somX2,'.1f'))
        tabel.append(lijn)
        ad(tabulate(tabel,headers=['Klassen','xi','fi','cfi','xifi','xi²','fixi²']))
        ad('\nMet indeling in klassen   :')
```

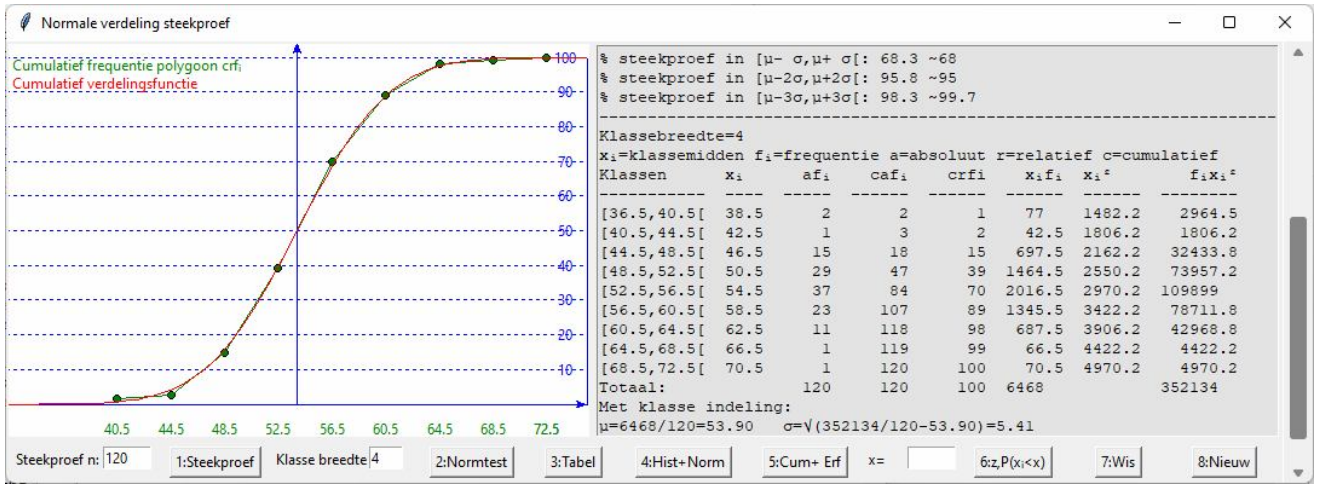
```
        ad(' \u03BC='+format(gem,'.2f')+' \u03C3='+format(staf,'.2f')+'\n')
        ad('Zonder indeling in klassen:')
        ad(' \u03BC='+format(m1,'.2f')+' \u03C3='+format(staf1,'.2f')+'\n')
def asrooster():
# assen
    X=transx(0);Y=transy(0);C.create_line(X,Y,X,0,fill='blue',arrow='last')
    Y=transy(0);C.create_line(0,Y,breedte,Y,fill='blue',arrow='last')
def histog():
    if l!=[]:
        global xmi,xma,ymi,yma
        xmi,xma,ymi,yma=-4,4,-0.075,1.5/sqrt(2*pi)+0.075
        asrooster();lis=[]
        for i in range(0,akl):
            Z_ond=(klgr[i]-gem)/staf;Z_bov=(klgr[i+1]-gem)/staf
            # breedte en hoogte aanpassen zodat de oppervlakte van het histogram=1
            # overeenstemming met standaardnormale kromme,waarbij staf=eenheid
            h=staf/klbr*klfr[i]/n;X=transx(Z_ond)
            C.create_text(X,hoogteC-5,text=format(klgr[i],'.1f'),fill='green')
            Y=transy(0);lis.append(X);lis.append(Y)
            Y=transy(h);lis.append(X);lis.append(Y)
            X=transx(Z_bov);lis.append(X);lis.append(Y)
        C.create_line(lis,fill='green')
    Y=transy(0);lis.append(X);lis.append(Y)
    C.create_line(lis,fill='green')
    C.create_text(transx(Z_bov),hoogteC-5,text=format(klgr[akl],'.1f'),fill='green')
    C.create_text(30,10,text='Histogram',fill='green')
    return
def normverd():
    global xmi,xma,ymi,yma
    xmi,xma,ymi,yma=-4,4,-0.075,1.5/sqrt(2*pi)+0.075
# kromme
    lis=[];stap=0.05;x=xmi
    while x<xma:
        x=x+stap;y=snd(x);X=transx(x);Y=transy(y)
        lis.append(X);lis.append(Y)
    C.create_line(lis,fill='red')
    asrooster()
    # tekst
    for i in range(1,8):
        xpos=i*breedte/8;tek=format(gem+(i-4)*staf,'.1f')
        C.create_text(xpos,hoogteC-25,text=tek,fill='red')
    # rooster
    for x in range(int(xmi),int(xma+1)):
        X=transx(x);Y1=transy(0);Y2=transy(snd(x));
        C.create_line(X,Y1,X,Y2,fill='red',dash=[1,2])
    C.create_text(50,25,text='Normale verdeling',fill='red')
def wis():C.delete(ALL);tex.delete('1.0',END)
def nieuw():global l;C.delete(ALL);tex.delete('1.0',END);l=[]
# hoofdprogramma
breedte=480;hoogte=365;hoogteC=hoogte-40;hoInv=hoogte-30
mu='/u03BC';si='/u03C3'
form=Tk();form.geometry(str(1010)+'x'+str(hoogte))
form.title('Normale verdeling steekproef')
C = Canvas(form, bg="white", height=hoogteC, width=breedte)
C.place(x=0,y=0)
```

```
tex=Text(form,bg='grey89',height=20,width=60,wrap=WORD);tex.place(x=breedte+8,y=3)
vsb=Scrollbar(form,orient=VERTICAL,command=tex.yview,width=25)
tex['yscrollcommand']=vsb.set;vsb.pack(side=RIGHT,fill='y')
B1=Button(form,text='1:Lijst',command=toonlijst);B1.place(x=20,y=hoInv)
B2=Button(form,text='2:Bereken',command=bereken);B2.place(x=90,y=hoInv)
B3=Button(form,text='3:Histog',command=histog);B3.place(x=170,y=hoInv)
B4=Button(form,text='4:NormV',command=normverd);B4.place(x=250,y=hoInv)
B5=Button(form,text='5:Wissen',command=wis);B5.place(x=330,y=hoInv)
B6=Button(form,text='6:Nieuw',command=nieuw);B6.place(x=410,y=hoInv)
form.mainloop()
```



86. Steekproef 2 steekproefTK2

Het programma 'Steekproef' werd uitgebreid met een 'Normtest' :
een 68-95-99.7 proef voor het % gegevens in het interval $[\mu - i\cdot\sigma, \mu + i\cdot\sigma]$ i=1,2,3
een cumulatief frequentiepolygoon, en een $P(x_i<x)$ berekening.



Programma:

```
# Steekproef - normale dichtheidsfunctie
from random import randint as ri,normalvariate as nv;from math import sqrt,pi,e
from tkinter import *;from tabulate import tabulate
def transx(x):return (x-xmi)*wid/(xma-xmi)
```

```
def transy(y):return heigC-heigC*(y-ymi)/(yma-ymi)
def snd(x):return e**(-x**2/2)/sqrt(2*pi)
def ad(lin):tex.insert(INSERT,lin+'')
def integr(a,b,n):
    h=(b-a)/n;x=a;sum=0
    for j in range(1,n):
        x=x+h
        if j%2==0:t=2
        else:t=4
        sum=sum+t*snd(x)
    sum=sum+snd(a)+snd(b);integ=sum*h/3
    return(integ)
def showlist():
    # lijst en berekening: gemiddelde en standaarddeviatie
    global l,n,mean1,stdev1,ncl
    mu=ri(40,60);si=ri(5,10);C.delete(ALL);varwid=0;ncl=10
    while varwid<36:
        s=0;n=int(E1.get());l=[]
        for i in range(0,n):x=int(nv(mu,si)+0.5);l.append(x);s=s+x
        varwid=max(l)-min(l)+1
    klbr=int(varwid/ncl+0.5);E2.delete(0,len(E2.get()));E2.insert(0,str(klbr))
    mean1=s/n;afw=0
    for i in range(0,n):afw=afw+(l[i]-mean1)**2
    stdev1=sqrt(afw/n)
    ad(lin);ad('Steekproef: lijst\n'+str(l)+'\n')
    ad('Min='+str(min(l))+'  Max='+str(max(l))+'  n='+str(n))
    ad('\nZonder indeling in klasse :')
    ad(' \u03BC='+format(mean1,'.2f')+' \u03C3='+format(stdev1,'.2f')+'\n');ad(lin)
def normtest():
    s1,s2,s3=0,0,0
    for i in range(0,n):
        if l[i]>=mean1-stdev1 and l[i]<mean1+stdev1:s1=s1+1
        if l[i]>=mean1-2*stdev1 and l[i]<mean1+2*stdev1:s2=s2+1
        if l[i]>=mean1-3*stdev1 and l[i]<mean1+3*stdev1:s3=s3+1
    s1pc,s2pc,s3pc=s1*100/n,s2*100/n,s3*100/n
    ad('Normale verdeling test')
    ad('\n% steekproef in [\u03BC- \u03C3,\u03BC+ \u03C3[: '+format(s1pc,'.1f')+' ~68')
    ad('\n% steekproef in [\u03BC-2\u03C3,\u03BC+2\u03C3[: '+format(s2pc,'.1f')+' ~95')
    ad('\n% steekproef in [\u03BC-3\u03C3,\u03BC+3\u03C3[: '+format(s3pc,'.1f')+' ~99.7\n' );ad(lin)
def calc():
# organize into classes; mean,stdev:
# mean and standard deviation with class centers and frequencies
    global l,n,m,mean,stdev,clmid,ncl,klgr,klfr,klbr,cf;C.delete(ALL)
    if l!=[]:
        l.sort();mal=max(l);mil=min(l);varwid=mal-mil+1
        klbr=int(E2.get());klgr=[];klfr=[];clmid=[]
        ncl=int((mal-mil)/klbr)+1
        for i in range(0,ncl+1):
            klgr.append(min(l)+i*klbr-0.5)
            clmid.append(min(l)+(i+0.5)*klbr-0.5);klfr.append(0)
        for i in range (0,ncl):
            for j in range(0,n):
                if l[j]>=klgr[i] and l[j]<klgr[i+1]:klfr[i]=klfr[i]+1
        klfr.pop()
        ad('Klassebreedte='+str(klbr)+'\n')
```

```
        ad('xi=klassemidden fi=frequentie a=absoluut r=relatief c=cumulatief\n')
        cf=[klfr[0]];table=[]
        for i in range(1,ncl):cf.append(cf[i-1]+klfr[i])
        for i in range(0,ncl):
           lijn=[];lijn.append('['+format(klgr[i],'.1f')+','+format(klgr[i+1],'.1f')+'[')
           lijn.append(round(clmid[i],1));lijn.append(round(klfr[i],1));
           lijn.append(cf[i]);lijn.append(int(cf[i]*100/n));lijn.append(clmid[i]*klfr[i])
           lijn.append(round(clmid[i]**2,1));lijn.append(round(clmid[i]**2*klfr[i],1))
           table.append(lijn)
        smX=0;smX2=0
        for i in range(0,ncl):
           smX=smX+clmid[i]*klfr[i]
           smX2=smX2+clmid[i]**2*klfr[i]
        mean=smX/n;stdev=sqrt(smX2/n-mean**2)
        lijn=[];lijn.append('Totaal:');lijn.append('')
        lijn.append(str(n));lijn.append(str(cf[ncl-1]));lijn.append('100')
        lijn.append(format(smX,'.1f'));lijn.append('')
        lijn.append(format(smX2,'.1f'));table.append(lijn);
        ad(tabulate(table,headers=['Klassen','xi','afi','cafi', 'crfi','xifi','xi²','fixi²']))
        ad('\nMet klasse indeling:')
        ad('\n\u03BC='+format(smX,'.0f')+'/'+str(n)+'='+format(mean,'.2f')+'   ')
        ad('\u03C3=\u221A('+format(smX2,'.0f')+'/'+str(n)+'-'+format(mean,'.2f')+')='+format(stdev,'.2f')+'\n')
        ad(lin)
    else:ad('Geen steekproef!!\n')
def hor_grid(t):
# axes
    X=transx(0);Y=transy(0);C.create_line(X,Y,X,0,fill='blue',arrow='last')
    Y=transy(0);C.create_line(0,Y,wid,Y,fill='blue',arrow='last')
#test
    ym=yma-.05
    lis=[transx(xma),transy(0),transx(xma),transy(ym)];C.create_line(lis,fill='blue')
    for i in range(1,t+1):
       lis=[transx(xmi),transy(i*ym/t),transx(xma),transy(i*ym/t)];C.create_line(lis,fill='blue',dash=[1,4])
       C.create_text(transx(xma-0.3),transy(i*ym/t),text=str(10*i),fill='blue')
def histog():
    C.delete(ALL)
    if l!=[]:
       global xmi,xma,ymi,yma
       xmi,xma,ymi,yma=-4,4,-0.1,stdev/klbr*50/n+.05
       hor_grid(5);lis=[]
       for i in range(0,ncl):
          Z_und=(klgr[i]-mean)/stdev;Z_upp=(klgr[i+1]-mean)/stdev
# pas breedte en hoogte aan zodat het gebied van het histogram =1
# voldoet aan de standaard normale kromme, waarbij stdev = eenheid
          h=stdev/klbr*klfr[i]/n;X=transx(Z_und)
          C.create_text(X,heigC-5,text=format(klgr[i],'.1f'),fill='green')
          Y=transy(0);lis.append(X);lis.append(Y)
          Y=transy(h);lis.append(X);lis.append(Y)
          X=transx(Z_upp);lis.append(X);lis.append(Y)
       C.create_line(lis,fill='green')
    Y=transy(0);lis.append(X);lis.append(Y)
    C.create_line(lis,fill='green')
    C.create_text(transx(Z_upp),heigC-5,text=format(klgr[ncl],'.1f'),fill='green')
    C.create_text(5,18,text='Histogram afi',fill='green',anchor='nw')
# normdis
```

```
# curve
   lis=[];stap=0.05;x=xmi
   while x<xma:
      x=x+stap;y=snd(x);X=transx(x);Y=transy(y)
      lis.append(X);lis.append(Y)
   C.create_line(lis,fill='red')
   hor_grid(5)
# text
   for i in range(1,8):
      xpos=i*wid/8;tek=format(mean+(i-4)*stdev,'.1f')
      C.create_text(xpos,heigC-20,text=tek,fill='red')
# grid stdev
   for x in range(int(xmi),int(xma+1)):
      X=transx(x);Y1=transy(0);Y2=transy(snd(x));
      C.create_line(X,Y1,X,Y2,fill='red',dash=[1,2])
   C.create_text(5,33,text='Normale dichtheidsfunctie ',fill='red',anchor='nw')
def cumfreq():
   C.delete(ALL)
   if l!=[]:
      global xmi,xma,ymi,yma
      xmi,xma,ymi,yma=-4,4,-0.1,stdev/klbr*100/n+.05
      hor_grid(10);lis=[]
      for i in range(0,ncl):
         Z_upp=(klgr[i+1]-mean)/stdev
         h=stdev/klbr*cf[i]*100/n**2;X,Y=transx(Z_upp),transy(h)
         C.create_oval(X-3,Y-3,X+3,Y+3,fill='green')
         C.create_text(X,heigC-5,text=format(klgr[i+1],'.1f'),fill='green')
         Y=transy(h);lis.append(X);lis.append(Y)
   C.create_line(lis,fill='green')
   C.create_text(transx(Z_upp),heigC-5,text=format(klgr[ncl],'.1f'),fill='green')
   C.create_text(5,12,text='Cumulatief frequentie polygoon crf_i',fill='green',anchor='nw')
# erf function
   lis=[];t=-4.1;opp=0
   while t<4:
      t=t+0.1
      X=transx(t);opp=opp+integr(t-0.1,t,10);Y=transy(stdev/klbr*opp*100/n)
      lis.append(X);lis.append(Y)
   C.create_line(lis,fill='red')
   C.create_text(5,27,text='Cumulatief verdelingsfunctie',fill='red',anchor='nw')
def calczpc():
   x=float(E3.get());z=(x-mean)/stdev;crf_x=100*integr(-4,z,100)
   ad('x='+str(x)+' z='+format(z,'.2f')+'  '+'P(x_i<'+str(x)+')='+format(crf_x,'.0f')+'%  (als normale
verdeling)\n')
   return
def wis():C.delete(ALL);tex.delete('1.0',END)
def clr(E):E.delete(0,len(E.get()))
def nieuw():
   global l;C.delete(ALL);tex.delete('1.0',END);l=[]
   for ent in [E1,E2,E3]:clr(ent)
   E1.insert(0,'120')
# hoofdprogramma
wid=480;heig=365;heigC=heig-40;hoInv=heig-30;lin='-'*70+'\n'
mu='/u03BC';si='/u03C3'
form=Tk();form.geometry(str(1080)+'x'+str(heig))
form.title('Normale verdeling steekproef ')
```

```
C = Canvas(form, bg="white", height=heigC, width=wid);C.place(x=0,y=0)
tex=Text(form,bg='grey89',height=20,width=70,wrap=WORD);tex.place(x=wid+8,y=3)
vsb=Scrollbar(form,orient=VERTICAL,command=tex.yview,width=25)
tex['yscrollcommand']=vsb.set;vsb.pack(side=RIGHT,fill='y')
Label(form,text='Steekproef n:').place(x=5,y=hoInv)
E1=Entry(form,width=6);E1.place(x=80,y=hoInv);E1.insert(0,'120')
Label(form,text='Klasse breedte :').place(x=220,y=hoInv)
E2=Entry(form,width=4);E2.place(x=300,y=hoInv)
B1=Button(form,text='1:Steekproef',command=showlist);B1.place(x=135,y=hoInv)
B2=Button(form,text='2:Normtest',command=normtest);B2.place(x=350,y=hoInv)
B3=Button(form,text='3:Tabel',command=calc);B3.place(x=445,y=hoInv)
B4=Button(form,text='4:Hist+Norm',command=histog);B4.place(x=520,y=hoInv)
B5=Button(form,text='5:Cum+ Erf',command=cumfreq);B5.place(x=625,y=hoInv)
Label(form,text='x=').place(x=710,y=hoInv);E3=Entry(form,width=6);E3.place(x=745,y=hoInv)
B6=Button(form,text='6:z,P(xi<x)',command=calczpc);B6.place(x=800,y=hoInv)
B7=Button(form,text='7:Wis',command=wis);B7.place(x=900,y=hoInv)
B8=Button(form,text='8:Nieuw',command=nieuw);B8.place(x=980,y=hoInv)
form.mainloop()
```

87. Invnorm invnormTK

Stel gegeven een oppervlakte opp. We willen een x vinden zodat de oppervlakte onder de kromme y=f(x) tussen a en x gelijk is aan deze oppervlakte.
Als we een primitieve functie F(t) kunnen bepalen ,dan is F(x)-F(a)=opp
Lossen we deze vergelijking op naar x, dan vinden we 1 of meerdere oplossingen die voldoen aan

$$\int_a^x f(t)dt = opp$$

Kunnen we analytisch geen primitieve functie bepalen, dan moeten we het vraagstuk numeriek oplossen.

Vb. bij de snd-functie $f(x)=\frac{1}{\sqrt{2\pi}}\cdot e^{\frac{-x^2}{2}}$

Hiervan weten we dat $\int_{-\infty}^{+\infty} f(x)dx = 1$ en $\int_{-\infty}^{z} f(x)dx = p$ waarbij p de waarschijnlijkheid is dat een gestandaardiseerd gegeven $z_i < z$.
We willen nu z berekenen in functie van p. Noem deze functie invN, dan is z= invN(p).
Omdat de snd-functie symmetrisch is t.o.v. de y-as, is invN(p) =-invN(1-p)
Deze laatste formule passen we toe als p< 0.5. Bovendien is 0=invN(0.5)

Stel bv. p=0.77, dan is $\int_{-\infty}^{0} f(x)dx + \int_{0}^{z} f(x)dx = p$ zodat $\int_{0}^{z} f(x)dx = p - 0.5 = 0.27$

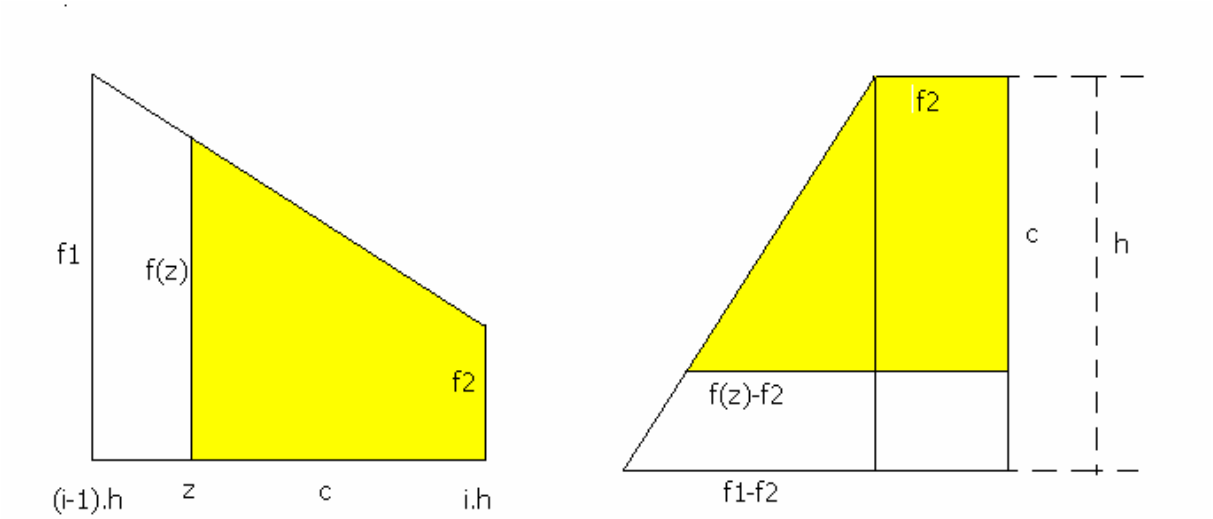
We nemen nu stapsgewijze (stap=h) de trapeziumregel vanuit 0 .

De eerste functiewaarde is $f(0) = \frac{1}{\sqrt{2\pi}}$. Stel f_1= f(0), f_2=f(h)

Stel dan opp= 0.5 en tel er bij op: $\frac{f_1 + f_2}{2}\cdot h$: dus opp=opp+$\frac{f_1 + f_2}{2}\cdot h$

Stel nu $f_1=f_2$ en f_2=f(2.h) en herhaal opp=opp+..... ... f_2=f(i.h) opp=opp+.....
We herhalen dit tot als opp>p: nu zijn we zeker dat de gezochte z gelegen is tussen (i-1).h en i.h
We moeten van de opp iets aftrekken (de gearceerde oppervlakte in de figuur) om p uit te komen, namelijk opp-p.
Dus moeten we evengoed van i.h iets aftrekken om z uit te komen: z=i.h-c
Het is die **c** (**c**orrectie die we zoeken)

Met de trapeziumregel veronderstellen we dat de oppervlakte onder de functie f(x) in het interval (i-1).h tot i.h gegeven wordt door een trapezium. We hebben reeds deze oppervlakte $\frac{f_1+f_2}{2}\cdot h$ opgeteld maar we moeten daar de gearceerde oppervlakte aftrekken, om $\frac{f_1+f(z)}{2}.(h-c)$ over te houden.



(2de fig. is 1ste fig. gekanteld naar links over 90°)

De gearceerde oppervlakte = opp - p = het verschil van de oppervlakte van de trapeziums (tussen f_1 en f_2) en (tussen f_1 en f(z)) $= \frac{f_1+f_2}{2}\cdot h-\frac{f_1+f(z)}{2}.(h-c)$

Hierin kunnen we f(z) berekenen in functie van c: $\frac{f(z)-f_2}{f_1-f_2}=\frac{c}{h}$ (gelijkvormige Δ)

waaruit: $f(z)=f_2+(f_1-f_2).\frac{c}{h}$

Samen hebben we: $\frac{f_1+f_2}{2}\cdot h-\frac{f_1+f_2+(f_1-f_2)\frac{c}{h}}{2}.(h-c)=\text{opp - p}$ x 2h

$(f_1+f_2)\cdot h^2-\left[(f_1+f_2)\cdot h+(f_1-f_2)\cdot c\right]\cdot(h-c)-2h.(opp-p)=0$

$(f_1+f_2)\cdot h\cdot c-(f_1-f_2)\cdot h\cdot c+(f_1-f_2)\cdot c^2-2h\cdot(opp-p)=0$

$(f_1-f_2)\cdot c^2+2h\cdot f_2.c-2h\cdot(opp-p)=0$ (vierkantsvergelijking in c)

Stellen we nu $A=f_1-f_2$ $B=2h.f_2$ $C=-2h.(opp-p)$ $D=B^2-4AC$

Omdat C<0 en A>0 is $\frac{C}{A}<0$ zodat we een positieve en negatieve oplossing hebben.

De positieve is de correctie die we zochten, dwz: **c** = $\frac{-B+\sqrt{D}}{2A}$.

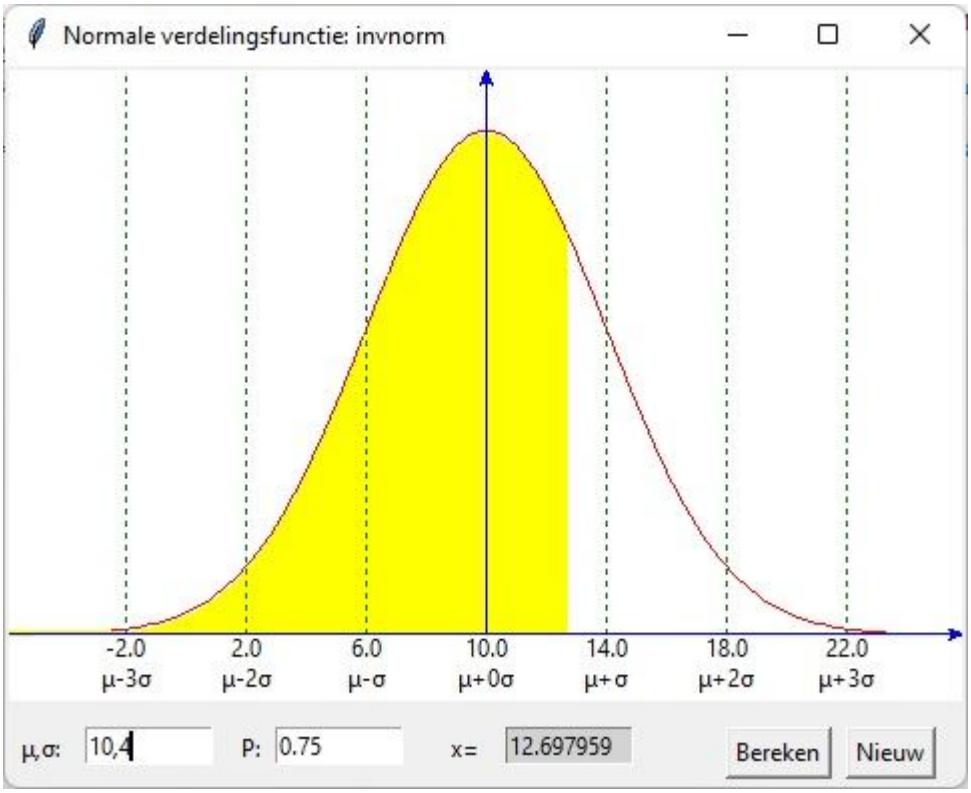
Uiteindelijk is z=i.h-c

Programma:

```
# normale verdeling invnorm
from math import *;from tkinter import *
def mult(s):
   l=s.split(',');co=[]
```

```
    for i in l:co.append(float(i))
    return co
def transx(x):return (x-xmi)*breedte/(xma-xmi)
def transy(y):return hoogteC-hoogteC*(y-ymi)/(yma-ymi)
def snd(x):return e**(-x**2/2)/sqrt(2*pi)
def normalcdf(p):
    z=0;h=0.001;opp=0.5;f1=1/sqrt(2*pi)
    while opp<=p:
        z=z+h;f2=snd(z);opp=opp+h*(f1+f2)/2;f1oud=f1;f1=f2
    # berekening correctie
    A=f1oud-f2;B=2*h*f2;C=-2*h*(opp-p);D=B*B-4*A*C
    c=(-B+sqrt(D))/2/A
    z=z-c
    return(z)
def bereken():
    global xmi,xma,ymi,yma,m,s,n
    C.delete(ALL)
    m,s=mult(E1.get());p=float(E2.get())
    if p<0.5:z=-normalcdf(1-p)
    else:z=normalcdf(p)
    x=s*z+m
    wis(E3);E3.insert(0,format(x,'.6f'))
    xmi,xma,ymi,yma=-4,4,-0.05,1/sqrt(2*pi)+0.05
# kromme
    lis=[];stap=0.05;x=xmi
    while x<xma:
        x=x+stap;y=snd(x);X=transx(x);Y=transy(y)
        lis.append(X);lis.append(Y)
    C.create_line(lis,fill='red')
# arcering
    sa=-4;sb=z
    x=sa;stap=0.01
    while x<=sb:
        lis=[];y=0.002;X=transx(x);Y=transy(y);lis.append(X)
        lis.append(Y);y=snd(x)-0.002;X=transx(x);Y=transy(y)
        lis.append(X);lis.append(Y);x=x+stap
        C.create_line(lis,fill='yellow' )
# rooster
    for x in range(int(xmi),int(xma+1)):
        X=transx(x);Y=transy(0)
        C.create_line(X,Y,X,0,fill='green',dash=[1,2])
    for y in range(int(ymi),int(yma+1)):
        Y=transy(y)
        C.create_line(0,Y,breedte,Y,fill='green',dash=[1,2] )
# assen
    X=transx(0);Y=transy(0)
    C.create_line(X,Y,X,0,fill='blue',arrow='last')
    Y=transy(0);C.create_line(0,Y,breedte,Y,fill='blue',arrow='last')
# tekst
    for i in range(1,8):
        xpos=i*breedte/8;tek=format(m+(i-4)*s,'.1f')
        C.create_text(xpos,hoogteC-25,text=tek)
        C.create_text(xpos,hoogteC-10,text='\u03BC'+vc(i-4)+'\u03C3')
def vc(x):
    if x==1:s='+'
```

```
    elif x==-1:s='-'
    elif x>=0:s='+'+str(x)
    else:s=str(x)
    return s
def wis(E):E.delete(0,len(E.get()))
def nieuw():wis(E1);wis(E2);wis(E3);C.delete(ALL)
# hoofdprogramma
breedte=480;hoogte=360;hoogteC=hoogte-45;hoInv=hoogte-30
#m=0;s=1
form=Tk();form.geometry(str(breedte)+'x'+str(hoogte))
form.title('Normale verdelingsfunctie: invnorm')
C=Canvas(form, bg="white", height=hoogteC, width=breedte);C.pack()
L1=Label(form,text='\u03BC,\u03C3:',);L1.place(x=5,y=hoInv)
E1=Entry(form,width=10);E1.place(x=40,y=hoInv);E1.insert(0,'0,1')
L2=Label(form,text='P:',);L2.place(x=115,y=hoInv)
E2=Entry(form,width=10);E2.place(x=135,y=hoInv);E2.insert(0,'0.75')
L3=Label(form,text='x=',);L3.place(x=220,y=hoInv)
E3=Entry(form,bg='lightgrey',width=10);E3.place(x=250,y=hoInv)
B1=Button(form,text='Bereken',command=bereken);B1.place(x=360,y=hoInv)
B2=Button(form,text='Nieuw',command=nieuw);B2.place(x=420,y=hoInv)
form.mainloop()
```



88. Betrouwbaarheidsintervallen bij een normale verdeling. betrouwbaarheidsinterval

1. Stel dat we een steekproef nemen van **n** elementen waarvan **x** successen.

We noemen de steekproefproportie $\hat{p} = \frac{x}{n}$

Dan zijn de proporties $\hat{p}$ eveneens normaal verdeeld met $\mu=\hat{p}$ en $\sigma=\sqrt{\frac{\hat{p}\cdot(1-\hat{p})}{n}}$

2. Zijn we geïnteresseerd in het gemiddelde van een populatie, dan hebben we behalve n ook het gemiddelde $\overline{x}$ en de standaardafwijking s van de steekproef nodig.

De gemiddelden $\overline{x}$ zijn dan eveneens normaal verdeeld met $\mu=\overline{x}$ en $\sigma=\frac{s}{\sqrt{n}}$

3. Stel nu bp = betrouwbaarheidspercentage, bvb. bp is 95
In de standaardnormale verdeling zit dan 95% van de gegevens in [-z, z] waarbij

$P(z_i\leq z)=\frac{1+\frac{bp}{100}}{2}=0.975$. Met invnorm vinden we $z\approx 1.96$

Een betrouwbaarheidsinterval met betrouwbaarheid bp is $[\mu-z\cdot\sigma\ ,\ \mu+z\cdot\sigma]$, dwz.:

- voor de populatieproportie: $[\hat{p}-z\cdot\sigma\ ,\ \hat{p}+z\cdot\sigma]=\left[\hat{p}-z\cdot\sqrt{\frac{\hat{p}\cdot(1-\hat{p})}{n}},\hat{p}+z\cdot\sqrt{\frac{\hat{p}\cdot(1-\hat{p})}{n}}\right]$

- voor het populatiegemiddelde: $\left[\overline{x}-z.\sigma,\overline{x}+z.\sigma\right]=\left[\overline{x}-z\cdot\frac{s}{\sqrt{n}},\overline{x}+z\cdot\frac{s}{\sqrt{n}}\right]$

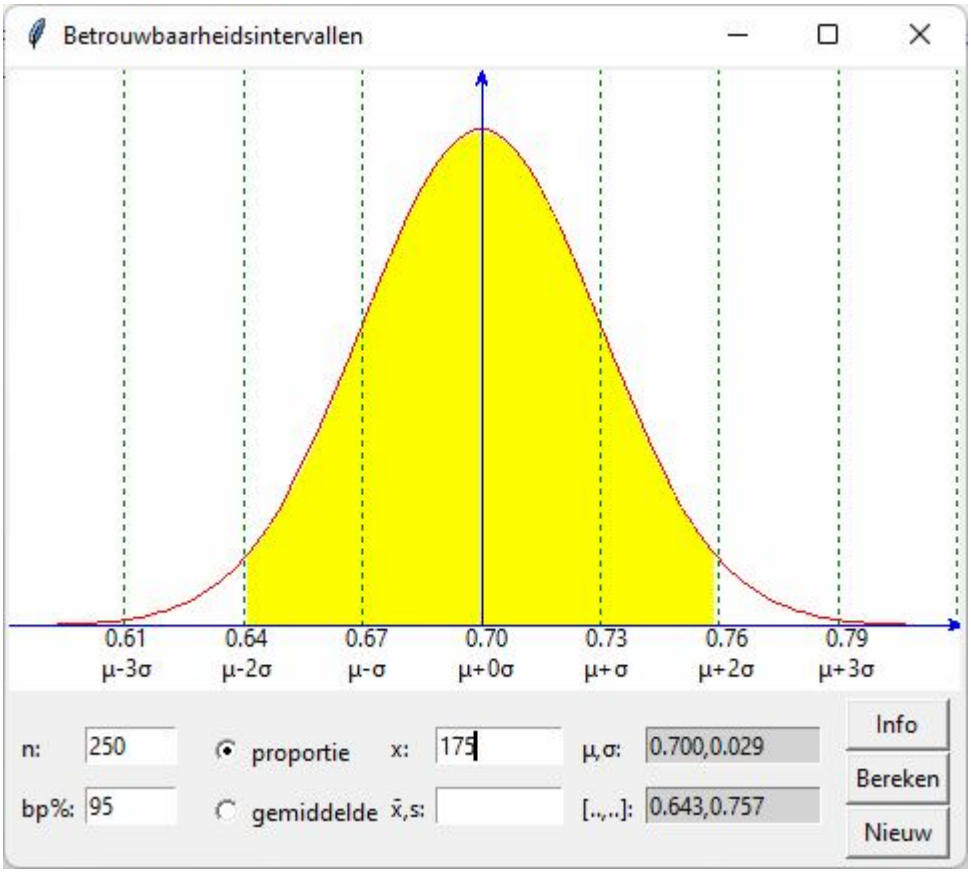
Het programma is een uitbreiding van het invnorm- programma. De gebruiker moet steeds n en bp% invullen. Kiest hij 'proportie', dan moet ook x ingevuld worden. Als hij 'gemiddelde' kiest moet hij $\overline{x}$ en s invullen. Het programma berekent μ , σ en het betrouwbaarheidsinterval. Het canvas toont dit interval in de grafiek van een standaardnormale functie, waarin op de x-as een omzetting is gedaan van z naar $\mu+z\cdot\sigma$

Programma

```
# betrouwbaarheidsintervallen
from math import *;from tkinter import *;from tkinter import messagebox
def mult(s):
   l=s.split(',');co=[]
   for i in l:co.append(float(i))
   return co
def transx(x):return (x-xmi)*breedteC/(xma-xmi)
def transy(y):return hoogteC-hoogteC*(y-ymi)/(yma-ymi)
def snd(x):return e**(-x**2/2)/sqrt(2*pi)
def normalcdf(p):
   z=0;h=0.001;opp=0.5;f1=1/sqrt(2*pi)
   while opp<=p:
      z=z+h;f2=snd(z);opp=opp+h*(f1+f2)/2;f1_oud=f1;f1=f2
   # berekening correctie
   A=f1_oud-f2;B=2*h*f2;C=-2*h*(opp-p);D=B*B-4*A*C
   c=(-B+sqrt(D))/2/A
   z=z-c
   return(z)
def bereken():
   global xmi,xma,ymi,yma,m,s,n
   C.delete(ALL)
   n=int(E1.get());bp=float(E2.get())/100;z=normalcdf((1+bp)/2)
# proportie
   if k==0:
```

```
        wis(E3);x=int(E4.get());p=x/n
        mp=p;sp=sqrt(p*(1-p)/n) # gemiddelde mp en staf sp aanpassen
# gemiddelde
    else:
        wis(E4);m,s=mult(E3.get())
        mp=m;sp=s/sqrt(n)
# marges
    marg=z*sp;ond=mp-marg;bov=mp+marg
    wis(E5);E5.insert(0,format(mp,'.3f')+','+format(sp,'.3f'))
    wis(E6);E6.insert(0,format(ond,'.3f')+','+format(bov,'.3f'))
    xmi,xma,ymi,yma=-4,4,-0.05,1/sqrt(2*pi)+0.05
# kromme
    lis=[];stap=0.05;x=xmi
    while x<xma:
        x=x+stap;y=snd(x);X=transx(x);Y=transy(y)
        lis.append(X);lis.append(Y)
    C.create_line(lis,fill='red')
# arcering
    x=-z;stap=0.01
    while x<=z:
        lis=[];y=0.002;X=transx(x);Y=transy(y);lis.append(X)
        lis.append(Y);y=snd(x)-0.002;X=transx(x);Y=transy(y)
        lis.append(X);lis.append(Y);x=x+stap
        C.create_line(lis,fill='yellow' )
# rooster
    for x in range(int(xmi),int(xma+1)):
        X=transx(x);Y=transy(0)
        C.create_line(X,Y,X,0,fill='green',dash=[1,2])
# assen
    X=transx(0);Y=transy(0)
    C.create_line(X,Y,X,0,fill='blue',arrow='last')
    Y=transy(0);C.create_line(0,Y,breedte,Y,fill='blue',arrow='last')
# tekst
    for i in range(1,8):
        xpos=i*breedte/8;tek=format(mp+(i-4)*sp,'.2f')
        C.create_text(xpos,hoogteC-25,text=tek)
        C.create_text(xpos,hoogteC-10,text=mu+vc(i-4)+si)
def vc(x):
    if x==1:s='+'
    elif x==-1:s='-'
    elif x>=0:s='+'+str(x)
    else:s=str(x)
    return s
def selec():global k;k=var.get()
def info():h=messagebox.showinfo(tit,infstr)
def wis(E):E.delete(0,len(E.get()))
def nieuw():
    for E in(E1,E2,E3,E4,E5,E6):wis(E)
    C.delete(ALL)
# hoofdprogramma
breedte=480;hoogte=400;hoogteC=hoogte-90;breedteC=breedte-5;hoInv1=hoogte-40;hoInv2=hoogte-70
```

```
mu='\u03BC';si='\u03C3'
form=Tk();form.geometry(str(breedte)+'x'+str(hoogte))
tit='Betrouwbaarheidsintervallen';form.title(tit)
C=Canvas(form, bg="white", height=hoogteC, width=breedteC);C.pack()
Label(form,text='n:').place(x=5,y=hoInv2)
E1=Entry(form,width=7);E1.place(x=40,y=hoInv2);E1.insert(0,'250')
L2=Label(form,text='bp%:',);L2.place(x=5,y=hoInv1)
E2=Entry(form,width=7);E2.place(x=40,y=hoInv1);E2.insert(0,'95')
Label(form,text='x̄,s:').place(x=190,y=hoInv1)
E3=Entry(form,width=10);E3.place(x=215,y=hoInv1);E3.insert(0,'10,2')
Label(form,text='x:').place(x=190,y=hoInv2)
E4=Entry(form,width=10);E4.place(x=215,y=hoInv2);E4.insert(0,'175')
Label(form,text=mu+','+si+':').place(x=285,y=hoInv2)
E5=Entry(form,bg='lightgrey',width=14);E5.place(x=320,y=hoInv2)
Label(form,text='[..,..]:').place(x=285,y=hoInv1)
E6=Entry(form,bg='lightgrey',width=14);E6.place(x=320,y=hoInv1)
B1=Button(form,text='Info',command=info,width=6);B1.place(x=420,y=hoogteC+6)
B2=Button(form,text='Bereken',command=bereken,width=6);B2.place(x=420,y=hoogteC+33)
B3=Button(form,text='Nieuw',command=nieuw,width=6);B3.place(x=420,y=hoogteC+60)
var=IntVar();k=0;keuze=['proportie','gemiddelde']
for i in range(0,2):
    rb=Radiobutton(form,text=keuze[i],variable=var,value=i,command=selec)
    rb.place(x=100,y=i*30+20+hoogteC)
infstr='n=omvang steekproef bp%=betrouwbaarheids-%\n'
infstr=infstr+'Kies je voor proporties dan is x= #successen\n'
infstr=infstr+'p̂=x/n is dan de steekproefproportie\n'
infstr=infstr+'Kies je voor gemiddelde,dan zijn x̄ en s gemiddelde\n'
infstr=infstr+'en standaardafwijking van de steekproef\n'
infstr=infstr+"Van de populatie zijn resp.de p̂'s of x̄'s\n"
infstr=infstr+'terug normaal verdeeld met nieuwe '+ mu+','+si+'\n'
infstr=infstr+'Proporties:    '+mu+' =p̂ en '+si+'=√((1-p̂).p̂/n)\n'
infstr=infstr+'Gemiddelde: '+mu+' =x̄ en '+si+'=√(s/n)\n'
infstr=infstr+'Van de populatie kan men met een zekerheid van\n'
infstr=infstr+'bp% zeggen dat de proportie, resp. gemiddelde in\n'
infstr=infstr+'het betrouwbaarheidsinterval [..,..] liggen'
form.mainloop()
```



89. Toetsen van hypothesen toetsen_hypothesen

Met dit programma worden enkel die gevallen behandeld die kunnen opgelost worden met de normale verdeling. Dwz. toetsen voor de populatieproportie p en toetsen voor μ bij bekende σ.

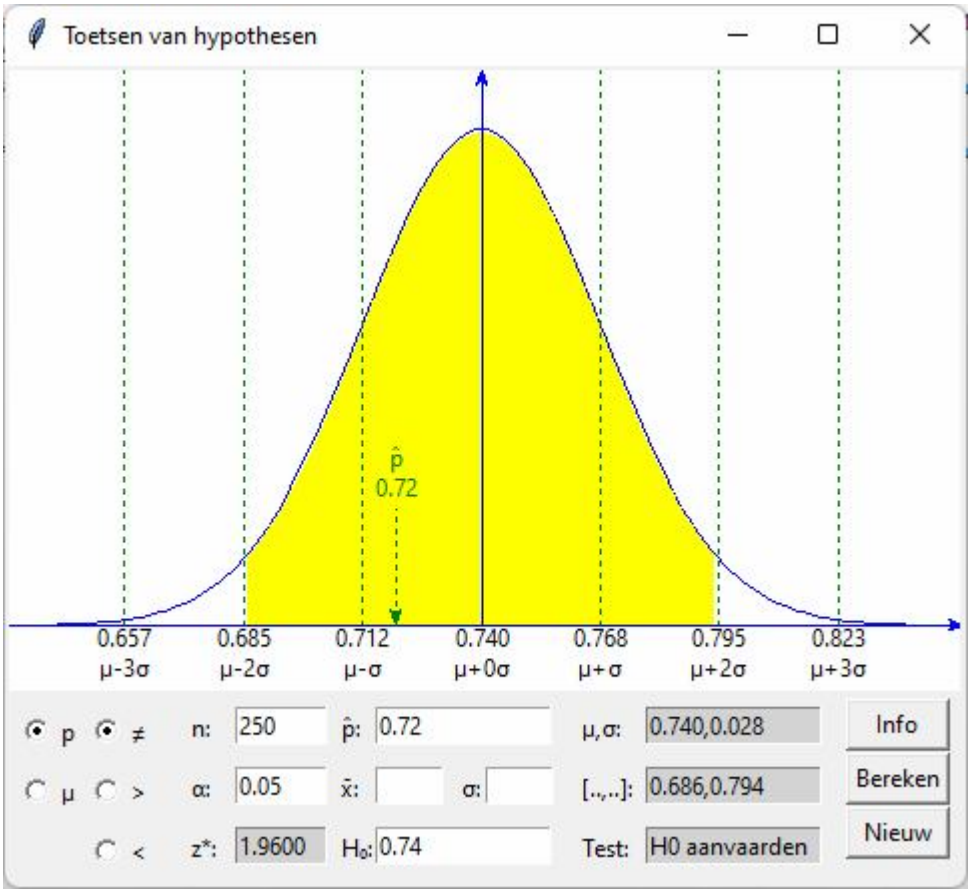
Programma:

```
# toetsen van hypothesen (enkel met normale verdeling)
from math import *;from tkinter import *;from tkinter import messagebox
def transx(x):return (x-xmi)*breedteC/(xma-xmi)
def transy(y):return hoogteC-hoogteC*(y-ymi)/(yma-ymi)
def snd(x):return e**(-x**2/2)/sqrt(2*pi)
def normalcdf(p):
    z=0;h=0.001;opp=0.5;f1=1/sqrt(2*pi)
    while opp<=p:
        z=z+h;f2=snd(z);opp=opp+h*(f1+f2)/2;f1_oud=f1;f1=f2
    # berekening correctie
    A=f1_oud-f2;B=2*h*f2;C=-2*h*(opp-p);D=B*B-4*A*C
    c=(-B+sqrt(D))/2/A;z=z-c
    return(z)
def bereken():
    global xmi,xma,ymi,yma,m,s,n
    C.delete(ALL);n=int(E1.get())
    alfa=float(E2.get());bp=1-alfa
    if k2==0:z=normalcdf((1+bp)/2)
    else:z=normalcdf(bp)
```

```
    hypotS=E7.get();hypot=eval(hypotS)
    wis(E9);E9.insert(0,format(z,'.4f'))
# proportie
    if k==0:
        wis(E3);p_hat=eval(E4.get());p=hypot
        mp=p_hat;sp=sqrt(p*(1-p)/n) # gemiddelde mp en staf sp aanpassen
# gemiddelde
    else:
        wis(E4);m=float(E3.get());s=float(E10.get())
        mp=m;sp=s/sqrt(n)
# marges
    marg=z*sp
    ond=format(hypot-marg,'.3f');bov=format(hypot+marg,'.3f')
    if k2==1:ond='-∞'
    if k2==2:bov='+∞'
    wis(E5);E5.insert(0,format(hypot,'.3f')+','+format(sp,'.3f'))
    wis(E6);E6.insert(0,ond+','+bov)
    xmi,xma,ymi,yma=-4,4,-0.05,1/sqrt(2*pi)+0.05
# kromme
    lis=[];stap=0.05;x=xmi
    while x<xma:
        x=x+stap;y=snd(x);X=transx(x);Y=transy(y)
        lis.append(X);lis.append(Y)
    C.create_line(lis,fill='blue')
# arcering
    if k2==0:beg=-z;end=z
    elif k2==1:beg=-4;end=z
    else: beg=-z;end=4
    x=beg;stap=0.01
    while x<end:
        lis=[];y=0.002;X=transx(x);Y=transy(y);lis.append(X)
        lis.append(Y);y=snd(x)-0.002;X=transx(x);Y=transy(y)
        lis.append(X);lis.append(Y);x=x+stap
        C.create_line(lis,fill='yellow' )
# rooster
    for x in range(-4,4):
        X=transx(x);Y=transy(0)
        C.create_line(X,Y,X,0,fill='green',dash=[1,2])
# assen
    X=transx(0);Y=transy(0)
    C.create_line(X,Y,X,0,fill='blue',arrow='last')
    Y=transy(0);C.create_line(0,Y,breedte,Y,fill='blue',arrow='last')
# tekst
    xpos=0;i=0
    while xpos<breedteC-breedteC/8:
        xpos=xpos+breedteC/8;i=i+1
        tek=format(hypot+(i-4)*sp,'.3f')
        C.create_text(xpos,hoogteC-25,text=tek)
        C.create_text(xpos,hoogteC-10,text='μ'+vc(i-4)+'σ')
    z=(mp-hypot)/sp;xpos=(z+4)*breedteC/8
    if beg==-4:beg=-1E6
```

```
    if end==4:end=1E6
    if z<beg or z>end:tek='H0 verwerpen';col='red'
    else:tek='H0 aanvaarden';col='green'
    C.create_text(xpos,hoogteC-100,text=mp,fill=col)
    if k==0:naam='p̂'
    else:naam='x̄'
    C.create_text(xpos,hoogteC-115,text=naam,fill=col)
    C.create_line(xpos,hoogteC-30,xpos,hoogteC-90,dash=[1,2],arrow='first',fill=col)
    wis(E8);E8.insert(0,tek)
def vc(x):
    if x==1:s='+'
    elif x==-1:s='-'
    elif x>=0:s='+'+str(x)
    else:s=str(x)
    return s
def selec():
    global k;k=var.get()
    if k==0:wis(E3);wis(E10)
    if k==1:wis(E4)
def selec2():global k2;k2=var2.get()
def info():h=messagebox.showinfo(tit+': info',infstr)
def wis(E):E.delete(0,len(E.get()))
def nieuw():
    for E in(E1,E2,E3,E4,E5,E6,E7,E8,E9,E10):wis(E)
    C.delete(ALL)
# hoofdprogramma
breedte=480;hoogte=410;hoogteC=hoogte-100;breedteC=breedte-5;hoInv1=hoogte-60;hoInv2=hoogte-
90;hoInv3=hoogte-30
lg='lightgrey'
form=Tk();form.geometry(str(breedte)+'x'+str(hoogte))
tit='Toetsen van hypothesen';form.title(tit)
C=Canvas(form, bg="white", height=hoogteC, width=breedteC);C.pack()
Label(form,text='n:').place(x=90,y=hoInv2)
E1=Entry(form,width=7);E1.place(x=115,y=hoInv2);E1.insert(0,'250')
L2=Label(form,text='α:',);L2.place(x=90,y=hoInv1)
E2=Entry(form,width=7);E2.place(x=115,y=hoInv1);E2.insert(0,'0.05')
Label(form,text='x̄:').place(x=165,y=hoInv1)
E3=Entry(form,width=5);E3.place(x=185,y=hoInv1)
Label(form,text='p̂:').place(x=165,y=hoInv2)
E4=Entry(form,width=14);E4.place(x=185,y=hoInv2);E4.insert(0,'0.72')
Label(form,text='H0:').place(x=165,y=hoInv3)
E7=Entry(form,width=14);E7.place(x=185,y=hoInv3);E7.insert(0,'0.74')
Label(form,text='μ,σ:').place(x=285,y=hoInv2)
E5=Entry(form,bg=lg,width=14);E5.place(x=320,y=hoInv2)
Label(form,text='[..,..]:').place(x=285,y=hoInv1)
E6=Entry(form,bg=lg,width=14);E6.place(x=320,y=hoInv1)
Label(form,text='Test:').place(x=285,y=hoInv3)
E8=Entry(form,bg=lg,width=14);E8.place(x=320,y=hoInv3)
Label(form,text='z*:').place(x=90,y=hoInv3)
E9=Entry(form,bg=lg,width=7);E9.place(x=115,y=hoInv3)
Label(form,text='σ:').place(x=225,y=hoInv1)
```

```
E10=Entry(form,width=5);E10.place(x=240,y=hoInv1)
B1=Button(form,text='Info',command=info,width=6);B1.place(x=420,y=hoogteC+6)
B2=Button(form,text='Bereken',command=bereken,width=6);B2.place(x=420,y=hoogteC+33)
B3=Button(form,text='Nieuw',command=nieuw,width=6);B3.place(x=420,y=hoogteC+60)
var=IntVar();k=0;keuze=['p','μ']
for i in range(0,2):
   rb=Radiobutton(form,text=keuze[i],variable=var,value=i,command=selec)
   rb.place(x=5,y=i*30+hoInv2)
var2=IntVar();k2=0;keuze2=['≠','>','<']
for i in range(0,3):
   rb2=Radiobutton(form,text=keuze2[i],variable=var2,value=i,command=selec2)
   rb2.place(x=40,y=i*30+hoInv2)
infstr='n=omvang steekproef α= significantie\n'
infstr=infstr+'Kies je voor proporties p dan is\n'
infstr=infstr+'p̂=x/n de steekproefproportie\n'
infstr=infstr+'Kies je voor gemiddelde μ, dan is x̄ het\n'
infstr=infstr+'gemiddelde van de steekproef en σ de\n'
infstr=infstr+'standaardafwijking van de populatie\n'
infstr=infstr+'Kies ook H1≠H0 of H1>H0 of H1<H0\n'
infstr=infstr+"Van de populatie zijn resp.de p̂'s of x̄'s\n"
infstr=infstr+'terug normaal verdeeld met nieuwe μ,σ\n'
infstr=infstr+'Proporties: μ =H0 en σ=√((1-p̂).p̂/n)\n'
infstr=infstr+'Gemiddelde: μ =H0 en σ=√(s/n)\n'
infstr=infstr+'H0=de hypothese die we testen\n'
infstr=infstr+'Ze zal aanvaard worden als p̂ of x̄\n'
infstr=infstr+'in het gebied [μ-z*.σ,μ+z*.σ] (≠) of\n'
infstr=infstr+']-∞,μ+z*.σ] (>) of [μ-z*.σ,+∞[ (<) ligt'
form.mainloop()
```



90. Discrete kansverdelingen kans

In het programma worden enkele kansverdelingen van de TI84 overgenomen.

binompdf(n,p,i): $P(X = i) = C_n^i \cdot p^i \cdot (1-p)^{n-i}$

binomcdf(n,p,i): $P(X \le i) = \sum_{j=0}^{i} C_n^j \cdot p^j \cdot (1-p)^{n-j}$

invbinom('kans',n,p): binomcdf berekenen tot als $P(X \le i) >$ kans.

geometpdf(p,k) = P(n=k)= $(1- p)^{k-1} \cdot p$

geometcdf(p,k) = P(n≤k)= $\sum_{j=1}^{k} (1-p)^{j-1} \cdot p$

poissonpdf(λ, k) = P(x=k) = $\frac{e^{-\lambda}.2^k}{k!}$

poissoncdf(λ, k) = P(x≤ k) = $\sum_{j=0}^{k} \frac{e^{-\lambda}.2^j}{j!}$

Hoe werk je ermee: selecteer eerst een kansverdeling in de lijst, klik op 'Invoer': dan zie je de parameters die je moet invoeren, klik dan op 'Bereken'

Programma:

```
# discrete kansfuncties
from math import *; from tkinter import*
```

```
def mult(s):
   l=s.split(',');co=[]
   for i in l:co.append(float(i))
   return co
def P(n):
   if n==1 or n==0:return 1
   else:return n*P(n-1)
def C(n,p):
   c=1;n=int(n);p=int(p)
   for i in range(0,p):c=c*(n-i)//(i+1)
   return c
def binompdf(n,p,i):return C(n,i)*p**i*(1-p)**(n-i)
def binomcdf(n,p,i):
   som=0;i=int(i)
   for j in range(0,i+1):som=som+binompdf(n,p,j)
   return som
def invbinom(k,n,p):
   som=0;i=0
   while som<k:i=i+1;som=som+binompdf(n,p,i)
   return i
def geometpdf(p,k):return (1-p)**(k-1)*p
def geometcdf(p,k):
   som=0;k=int(k)
   for j in range(1,k+1):som=som+geometpdf(p,j)
   return som
def poissonpdf(lamb,k):return exp(-lamb)*lamb**k/P(k)
def poissoncdf(lamb,k):
   som=0;k=int(k)
   for j in range(0,k+1):som=som+poissonpdf(lamb,j)
   return som
def invoer():
   global ke;Label(form,text='          ').place(x=180,y=hoInv)
   ke=Lb.curselection()[0];Label(form,text=param[ke]+':').place(x=180,y=hoInv)
   nieuw();ad(algemeen[0]);ad(welke[ke])
def bereken():
   inv=E1.get();ke=Lb.curselection()[0]
   if ke==0:n,p,i=mult(inv);ant=binompdf(n,p,i)
   if ke==1:n,p,i=mult(inv);ant=binomcdf(n,p,i)
   if ke==2:k,n,p=mult(inv);ant=invbinom(k,n,p)
   if ke==3:p,k=mult(inv);ant=geometpdf(p,k)
   if ke==4:p,k=mult(inv);ant=geometcdf(p,k)
   if ke==5:lamb,k=mult(inv);ant=poissonpdf(lamb,k)
   if ke==6:lamb,k=mult(inv);ant=poissoncdf(lamb,k)
   wis(E2);E2.insert(0,format(ant,'.6f'))
   return
def wis(E):E.delete(0,len(E.get()))
def ad(lin):tex.insert(INSERT,lin+'\n')
def nieuw():
   wis(E1);wis(E2);tex.delete('1.0',END)
#main
breedte=330;hoogte=140;hoInv=5;ke=0;j=0
```

```
form=Tk();form.geometry(str(breedte)+'x'+str(hoogte));form.title('Kansrekenen')
nml=['binompdf','binomcdf','invbinom','geometpdf','geometcdf','poissonpdf','poissoncdf']
param=['n,p,i','n,p,i','k,n,p','p,k','p,k','λ,k','λ,k']
welke=['kans op i-keer succes P(x=i)','kans op ten hoogste i-keer succes P(x≤k)']
welke.append("k=kans op minder dan 'antw' succes");welke.append('kans op succes na k pogingen P(N=k)')
welke.append('kans op succes bij hoogstens k pogingen P(N≤k)');welke.append('kans op k/T als gemiddeld
λ/T')
welke.append('kans op ten hoogste k/T als gemiddeld λ/T')
algemeen=['n=aantal p=kans i=#succes']
tupfun=tuple(nml)
Lb=Listbox(form,listvariable=StringVar(value=tupfun),selectmode='Single',height=3,width=12,bg='lightgre
y')
Lb.place(x=10,y=hoInv)
vsb=Scrollbar(form,orient=VERTICAL,command=Lb.yview,width=25);Lb['yscrollcommand']=vsb.set;vsb.p
lace(x=90,y=hoInv)
tex=Text(form,bg='grey89',height=3,width=38,wrap=WORD);tex.place(x=10,y=70)
E1=Entry(form,width=16);E1.place(x=220,y=hoInv)
B1=Button(form,text='Invoer',width=6,command=invoer);B1.place(x=120,y=hoInv)
E2=Entry(form,width=16,bg='lightgrey');E2.place(x=220,y=hoInv+30)
B2=Button(form,text='Bereken',width=6,command=bereken);B2.place(x=120,y=hoInv+30)
form.mainloop()
```

91. Verloop van veeltermfuncties

veeltermfunctiesTk

Gegeven een veeltermfunctie $y = f(x) = a_0x^n + a_1x^{n-1} + ... + a_{n-1}x^1 + a_nx^0$.

De afgeleide is dan : $f'(x) = n.a_0x^{n-1} + (n-1)a_1x^{n-2} + ... + 2.a_{n-2}.x^1 + a_{n-1}x^0$

En de 2de afgeleide : $f''(x) = n.(n-1).a_0x^{n-2} + (n-1).(n-2).a_1x^{n-3} + ... + 2.a_{n-2}x^0$

Als cof de lijst coëfficiënten van f(x) is, dan kunnen we met het volgende stukje programma de lijst codf van de coëfficiënten van f ' en de lijst cod2f van de coëfficiënten van f '' bepalen:

```
codf=[];cod2f=[]
for i in range(0,n):
    codf.append((n-i-1)*cof[i])
    cod2f.append((n-i-2)*codf[i])
codf.pop();cod2f.pop();cod2f.pop()
```

Om het verloop van de functie, d.w.z. ligging t.o.v. x-as, stijgen en dalen, en de buigpunten te bepalen, moeten we van de 3 functies f, f ' en f '' eerst de reële nulpunten berekenen. Hiervoor benaderen we met

Newton delers van de vorm $x-x_1$, bepalen we het quotiënt met Horner, waarvoor we dan terug Newton toepassen, enz...

De benaderingsfunctie $x - \frac{f(x)}{f'(x)} = x - \frac{a_0x^n + a_1x^{n-1} + ... + a_{n-1}x^1 + a_nx^0}{n.a_0x^{n-1} + (n-1)a_1x^{n-2} + ... + 2.a_{n-2}.x^1 + a_{n-1}x^0}$

$$= \frac{(n-1).a_0.x^n + (n-2).a_1.x^{n-1} + (n-3).a_2.x^{n-2} + ...1.a_{n-2}.x^2 + 0.a_{n-1}.x^1 - 1.a_n.x^0}{n.a_0x^{n-1} + (n-1)a_1x^{n-2} + ... + 2.a_{n-2}.x^1 + a_{n-1}x^0}$$

Met het volgende programma-onderdeel bereken je deze breuk:

```
T=0;N=0
for i in range(n,1,-1):
    T=T+(i-1)*a[n-i]*x**i
    N=N+i*a[n-i]*x**(i-1)
T=T-a[n];N=N+a[n-1]
if N!=0:x=T/N
```

Om een deler van de vorm $x-x_1$ te zoeken, berekenen we bvb. functiewaarden van -10 tot 9 en nemen als startwaarde het argument waarvoor de functiewaarde van teken verandert, dus als we in de buurt van een nulpunt zijn. Deze methode is zeker niet volmaakt, maar zal wel voldoen voor bijna alle mogelijke veeltermfuncties.

```
for x in range(-10,10):
        x=x+0.1
        if val(x,a,len(a))*val(x+1,a,len(a))<0:st=x
```

Als we de regel van Horner toepassen, vervangen we eigenlijk de coëfficiëntenlijst a van het deeltal door de coëfficiëntenlijst van het quotiënt om hiervoor dan terug een deler te zoeken van de vorm $x-x_2$.

```
na=[a[0]]
for i in range(1,m):na.append(a[i]+na[i-1]*x)
na.pop();a=na;m=len(a)
```

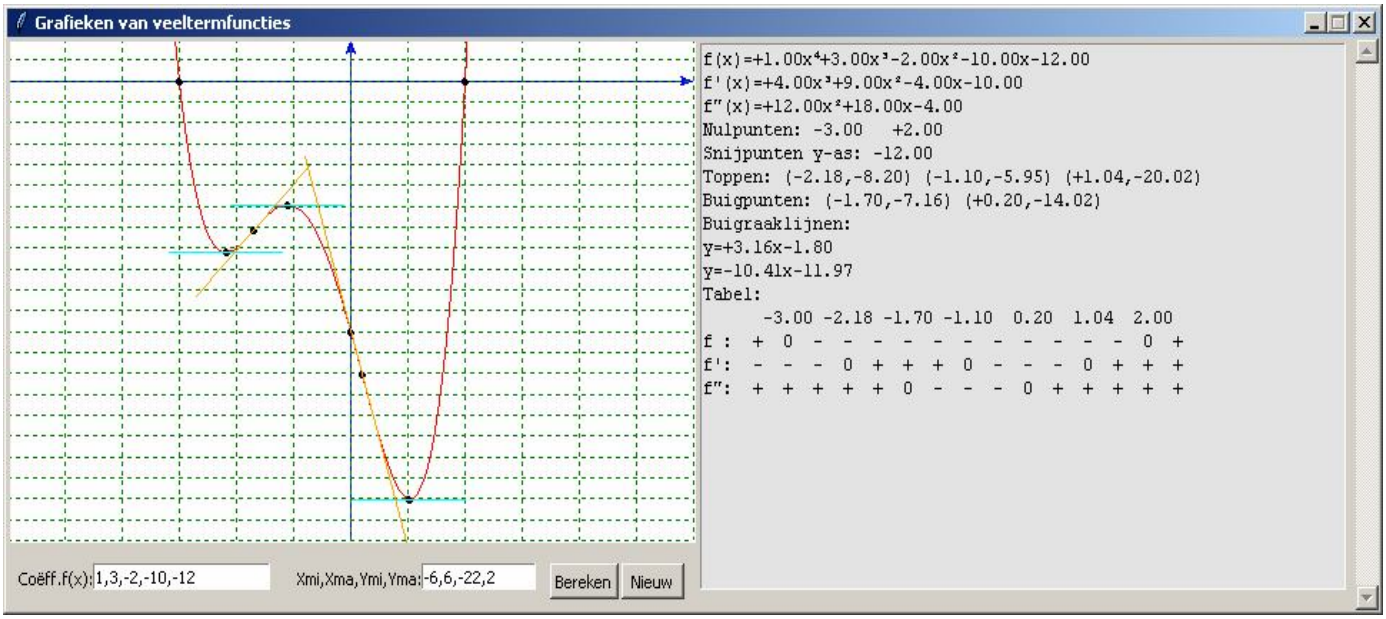
Programma:

```
# Verloop van veeltermfuncties
from math import sqrt;from tkinter import *
def mult(s):
   l=s.split(',');co=[]
   for i in l:co.append(float(i))
   return co
def transx(x):return (x-xmi)*wid/(xma-xmi)
def transy(y):return heigC-heigC*(y-ymi)/(yma-ymi)
def newton(a):
   global it
   dx=0.0001;x=st+1;ox=x+1;n=len(a)-1;it=0
   while abs(x-ox)>dx and it<50:
      ox=x;T=0;N=0;it=it+1
      for i in range(n,1,-1):
         T=T+(i-1)*a[n-i]*x**i
         N=N+i*a[n-i]*x**(i-1)
```

```
        T=T-a[n];N=N+a[n-1]
        if N!=0:x=T/N
    return x
def val(x,lico,n):
    value=0
    for i in range(0,n):value=value+lico[i]*x**(n-i-1)
    return value
def f(x):return val(x,cof,n)
def df(x): return val(x,codf,n-1)
def d2f(x):return val(x,cod2f,n-2)
def ad(lin):tex.insert(INSERT,lin+' ')
def v(x):
    if x>=0:sgn='+'
    else:sgn=''
    return sgn+format(x,'.2f')
def sg(a):
    if abs(a)<0.0001:t=' 0'
    elif a>0:t=' +'
    elif a<0:t=' -'
    return t
def vergel(lis):
    equ='';n=len(lis)
    for i in range(0,n):
        p=n-i-1;equ=equ+v(lis[i])
        if p>1:equ=equ+'x'+supdig[p]
        elif p==1:equ=equ+'x'
    return equ
def factor(a):
    global it,st
    m=len(a);sol=[];it=0
     # afsplitsen factoren x-x1
    while it<50 and m>3:
        st=100
        for x in range(-10,10):
            x=x+0.1
            if val(x,a,len(a))*val(x+1,a,len(a))<0:st=x
        x=newton(a)
        if it<50:
            na=[a[0]]
            for i in range(1,m):na.append(a[i]+na[i-1]*x)
            na.pop();a=na;m=len(a);sol.append(x)
    if m==3:
        p=a[1]/a[0];q=a[2]/a[0]
        d=p*p-4*q
        if d>0:
            root=sqrt(d);x1,x2=(-p-root)/2,(-p+root)/2
            sol.append(x1);sol.append(x2)
        elif d==0:
            x1=x2=-p/2
            sol.append(x1)
    if m==2:x=-a[1]/a[0];sol.append(x)
```

```
    return simp(sol)
def simp(lis):
    lis.sort();ind=[]
    for j in range(1,len(lis)):
        if abs(lis[j]-lis[j-1])<0.001:ind.append(j)
    sol2=[]
    for j in range(0,len(lis)):
        if j not in ind:sol2.append(lis[j])
    return sol2
def graf():
    global cof,codf,cod2f,n,nulf,nuldf,nuld2f,xli,sol
    cof=mult(E1.get());n=len(cof);codf=[];cod2f=[];sol=[]
    tex.delete('1.0',END)
    for i in range(0,n):
        codf.append((n-i-1)*cof[i])
        cod2f.append((n-i-2)*codf[i])
    codf.pop();cod2f.pop();cod2f.pop()
    ad('f(x)='+vergel(cof))
    ad('\nf'+chr(39)+'(x)='+vergel(codf))
    ad('\nf'+chr(34)+'(x)='+vergel(cod2f))
    nulf,nuldf,nuld2f=factor(cof),factor(codf),factor(cod2f)
    ad('\nNulpunten:');nulf.sort()
    for x in nulf:ad(v(x)+'  ')
    ad('\nSnijpunten y-as:')
    ad(v(f(0)))
    ad('\nToppen:')
    for x in nuldf:ad('('+v(x)+','+v(f(x))+')')
    ad('\nBuigpunten:')
    for x in nuld2f:ad('('+v(x)+','+v(f(x))+')')
    ad('\nBuigraaklijnen:')
    for x in nuld2f:
        rl=v(df(x))+'x'+v(f(x)-x*df(x))
        ad('\ny='+rl)
    ad('\nTabel:\n     ')
    xli=nulf+nuldf+nuld2f;xli=simp(xli)
    for x in xli:
        xs=format(x,'.2f');l=len(str(xs))
        ad(xs.rjust(13-2*l))
    eps=0.01;xtab=[min(xli)-eps]
    for x in xli:xtab.append(x);xtab.append(x+eps)
    ad('\nf :')
    for x in xtab:ad(sg(f(x)))
    ad('\nf'+chr(39)+':')
    for x in xtab:ad(sg(df(x)))
    ad('\nf'+chr(34)+':')
    for x in xtab:ad(sg(d2f(x)))
    teken()
def teken():
    global xmi,xma,ymi,yma
    xmi,xma,ymi,yma=mult(E2.get());C.delete(ALL)
#assen
```

```
    X=transx(0);line = C.create_line(X,0,X,heigC,fill='blue',arrow='first')
    Y=transy(0);line = C.create_line(0,Y,wid,Y,fill='blue',arrow='last')
# rooster
    for x in range(int(xmi),int(xma+1)):
        X=transx(x);line = C.create_line(X,0,X,heigC,fill='green',dash=[1,2])
    for y in range(int(ymi),int(yma+1)):
        Y=transy(y);line = C.create_line(0,Y,wid,Y,fill='green',dash=[1,2] )
#kromme
    krom(f,'red')
def krom(fu,col):
    lis=[];stap=0.05;x=xmi
    while x<xma:
        x=x+stap;y=fu(x);X=transx(x);Y=transy(y);lis.append(X);lis.append(Y)
    line = C.create_line(lis,fill=col)
    for x in xli:#punten
        y=f(x);X,Y=transx(x),transy(y);C.create_oval(X-2,Y-2,X+2,Y+2,fill='black')
    X,Y=transx(0),transy(f(0));C.create_oval(X-2,Y-2,X+2,Y+2,fill='black')#snijpunt y-as
    for x in nuldf:#raaklijnen in toppen
        y=f(x);X1=transx(x-1);X2=transx(x+1);Y1=transy(y)
        line = C.create_line(X1,Y1,X2,Y1,fill='cyan')
    for x in nuld2f:#buigraaklijnen
        X1=transx(x-1);Y1=transy(f(x)-df(x));X2=transx(x+1);Y2=transy(f(x)+df(x))
        line = C.create_line(X1,Y1,X2,Y2,fill='orange')
def nieuw():
     E2.delete(0,len(E2.get()));E1.delete(0,len(E1.get()));C.delete(ALL);tex.delete('1.0',END)
#hoofdprogramma
supdig=['\u2070','\u00B9','\u00B2','\u00B3','\u2074','\u2075','\u2076','\u2077','\u2078','\u2079']
wid=480;heig=365;heigC=heig-45;hoInv=heig-30
form=Tk();form.geometry(str(960)+'x'+str(heig));form.title('Grafieken van veeltermfuncties')
form.resizable(False,False)
C = Canvas(form, bg="white", height=heigC, width=wid);C.place(x=0,y=0)
L1=Label(form,text='Coëff.f(x):',);L1.place(x=5,y=hoInv)
E1=Entry(form);E1.place(x=60,y=hoInv);E1.insert(0,'1,3,-2,-10,-12')
L2=Label(form,text='Xmi,Xma,Ymi,Yma:',);L2.place(x=200,y=hoInv)
E2=Entry(form);E2.place(x=290,y=hoInv,width=80);E2.insert(0,'-6,6,-25,5')
tex=Text(form,bg='grey89',height=23,width=64,wrap=WORD);tex.place(x=wid+5,y=3)
vsb=Scrollbar(form,orient=VERTICAL,command=tex.yview,width=35)
tex['yscrollcommand']=vsb.set;vsb.pack(side=RIGHT,fill='y')
B1=Button(form,text='Bereken',command=graf);B1.place(x=380,y=hoInv)
B2=Button(form,text=' Nieuw ',command=nieuw);B2.place(x=430,y=hoInv)
form.mainloop()
```



92. Kansrekenen: simulatie van het gooien met 2 dobbelstenen dobbelsteen

Met dit programma kan de gebruiker het opgooien van 2 dobbelstenen simuleren. De uitkomstenverzameling van dit kansexperiment is {2,3,4,5,6,7,8,9,10 11,12}.
De wiskundige kans op elk van deze uitkomsten kan gemakkelijk berekend worden met een uitkomstentabel. De kansen zijn:

$$P(2)=P(12)=\frac{1}{36} \quad P(3)=P(11)=\frac{2}{36} \quad P(4)=P(10)=\frac{3}{36} \quad P(5)=P(9)=\frac{4}{36} \quad P(6)=P(8)=\frac{5}{36} \quad P(7)=\frac{6}{36}$$

Deze kansen worden onder de grafiek per kolom afgedrukt. Bovendien wordt hiermee een histogram getekend, dat overeenstemt met 500 opgooien.

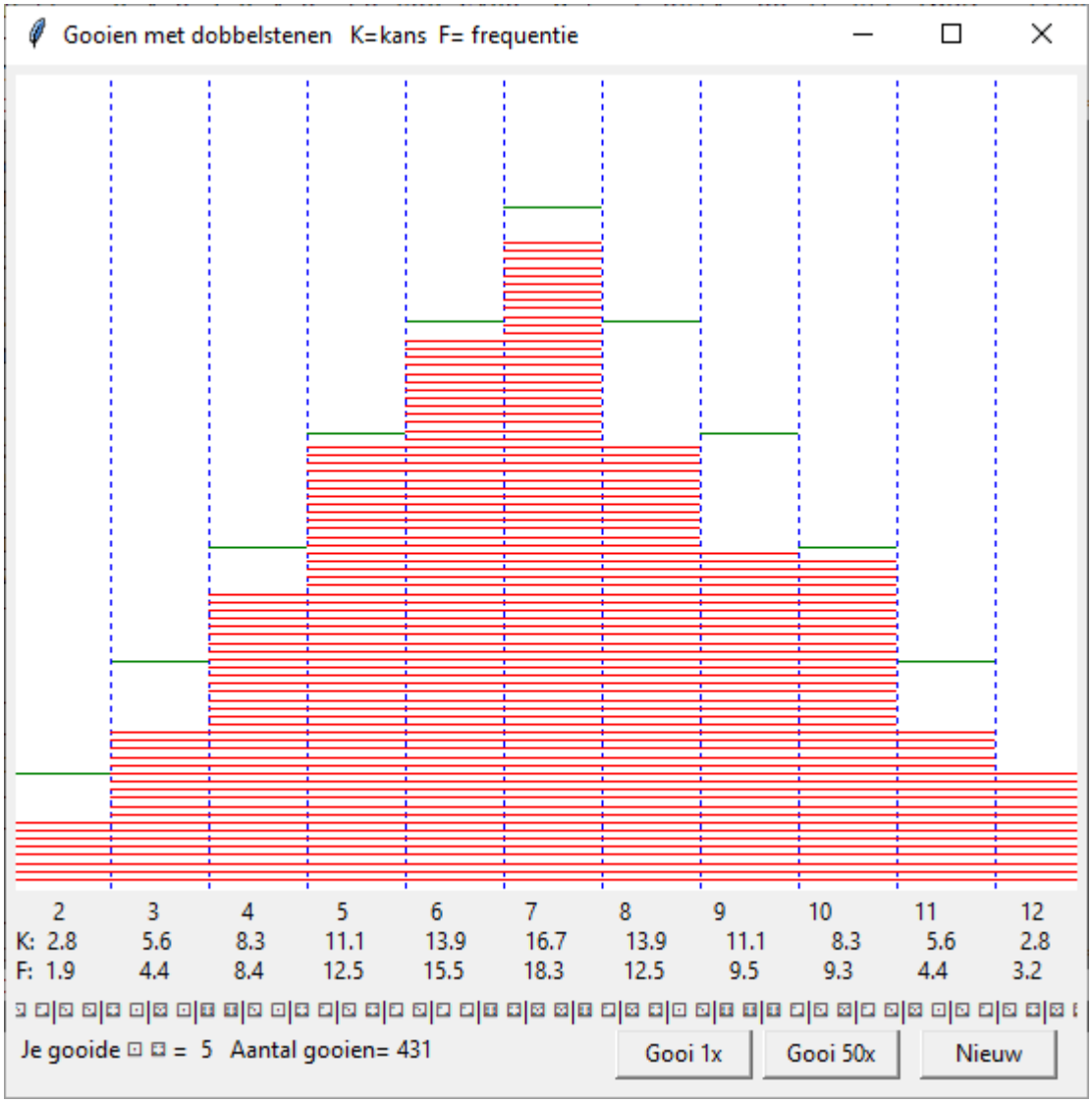
Kiest hij voor 1 gooi, dan zie je dat de gooi wordt voorgesteld door 2 dobbelsteentjes , en wordt er een streepje bijgetekend in de juiste kolom. Bovendien komt er onder de kanslijn een lijn met de relatieve frequenties. Je kan ook telkens kiezen voor 50 gooien. Zelfs als er bvb. 431 keer gegooid wordt, is er nog steeds een verschil tussen de relatieve frequentie en de wiskundige kans. Maar geleidelijk zie je wel een overeenkomst ontstaan tussen beide histogrammen.

Programma:

```
# opgooien dobbelstenen
from random import *; from math import *; from tkinter import *
def mult(s):
        l=s.split(',')
        co=[]
        for i in l:co.append(float(i))
        return co
def transx(x):return (x-xmi)*breedte/(xma-xmi)
def transy(y):return hoogteC-hoogteC*(y-ymi)/(yma-ymi)
def lijn(sc,kl):
    xl[sc]=xl[sc]+1
    x1=transx(sc-1);x2=transx(sc)
    y1=transy(xl[sc]);y2=y1
    line = C.create_line(x1,y1,x2,y2,fill=kl)
    freqtek='F: '
    for i in range(2,13):
```

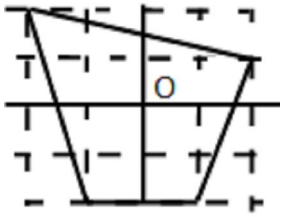
```
            freqst=format(xl[i]*100/aantal,'.1f')
            if len(freqst)<4:freqst=' '+freqst
            freqtek=freqtek+freqst+'          '
      L4=Label(form,text=freqtek)
      L4.place(x=2,y=honv)
def kans():
      #volgens de kansberekening
      for i in range(1,7):
            x=i;x1=transx(x);x2=transx(x+1)
            y=i*500/36;y1=transy(y);y2=transy(y)
            line = C.create_line(x1,y1,x2,y2,fill='green')
            x=12-i;x1=transx(x);x2=transx(x+1)
            line = C.create_line(x1,y1,x2,y2,fill='green')
            kanstek='K: '
      for i in range(2,13):
            if i<8:kans=100*(i-1)/36
            else:kans=100*(13-i)/36
            kt=format(kans,'.1f')
            if len(kt)==3:kt=' '+kt
            kanstek=kanstek+kt+'          '
      L3=Label(form,text=kanstek)
      L3.place(x=2,y=honv-15)
def dashen():
            for x in range(1,12):
                  x1=transx(x);x2=x1
                  y1=transy(ymi);y2=transy(yma)
                  line = C.create_line(x1,y1,x2,y2,fill='blue',dash=[1,2])
def gooi():
            global a,b,xmi,xma,ymi,yma,aantal,gtek
            a=randint(1,6);b=randint(1,6)
            sc=a+b
            g=d[a]+' '+d[b]
            gtek=gtek+'|'+g
            L2=Label(form,text=gtek)
            L2.place(x=5,y=honv+20,width=breedte-10)
            aantal=aantal+1;stsc=str(sc)
            if len(stsc)<2:stsc=' '+stsc
            L5=Label(form,text='Je gooide '+g+' = '+stsc+'   Aantal gooien= '+str(aantal))
            L5.place(x=5,y=honv+40)
            #rechte
            lijn(sc,'red')
def wis():
      global xl,ymi,yma,aantal,gtek;ymi=0;yma=100;aantal=0;gtek=''
      C.delete(ALL)
      L2=Label(form,text=(' ').ljust(breedte-10))
      L2.place(x=5,y=honv+20,width=breedte-10)
      L5=Label(form,text=(' ').ljust(100))
      L5.place(x=5,y=honv+40)
      xl=[0,0,0,0,0,0,0,0,0,0,0,0]
def spel():
            i=0
            while i<50:
                  i=i+1
                  gooi()
def nieuw():
```

```
    wis();dashen();kans()
#hoofdprogramma
xl=[0,0,0,0,0,0,0,0,0,0,0,0,0]
d=['','\u2680','\u2681','\u2682','\u2683','\u2684','\u2685']
breedte=550;hoogte=525;hoogteC=hoogte-110;honv=hoogte-75
form=Tk();form.geometry(str(breedte)+'x'+str(hoogte))
form.title('Gooien met dobbelstenen   K=kans  F= frequentie')
C = Canvas(form, bg="white",height=hoogteC, width=breedte-10);C.place(x=3,y=3);sctek='     '
for sci in range(2,13):sctek=sctek+str(sci)+'              '
L0=Label(form,text=sctek);L0.place(x=3,y=honv-30)
xmi=1;xma=12;ymi=0;yma=100;aantal=0;gtek=''
B1=Button(form,text='Gooi 1x',command=gooi);B1.place(x=310,y=honv+40,width=70)
B2=Button(form,text='Nieuw',command=nieuw);B2.place(x=465,y=honv+40,width=70)
B3=Button(form,text='Gooi 50x',command=spel);B3.place(x=385,y=honv+40,width=70)
kans();dashen()
form.mainloop()
```

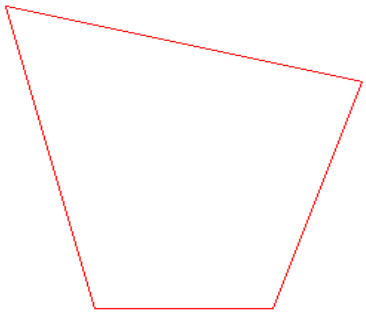


93. Rotatie van een veelhoek rotatieveelhoek

In dit eerste rotatieprogramma moet de gebruiker de koppels coördinaten van enkele punten invoeren



Om de vierhoek hiernaast te tekenen, voeren we in:
2,1,-2,2,-1,-2,1,-2 en met **Teken** krijgen we de vierhoek op het scherm.



We kunnen een rotatiehoek invoeren, bvb 20
Met de knop **Draai** wordt de grafiek volledig rondgedraaid
De rotatieformules om de nieuwe coördinaten te berekenen zijn
x_1 = x.cos(t)-y.sin(t) en y_1=x.sin(t)+y.cos(t)
Deze formule moet op elk van de hoekpunten worden uitgevoerd.

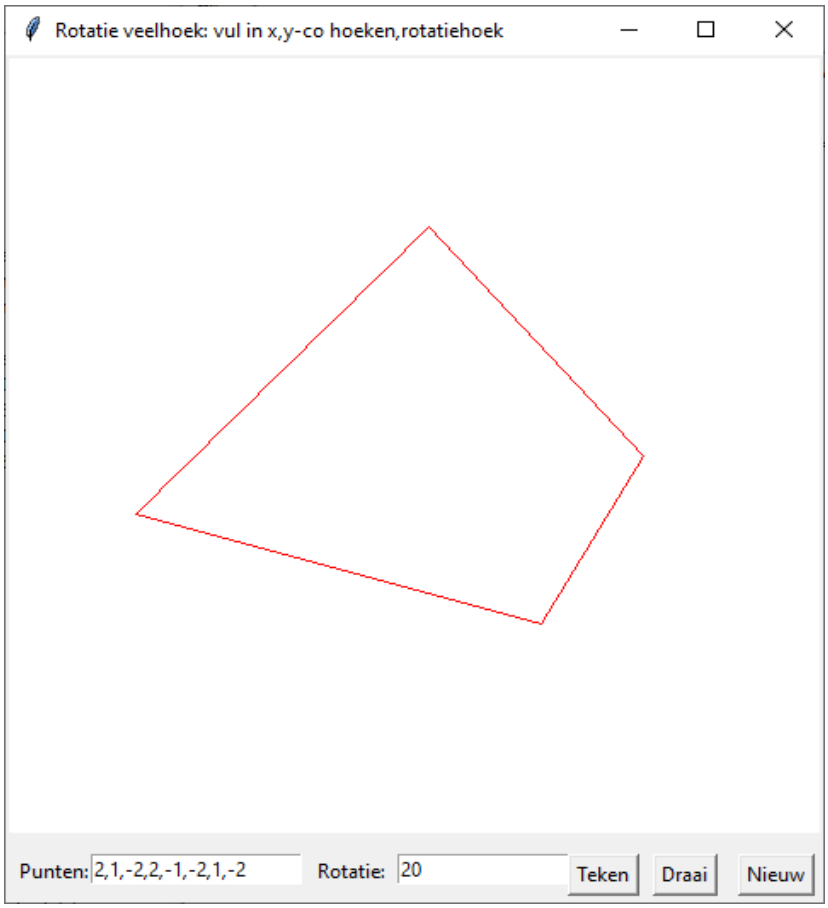
De functie **kromme(lijst)** vult de 'lijst' van ingevulde koppels coördinaten eerst aan met de coördinaten van het eerste punt. Deze worden dan met transx en transy omgezet naar schermcoördinaten en ingevuld in een nieuwe lijst die met C.create_line getekend wordt.

De functie **trans()** gebruikt de rotatieformules (zie hierboven) om de lijst lis om te zetten naar een nieuwe lijst listdrn van de coördinaten van de hoekpunten van de gedraaide rechthoek. Telkens we op de knop **Draai** klikken wordt f met 1 verhoogd en draait de vierhoek een stand verder.

Programma:

```
# draaien van een meetkundige figuur
from math import *; from tkinter import *
def mult(s):
    l=s.split(',')
    co=[]
    for i in l:co.append(float(i))
    return co
def transx(x): return (x-xmi)*breedte/(xma-xmi)
def transy(y): return hoogteC-hoogteC*(y-ymi)/(yma-ymi)
def kromme(lijst):
    lijst.append(lijst[0]);lijst.append(lijst[1])
    lis2=[];i=0
    for co in lijst:
        if i % 2==0:lis2.append(transx(co))
        else:lis2.append(transy(co))
        i=i+1
    C.delete(ALL)
    line = C.create_line(lis2,fill='red')
def teken():
    global lis,ldr,xmi,xma,ymi,yma,f
    lis=mult(E1.get());d=max(min(lis),max(lis))+2
    xmi,ymi=-d,-d;xma,yma=d,d;kromme(lis);f=0
def trans():
    t=radians(float(E2.get()))
    le=len(lis);lisdrn=[]
    for i in range(0,le-2,2):
        lisdrn.append(lis[i]*cos(f*t)-lis[i+1]*sin(f*t))
        lisdrn.append(lis[i]*sin(f*t)+lis[i+1]*cos(f*t))
    kromme(lisdrn)
def draai():
    global f;f=f+1;trans()
```

```
def nieuw():
    E1.delete(0,len(E1.get()));E2.delete(0,len(E2.get()));C.delete(ALL)
#hoofdprogramma
breedte=480;hoogte=525;hoogteC=hoogte-45;hoInv=hoogte-30
form=Tk();form.geometry(str(breedte)+'x'+str(hoogte))
form.title('Rotatie veelhoek: vul in x,y-co hoeken,rotatiehoek')
C = Canvas(form, bg="white", height=hoogteC, width=breedte);C.pack()
L1=Label(form,text='Punten:',);L1.place(x=5,y=hoInv)
E1=Entry(form);E1.place(x=50,y=hoInv)
L2=Label(form,text='Rotatie:',);L2.place(x=180,y=hoInv)
E2=Entry(form);E2.place(x=230,y=hoInv)
B1=Button(form,text='Teken',command=teken);B1.place(x=330,y=hoInv)
B3=Button(form,text='Draai',command=draai);B3.place(x=380,y=hoInv)
B2=Button(form,text='Nieuw',command=nieuw);B2.place(x=430,y=hoInv)
form.mainloop()
```



94. Rotatie van de grafiek van een poolvergelijking rotatie_pool

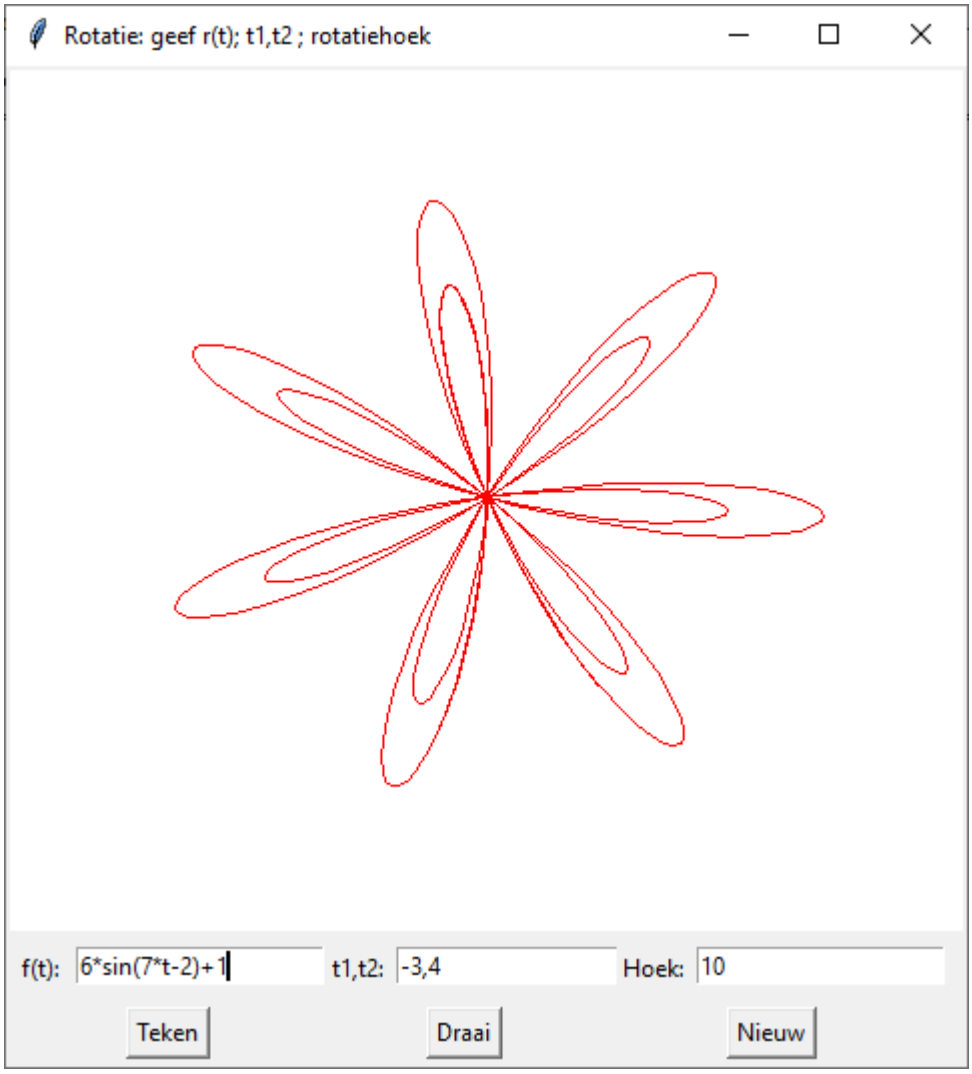
In dit programma worden de rotatieformules toegepast op de lijst van de coördinaten van de poolgrafiek. Het is +/- de samenvoeging van 2 vorige programma's

Programma:

```
#tekenen van de grafiek van een poolvergelijking + draaiing
from math import *; from tkinter import *
```

```
def mult(s):
    l=s.split(',')
    co=[]
    for i in l:co.append(float(i))
    return co
def transx(x): return (x-xmi)*breedte/(xma-xmi)
def transy(y): return hoogteC-hoogteC*(y-ymi)/(yma-ymi)
def r(t): return eval(E1.get())
def kromme(lijst):
    lis2=[];i=0
    for co in lijst:
        if i % 2==0:lis2.append(transx(co))
        else:lis2.append(transy(co))
        i=i+1
    line = C.create_line(lis2,fill='red')
def teken():
    global lis,ldr,xmi,xma,ymi,yma,f
    tmi,tma=mult(E2.get())
    lis=[]
    stap=0.02;t=tmi
    xmi,xma,ymi,yma=-10,10,-10,10
    while t<tma:
        t=t+stap
        x=r(t)*cos(t);y=r(t)*sin(t)
        lis.append(x);lis.append(y)
    C.delete(ALL)
    kromme(lis)
    f=0
def trans():
    ldr=lis;hoek=radians(float(E3.get()))
    le=len(lis);lisdrn=[]
    print(f)
    for i in range(0,le-2,2):
        lisdrn.append(lis[i]*cos(f*hoek)-lis[i+1]*sin(f*hoek))
        lisdrn.append(lis[i]*sin(f*hoek)+lis[i+1]*cos(f*hoek))
    C.delete(ALL)
    kromme(lisdrn)
def draai():
    global f
    f=f+1
    trans()
def nieuw():
     E1.delete(0,len(E1.get()));E2.delete(0,len(E2.get()))
     E3.delete(0,len(E3.get()));C.delete(ALL)
#hoofdprogramma
breedte=480;hoogte=500;hoogteC=hoogte-70;hoInv=hoogte-60;hoknop=hoogte-30
form=Tk();form.geometry(str(breedte)+'x'+str(hoogte));form.title('Rotatie: geef r(t); t1,t2 ; rotatiehoek ')
C = Canvas(form, bg="white", height=hoogteC, width=breedte);C.pack()
L1=Label(form,text='f(t):',);L1.place(x=5,y=hoInv)
E1=Entry(form);E1.place(x=35,y=hoInv)
L2=Label(form,text='t1,t2:',);L2.place(x=160,y=hoInv)
```

```
E2=Entry(form);E2.place(x=195,y=hoInv)
L3=Label(form,text='Hoek:',);L3.place(x=305,y=hoInv)
E3=Entry(form);E3.place(x=345,y=hoInv)
B1=Button(form,text='Teken',command=teken);B1.place(x=60,y=hoknop)
B2=Button(form,text='Draai',command=draai);B2.place(x=210,y=hoknop)
B3=Button(form,text='Nieuw',command=nieuw);B3.place(x=360,y=hoknop)
form.mainloop()
```



95. Een wetenschappelijk rekentoestel met tkinter rekenmachine

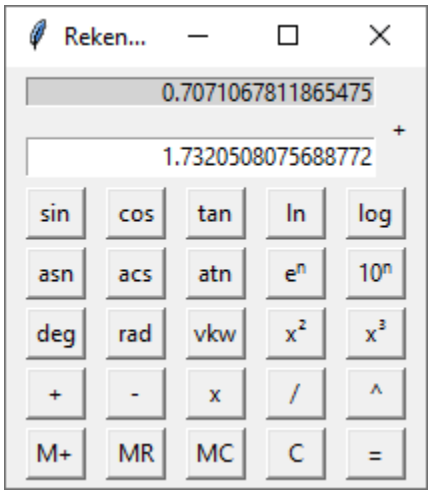
Bij dit rekentoestel wordt gebruik gemaakt van 2 entries, het onderste met witte achtergrond waarin getallen rechtstreeks ingetikt kunnen worden, het bovenste in lichtgrijs waarin eigenlijk niets mag ingevoerd worden. Als je een getal intikt en op één van de functietoetsen tikt,wordt het getal vervangen door de functiewaarde. Tik je op een bewerkingstoets (+,-,*,/,^),dan wordt het getal in het witte venster verplaatst naar het grijze venster. Je kan in het witte venster een tweede getal invoeren. Als je op = tikt wordt de bewerking uitgevoerd: het resultaat van 'grijs' 'bewerking' 'wit' komt in het witte venster.

Programma:

```
#rekentoestel
from random import *; from math import *; from tkinter import *
```

```
global a,b,c,m,bew;a='0.0';b='0.0';c='0.0';m='0.0'
breedte=210;hoogte=210;bew="
form=Tk();form.geometry(str(breedte)+'x'+str(hoogte));form.title('Rekentoestel')
E1=Entry(form,justify=RIGHT);E1.place(x=10,y=35,width=175,height=20)
E2=Entry(form,bg='lightgrey',justify=RIGHT);E2.place(x=10,y=5,width=175,height=15)
def lab(x):Label(form,text=x).place(x=190,y=20)
def inv(x):
    global a,bew;a=E1.get();bew=x
    deE1();deE2();E2.insert(0,a)
    lab('  ');lab(x)
def plus():inv('+')
def mi():inv('-')
def maal():inv('*')
def deel():inv('/')
def macht():inv('**')
def deE1():
    E1.delete(0,len(E1.get()))
def deE2():
    E2.delete(0,len(E2.get()))
def res():
    global a,b,bew;b=E1.get();deE1();deE2()
    lab('  ')
    if b!='' and bew!='':
        c=eval(a+bew+b);E1.insert(0,c)
    else:E1.insert(0,b)
    E2.insert(0,'')
    if bew!='':bew=''
def fun(f):
    a=E1.get();deE1()
    E1.insert(0,eval(f+'(float(a))'))
def si():fun('sin')
def co():fun('cos')
def ta():fun('tan')
def ln():fun('log')
def log10():fun('1/log(10)*log')
def ex():fun('exp')
def xt():fun('10**')
def asi():fun('asin')
def ac():fun('acos')
def at():fun('atan')
def vk():fun('sqrt')
def deg():fun('180/pi*')
def rad():fun('pi/180*')
def kwad():fun('(float(a))*')
def tri():fun('(float(a))**2*')
def memplus():
    global m;a=E1.get();m=str(float(m)+float(a))
def memrec():
    global m;deE1();a=m;E1.insert(0,a)
def memcl():
    global m;m='0.0'
```

```
def nieuw():
   deE1();deE2();m='0.0'
def but(xp,yp,t,co):
   Button(form,text=t,command=co).place(x=xp,y=yp,width=30)
but(10,60,'sin',si);but( 50,60,'cos',co);but(90,60,'tan',ta)
but(130,60,'ln',ln);but(170,60,'log',log10)
# superscript n='u207F'
but(130,90,'e\u207F',ex);but(170,90,'10\u207F',xt);but(10,90,'asn',asi)
but(50,90,'acs',ac);but(90,90,'atn',at)
but(10,120,'deg',deg);but(50,120,'rad',rad);but(90,120,'vkw',vk)
but(130,120,'x²',kwad);but(170,120,'x³',tri)
but(10,150,'+',plus);but(50,150,'-',mi);but(90,150,'x',maal)
but(130,150,'/',deel);but(170,150,'^',macht)
but(10,180,'M+',memplus);but(50,180,'MR',memrec);but(90,180,'MC',memcl)
but(130,180,'C',nieuw);but(170,180,'=',res)
form.mainloop()
```



96. Rekentoestel 2 rekenmachine 2

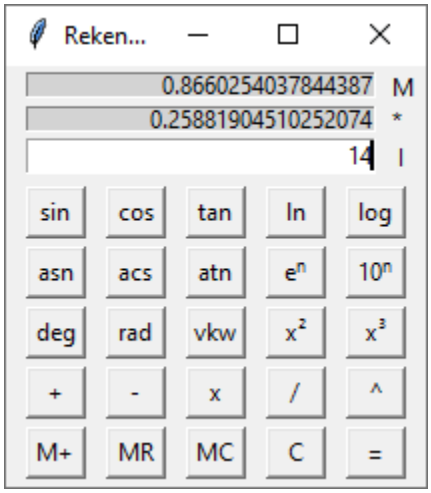
In deze 2de versie is er een venster voorzien om steeds de inhoud van het geheugen te zien. Een 2de verandering is het veel compacter maken van het programma. Het is bijvoorbeeld mogelijk om bij de definitie van een button een functie met een parameter in te geven: command=fun(co) lukt niet maar wel: command=lambda:fun(co) waarbij co een parameter is, bvb. met waarde 'tan'

def but(xp,yp,t,co):Button(form,text=t,command=lambda: fun(co)).place(x=xp*sc,y=yp*sc,width=3*sc)
kan opgeroepen worden met but(9,6,'tan','tan')
Ook Entries kunnen als parameter worden meegegeven, zoals:
def de(en):en.delete(0,len(en.get()))
de(E1)
sc = schaal: voor **python** (onder Windows) moet je een andere keuze maken dan voor **pydroid** (onder Android). Je kan best experimenteren met de waarden van sc.(pydroid is een recente versie van python op de smartphone,hij is gratis als je er de reclameboodschappen bij wilt nemen)

Programma:

```
#rekentoestel 2
from math import *; from tkinter import *
```

```
global a,b,c,m,bew,sc;a='0.0';b='0.0';c='0.0';m='0.0';sc=10
breedte=21*sc;hoogte=21*sc;bew="
form=Tk();form.geometry(str(breedte)+'x'+str(hoogte));form.title('Rekentoestel')
E1=Entry(form,justify=RIGHT);E1.place(x=sc,y=3.6*sc,width=17.5*sc,height=1.8*sc)
E2=Entry(form,bg='lightgrey',justify=RIGHT);E2.place(x=sc,y=2*sc,width=17.5*sc,height=1.3*sc)
E3=Entry(form,bg='lightgrey',justify=RIGHT);E3.place(x=sc,y=.3*sc,width=17.5*sc,height=1.3*sc)
def lab(t,h):Label(form,text=t).place(x=19*sc,y=h*sc)
lab(' I',3.5);lab('M',.0);lab('B ',1.7)
def inv(x):
 global a,bew;a=E1.get();bew=x
 de(E1);de(E2);E2.insert(0,a)
 lab('B ',1.7);lab(x,1.7)
def de(en):en.delete(0,len(en.get()))
def res():
 global a,b,bew;b=E1.get();de(E1);de(E2)
 lab('B ',1.7)
 if b!='' and bew!='':
    c=eval(a+bew+b);E1.insert(0,c)
 else:E1.insert(0,b)
 E2.insert(0,'')
 if bew!='':bew=''
def fun(f):
 a=E1.get();de(E1)
 E1.insert(0,eval(f+'(float(a))'))
def memplus():
 global m;a=E1.get();m=str(float(m)+float(a));de(E3);E3.insert(0,m)
def memrec():
 global m;de(E1);a=m;E1.insert(0,a)
def memcl():
 global m;m='0.0';de(E3);E3.insert(0,m)
def nieuw():
 de(E1);de(E2)
def but(xp,yp,t,co):Button(form,text=t,command=lambda: fun(co)).place(x=xp*sc,y=yp*sc,width=3*sc)
def but2(xp,yp,t,co):Button(form,text=t,command=lambda: inv(co)).place(x=xp*sc,y=yp*sc,width=3*sc)
def but3(xp,yp,t,co):Button(form,text=t,command=co).place(x=xp*sc,y=yp*sc,width=3*sc)
but(1,6,'sin','sin');but( 5,6,'cos','cos');but(9,6,'tan','tan')
but(13,6,'ln','log');but(17,6,'log','1/log(10)*log')
# superscript n='u207F'
but(1,9,'asn','asin');but(5,9,'acs','acos');but(9,9,'atn','atan')
but(13,9,'e\u207F','exp');but(17,9,'10\u207F','10**')
but(1,12,'deg','180/pi*');but(5,12,'rad','pi/180*');but(9,12,'vkw','sqrt')
but(13,12,'x²','(float(a))*');but(17,12,'x³','(float(a))**2*')
but2(1,15,'+','+');but2(5,15,'-','-');but2(9,15,'x','*')
but2(13,15,'/','/');but2(17,15,'^','**')
but3(1,18,'M+',memplus);but3(5,18,'MR',memrec);but3(9,18,'MC',memcl)
but3(13,18,'C',nieuw);but3(17,18,'=',res)
nieuw();memcl()
form.mainloop()
```

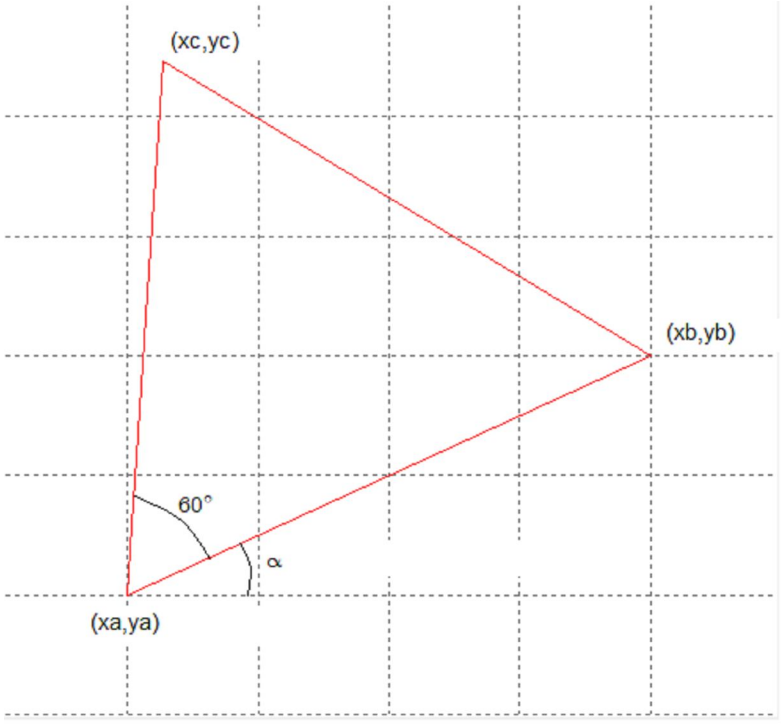


97. Fractalen: Koch, Sierpinski, Julia, Mandelbrot

Een fractaal is een meetkundige figuur die zelf-gelijkend is. M.a.w. we kunnen een fractaal opsplitsen in steeds kleinere figuren die gelijkvormig zijn aan het originele: bekende fractalen zijn deze van Koch, Sierpinski, Julia en Mandelbrot. De meeste functiedefinities zijn gemeenschappelijk voor de verschillende fractalen. Vandaar dat ze opgenomen zijn in een module 'fractbasis.py'
Opmerking: heel wat programma's in deze cursus bevatten heel wat gemeenschappelijke code. Je zou dit in afzonderlijke modules kunnen opnemen. Voordeel is dat de listings van de programma's veel korter worden. Een nadeel is misschien een verminderde leesbaarheid.

Berekenen van de coördinaten van het 3de hoekpunt van een gelijkzijdige driehoek.

Zowel voor Koch als Sierpinski hebben we een formule nodig die analytisch de coördinaten berekent van het 3de hoekpunt van een gelijkzijdige driehoek. Uitgaande van 2 punten A(xa,ya) en B(xb,yb) berekenen we de coördinaten van het 3de punt C(xc,yc).
In de module 'fractbasis' wordt daarvoor de functie hoek3(xa,ya,xb,yb) gebruikt.



Stel w=tan 60°=1.732 en $t = \tan(\alpha) = \frac{yb - ya}{xb - xa}$

$$p = \tan(\alpha + 60°) = \frac{\tan\alpha + \tan 60°}{1 - \tan\alpha.\tan 60°} = \frac{t + w}{1 - tw}$$

$$q = \tan(\alpha + 120°) = \frac{\tan\alpha + \tan 120°}{1 - \tan\alpha.\tan 120°} = \frac{\tan\alpha - \tan 60°}{1 + \tan\alpha.\tan 60°} = \frac{t - w}{1 + tw}$$

We vinden de coördinaten van het 3de punt (xc,yc) met het stelsel van de 2 vergelijkingen van de rechten ac en bc:

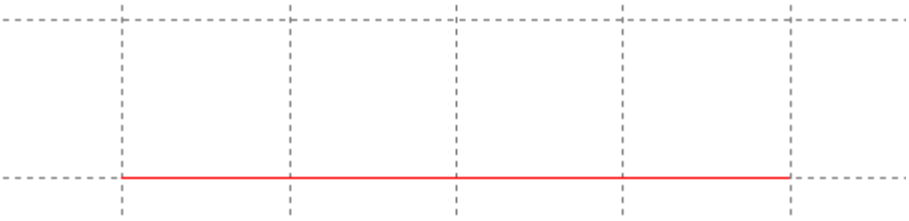
$$\begin{cases} yc - ya = p.(xc - xa) \,|\, 1 \\ yc - yb = q.(xc - xb) \,|\, -1 \end{cases} \begin{vmatrix} -q \\ p \end{vmatrix} \begin{vmatrix} yb - ya = (p - q).xc - p.xa + q.xb \\ (p - q).yc + q.ya - p.yb = p.q.(xa - xb) \end{vmatrix}$$

$$xc = \frac{yb - ya + p.xa - q.xb}{p - q} \qquad yc = \frac{p.q.(xa - yb) - q.ya + p.yb}{p - q}$$

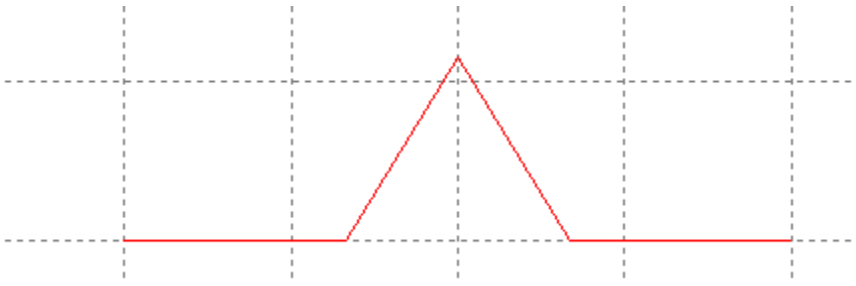
98. Koch

We starten van een lijnstuk, verdelen dit in 3 gelijke stukken,waarvan we het midden vervangen door de 2 overige zijden van een gelijkzijdige driehoek. Op deze 4 lijnstukken passen we weer dezelfde regel toe: in de volgende 4 tekeningen zien we stapsgewijze de fractaal van Koch ontstaan: theoretisch kan dit altijd verder verdeeld worden

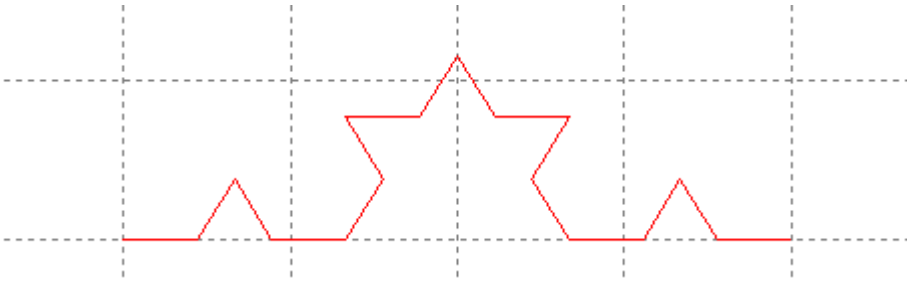
k=0



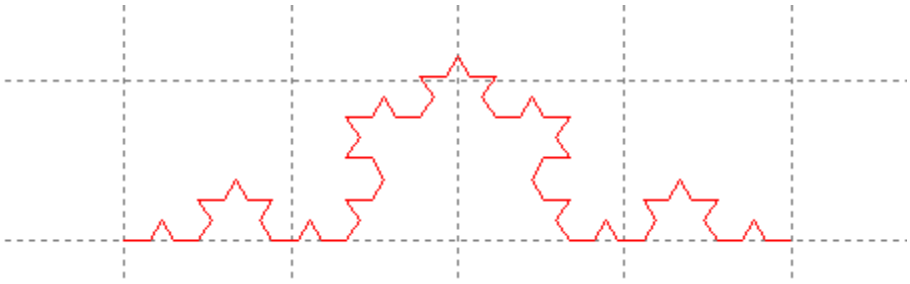
k=1



k=2



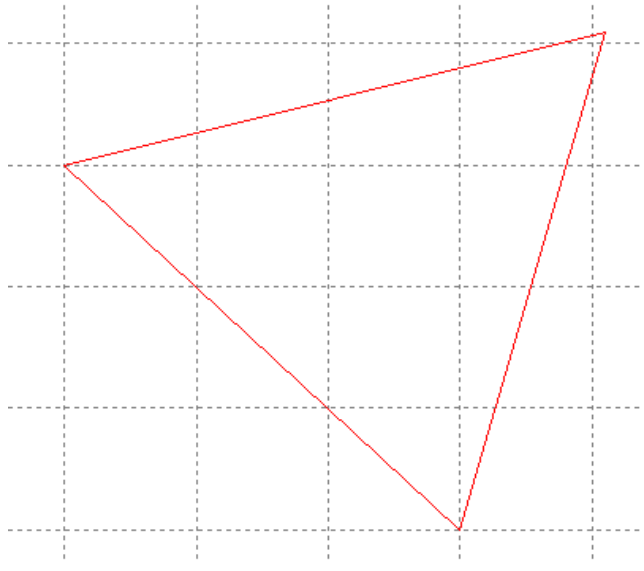
k=3

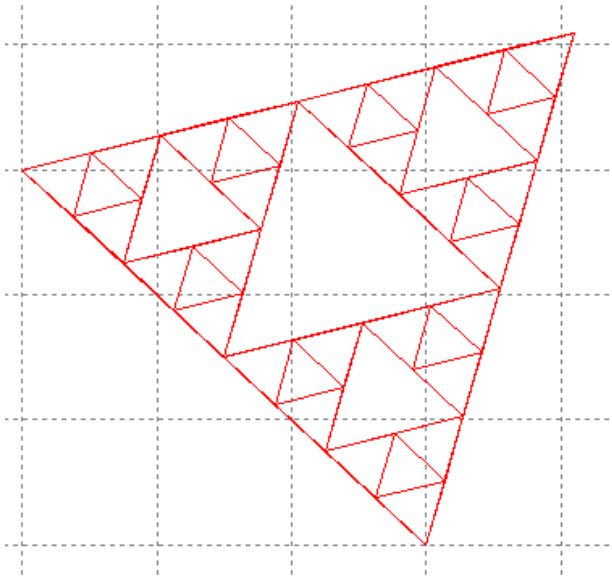


Hoe kunnen we dit programmeren: we starten met een lijst [x1,y1,x2,y2]
De grafiek met tkinter is een lijnstuk
We gaan hier telkens 3 punten x3,y3,x5,y5,x4,y4 tussenvoegen.
x3,y3 ligt op 1/3 van x1,y1 en 2/3 van x2,y2: x3=(2*x1+x2)/3 , y3=....
x4,y4 ligt op 2/3 van x1,y1 en 1/3 van x2,y2: x4=(x1+2*x2)/3 , y4=...
x5,y5 worden berekend als 3de hoekpunt van een gelijkzijdige driehoek met de functie hoek3(....)
De eerste 3 punten worden op positie 2 toegevoegd, de volgende op positie 10 (i=i+8)
Voor de volgende k-waarde wordt de rij opnieuw doorlopen en worden bij elke 2 opeenvolgende punten terug 3 punten tussengevoegd.

99. Sierpinski

We vertrekken ook hier van een lijnstuk [x1,y1,x2,y2]
We berekenen x3,y3 met hoek3(...)
De middens van de 3-zijden dienen als hoekpunten om opnieuw 3 driehoeken te tekenen:
door het recursief oproepen van dezelfde functieprocedure tkdrh voor l_1,l_2,l_3 waarbij telkens k=k-1 blijft dit doorgaan tot als k=0





100. Julia

Stel een functie $f(z) = z^2+c$ waarbij $z,c \in C$. Als we de recursieve rij z_1 , z_2 , z_3 ,.... opstellen waarbij $z_2 = z_1^2+c$
$z_3 = z_2^2+c$,... dan blijft het beeld binnen een zeker gebied in het complexe vlak, gaat dus niet naar ∞ .
De grens van het gebied waar de z_i niet naar ∞ gaan, is het Julia-fractaal.
Stel $z_1=x_1+y_1.i$ en $c_1=c_{x1}+c_{y1}.i$ ==> $z_1^2+c_1 = (x_1+y_1.i)^2 + c_{x1}+c_{y1}.i = x_1^2 - y_1^2 + c_{x1} + (2x_1y_1 + c_{y1}).i$
Stellen we de nieuwe $z_2=x_2+ y_2.i$ hieraan gelijk, dan is:

$x_2 = x_1^2 - y_1^2 + c_{x1}$ en $y_2 = 2x_1y_1 + c_{y1}$
Als we $x_2 - c_{x1} = a$ en $y_2 - c_{y1} = b$ stellen, dan zijn x_1 en y_1 oplossingen van het stelsel:

$$\begin{cases} x^2 - y^2 = a \\ 2xy = b \end{cases} \qquad y = \frac{b}{2x} \qquad x^2 - \left(\frac{b}{2x}\right)^2 - a = 0 \qquad x^4 - ax^2 - \frac{b^2}{4} = 0$$

stel $x^2 = t$

$$t^2 - at - \frac{b^2}{4} = 0 \quad D = B^2 - 4AC = a^2 + b^2 \quad t_1 = \frac{a + \sqrt{a^2 + b^2}}{2} \geq 0 \quad t_2 \leq 0 \quad x_1 = \pm\sqrt{\frac{a + \sqrt{a^2 + b^2}}{2}} \quad y_1 = \pm\frac{b}{2x_1}$$

Opgepast, hiermee berekenen we $z_1 = x_1 + y_1.i$ in functie van $z_2 = x_2 + y_2.i$
De rij z_1 , z_2 , z_3 , … wordt dus achterstevoren berekend.
Gaan we uit van z = 0 en berekenen we random één van beide oplossingen, dan krijgen we punten uit de rand van het convergentiegebied. Dit toont ons het Julia-fractaal.

101. Fractbasis.py fractbasis
Om de 3 soorten fractalen te tekenen (met tkinter) maken we gebruik van een module met de gemeenschappelijke code van de 3 programma's.

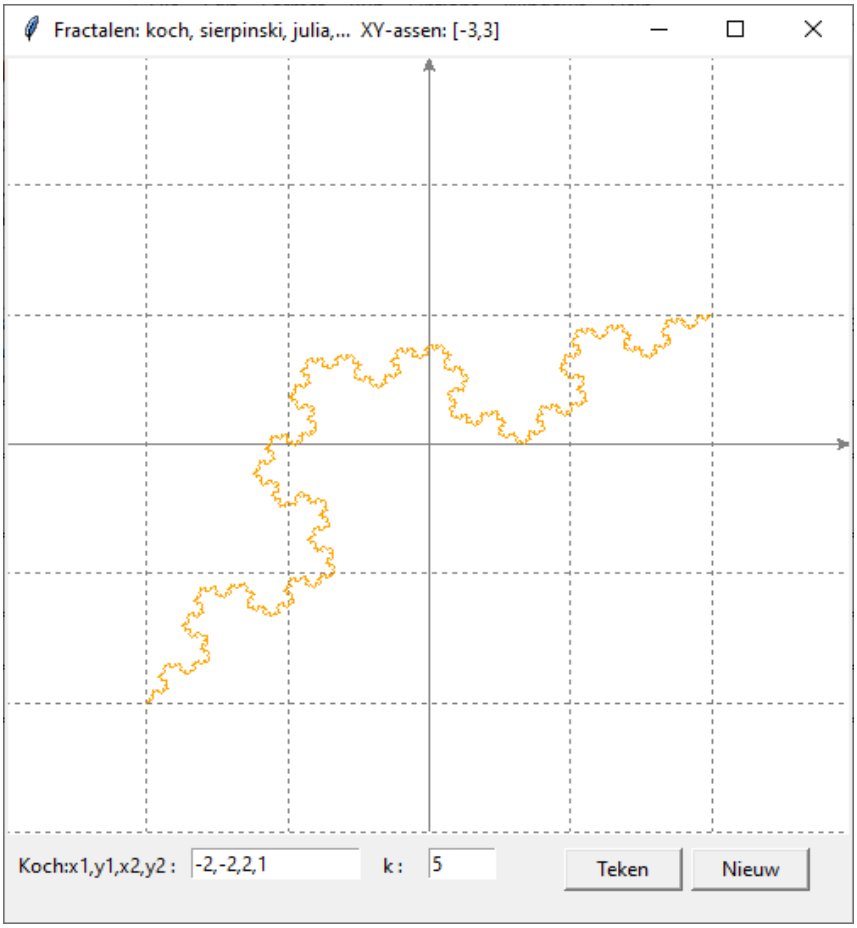
Programma:

```
# Basismodule fractalen: gemeenschappelijke code
w=1.732;xmi,ymi=-3,-3;xma,yma=3,3
kleur=['red','green','magenta','yellow','blue','orange']
from math import sqrt;from random import randint
from tkinter import *;from copy import deepcopy
def mult(s):
   l=s.split(',');co=[]
   for i in l:co.append(float(i))
   return co
def transx(x):return (x-xmi)*breedte/(xma-xmi)
def transy(y):return hoogteC-hoogteC*(y-ymi)/(yma-ymi)
def transd(l):
        tl=[]
        for i in range(0,len(l)):
                if i%2==0:tl.append(transx(l[i]))
                else:tl.append(transy(l[i]))
        tl.append(tl[0]);tl.append(tl[1])
        return tl
def hoek3(xa,ya,xb,yb):
   if xb-xa!=0:t=(yb-ya)/(xb-xa)
   else:t=100000
   p=(t+w)/(1-w*t);q=(t-w)/(1+w*t)
   xc=(yb-ya+p*xa-q*xb)/(p-q)
   yc=(p*q*(xa-xb)-q*ya+p*yb)/(p-q)
   return [xc,yc]
def assen():
        kl='grey';C.delete(ALL)
        X=transx(0);line = C.create_line(X,0,X,hoogteC,fill=kl,arrow='first')
```

```
        Y=transy(0);line = C.create_line(0,Y,breedte,Y,fill=kl,arrow='last')
        for x in range(int(xmi),int(xma+1)):
                X=transx(x);line = C.create_line(X,0,X,hoogteC,fill=kl,dash=[1,2])
        for y in range(int(ymi),int(yma+1)):
                Y=transy(y);line = C.create_line(0,Y,breedte,Y,fill=kl,dash=[1,2])
sc=1;breedte=500*sc;hoogte=525*sc;hoogteC=hoogte-55*sc;hoInv=hoogte-45*sc
form=Tk();form.geometry(str(breedte)+'x'+str(hoogte))
form.title('Fractalen: koch, sierpinski, julia  XY-assen: [-3,3]')
C=Canvas(form, bg="white", height=hoogteC, width=breedte);C.pack()
E1=Entry(form);E1.place(x=100*sc,y=hoInv,width=140*sc)
L2=Label(form,text='k:',);L2.place(x=250*sc,y=hoInv)
E2=Entry(form);E2.place(x=275*sc,y=hoInv,width=40*sc)
def nieuw():
    E1.delete(0,len(E1.get()));assen()
    E2.delete(0,len(E2.get()));
B2=Button(form,text='Nieuw',command=nieuw);B2.place(x=405*sc,y=hoInv,width=70*sc)
```

102. Koch programma koch

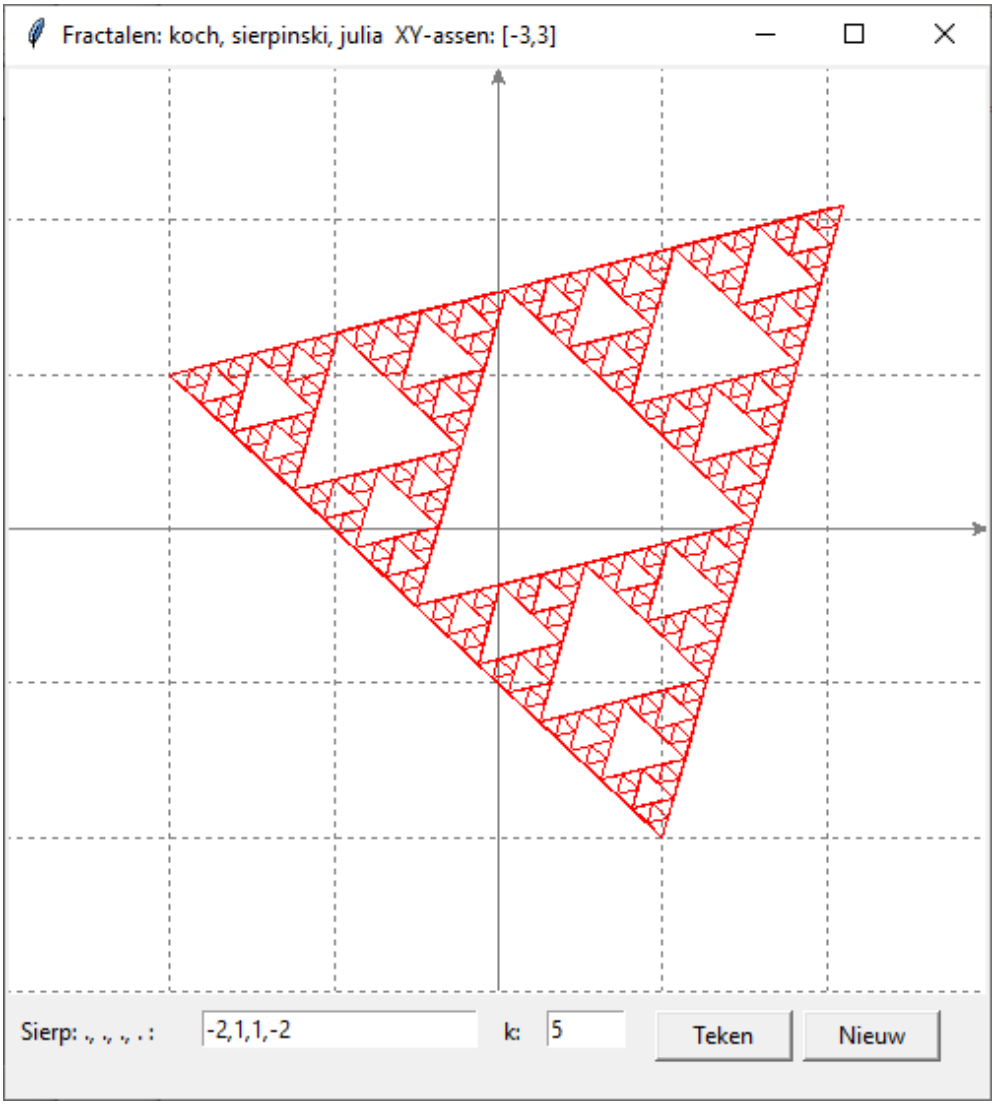
```
# koch
from fractbasis import *
def kromme(lijst):
        k=int(E2.get());l=lijst;kl=0
        while kl<k:
                kl=kl+1;i=0
                while i<len(l)-3:
                        x1,y1,x2,y2=l[i:i+4]
                        x3=(2*x1+x2)/3;y3=(2*y1+y2)/3
                        x4=(x1+2*x2)/3;y4=(y1+2*y2)/3
                        x5,y5=hoek3(x3,y3,x4,y4)
                        l.insert(i+2,x3);l.insert(i+3,y3)
                        l.insert(i+4,x5);l.insert(i+5,y5)
                        l.insert(i+6,x4);l.insert(i+7,y4)
                        i=i+8
        lis2=[]
        for i in range(0,len(l)):
          if i%2==0:lis2.append(transx(l[i]))
          else:lis2.append(transy(l[i]))
        assen();line = C.create_line(lis2,fill=kleur[k%6])
def teken():
   global lis,k;lis=mult(E1.get());kromme(lis)
L1=Label(form,text='Koch:x1,y1,x2,y2 :',);L1.place(x=5,y=hoInv)
L2=Label(form,text='k :');L2.place(x=220*sc,y=hoInv)
B1=Button(form,text='Teken',command=teken);B1.place(x=330*sc,y=hoInv,width=70*sc)
E1.insert(0,'-2,-2,2,1');E2.insert(0,'5');assen()
form.mainloop()
```



103. Sierpinski programma sierpinski

```
#sierpinski
from fractbasis import *
def tekdrh(l,k):
      tl=transd(l);line = C.create_line(tl,fill='red')
      lm=deepcopy(l);p=0.5;q=1-p
      if k>0:
            lm[0]=p*l[0]+q*l[2];lm[1]=p*l[1]+q*l[3]
            lm[2]=p*l[2]+q*l[4];lm[3]=p*l[3]+q*l[5]
            lm[4]=p*l[4]+q*l[0];lm[5]=p*l[5]+q*l[1]
            l1=[l[0],l[1],lm[0],lm[1],lm[4],lm[5]]
            l2=[l[2],l[3],lm[0],lm[1],lm[2],lm[3]]
            l3=[l[4],l[5],lm[2],lm[3],lm[4],lm[5]]
            k=k-1;tekdrh(l1,k);tekdrh(l2,k);tekdrh(l3,k)
def teken():
   global lis,k,p;lis=mult(E1.get());k=int(E2.get());assen()
   x3,y3=hoek3(lis[0],lis[1],lis[2],lis[3])
   lis.append(x3);lis.append(y3)
   tekdrh(lis,k)
#hoofdprogramma
B1=Button(form,text='Teken',command=teken);B1.place(x=330*sc,y=hoInv,width=70*sc)
L1=Label(form,text='Sierp: x1,y1,x2,y2 :',);L1.place(x=5,y=hoInv)
```

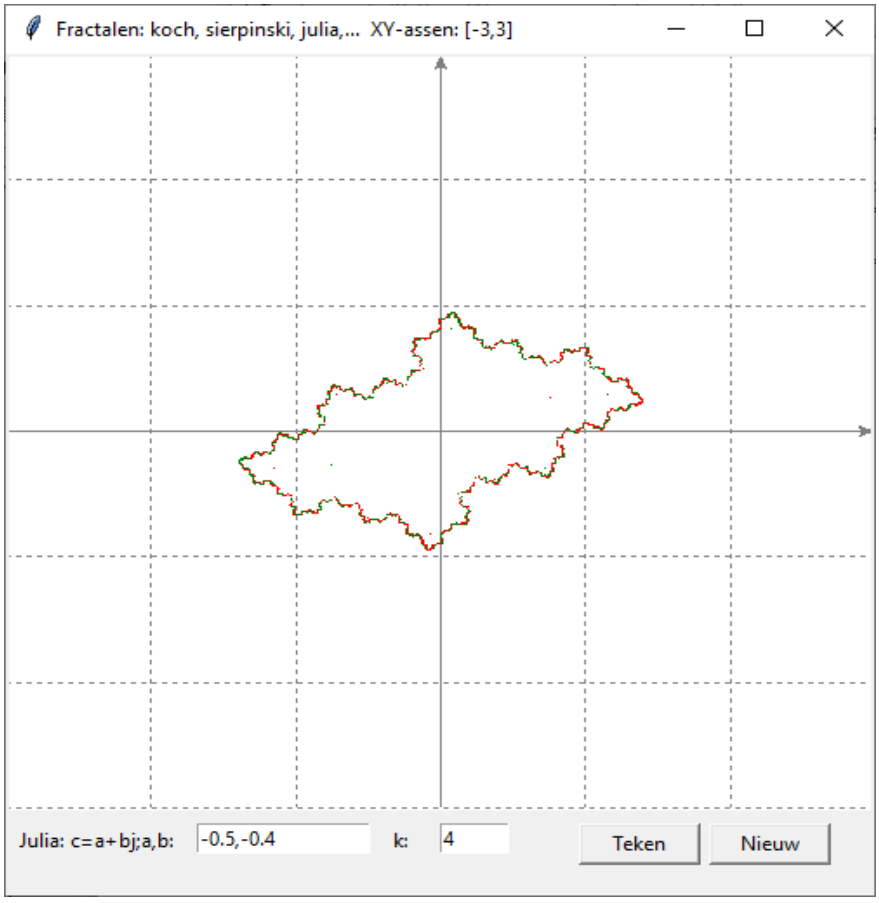
```
L2=Label(form,text='k :');L2.place(x=220*sc,y=hoInv)
E1.insert(0,'-2,1,1,-2');E2.insert(0,'4');assen()
form.mainloop()
```



104. Julia programma julia

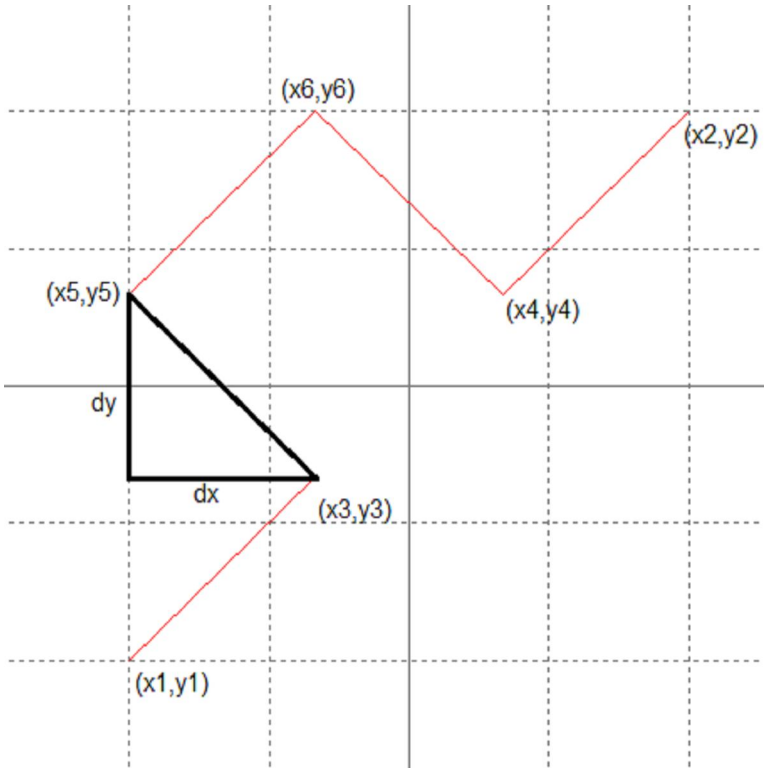
```
# julia
from fractbasis import *
def ap(x,y):
        X=transx(x);Y=transy(y);l=[]
        l.append(X);l.append(Y)
        l.append(X+1);l.append(Y+1)
        return l
def kromme(lijst,k):
        cx,cy=lijst
        i=0;max=10**k;zx=0;zy=0
        while i<max:
                lp=[];lq=[]
                i=i+1;a=zx-cx;b=zy-cy
                zx=(-1)**randint(1,2)*sqrt((a+sqrt(a*a+b*b))/2)
                zy=b/2/zx
```

```
                lp=ap(zx,zy);lq=ap(-zx,-zy)
                line = C.create_line(lp,fill='red')
                line = C.create_line(lq,fill='green')
def teken():
    global lis,k;lis=mult(E1.get());tl=transd(lis);k=float(E2.get());assen()
    line = C.create_line(tl,fill='red');kromme(lis,k)
#hoofdprogramma
L1=Label(form,text='Julia: c=a+bj;a,b:',);L1.place(x=5,y=hoInv)
L2=Label(form,text='k:');L2.place(x=220*sc,y=hoInv)
B1=Button(form,text='Teken',command=teken);B1.place(x=330*sc,y=hoInv,width=70*sc)
E1.insert(0,'-0.5,-0.4');E2.insert(0,'4');assen()
form.mainloop()
```



105. Het 'broccoli'-fractaal ? (Koch 2) koch 2

We denken terug aan het fractaal van Koch. Nog steeds zullen we elke zijde verdelen in 3 gelijke stukken, waarbij we nu het middenste stuk vervangen door de 3 zijden van een rechthoek.
Oorspronkelijk zullen we de berekening maken voor een vierkant.
Uitgaande van het middenste lijnstuk, berekenen we de coördinaten van de 2 overblijvende hoekpunten.



Stel dx1= x4-x3 dy1= y4-y3

dx = x5-x3 dy = y5-y3 zkw=$(dx1)^2+(dy1)^2$ $m = \frac{dy1}{dx1}$

Dan geldt ook dx=x6-x4 en dy=y6-y4
Loodrechte stand: dx=-m.dy
Vierkant: $dx^2+dy^2=(dx1)^2+(dy1)^2$= zkw

Dus: zkw = $m^2.dy^2+dy^2=(m^2+1).dy^2$, waaruit volgt: $dy = \pm\sqrt{\frac{zkw}{m^2+1}}$

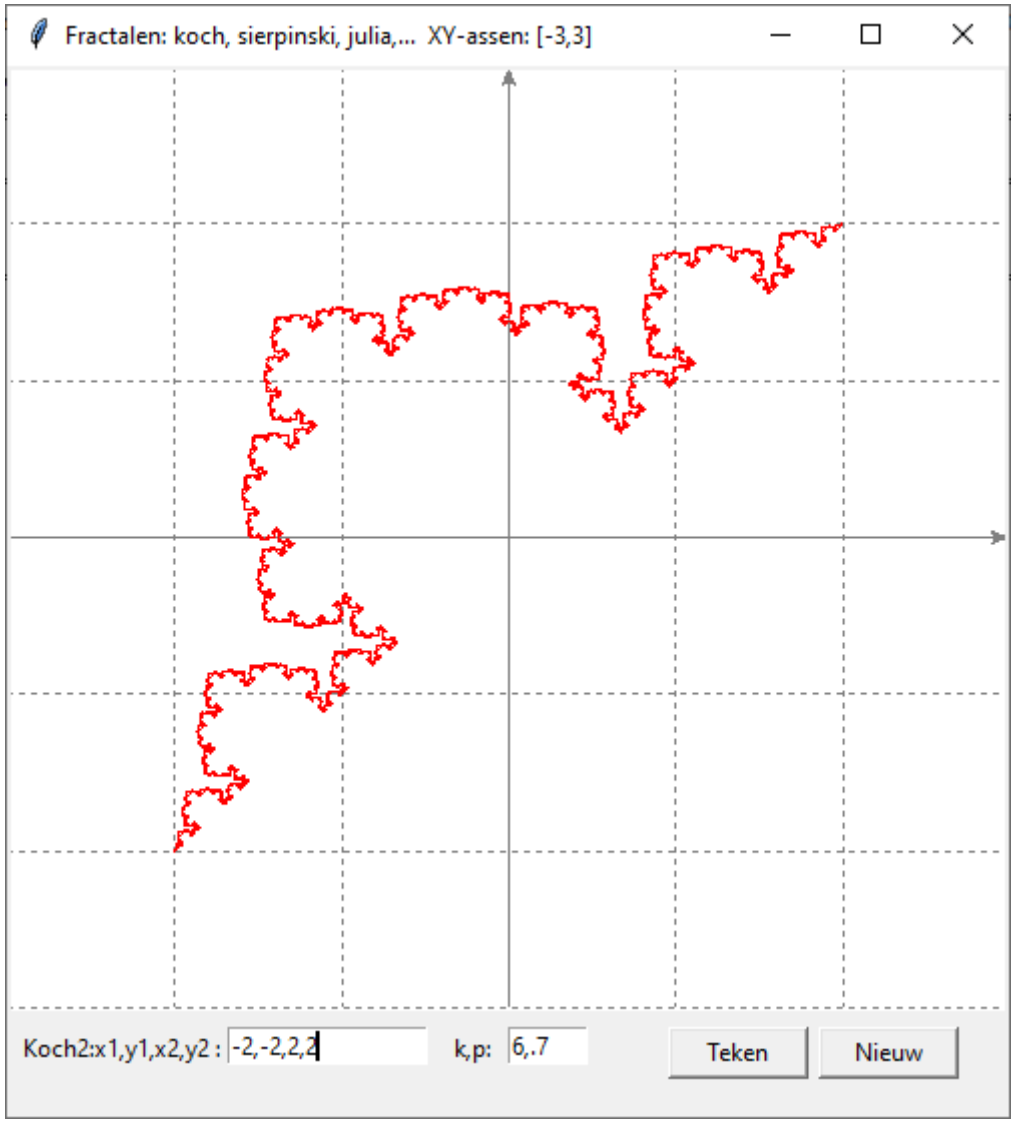
Welk teken ?: is x4>x3 dan is dx1 >0 :uit de tekening volgt:+, is dx1<=0 dan –

```
dx1=x4-x3;dy1=y4-y3
if dx1!=0:m=dy1/dx1
else:m=100000
zkw=dx1**2+dy1**2
if dx1>0:t=1
else: t=-1
dy=p*sqrt(zkw/(m*m+1))*t;dx=-m*dy
x5=x3+dx;y5=y3+dy
x6=x4+dx;y6=y4+dy
```

De opmerkzame lezer ziet dat dy=**p***sqrt(..... Die **p*** is toegevoegd om niet alleen vierkanten te tekenen maar ook rechthoeken. Is p=1, dan hebben we effectief vierkanten, nemen we p<1, dan krijgen we lage rechthoeken. p>1 geeft hoge rechthoeken.
In het 2de invoervenster moeten we 2 getallen k en p ingeven:
k= aantal keer dat elk lijnstukje vervangen wordt door een trapje.
Kies voor p waarden tussen 0.5 en 1.5

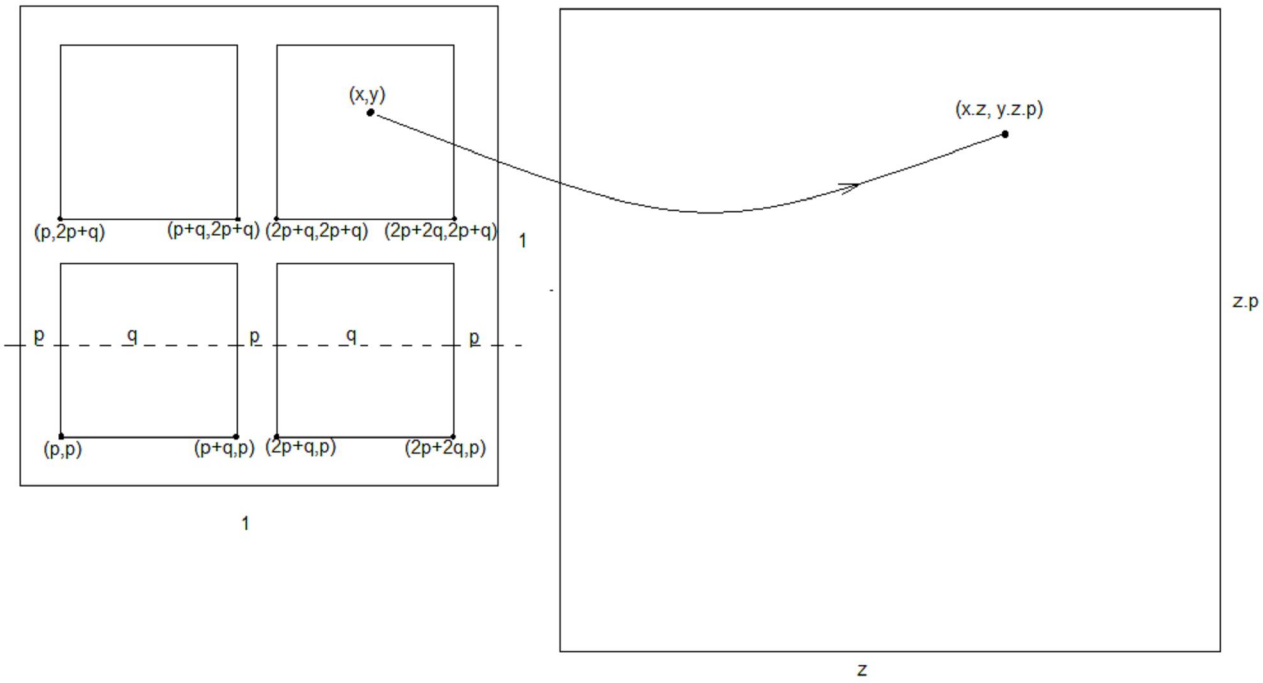
Programma:

```
# koch 2
from fractbasis import *
def hoek3en4(l):
   xa,ya,xb,yb=l
   dx1=xb-xa;dy1=yb-ya
   if dx1!=0:m=dy1/dx1
   else:m=100000
   zkw=dx1**2+dy1**2
   if dx1>0:t=1
   else: t=-1
   dy=p*sqrt(zkw/(m*m+1))*t;dx=-m*dy
   xc=xa+dx;yc=ya+dy
   xd=xb+dx;yd=yb+dy
   return [xc,yc,xd,yd]
def kromme(lijst):
   global k,p;k,p=mult(E2.get());k=int(k);l=lijst;kl=0
   while kl<k:
      kl=kl+1;i=0
      while i<len(l)-3:
         x1,y1,x2,y2=l[i:i+4]
         x3=(2*x1+x2)/3;y3=(2*y1+y2)/3
         x4=(x1+2*x2)/3;y4=(y1+2*y2)/3
         x5,y5,x6,y6=hoek3en4([x3,y3,x4,y4])
         l.insert(i+2,x3);l.insert(i+3,y3)
         l.insert(i+4,x5);l.insert(i+5,y5)
         l.insert(i+6,x6);l.insert(i+7,y6)
         l.insert(i+8,x4);l.insert(i+9,y4)
         i=i+10
      lis2=[]
      for i in range(0,len(l)):
         if i%2==0:lis2.append(transx(l[i]))
         else:lis2.append(transy(l[i]))
      assen();line = C.create_line(lis2,fill=kleur[k%6])
def teken():
   global lis,k;lis=mult(E1.get());kromme(lis)
L1=Label(form,text='Koch2:x1,y1,x2,y2 :',);L1.place(x=5,y=hoInv)
L2=Label(form,text='k,p:');L2.place(x=220*sc,y=hoInv)
B1=Button(form,text='Teken',command=teken);B1.place(x=330*sc,y=hoInv,width=70*sc)
E1.insert(0,'-2,-2,2,1');E2.insert(0,'5,.7');assen()
form.mainloop()
```



106. Sierpinski 2 sierpinski 2

Bij het Sierpinski fractaal wordt een gelijkzijdige driehoek telkens verder opgedeeld in kleinere driehoeken. Je zou dus eigenlijk evengoed een rechthoek steeds verder opdelen in gelijkvormige rechthoeken.
We zorgen ook voor een gelijke marge tussen de rechthoekjes onderling en de grote rechthoek.
We starten eenvoudig met een vierkant met zijde 1. Hierin is de marge =q, de breedte van de kleinere rechthoek is p. Dan geldt 3q+2p=1. Bij elke keuze van q <1/3 is p=(1-3q)/2
In 2 stappen gaan we de coördinaten van een punt P(x,y) omzetten naar de coördinaten P(X,Y) van ditzelfde punt in het assenstelsel.
Deze transformatieformules kunnen we dan gebruiken om de basissen
(p,p) – (p+q,p) , (2p+q,p) – (2p+2q,p), … om te zetten naar de (X,Y)-coördinaten van diezelfde punten in het assenstelsel. Op die basissen kunnen we dan rechthoeken oprichten met de functieprocedure hoek3en4(...) die we bij Koch 2 besproken hebben.
Opmerking: deze '(X,Y)'-coördinaten moeten uiteindelijk ook nog eens omgezet worden naar de schermcoördinaten



1ste stap: zet P(x,y) in het eenheidsvierkant om naar P(x.z,y.z.p) in de rechthoek met breedte z en lengte z.p (zie eerste tekening). Dit is evident.

2de stap: het punt P heeft in de 2de tekening coördinaten (X,Y) t.o.v. het assenstelsel en coördinaten (x.z , y.z.p) t.o.v. de rechthoek.
De volgende formules kunnen gemakkelijk geverifieerd worden:

$$z = \sqrt{(xb - xa)^2 + (yb - ya)^2} \quad r = \sqrt{x^2 z^2 + y^2 z^2 p^2}$$

$$\cos\alpha = \frac{xb - xa}{z} \quad \sin\alpha = \frac{yb - ya}{z}$$

$$\cos\beta = \frac{x.z}{r} \quad \sin\beta = \frac{y.z.p}{r}$$

$$dx = \cos(\alpha + \beta).r \quad dy = \sin(\alpha + \beta).r$$

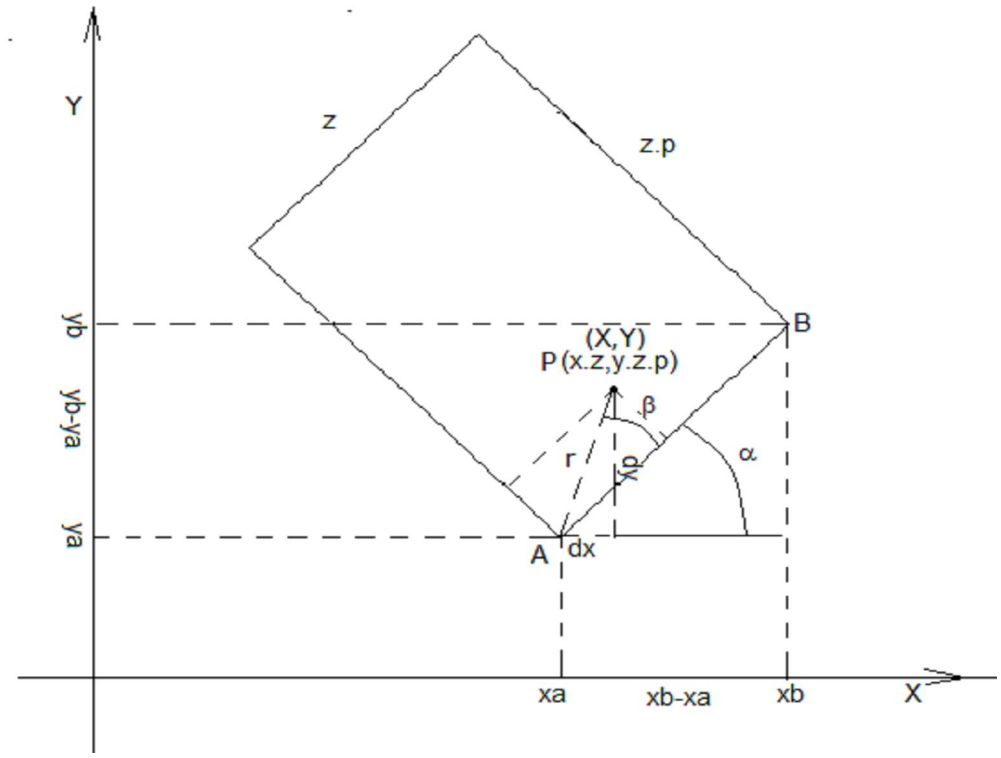
met $\cos(\alpha+\beta) = \cos\alpha \cdot \cos\beta - \sin\alpha \cdot \sin\beta$ en $\sin(\alpha+\beta) = \sin\alpha.\cos\beta + \cos\alpha.\sin\beta$

$$dx = \left(\frac{xb - xa}{z}.\frac{x.z}{r} - \frac{yb - ya}{z}.\frac{y.z.p}{r}\right).r = (xb - xa).x - (yb - ya).y.p$$

$$dy = \left(\frac{yb - ya}{z}.\frac{x.z}{r} + \frac{xb - xa}{z}.\frac{y.z.p}{r}\right).r = (yb - ya).x - (xb - xa).y.p$$

$$z=\sqrt{(xb-xa)^2+(yb-ya)^2} \qquad dx=(xb-xa).x-(yb-ya).y.p \qquad dy=(yb-ya).x+(xb-xa).y.p$$

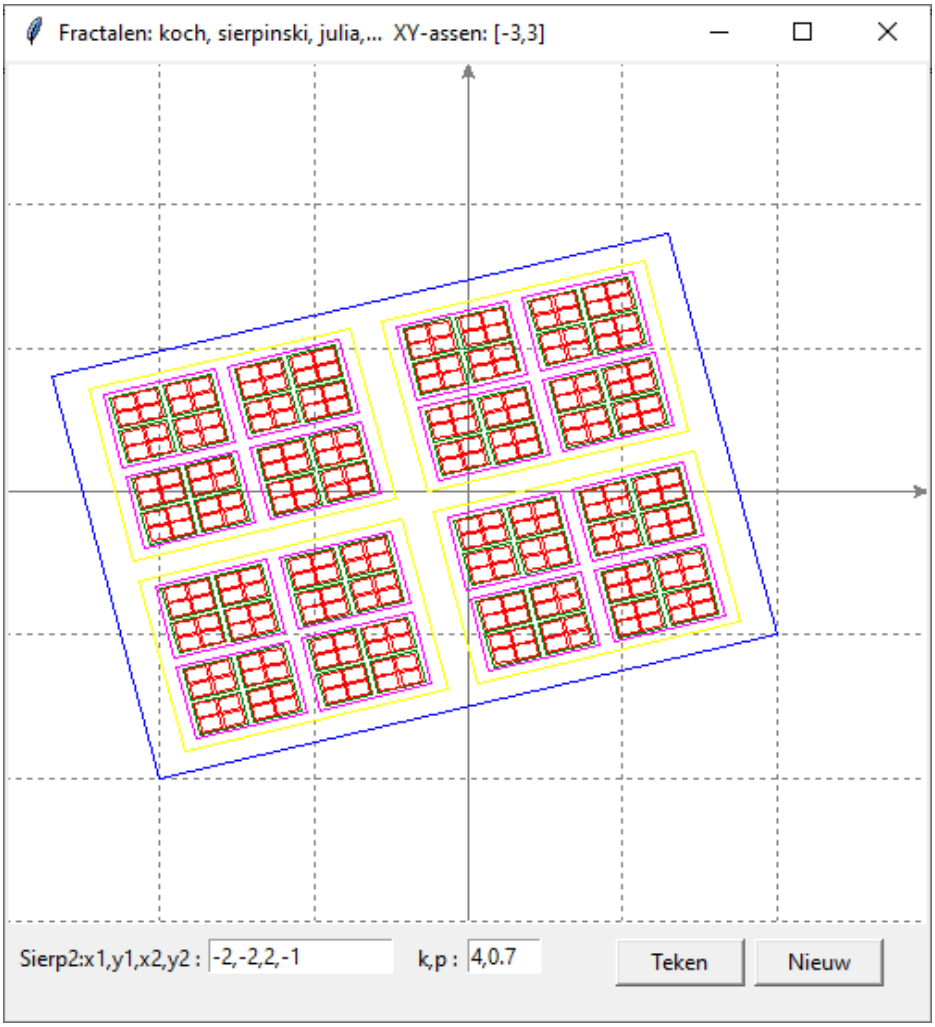
Waarmee we uitgaande van P(x,y) in het eenheidsvierkant de (X,Y)-coördinaten van het punt P in het assenstelsel kunnen berekenen X=xa+dx en Y=ya+dy



Programma:

```
#sierpinski 2 (rechthoeken)
from fractbasis import *
def hoek3en4(l):
    xa,ya,xb,yb=l
    dx1=xb-xa;dy1=yb-ya
    if dx1!=0:m=dy1/dx1
    else:m=100000
    zkw=dx1**2+dy1**2
    if dx1>0:t=1
    else: t=-1
    dy=p*sqrt(zkw/(m*m+1))*t;dx=-m*dy
    xc=xa+dx;yc=ya+dy
    xd=xb+dx;yd=yb+dy
    return [xc,yc,xd,yd]
def punt(x,y):
    if xb==xa:
        dx=y*abs(yb-ya)*p
        dy=(yb-ya)*x
    else:
        dx=(xb-xa)*x-(yb-ya)*y*p
        dy=(xb-xa)*y*p+(yb-ya)*x
    return [xa+dx,ya+dy]
def tekvk(l,k):
    global xa,ya,xb,yb,z
    xa,ya,xb,yb=l
    z=sqrt((xb-xa)**2+(yb-ya)**2)
    x3,y3,x4,y4=hoek3en4(l)
```

```
    l.append(x4);l.append(y4);l.append(x3);l.append(y3)
    tl=transd(l);line = C.create_line(tl,fill=kleur[k%6])
   #basislijn van 4 kleinere rechthoekjes berekenen
    if k>0:
        l1=punt(q,q)+punt(q1,q);l2=punt(q2,q)+punt(q3,q)
        l3=punt(q,q2)+punt(q1,q2);l4=punt(q2,q2)+punt(q3,q2)
        k=k-1
        tekvk(l1,k);tekvk(l2,k);tekvk(l3,k);tekvk(l4,k)
def teken():
    global lis,k,p,q,q1,q2,q3
    q=0.05
    r=(1-3*q)/2;q1=q+r;q2=2*q+r;q3=2*q+2*r
    lis=mult(E1.get());k,p=mult(E2.get());k=int(k);assen();tekvk(lis,k)

B1=Button(form,text='Teken',command=teken);B1.place(x=330*sc,y=hoInv,width=70*sc)
L1=Label(form,text='Sierp2:x1,y1,x2,y2 :',);L1.place(x=5,y=hoInv)
L2=Label(form,text='k,p :');L2.place(x=220*sc,y=hoInv)
E1.insert(0,'-2,-2,2,-1');E2.insert(0,'4,0.7');assen()
form.mainloop()
```



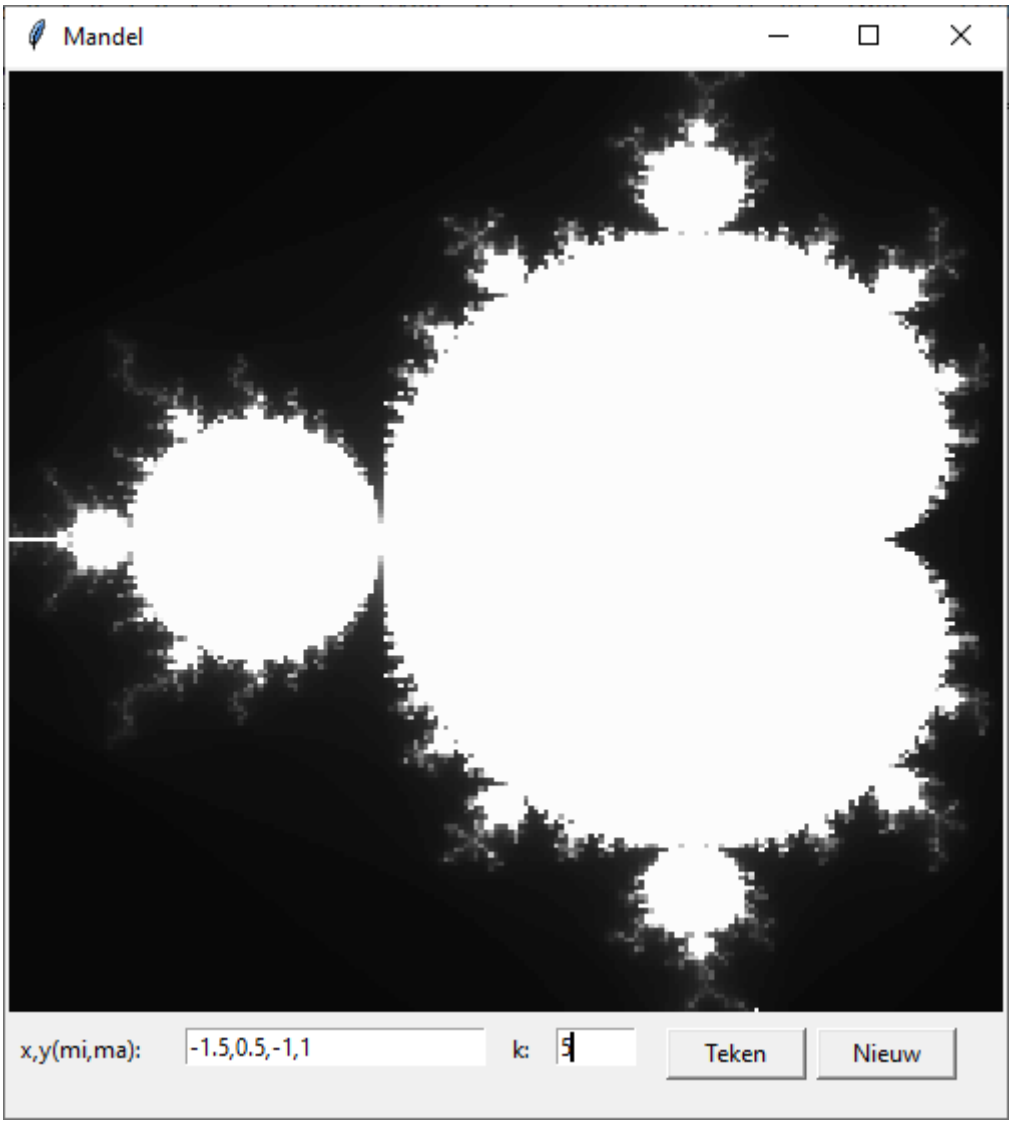
107. Mandelbrot mandelbrot

Net zoals bij Julia-fractalen gaan we uit van de complexe vergelijking: $z_{i+1}=z_i^2 + c$
Eigenlijk zoeken we nu voor welke waarden van het complex getal
c = x+y.j , abs(z) < 2 blijft.
c, dus (x,y) doorloopt het vlak [-1.5,0.5] x [-1,1].
Voor elke van deze complexe getallen c onderzoeken we:
een teller it houdt bij hoeveel keer we de vergelijking $z_{i+1}=z_i^2 + c$ kunnen toepassen, zonder dat abs(z)>2 wordt. Gebeurt dit snel, bvb, na 5 keer, dan wordt dit punt ingekleurd met 'grey5', d.i.+/- zwart.
Naarmate it groter wordt 'grey..' bleker
Wordt it > (maximum it) =mait=99, dan is abs(z) steeds kleiner gebleven dan 2 en hebben we 'grey99' = wit

Programma:

```
# mandelbrot
w=1.73
from math import sqrt
from random import randint
from tkinter import *
def mult(s):
        l=s.split(',')
        co=[]
        for i in l:co.append(float(i))
        return co
def transx(x):
   return (x-xmi)*breedte/(xma-xmi)
def transy(y):
   return hoogteC-hoogteC*(y-ymi)/(yma-ymi)
def ap(x,y): # rechthoekje maken
   X=transx(x);Y=transy(y);li=[];li.append(X-b);li.append(Y+h)
       li.append(X-b);li.append(Y-h);li.append(X+b);li.append(Y+h)
       li.append(X-b);li.append(Y+h);li.append(X+b);li.append(Y-h)
       return li
def color_points():
   C.delete(ALL);x_p = k;y_p = k;mait=99
   for yi in range(y_p):
      for xi in range(x_p):
         xz=0;yz=0;it=0
         x=xmi+(xma-xmi)*xi/x_p
         y=ymi+(yma-ymi)*yi/y_p
         while it<mait and xz*xz+yz*yz<4:
             xzo=xz
             it=it+1;xz=xz*xz-yz*yz+x
             yz=2*xzo*yz+y
         lp=ap(x,y)
         line = C.create_line(lp,fill='grey'+str(it))
sc=1;breedte=500*sc;hoogte=525*sc;hoogteC=hoogte-55*sc;hoInv=hoogte-45*sc
form=Tk();form.geometry(str(breedte)+'x'+str(hoogte))
form.title('Mandel')
C = Canvas(form, bg="white", height=hoogteC, width=breedte)
C.pack()
L1=Label(form,text='x,y(mi,ma):',);L1.place(x=5,y=hoInv)
```

```
E1=Entry(form);E1.place(x=90*sc,y=hoInv,width=150*sc)
L2=Label(form,text='k:',);L2.place(x=250*sc,y=hoInv)
E2=Entry(form);E2.place(x=275*sc,y=hoInv,width=40*sc)
def teken():
        global xmi,xma,ymi,yma,k,b,h
        xmi,xma,ymi,yma =mult(E1.get())
        k=40*int(E2.get());b=breedte/k/2;h=hoogteC/k/2
        C.delete(ALL);color_points()
def nieuw():
        E1.delete(0,len(E1.get()));C.delete(ALL)
        E2.delete(0,len(E2.get()));
B1=Button(form,text='Teken',command=teken);B1.place(x=330*sc,y=hoInv,width=70*sc)
B2=Button(form,text='Nieuw',command=nieuw);B2.place(x=405*sc,y=hoInv,width=70*sc)
E1.insert(0,'-1.5,0.5,-1,1')
E2.insert(0,'2')
form.mainloop()
```

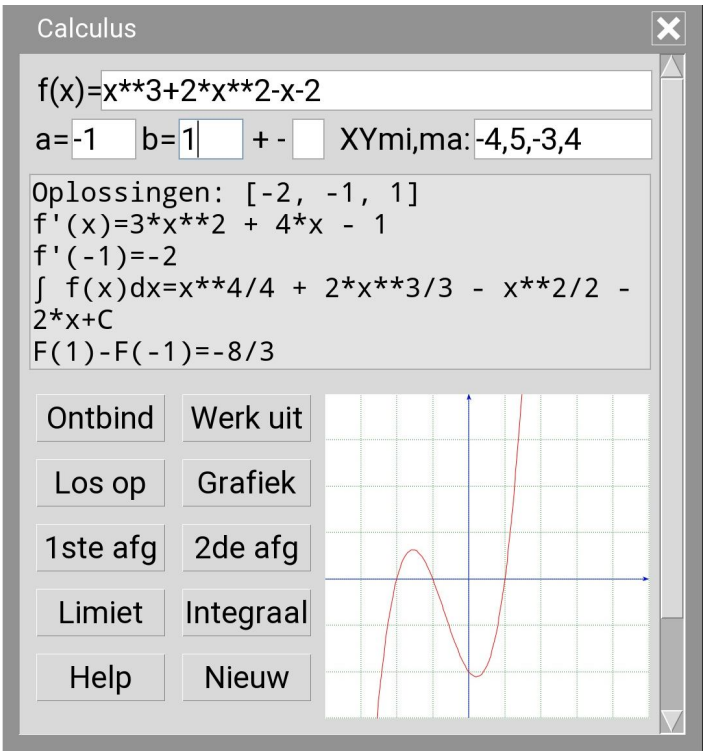


108. Een calculus voor analyse (met 'sympy') calculus 1

Om dit programma uit te voeren moeten we de modules '**sympy**' en '**mathplotlib**' installeren. Dit lukt alleen bij de nieuwere versies van python.

In het functievenster mogen willekeurige functies van x ingevoerd worden.
Met **Ontbind** en **Werk uit** kan je de ingevoerde functie vervangen door de ontbonden of uitgewerkte vorm.
Parameter a wordt gebruikt voor het berekenen van een limiet of een afgeleide in a. Je mag ook oneindig (oo of -oo) invoeren.(2 kleine o-tjes)
Het veld + of – mag je invullen als je een rechter- of linkerlimiet wenst. Parameter b wordt in combinatie met a gebruikt, om een bepaalde integraal te berekenen.
Dank zij de functies Derivative en Integral wordt ook de afgeleide, respectievelijk primitieve functie getoond.
Je kan ook de nulpunten laten berekenen met **Los op**.
Al deze resultaten worden in een tekstveld onder elkaar afgedrukt.
Om de laatste resultaten te zien, moet je naar beneden scrollen.
Maar je kunt een antwoord ook tussenvoegen, door de cursor op de gewenste plaats te zetten.
Om een grafiek te tekenen, moeten xmi,xma,ymi,yma ingevuld zijn.
Er is een helptekstje voorzien, die in een voorafgemaakt tekstbestandje **helpcalc.txt** zit.
Vergeet je iets in te voeren, dan komt er een foutmelding in de shell maar meestal kan je het programma gewoon verder zetten.

Er zijn van dit programma 2 versies: enkel van het 2de is de listing opgenomen in de tekst. Het 1ste zit wel in de map 'programmas'
Calculus 1 tekent de grafiek in een deel van het formulier, een canvas. De functieprocedure om de grafiek te tekenen is overgenomen uit het eerste grafisch tkinter-programma.



109. Calculus 2 (met mathplotlib) calculus 2

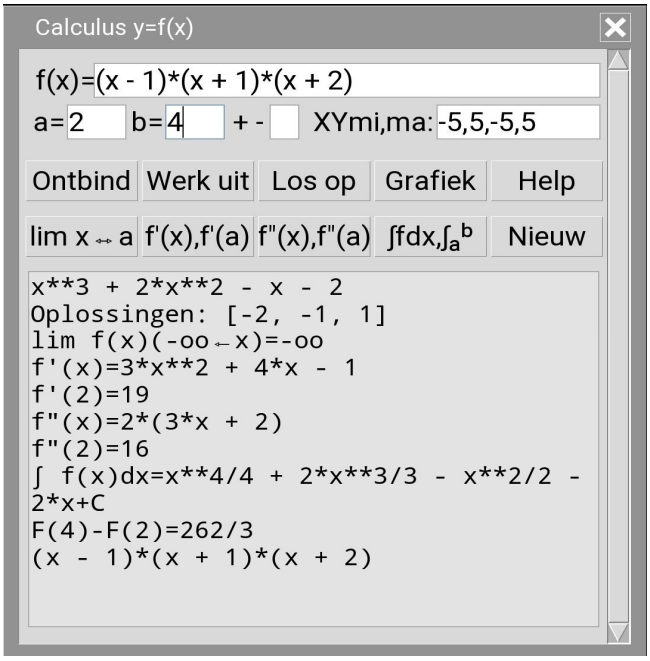
Calculus 2 maakt gebruik van de uitgebreide mogelijkheden van mathplotlib

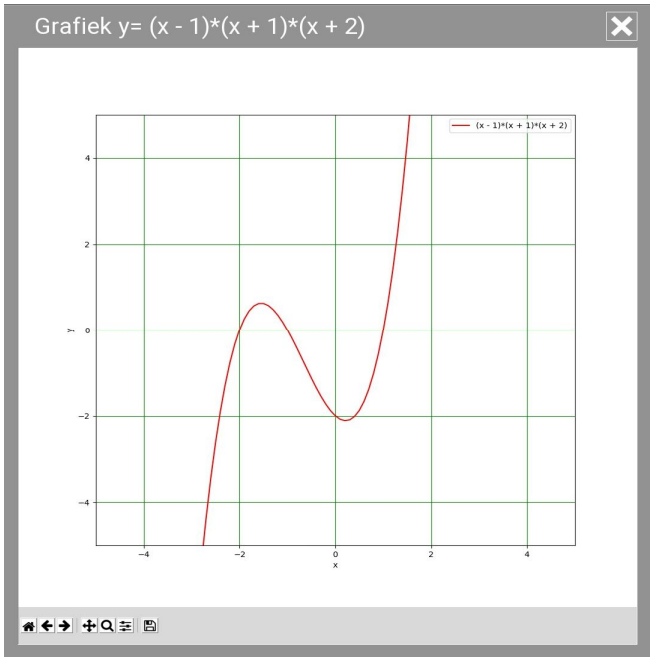
Programma

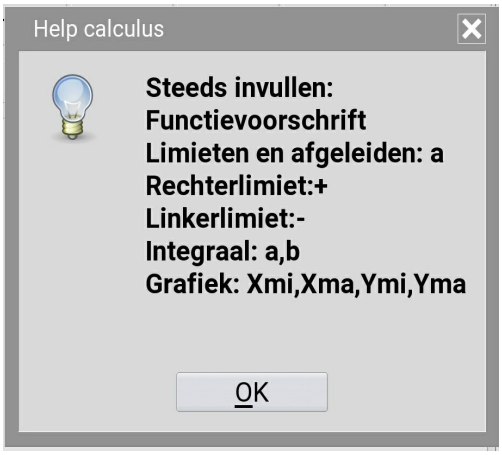
```
# Calculus
from tkinter import *
from tkinter import messagebox
from sympy import *
from matplotlib import pyplot as plt
#unicodes
pl='\u21FDx';pr='x\u21FE';pd='x\u21FF'
V='\u221A';I='\u222B'
sa='\u2090';sb='\u1D47';BI=I+sa+sb
def mult(s):
    l=s.split(',');co=[]
    for i in l:co.append(float(i))
    return co
def transx(x):
    return (x-xmi)*breedte/(xma-xmi)
def transy(y):
    return hoogteC-hoogteC*(y-ymi)/(yma-ymi)
def rich(ri,a):
    if ri=='+' :ls=a+pl
    elif ri=='-':ls=pr+a
    else:ls=pd+a
    if a=='-oo':ls=a+pl
    if a=='oo' or a=='+oo':ls=pr+a
    ls='('+ls+')'
    return ls
def voegtoe(lijn):tex.insert(INSERT, lijn+'\n')
def wisantw():tex.delete('1.0',END)
def wis(ent):ent.delete(0,len(ent.get()))
def nieuw():
    entries=[E1,E2,E3,E4,E5]
    for ent in entries:wis(ent)
    wisantw()
def ontb():
    fac=str(factor(E1.get()));wis(E1);E1.insert(0,fac);voegtoe(fac)
def werkuit():
    uitw=str(expand(E1.get()));wis(E1);E1.insert(0,uitw);voegtoe(uitw)
# tekening met mathplotlib
def teken():
    global xmi,xma,ymi,yma,fx
    xmi,xma,ymi,yma=mult(E4.get());fx=E1.get()
    lx=[];ly=[];x=2*xmi
    while x<2*xma:
            x=x+0.1
            lx.append(x);ly.append(f(x))
    lya=[ymi,yma];lxa=[xmi,xma]
```

```
    plt.figure(figsize=(10.3,10.1),num='Grafiek y= '+fx)
    plt.grid(axis='both',color='g',linestyle='-')
    plt.xlim(lxa);plt.ylim(lya)
    plt.plot(lx,ly,label=fx,color='r')
    plt.xlabel('x');plt.ylabel('y')
    plt.legend();plt.show()
def f(x):
    return eval(fx)
def hlp():
    hc=open('helpcalc.txt');hptek=hc.read();hc.close
    h=messagebox.showinfo('Help calculus',hptek)
def afg():
    global fx,arg,ri;fx=E1.get(); arg=E2.get();ri=E3.get()
    x= Symbol('x')
    fx=sympify(fx)
    df= Derivative(fx,x).doit()
    d2f=Derivative(fx,x,2).doit()
    return df,d2f,x
def afg1():
    df,d2f,x=afg();voegtoe('f\'(x)='+str(df))
    if ri!='':li= Limit(df,x,arg,dir=ri).doit()
    else:li=Limit(df,x,arg).doit()
    voegtoe('f\'('+arg+')='+str(li))
def afg2():
    df,d2f,x=afg();voegtoe('f\''(x)='+str(d2f))
    if ri!='':li= Limit(d2f,x,arg,dir=ri).doit()
    else:li=Limit(d2f,x,arg).doit()
    voegtoe('f\''('+arg+')='+str(li))
def losop():
    x=Symbol('x');fx=E1.get()
    nlp=str(solve(fx))
    n=nlp.replace('sqrt',V);voegtoe('Oplossingen: '+n)
def lim():
    global fx,arg,ri
    fx=E1.get();arg=E2.get();ri=E3.get()
    x= Symbol('x');fx=sympify(fx)
    if ri!='':li= Limit(fx,x,arg,dir=ri).doit()
    else:li=Limit(fx,x,arg).doit()
    voegtoe('lim f(x)'+rich(ri,arg)+'='+str(li))
def integ():
    global fx,a,b;fx=E1.get()
    sa=E2.get();sb=E5.get()
    x= Symbol('x');fx=sympify(fx)
    pri=Integral(fx,x).doit();bpi=Integral(fx,(x,sa,sb)).doit()
    voegtoe(I+' f(x)dx='+str(pri)+'+C')
    voegtoe('F('+sb+')-F('+sa+')='+str(bpi))
# Formulier opstellen
sc=2;dx=205*sc;dy=215*sc;d=str(dx)+'x'+str(dy)
form=Tk();form.geometry(d);form.title('Calculus y=f(x)')
Label(form,text='f(x)=').place(x=5*sc,y=5*sc)
E1=Entry(form);E1.place(x=25*sc,y=5*sc,width=160*sc)
Label(form,text='a=').place(x=5*sc,y=20*sc,width=10*sc)
```

```
E2=Entry(form);E2.place(x=16*sc,y=20*sc,width=20*sc)
Label(form,text='b=').place(x=38*sc,y=20*sc,width=10*sc)
E5=Entry(form);E5.place(x=49*sc,y=20*sc,width=20*sc)
Label(form,text='+ -').place(x=71*sc,y=20*sc)
E3=Entry(form);E3.place(x=84*sc,y=20*sc,width=10*sc)
Label(form,text='XYmi,ma:').place(x=98*sc,y=20*sc)
E4=Entry(form);E4.place(x=130*sc,y=20*sc,width=55*sc)
hoogteC=100*sc;breedte=100*sc
tex= Text(form,bg='grey89',height=16,width=45);tex.place(x=3*sc,y=80*sc)
vscbar=Scrollbar(form,orient=VERTICAL,command=tex.yview,width=25)
tex['yscrollcommand'] = vscbar.set;vscbar.pack(side=RIGHT,fill=Y)
# Buttons toevoegen
def but(i,com):
    xp=(2+18*(i%5))*sc;yp=(18+9*(i//5))*sc
Button(form,text=com[0],command=com[1]).place(x=xp*sc,y=yp*sc,width=35*sc,height=15*sc)
lbut=[('Ontbind',ontb),('Werk uit',werkuit),('Los op',losop),('Grafiek',teken),('Help',hlp),('lim
'+pd+'a',lim),('f\'(x),f\'(a)',afg1) ,('f\"(x),f\"(a)',afg2),(I+'fdx,'+BI,integ),('Nieuw',nieuw)]
for i in range(0,10):
    but(i,lbut[i])
# Invullen entries
E1.insert(0,'x**3+2*x**2-x-2');E2.insert(0,'1');E5.insert(0,'2') ;E3.insert(0,'');E4.insert(0,'-5,5,-5,5')
form.mainloop()
```

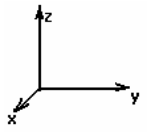




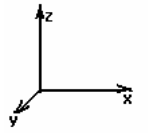


110. 3-dimensionale grafieken grafiek_3dimensies

De grafiek van een functie y=f(x) in een xy- vlak is steeds een kromme.
(is f(x) van de eerste graad,dan is de grafiek een rechte)
Daarentegen is de grafiek van een functie z=f(x,y) een 'meestal' golvende oppervlakte in de 3-dimensionale ruimte Σ_0. (is f(x,y) van de eerste graad, dan is de grafiek een vlak)
In de ruimte hebben we 3 assen die 2 aan 2 loodrecht op elkaar staan. Conventioneel is de x-as horizontaal naar voor gericht, de y-as samenvallend met de x-as uit de vlakke meetkunde, de z-as i.p.v. de y-as uit de vlakke meetkunde, symbolisch voorgesteld door :



Maar bij het volgende programma's over 'oppervlakken in de ruimte' heb ik voor een andere opstelling gekozen: de x-as blijft de x-as uit de vlakke meetkunde, de y-as is horizontaal naar voor gericht. Dit wordt symbolisch voorgesteld door



Waarom?:

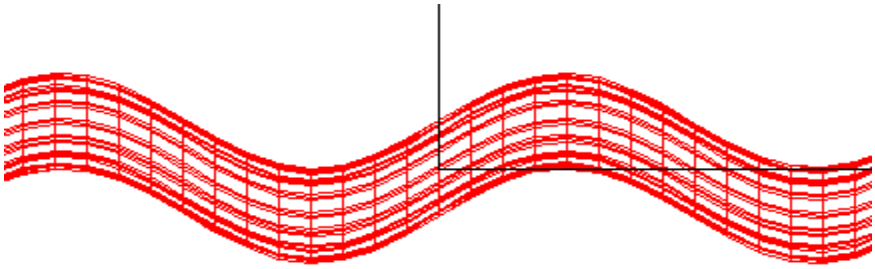
Als we het in de analyse hebben over omwentelingsoppervlakken (inhoud en zijdelingse oppervlakte) , wordt een kromme y=f(x), 360° gewenteld rond de x-as.

Om de integraal te berekenen, kunnen we ons dit het best voorstellen als de x-as hetzelfde blijft. Maar ook bij andere ruimtelijke oppervlakken (vb. torus) lijkt me deze opstelling het beste. Wens je toch de 1ste (conventionele) opstelling, dan moet je vóór de omzetting naar vlakke coördinaten en nadien naar schermcoördinaten, x en y verwisselen. Verderop in deze notities in het programma 'ruimtemeetkunde' (punten, rechten en vlakken) kies ik ook voor de 'conventionele' opstelling.

Van een punt met coördinaten (x,y,z) t.o.v. dit assenstelsel kunnen we maar 2 coördinaten tonen op het scherm: de x en de z-coördinaat.
Om ons oppervlak te tonen kunnen we een raster van niveaulijnen tekenen:

1. lijnen die op vaste afstand van elkaar evenwijdig met het YZ-vlak lopen
   (per lijn x constant houden en y laten variëren van ymi tot yma)
2. lijnen die op vaste afstand van elkaar evenwijdig met het XZ-vlak lopen
   (per lijn y constant houden en x laten variëren van xmi tot xma)

Per niveaulijn nemen we een vast aantal punten (x,y,z=f(x,y)) die we omzetten naar de overeenkomstige (X,Z): dit koppel wordt aangevuld in een lijst. Als alle punten van de niveaulijn ingevuld zijn kunnen we de lijst met C.create_line in het canvas C tekenen.
Maar dan krijgen we bvb. voor de functie z=sin(x)+sin(y) een tekening zoals hieronder.
Alhoewel wiskundig correct, lijkt dit meer op een vlakke band i.p.v. een 3-dimensionaal oppervlak.



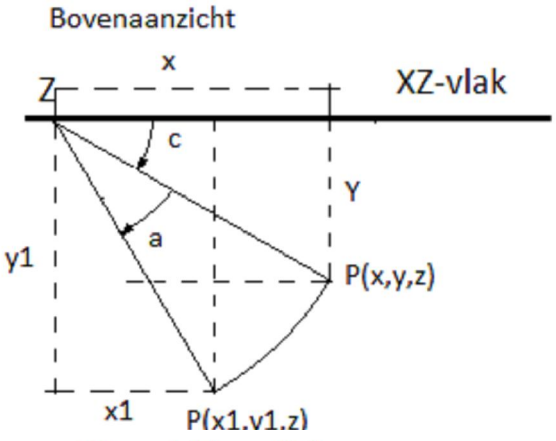
Om een 3-dimensionale ervaring te hebben, moeten we in de tekening een soort van perspectief steken. Daarom zullen we op elk punt P(x,y,z) 2 draaiingen toepassen:

1. We laten z verticaal maar draaien het punt P(x,y,z) over een hoek a rond de Z-as naar voor. Het punt P(x ,y ,z) behoudt t.o.v XYZ zijn z-coordinaat maar krijgt een nieuwe x_1 en y_1
2. Nu passen we een tweede draaiing toe op het punt P(x_1 ,y_1 ,z) over een hoek b rond de X-as naar voor. Deze keer blijft x_1 gelijk, maar y_1 en z veranderen:het punt P heeft t.o.v. XYZ coördinaten (x_1,y_2,z_1) gekregen. Het punt (x_1,z_1) moet dan na omzetting met transx en transy (zie gebruik in vorige programma's) aangevuld worden in de lijst (die de schermcoördinaten van de niveaulijn bevat.

Om te beginnen moeten we formules opstellen voor (x_1 ,y_2 ,z_1) in functie van (x ,y ,z)

1_{ste} draaiing

De benaming 1_{ste} kwadrant komt niet overeen met 1_{ste} kwadrant van de goniometrische cirkel.(met een onderaanzicht zou er wel overeenstemming zijn met de goniometrische cirkel)
Dit is de situatie in het 1ste kwadrant: een horizontale draaiing a van het punt P om de Z-as.
Om c te berekenen, vinden we in het 'eerste kwadrant' tan(c)=y/x, dus c =atan(y/x).
In 2_{de} en 3_{de} kwadrant moeten we hier pi bij optellen.(zie straks)

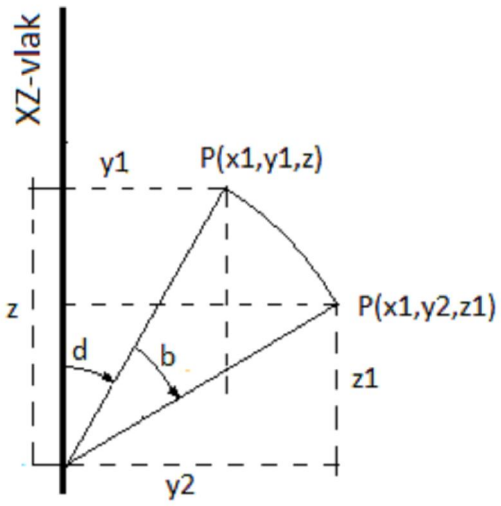


$$r=\sqrt{x^2+y^2}$$

x_1 = cos(a+c).r y_1 = sin(a+c).r
z_1 = z

2_{de} draaiing
Beschouw ook dit als 1_{ste} kwadrant

Om d te berekenen, vinden we in dit 'eerste' kwadrant' tan(d)=y1/z, dus d =atan(y_1/z).
In 2_{de} en 3_{de} kwadrant moeten we hier pi bij optellen.(zie straks)
De 2_{de} draaiing van P is verticaal over een hoek b rond de X-as.

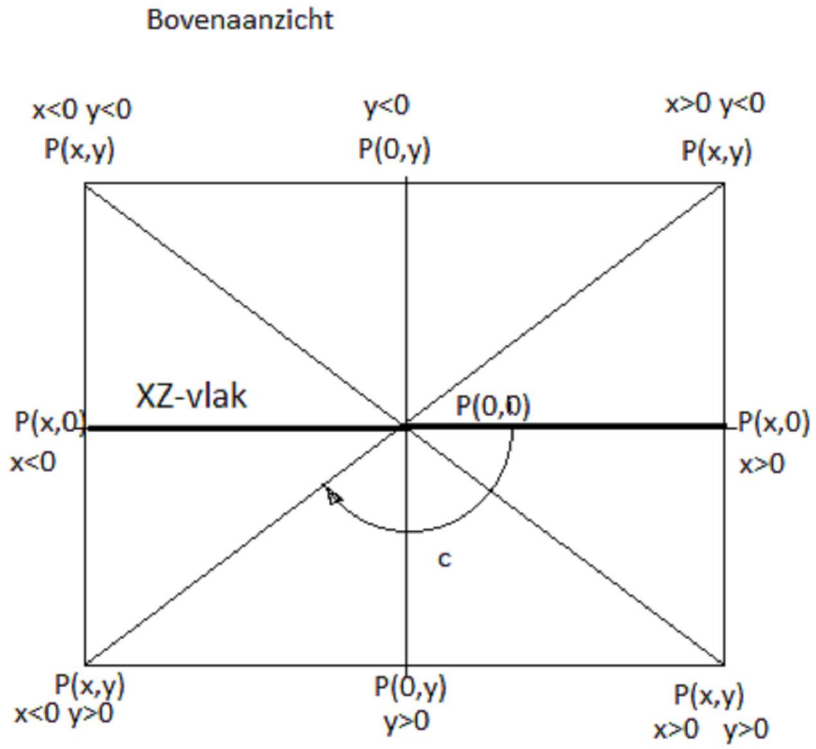


$$r_1=\sqrt{y_1^2+z^2}$$

z_1 = cos(b+d).r_1 y_2 = sin(b+d).r_1
x_1 blijft gelijk

Correcte berekening van c en d in elke situatie

Zit P bvb. in het 2_{de} kwadrant, dan is y/x<0, en geeft c=atan(y/x) een waarde in]-pi/2,0[. Dit is correct voor het 4_{de} kwadrant maar niet voor het 2_{de}. De correcte waarde vind je door pi op te tellen.
Zit P in het 3_{de} kwadrant, dan is y/x>0, en geeft c=atan(y/x) een waarde in]0,pi/2[. Ook hier moeten we dus pi optellen. Kort gezegd, is x<0,dan moeten we bij c pi optellen.
Als x of y 0 is, dan kan je gemakkelijk in de figuur de waarde van c aflezen.
Dezelfde methode is toepasselijk op de 2_{de} rotatie



Om de juiste hoek c of d te berekenen, volstaat dus de volgende functie:

```
def hoek(y,x):
    if y==0 and x>=0:c=0
    elif y==0 and x<0:c=pi
    elif x==0 and y>0:c=pi/2
    elif x==0 and y<0:c=3*pi/2
    else:
        c=atan(y/x)
        if x<0:c=c+pi
    return(c)
```

De formules van de 2 draaiingen worden berekend met de omzetfunctie

```
def omzet(x,y,z):
    c=hoek(y,x); r=sqrt(x*x+y*y); x1=cos(a+c)*r; y1=sin(a+c)*r
    d=hoek(y1,z); r1=sqrt(y1*y1+z*z); z1= cos(b+d)*r1
    X=transx(x1);Y=transy(z1)
    return [X,Y]
```

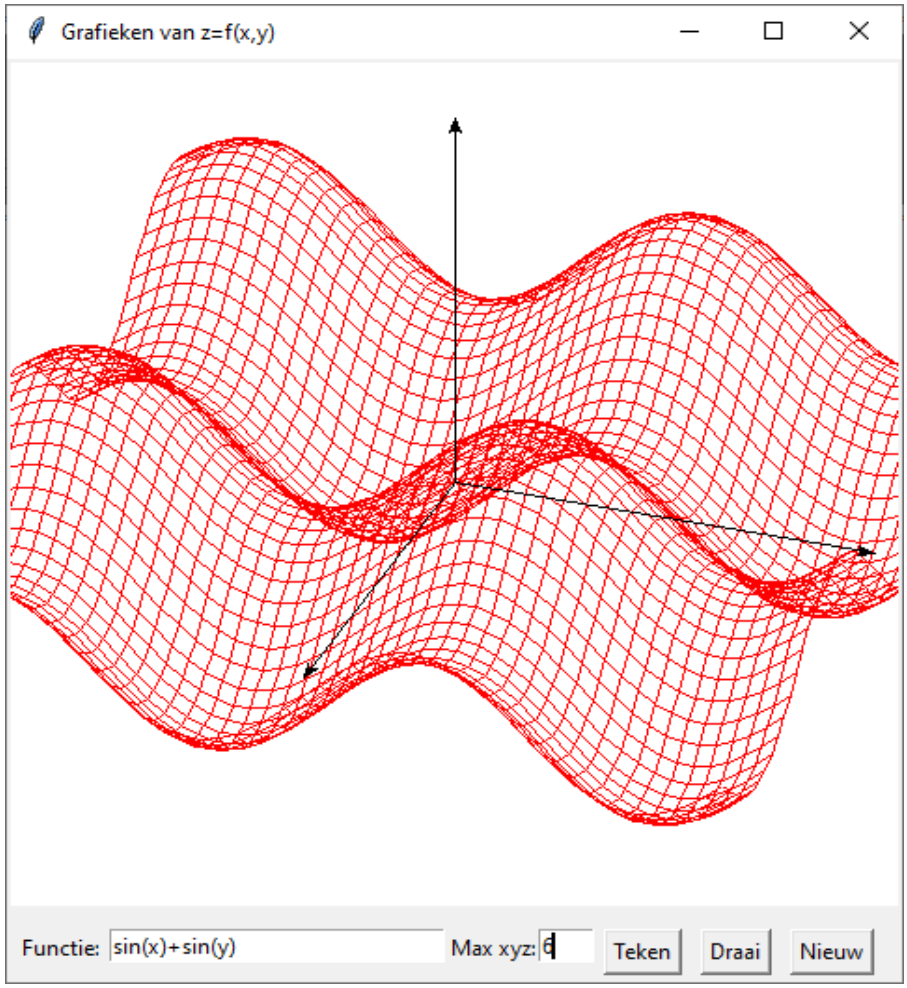
Het is deze (X,Y) die toegevoegd worden als schermcoördinaten aan de lijst om met C.create_line getekend te worden. Merk op dat het x_1 en z_1 zijn die omgezet worden naar de X en Y coördinaten van het scherm. Voor de programmalisting kunnen we enkele functies uit de vorige grafische Tkinter-programma's overnemen.

Programma:

```
# 3 dimensionale grafieken
from math import *
from tkinter import *
```

```
def transx(x):
   return (x-xmi)*breedte/(xma-xmi)
def transy(y):
   return hoogteC-hoogteC*(y-ymi)/(yma-ymi)
def f(x,y):
   return eval(E1.get())
from math import *
def hoek(y,x):
   if y==0 and x>=0:c=0
   elif y==0 and x<0:c=pi
   elif x==0 and y>0:c=pi/2
   elif x==0 and y<0:c=3*pi/2
   else:
      c=atan(y/x)
      if x<0:c=c+pi
   return(c)
def omzet(x,y,z):
   c=hoek(y,x)
   r=sqrt(x*x+y*y);y1=sin(a+c)*r
   d=hoek(y1,z)
   r1=sqrt(y1*y1+z*z)
   X=transx(cos(a+c)*r);Y=transy(cos(b+d)*r1)
   return [X,Y]
def asteken(x,y,z):
   lis=[]
   X,Y=omzet(0,0,0);lis.append(X);lis.append(Y)
   X,Y=omzet(x,y,z);lis.append(X);lis.append(Y)
   line = C.create_line(lis,fill='black',arrow='last')
breedte=480;hoogte=520;hoogteC=hoogte-45;hoInv=hoogte-30
form=Tk();form.geometry(str(breedte)+'x'+str(hoogte))
form.title('Grafieken van z=f(x,y)')
C = Canvas(form, bg="white", height=hoogteC, width=breedte)
C.pack();draaixy=20;draaiyz=30
L1=Label(form,text='Functie:',);L1.place(x=5,y=hoInv)
E1=Entry(form);E1.place(x=55,y=hoInv,width=180)
L2=Label(form,text='Max xyz:',);L2.place(x=235,y=hoInv)
E2=Entry(form);E2.place(x=285,y=hoInv,width=30)
def teken():
   global xmi,xma,ymi,yma,zmi,zma,a,b
   xma=float(E2.get());C.delete(ALL)
   yma=xma;zma=xma
   xmi=-xma
   ymi=xmi;zmi=xmi
   a=radians(draaixy);b=radians(draaiyz)
#kromme
   stap=xma/25
#niveaulijnen tekenen voor constante y
   y=ymi
   while y<yma:
      y=y+stap;lis=[];x=xmi
      while x<xma:
```

```
            x=x+stap;z=f(x,y);X,Y=omzet(x,y,z)
            lis.append(X);lis.append(Y)
        line = C.create_line(lis,fill='red')
#niveaulijnen tekenen voor constante x
    x=xmi
    while x<xma:
        x=x+stap;lis=[];y=ymi
        while y<yma:
            y=y+stap;z=f(x,y);X,Y=omzet(x,y,z)
            lis.append(X);lis.append(Y)
        line = C.create_line(lis,fill='red')
#assen
    asteken(xma,0,0);asteken(0,yma,0);asteken(0,0,zma)
def draai():
    global draaixy;draaixy=draaixy+30
    teken()
def nieuw():
    E2.delete(0,len(E2.get()));E1.delete(0,len(E1.get()));C.delete(ALL)
    global draaixy;draaixy=20
B1=Button(form,text='Teken',command=teken);B1.place(x=320,y=hoInv)
B2=Button(form,text='Draai',command=draai);B2.place(x=372,y=hoInv)
B3=Button(form,text='Nieuw',command=nieuw);B3.place(x=420,y=hoInv)
form.mainloop()
```



Opmerking:
Geef je er de voorkeur aan om x- en y-as te verwisselen, dus de x-as naar voor gericht, zoals conventioneel wordt aangenomen in de ruimtemeetkunde: dan moet je eigenlijk alleen in de omzet-functie, x en y verwisselen.

```
def omzet(x,y,z):
    c=hoek(x,y)                          <==  enkel in deze functie
    r=sqrt(x*x+y*y);x1=sin(a+c)*r
    d=hoek(x1,z)
    r1=sqrt(x1*x1+z*z)
    .....
```

111. Raakvlak in een punt Po(xo,yo,zo) aan een oppervlak z=f(x,y) graf_raakvlak_3dim

Raaklijn aan een kromme.

Een rechte in een vlak Π_0 wordt bepaald door een steunvector Po(xo,yo) en een richtingsvector A(a,b).

De vectoriële vergelijking is dan: $\vec{P} = \vec{P}_O + k.\vec{A}$

Als we deze vergelijking omzetten naar de coördinaten krijgen we (x,y)=(xo,yo)+k(a,b)
Is de rechte een raaklijn in een punt Po(xo,yo) van een kromme y=f(x), dan is (1,f'(xo)) een richtingsvector voor de rechte.
De vergelijking wordt dan (x,y)=(xo,yo)+k(1,f'(xo))
waaruit de parametervergelijkingen :

$$\begin{cases} x = x_0 + k \\ y = y_0 + k.f'(x_0) \end{cases}$$

Door eliminatie van de parameter volgt de cartesische vergelijking
y=yo+(x-xo).f'(xo).

Uitbreiding van 2 naar 3 dimensies: raakvlak aan een oppervlak.

Een vlak in de ruimte Σ_0 heeft één steunvector Po(xo,yo,zo) en twee richtingsvectoren A(a,b,c) en B(d,e,f).

Een willekeurig punt P(x,y,z) voldoet dan aan de vectoriële vergelijking: $\vec{P} = \vec{P}_O + k.\vec{A} + l.\vec{B}$

Is het vlak een raakvlak in het punt Po van een oppervlak z=f(x,y), dan kiezen we de richtingsvectoren volgens de niveaulijnen, resp. // yz (in de x-richting) en // xz (in de y-richting)
De coördinaten van deze richtingsvectoren zijn A(1,0,f'x) en B(0,1,f'y) waarbij dat f'x en f'y de partiële afgeleiden zijn van z=f(x,y) in (xo,yo).
De vectoriële vergelijking P=P+k.A+l.B wordt na omzetting naar de coördinaten:
(x,y,z)=(xo,yo,zo)+k(1,0, f'x)+l(0,1,f'y)
Parametervergelijkingen zijn:

$$\begin{cases} x = x_0 + k \\ y = y_0 + l \\ z = z_0 + k.f'_x + f'_y \end{cases}$$

Omdat Po(xo,yo,zo) het raakpunt is, dus in het raakvlak ligt, is zo=f(xo,yo). Als we de parameters k en l elimineren, vinden we de cartesische vergelijking van het raakvlak:
z=f(xo,yo)+(x-xo).f'x+(y-yo).f'y
Om onderscheid te maken met de functie z=f(x,y) noemen we de laatste de raakvlakfunctie r. In Python is dit gemakkelijk te programmeren.

```
d=0.00001
```

```
def f(x,y):
        return(eval(fs))
def dfx(x,y):
        return((f(x+d,y)-f(x,y))/d)
def dfy(x,y):
        return((f(x,y+d)-f(x,y))/d)
def r(x,y):
        return (f(xo,yo)+(x-xo)*dfx(xo,yo)+(y-yo)*dfy(xo,yo))
……….
fs=input('Geef de functie z=f(x,y) :')
```

Grafiek van een functie van 2 veranderlijken in Σ_0. Raakvlak in een punt P.

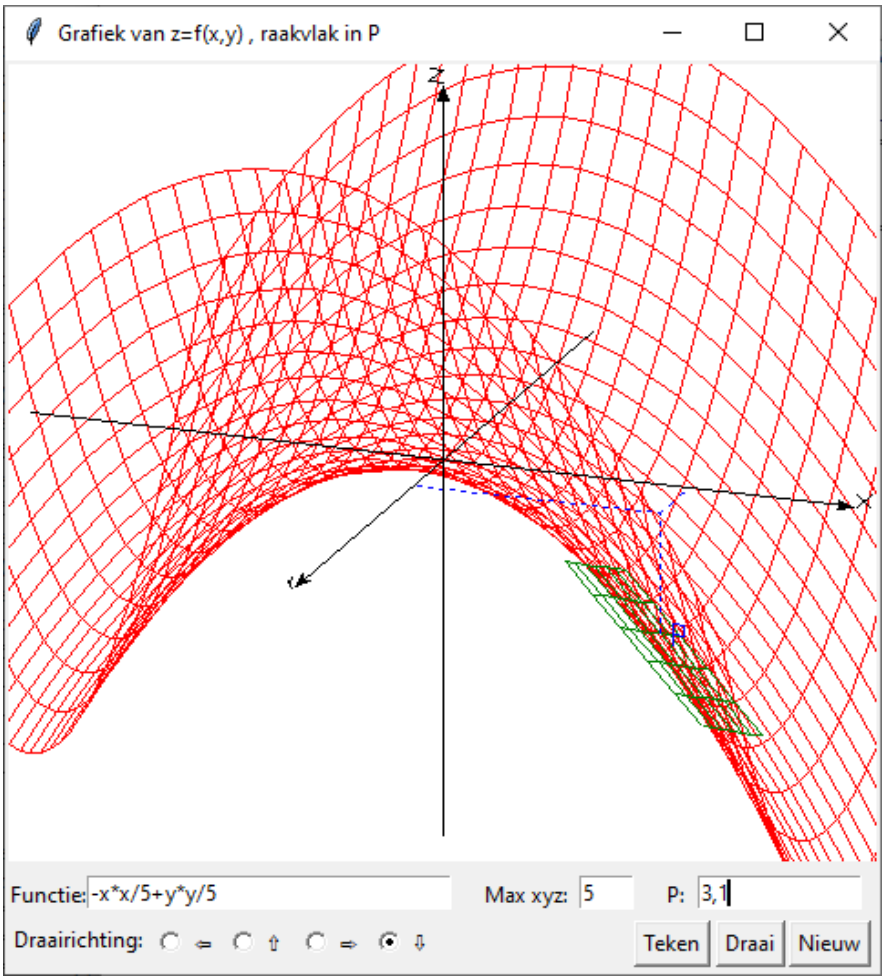
Het volgende programma is een uitbreiding van het vorige 3D-programma in een aantal opzichten. De gebruiker kan draaien in 4 richtingen:linksom, naar boven, rechtsom, naar beneden. Indien een punt P ingevoerd wordt, krijgen we ook het raakvlak in dit punt.

Programma:

```
# 3 dimensionale grafieken
from math import *
from tkinter import *
d=0.00001
def mult(s):
   l=s.split(',');co=[]
   for i in l:co.append(float(i))
   return co
def transx(x):return (x+ma)*breedte/ma/2
def transy(y):return hoogteC-hoogteC*(y+ma)/ma/2
def f(x,y):return eval(fs)
def dfx(x,y):return((f(x+d,y)-f(x,y))/d)
def dfy(x,y):return((f(x,y+d)-f(x,y))/d)
def r(x,y):return(z0+(x-x0)*dzx+(y-y0)*dzy)
def hoek(y,x):
   if y==0 and x>=0:c=0
   elif y==0 and x<0:c=pi
   elif x==0 and y>0:c=pi/2
   elif x==0 and y<0:c=3*pi/2
   else:
      c=atan(y/x)
      if x<0:c=c+pi
   return(c)
def omzet(x,y,z):
   c=hoek(y,x);r=sqrt(x*x+y*y);y1=sin(a+c)*r
   d=hoek(y1,z);r1=sqrt(y1*y1+z*z)
   X=transx(cos(a+c)*r);Y=transy(cos(b+d)*r1)
   return [X,Y]
def asteken(x,y,z):
   lis=[];X,Y=omzet(-x,-y,-z);lis.append(X);lis.append(Y)
   X,Y=omzet(x,y,z);lis.append(X);lis.append(Y)
   line = C.create_line(lis,fill='black',arrow='last');lis=[]

def niv0():
   y=-ma;stap=ma/15
   while y<ma:
```

```
        y=y+stap;lis=[];x=-ma
        while x<ma:
            x=x+stap
            z=f(x,y);X,Y=omzet(x,y,z)
            lis.append(X);lis.append(Y)
        line = C.create_line(lis,fill=kleur)
def niv1():
    x=-ma;stap=ma/15
    while x<ma:
        x=x+stap;lis=[];y=-ma
        while y<ma:
            y=y+stap
            z=f(x,y);X,Y=omzet(x,y,z)
            lis.append(X);lis.append(Y)
        line = C.create_line(lis,fill=kleur)
def niv2():
    afm=1;y=y0-afm;stap=ma/15
    while y<y0+afm:
        y=y+stap;lis=[];x=x0-afm
        while x<x0+afm:
            x=x+stap
            z=r(x,y);X,Y=omzet(x,y,z)
            lis.append(X);lis.append(Y)
        line = C.create_line(lis,fill=kleur)
def niv3():
    afm=1;x=x0-afm;stap=ma/15
    while x<x0+afm:
        x=x+stap;lis=[];y=y0-afm
        while y<y0+afm:
            y=y+stap
            z=r(x,y);X,Y=omzet(x,y,z)
            lis.append(X);lis.append(Y)
        line = C.create_line(lis,fill=kleur)
def tekenP():
    b=0.15;
    lis=[omzet(x0+b,y0,z0-b),omzet(x0+b,y0,z0+b),omzet(x0+2*b,y0,z0+b),
    omzet(x0+2*b,y0,z0),omzet(x0+b,y0,z0)]
    line = C.create_line(lis,fill='blue')
    lis=[omzet(x0,y0,z0),omzet(x0,y0,0),omzet(0,y0,0),omzet(x0,y0,0),omzet(x0,0,0)]
    line = C.create_line(lis,fill='blue',dash=[1,2])
def teken():
    global ma,a,b,x0,y0,z0,dzx,dzy,lis,x0,y0,kleur,fs
    fs=E1.get();ma=float(E2.get());C.delete(ALL)
    a=radians(draaixy);b=radians(draaiyz)
#kromme, niveaulijnen tekenen // xz en // yz
    kleur='red';niv0();niv1()
    corp=E3.get()
    if corp!='':
        x0,y0=mult(corp)
        z0=f(x0,y0);dzx=dfx(x0,y0);dzy=dfy(x0,y0)
        #raakvlak, niveaulijnen tekenen // xz en // yz
        kleur='green';niv2();niv3()
        tekenP()
#assen
    asteken(ma,0,0);asteken(0,ma,0);asteken(0,0,ma)
```

```
#x,y,z letters
   hl=0.17;st=ma+0.03
lis=[omzet(st,0,0),omzet(st+hl,0,hl),omzet(st+hl/2,0,hl/2),omzet(st,0,hl),omzet(st+hl,0,0)]
   line = C.create_line(lis,fill='black') #x
   lis=[omzet(0,st,0),omzet(0,st+hl,hl),omzet(0,st+hl/2,hl/2),omzet(0,st,hl)]
   line = C.create_line(lis,fill='black') #y
   lis=[omzet(0,0,st),omzet(-hl,0,st),omzet(0,0,st+hl),omzet(-hl,0,st+hl)]
   line = C.create_line(lis,fill='black') #z
def draai():
   global draaixy,draaiyz
   if rich==0:draaixy=draaixy+30
   if rich==1:draaiyz=draaiyz-30
   if rich==2:draaixy=draaixy-30
   if rich==3:draaiyz=draaiyz+30
   teken()
def nieuw():
   E1.delete(0,len(E1.get()));E2.delete(0,len(E2.get()))
   E3.delete(0,len(E3.get()));C.delete(ALL)
   global draaixy,draaiyz;draaixy=20;draaiyz=20
def sel():
   global rich;rich=var.get()
breedte=480;hoogte=510;hoogteC=hoogte-65;hoInv=hoogte-55;hoInv2=hoogte-30
lis=[]
form=Tk();form.geometry(str(breedte)+'x'+str(hoogte))
form.title('Grafiek van z=f(x,y) , raakvlak in P')
C = Canvas(form, bg="white", height=hoogteC, width=breedte)
C.pack();draaixy=20;draaiyz=20
L1=Label(form,text='Functie:',);L1.place(x=0,y=hoInv)
E1=Entry(form);E1.place(x=45,y=hoInv,width=200)
L2=Label(form,text='Max xyz:',);L2.place(x=260,y=hoInv)
E2=Entry(form);E2.place(x=315,y=hoInv,width=30)
L3=Label(form,text='P:',);L3.place(x=360,y=hoInv)
E3=Entry(form);E3.place(x=380,y=hoInv,width=90)
B1=Button(form,text='Teken',command=teken);B1.place(x=345,y=hoInv2)
B2=Button(form,text='Draai',command=draai);B2.place(x=390,y=hoInv2)
B3=Button(form,text='Nieuw',command=nieuw);B3.place(x=430,y=hoInv2)
pijl=['\u21E6','\u21E7','\u21E8','\u21E9'] #unicode pijlen
rich=0;var = IntVar()
L4=Label(form,text='Draairichting:');L4.place(x=2,y=hoInv2)
for i in range(0,4):
        rb=Radiobutton(form,text=pijl[i],variable=var,value=i,command=sel)
        rb.place(x=80+40*i,y=hoInv2)
form.mainloop()
```



112. 3 dimensionale grafieken van parametervergelijkingen grafiek_3dimensies_param_vgl

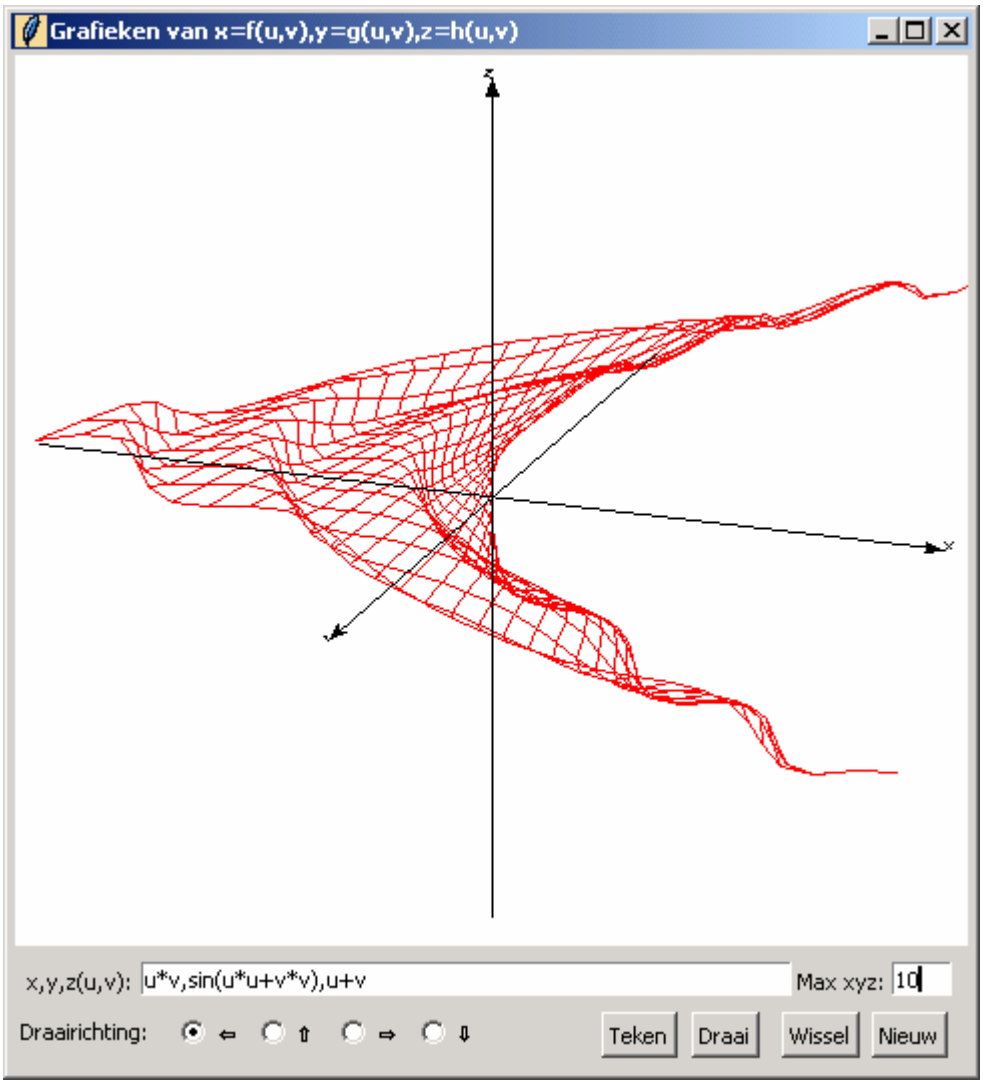
Dit programma is terug een variante van het 3-dim programma.
Hier moeten x, y en z ingevoerd worden als functies van 2 parameters u en v, gescheiden door komma's. Ook hier bestaat de mogelijkheid om de grafiek in 4 richtingen te wentelen. Bovendien kunnen de voorschriften van x, y en z verwisseld worden met de wisselknop.

Programma:

```
# 3 dimensionale grafieken van parametervergelijkingen
from math import *
from tkinter import *
def transx(x):
    return (x-xmi)*breedte/(xma-xmi)
def transy(y):
    return hoogteC-hoogteC*(y-ymi)/(yma-ymi)
def f(u,v):
    return eval(lijstfun[0])
def g(u,v):
    return eval(lijstfun[1])
def h(u,v):
    return eval(lijstfun[2])
def hoek(y,x):
    if y==0 and x>=0:c=0
    elif y==0 and x<0:c=pi
```

```
    elif x==0 and y>0:c=pi/2
    elif x==0 and y<0:c=3*pi/2
    else:
        c=atan(y/x)
        if x<0:c=c+pi
    return(c)
def omzet(x,y,z):
    c=hoek(y,x)
    r=sqrt(x*x+y*y);y1=sin(a+c)*r
    d=hoek(y1,z)
    r1=sqrt(y1*y1+z*z)
    X=transx(cos(a+c)*r);Y=transy(cos(b+d)*r1)
    return [X,Y]
def asteken(x,y,z):
    lis=[]
    X,Y=omzet(-x,-y,-z);lis.append(X);lis.append(Y)
    X,Y=omzet(x,y,z);lis.append(X);lis.append(Y)
    line = C.create_line(lis,fill='black',arrow='last')
def sel():
    global rich;rich=var.get()
#hoofdprogramma
breedte=480;hoogte=510;hoogteC=hoogte-65;hoInv=hoogte-55;hoInv2=hoogte-30
form=Tk();form.geometry(str(breedte)+'x'+str(hoogte))
form.title('Grafieken van x=f(u,v),y=g(u,v),z=h(u,v)')
C = Canvas(form, bg="white", height=hoogteC, width=breedte)
C.pack();draaixy=20;draaiyz=20
L1=Label(form,text='x,y,z(u,v):',);L1.place(x=5,y=hoInv)
E1=Entry(form);E1.place(x=65,y=hoInv,width=325)
L2=Label(form,text='Max xyz:',);L2.place(x=390,y=hoInv)
E2=Entry(form);E2.place(x=440,y=hoInv,width=30)
def teken():
    global xmi,xma,ymi,yma,zmi,zma,umi,vmi,uma,vma,a,b,lijstfun
    lijstfun=(E1.get()).split(',')
    xma=float(E2.get());C.delete(ALL)
    yma=xma;zma=xma
    xmi=-xma
    ymi=xmi;zmi=xmi
    umi,vmi=-pi,-pi
    uma,vma=pi,pi
    a=radians(draaixy);b=radians(draaiyz)
#kromme
    stap=vma/15
#niveaulijnen tekenen voor constante u
    u=umi
    while u<uma:
        u=u+stap
        lis=[]
        v=vmi
        while v<vma:
            v=v+stap;x=f(u,v);y=g(u,v);z=h(u,v)
            X,Y=omzet(x,y,z)
            lis.append(X);lis.append(Y)
        line = C.create_line(lis,fill='red')
#niveaulijnen tekenen voor constante v
    v=vmi
```

```
    while v<vma:
        v=v+stap
        lis=[]
        u=umi
        while u<uma:
            u=u+stap;x=f(u,v);y=g(u,v);z=h(u,v)
            X,Y=omzet(x,y,z)
            lis.append(X);lis.append(Y)
        line = C.create_line(lis,fill='red')
#assen
    asteken(xma,0,0);asteken(0,yma,0);asteken(0,0,zma)
#x,y,z letters
    hl=0.17;st=xma+0.03
    lis=[omzet(st,0,0),omzet(st+hl,0,hl),omzet(st+hl/2,0,hl/2) ,omzet(st,0,hl),omzet(st+hl,0,0)]
    line = C.create_line(lis,fill='black') #x
    lis=[omzet(0,st,0),omzet(0,st+hl,hl),omzet(0,st+hl/2,hl/2) ,omzet(0,st,hl)]
    line = C.create_line(lis,fill='black') #y
    lis=[omzet(0,0,st),omzet(-hl,0,st),omzet(0,0,st+hl),omzet(-hl,0,st+hl)]
    line = C.create_line(lis,fill='black') #z
def draai():
    global draaixy,draaiyz
    if rich==0:draaixy=draaixy+30
    if rich==1:draaiyz=draaiyz-30
    if rich==2:draaixy=draaixy-30
    if rich==3:draaiyz=draaiyz+30
    teken()
def wissel():
    lijstfun=(E1.get()).split(',')
    wissel=lijstfun[2];lijstfun[2]=lijstfun[0]
    lijstfun[0]=lijstfun[1];lijstfun[1]=wissel
    E1.delete(0,len(E1.get()))
    lf=lijstfun[0]+','+lijstfun[1]+','+lijstfun[2]
    E1.insert(0,lf)
    teken()
def nieuw():
    E2.delete(0,len(E2.get()));E1.delete(0,len(E1.get()));C.delete(ALL)
    global draaixy,draaiyz;draaixy=20;draaiyz=20
B1=Button(form,text='Teken',command=teken);B1.place(x=295,y=hoInv2)
B2=Button(form,text='Draai',command=draai);B2.place(x=340,y=hoInv2)
B3=Button(form,text='Wissel',command=wissel);B3.place(x=385,y=hoInv2)
B4=Button(form,text='Nieuw',command=nieuw);B4.place(x=430,y=hoInv2)
pijl=['\u21E6','\u21E7','\u21E8','\u21E9'] #unicode pijlen
rich=0;var = IntVar()
L4=Label(form,text='Draairichting:');L4.place(x=2,y=hoInv2)
for i in range(0,4):
    rb=Radiobutton(form,text=pijl[i],variable=var,value=i,command=sel)
    rb.place(x=80+40*i,y=hoInv2)
form.mainloop()
```



113. Kegelsneden: snijden van een vlak met een kegel kegelsneden_3dim

Door een rechte $y=k \cdot x$ te wentelen rond de x-as krijgen we een dubbele kegel met top in de oorsprong van een 3-dimensionaal assenstelsel. We snijden de kegel met een verticaal vlak $p \cdot x+q.y=s$.
Afhankelijk van de waarden van p,q en s zal de snijlijn een cirkel, ellips, een parabool of een hyperbool zijn.
Het volgende programma toont de kegel, het snijvlak en de kegelsnede in '3 dimensies' of in 'bovenaanzicht'

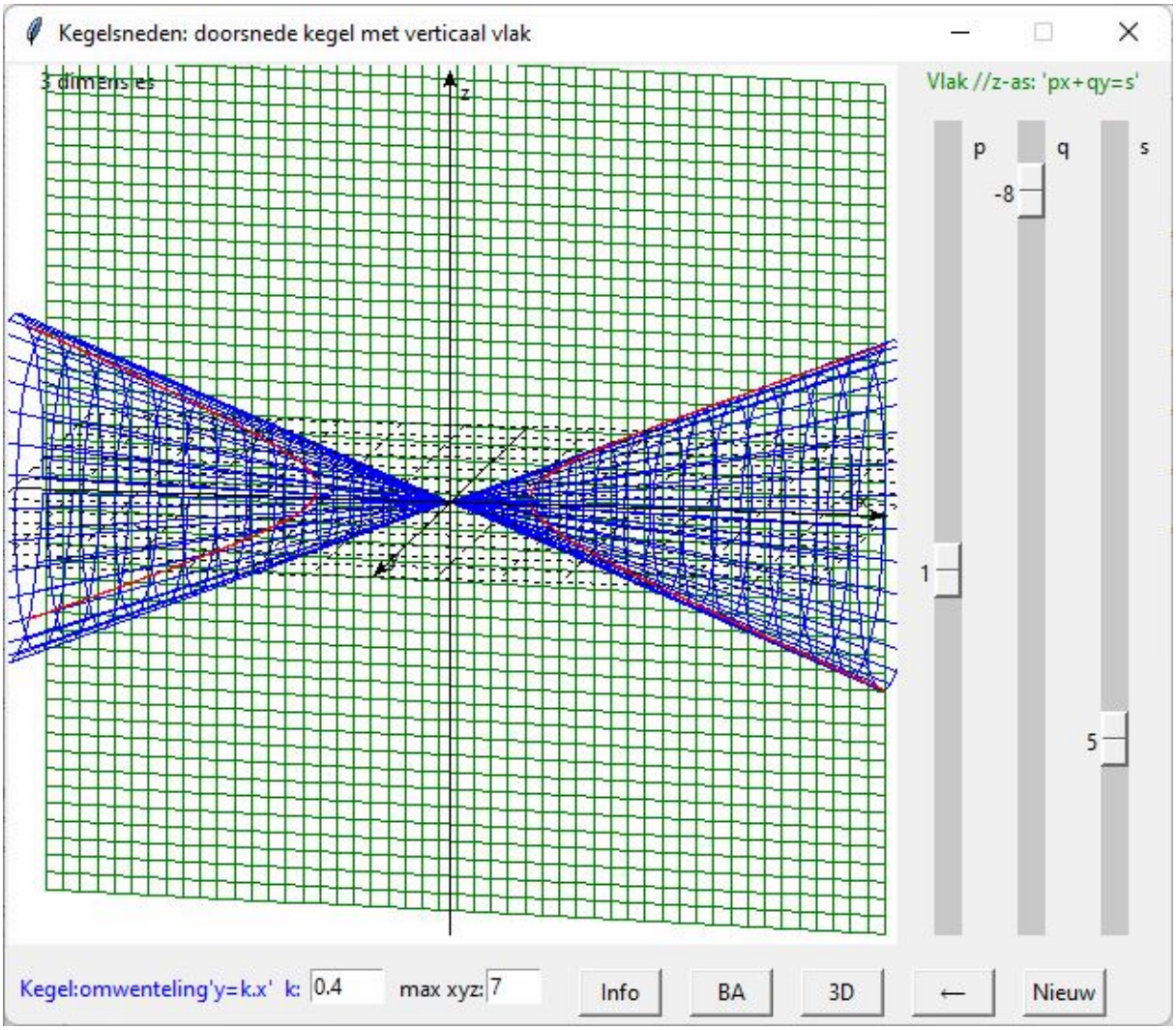
Programma:

```
# 3 dimensionale grafieken: kegelsneden
from math import *;from tkinter import *;from tkinter import messagebox
def transx(x):return (x-mi)*breedteC/(ma-mi)
def transy(y):return hoogteC-hoogteC*(y-mi)/(ma-mi)
def hoek(y,x):
    if y==0 and x>=0:c=0
    elif y==0 and x<0:c=pi
    elif x==0 and y>0:c=pi/2
    elif x==0 and y<0:c=3*pi/2
    else:
```

```
        c=atan(y/x)
        if x<0:c=c+pi
    return(c)
def omzet(x,y,z):
    c=hoek(y,x)
    r=sqrt(x*x+y*y);y1=sin(a+c)*r
    d=hoek(y1,z)
    r1=sqrt(y1*y1+z*z)
    X=transx(cos(a+c)*r);Y=transy(cos(b+d)*r1)
    return [X,Y]
def asteken(x,y,z):
    lis=[];ap(lis,-x,-y,-z);ap(lis,x,y,z)
    C.create_line(lis,fill=bl,arrow='last')
def teken():
    global mi,ma,a,b,p,q,s
    C.create_text(50,10,text=graftit,fill=bl)
    ma=float(E2.get());mi=-ma
    a=radians(draaixy);b=radians(draaiyz)
    p=float(var_p.get());q=float(var_q.get());s=float(var_s.get())
#vlak px+qy=s ( // z-as)
    stap=ma/25
    if q!=0:
        z=mi
        while z<ma:
            z=z+stap;lis=[];x=mi
            while x<ma:
                x=x+stap;y=(s-p*x)/q;ap(lis,x,y,z)
            C.create_line(lis,fill=gr)
        x=mi
        while x<ma:
            x=x+stap;lis=[];z=mi
            while z<ma:
                z=z+stap;y=(s-p*x)/q;ap(lis,x,y,z)
            C.create_line(lis,fill=gr)
    elif p!=0:
        z=mi
        while z<ma:
            z=z+stap;lis=[];y=mi
            while y<ma:
                y=y+stap;x=s/p;ap(lis,x,y,z)
            C.create_line(lis,fill=gr)
        y=mi
        while y<ma:
            y=y+stap;lis=[];z=mi
            while z<ma:
                z=z+stap;x=s/p;ap(lis,x,y,z)
            C.create_line(lis,fill=gr)
    else:messagebox.showinfo('Fout','p en q niet tegelijkertijd 0 stellen')
#kegel wenteling van de rechte y =kx rond de x-as C.V.: z²+y²=k²x²
    k=float(E1.get())
    stap=0.5;stapt=0.2;xi=mi-stap # cirkels
```

```
    while xi<ma:
        xi=xi+stap;t=0;lis=[]
        while t<2*pi:
            t=t+stapt;x=xi;z=k*xi*sin(t);y=k*xi*cos(t);ap(lis,x,y,z)
        C.create_line(lis,fill='blue')
    t=-stapt # rechten
    while t<2*pi:
        lis=[];t=t+stapt
        x=mi;z=k*mi*sin(t);y=k*mi*cos(t);ap(lis,x,y,z)
        x=ma;z=k*ma*sin(t);y=k*ma*cos(t);ap(lis,x,y,z)
        C.create_line(lis,fill='blue')
#doorsnede vlak en kegel:
#als p=0:cirkel, als q=0: hyperbool, als p en q niet 0:ellips, als q/p=k:parabool
    lis1,lis2,lis3,lis4=[],[],[],[];stap=0.01
    if q!=0:
        x=mi-stap
        while x<ma:
            x=x+stap;y=(s-p*x)/q;t=k*x
            if y*y<=t*t:
                z1=t*sqrt(1-y*y/t/t);z2=-z1
                if x<0:ap(lis1,x,y,z1);ap(lis2,x,y,z2)
                if x>=0:ap(lis3,x,y,z1);ap(lis4,x,y,z2)
        for lis in (lis1,lis2,lis3,lis4):
            if lis!=[]:C.create_line(lis,fill='red')
    elif p!=0:
        y=mi-stap
        while y<ma:
            y=y+stap;x=s/p;t=k*x
            if y*y<=t*t:
                z1=t*sqrt(1-y*y/t/t);z2=-z1
                ap(lis1,x,y,z1);ap(lis2,x,y,z2)
        if lis1!=[]:C.create_line(lis1,fill='red')
        if lis2!=[]:C.create_line(lis2,fill='red')
#assen 3d
    asteken(ma,0,0);asteken(0,ma,0);asteken(0,0,ma)
#xyz
    d=0.3
    lis=[];ap(lis,ma-d,d,d);C.create_text(lis,text='x',fill=bl)
    lis=[];ap(lis,d,ma-d,d);C.create_text(lis,text='y',fill=bl)
    lis=[];ap(lis,d,d,ma-d);C.create_text(lis,text='z',fill=bl)
#rooster
    for x in range(int(mi),int(ma+1)):
        lis=[];ap(lis,x,mi,0);ap(lis,x,ma,0);C.create_line(lis,fill=bl,dash=[1,2])
    for y in range(int(mi),int(ma+1)):
        lis=[];ap(lis,mi,y,0);ap(lis,ma,y,0);C.create_line(lis,fill=bl,dash=[1,2])
def boven():
    global draaiyz,graftit;graftit='bovenaanzicht';C.delete(ALL);draaiyz=90;teken()
def teken3d():
    global draaiyz,graftit;graftit='3 dimensies';C.delete(ALL);draaiyz=10;teken()
def ap(li,x,y,z):X,Y=omzet(x,y,z);li.append(X);li.append(Y)
def draai():
```

```
   global draaixy;draaixy=draaixy+7;C.delete(ALL);teken()
def info():messagebox.showinfo(tit+': info',infstr)
def nieuw():
   E2.delete(0,len(E2.get()));E1.delete(0,len(E1.get()));C.delete(ALL)
   global draaixy;draaixy=10
#hoofdprogramma
tit='Kegelsneden: doorsnede kegel met verticaal vlak';bl='black';gr='green'
breedte=630;hoogte=520;hoogteC=hoogte-45;hoInv=hoogte-30;breedteC=breedte-150
form=Tk();form.geometry(str(breedte)+'x'+str(hoogte));form.title(tit);form.resizable(False,False)
C = Canvas(form, bg="white", height=hoogteC, width=breedteC)
C.place(x=0,y=0);draaixy=10;draaiyz=10
var_p,var_q,var_s=DoubleVar(),DoubleVar(),DoubleVar()
wp=Scale(form,from_=-9,to=9,length=440,bd=0,orient='vertical', resolution=1,label='p',variable=var_p)
wp.set(3);wp.place(x=breedteC+5,y=30)
wq=Scale(form,from_=-9,to=9,length=440,bd=0,orient='vertical', resolution=1,label='q',variable=var_q)
wq.set(2);wq.place(x=breedteC+50,y=30)
ws=Scale(form,from_=-9,to=9,length=440,bd=0,orient='vertical', resolution=1,label='s',variable=var_s)
ws.set(5);ws.place(x=breedteC+95,y=30)
Label(form,text="Vlak //z-as: 'px+qy=s' ",fg=gr).place(x=breedteC+15,y=0)
Label(form,text="Kegel:omwenteling'y=k.x'  k:",fg='blue').place(x=5,y=hoInv)
E1=Entry(form);E1.place(x=165,y=hoInv,width=40);E1.insert(0,'0.4')
L2=Label(form,text='max xyz:',);L2.place(x=210,y=hoInv)
E2=Entry(form);E2.place(x=260,y=hoInv,width=30);E2.insert(0,'7')
B1=Button(form,text='Info',command=info,width=5);B1.place(x=310,y=hoInv)
B2=Button(form,text='BA',command=boven,width=5);B2.place(x=370,y=hoInv)
B3=Button(form,text='3D',command=teken3d,width=5);B3.place(x=430,y=hoInv)
B4=Button(form,text='←',command=draai,width=5);B4.place(x=490,y=hoInv)
B5=Button(form,text='Nieuw',command=nieuw,width=5);B5.place(x=550,y=hoInv)
infstr="Doorsnede van een verticaal vlak\n"
infstr=infstr+"'px+qy=s'met een kegel 'z²+y²=k²x²'\n"
infstr=infstr+'Wijzig eventueel k en max x,y,z.\n'
infstr=infstr+'Met de sliders p,q,s instellen \n'
infstr=infstr+"Met 'BA' zie je een bovenaanzicht,\n"
infstr=infstr+"met '3D' een 3-dimensionale grafiek\n"
infstr=infstr+'van vlak en kegel en eventueel de\n'
infstr=infstr+'doorsnede van beide(een kegelsnede):\n'
infstr=infstr+'een cirkel, ellips, parabool of hyperbool\n'
infstr=infstr+"Met '←' kan je de grafiek draaien."
form.mainloop()
```



114. Formule van de booglengte formule_booglengte

We verdelen een interval [a,b] in n gelijke stukken Δx.

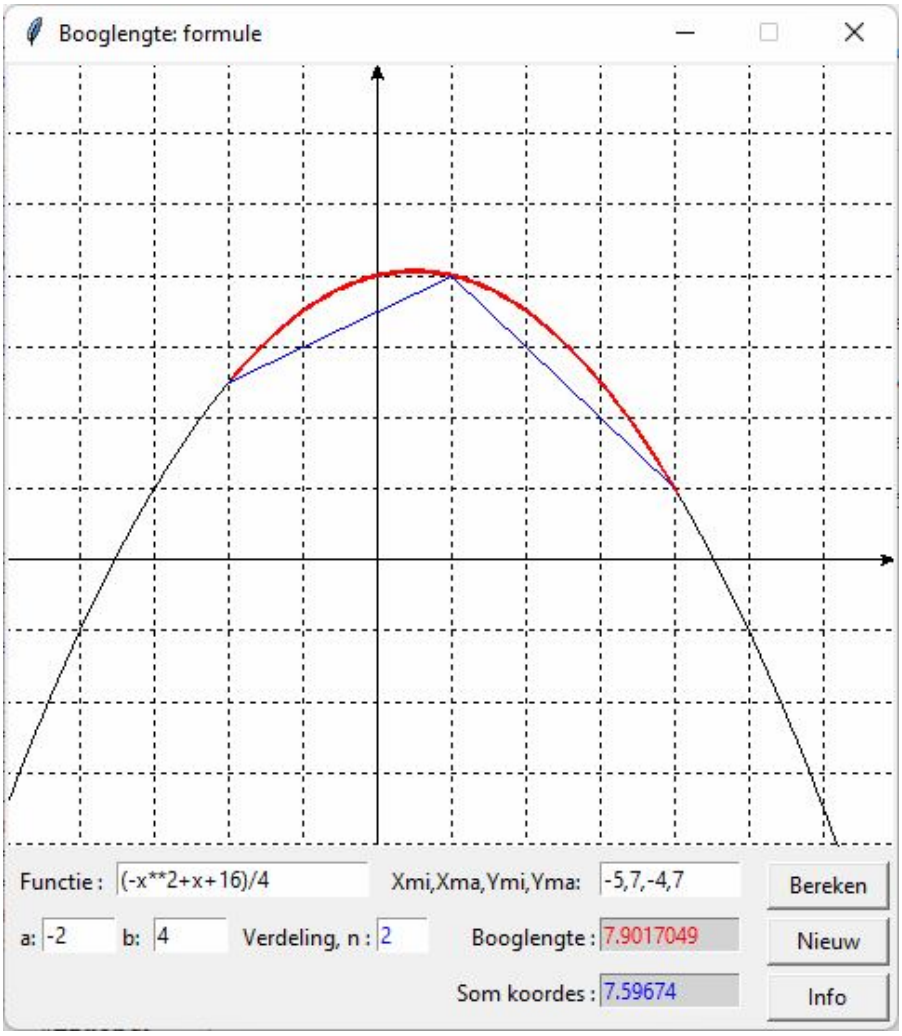
Met de formule $\sum_{i=1}^{n}\sqrt{\Delta x^2+[f(x_i+\Delta x)-f(x_i)]^2}$ waarbij $x_i = a+(i-1)\cdot\Delta x$ berekenen we de som van de lengtes van de koorden aan de kromme tussen (a, f(a)) en (b, f(b)). Als we $n\to\infty$ zien we met dit programma dat deze som nadert tot de booglengte tussen (a, f(a)) en (b, f(b)).

Programma:

```
# formule van de booglengte
from math import *;from tkinter import *;from tkinter import messagebox
def mult(s):
    l=s.split(',');co=[]
    for i in l:co.append(float(i))
    return co
def transx(x):return (x-xmi)*breedte/(xma-xmi)
def transy(y):return hoogteC-hoogteC*(y-ymi)/(yma-ymi)
def f(x):return eval(E1.get())
```

```
def df(x):return (f(x+dx)-f(x-dx))/dx/2
def bl(x):return sqrt(1+df(x)**2)
def integr(a,b): # integraalberekening booglengte met simpson
    n=100;h=(b-a)/n;x=a;som=0
    for j in range(1,n):
        x=x+h
        if j%2==0:t=2
        else:t=4
        som=som+t*bl(x)
    som=som+bl(a)+bl(b)
    integ=som*h/3
    return integ
def bereken():
    global xmi,xma,ymi,yma,a,b,n,h;nwk='.7f'
    xmi,xma,ymi,yma=mult(E2.get());C.delete(ALL)
    a=float(E3.get());b=float(E4.get())
    opp=integr(a,b)
    wis(E5);E5.insert(0,format(opp,nwk))
#assen
    X=transx(0);C.create_line(X,0,X,hoogteC,fill='black',arrow='first')
    Y=transy(0);C.create_line(0,Y,breedte,Y,fill='black',arrow='last')
# rooster
    for x in range(int(xmi),int(xma+1)):
        X=transx(x);C.create_line(X,0,X,hoogteC,fill='black',dash=[1,2])
    for y in range(int(ymi),int(yma+1)):
        Y=transy(y);C.create_line(0,Y,breedte,Y,fill='black',dash=[1,2] )
#kromme
    lis=[];stap=0.05;x=xmi
    while x<xma:
        x=x+stap;y=f(x);ap(lis,x,y)
    C.create_line(lis,fill='black')
#boog dik aanzetten
    for dik in (-0.02,0,0.02):
        lis=[];stap=0.05;x=a
        while x<b:
            x=x+stap;y=f(x)+dik;ap(lis,x,y)
        C.create_line(lis,fill='red')
#som koordes
    n=int(E6.get());del_x=(b-a)/n
    som=0;x=a;lis=[]
    for i in range(0,n):
        som=som+sqrt(del_x**2+(f(x+del_x)-f(x))**2)
        ap(lis,x,f(x));x=x+del_x
    ap(lis,x,f(x));C.create_line(lis,fill='blue')
    wis(E7);E7.insert(0,format(som,'.5f'))
def info():h=messagebox.showinfo(tit,infstr)
def halveerdel_x():del_x=float(E6.get());del_x=del_x/2;wis(E6);E6.insert(0,str(del_x))
def ap(li,x,y):X=transx(x);Y=transy(y);li.append(X);li.append(Y)
def wis(E):E.delete(0,len(E.get()))
def nieuw():
    for E in (E1,E2,E3,E4,E5,E6,E7):wis(E)
```

```
    C.delete(ALL)
#Hoofdprogramma
tit='Booglengte: formule';gr='lightgrey';dx=1E-6
breedte=480;hoogte=520;hoogteC=hoogte-100;hoInv1=hoogte-90;hoInv2=hoogte-60;hoInv3=hoogte-30
form=Tk();form.geometry(str(breedte)+'x'+str(hoogte))
form.title(tit);form.resizable(False,False)
C = Canvas(form, bg="white", height=hoogteC, width=breedte);C.pack()
L1=Label(form,text='Functie :',);L1.place(x=5,y=hoInv1)
E1=Entry(form,width=22);E1.place(x=60,y=hoInv1);E1.insert(0,'(-x**2+x+16)/4')
L2=Label(form,text='Xmi,Xma,Ymi,Yma:',);L2.place(x=205,y=hoInv1)
E2=Entry(form,width=12);E2.place(x=320,y=hoInv1);E2.insert(0,'-5,7,-4,7')
L3=Label(form,text='a:',);L3.place(x=5,y=hoInv2)
E3=Entry(form,width=6);E3.place(x=20,y=hoInv2);E3.insert(0,'-2')
L4=Label(form,text='b:',);L4.place(x=60,y=hoInv2)
E4=Entry(form,width=6);E4.place(x=80,y=hoInv2);E4.insert(0,'4')
L5=Label(form,text='Booglengte :',);L5.place(x=248,y=hoInv2)
E5=Entry(form,bg=gr,fg='red',width=12);E5.place(x=320,y=hoInv2)
L6=Label(form,text='Verdeling, n :',);L6.place(x=125,y=hoInv2)
E6=Entry(form,fg='blue',width=4);E6.place(x=200,y=hoInv2);E6.insert(0,'2')
L7=Label(form,text='Som koordes :',);L7.place(x=240,y=hoInv3)
E7=Entry(form,bg=gr,fg='blue',width=12);E7.place(x=320,y=hoInv3)
Button(form,text='Bereken',command=bereken,width=8).place(x=410,y=hoInv1)
Button(form,text='Nieuw',command=nieuw,width=8).place(x=410,y=hoInv2)
Button(form,text='Info',command=info,width=8).place(x=410,y=hoInv3)
infstr='Een formule voor de booglengte vinden we\ndoor eerst het interval [a,b] te verdelen in n\n'
infstr=infstr+'gelijke lijnstukken Δx  en de som van de\n koordes te nemen, dit is: Σ√(Δx²+Δy²)\n'
infstr=infstr+" Σ√ [1+(Δy/Δx)²]Δx. Als we n laten naderen\n tot ∞ nadert deze functie tot ∫ₐᵇ√(1+y'²)dx"
form.mainloop()
```



115. Formule voor de inhoud van een omwentelingslichaam formule_inhoud

Voor dit programma gebruiken we -nogmaals- de functies om een 3 dimensionale grafiek te tekenen. De x-as blijft, de y-as komt uit het vlak van het scherm, de z-as =verticaal. Voor het 3-dimensionaal effect voorzien we terug 2 kantelingen: *draaixy* rond de Z-as en *draaiyz* rond de x-as.
draaixy kan gewijzigd worden zodat de grafiek rond de z-as kan draaien.
Voor *draaiyz* kunnen we kiezen voor 15°: '3dim' -- of 90° , 'boven' die ons een 2-dimensionaal bovenaanzicht geeft.
Kiezen we de knop ∫.., dan wordt met simpson de integraal berekend van de functie $\pi \cdot f(x)^2$
Met de knop Σ.. wordt de som van de inhouden van '#cilind' tussen a en b berekend.
Als je dit aantal groot laat worden, nadert deze som tot de inhoud van het omwentelingslichaam

Programma:

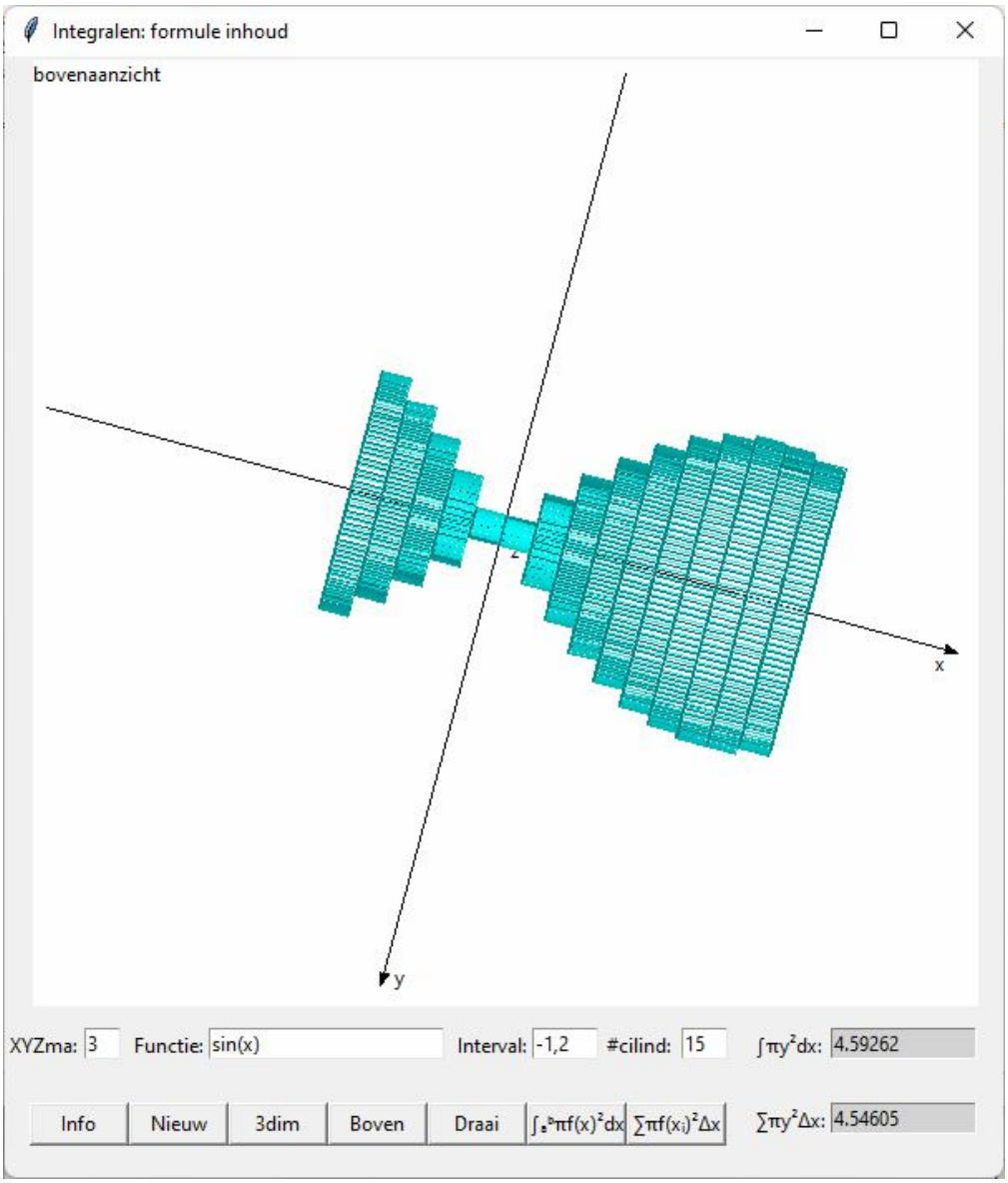
```
# integraal_grafiekTK: formule inhoud
from math import *;from tkinter import *;from copy import *;from tkinter import messagebox
def col(rgb):return '#%02x%02x%02x' % rgb
def f(x):return eval(E1.get())
def mult(s):
    l=s.split(',');co=[]
    for i in l:co.append(float(i))
```

```
    return co
def transx(xp):return (xp-xmi)*afm/(xma-xmi)
def transy(yp):return afm-afm*(yp-ymi)/(yma-ymi)
def det(mat):
    le=len(mat)
    if le==2:d=mat[0][0]*mat[1][1]-mat[0][1]*mat[1][0]
    else:
        d=0
        for i in range(0,le):
            s=[]
            for k in range(0,le):
                if i!=k:s.append(mat[k][1:])
            d=d+mat[i][0]*(-1)**i*det(s)
    return d
def prod(b,c): # product 2 matrices
    n=len(b);p=[]
    for i in range(0,n):
        som=0
        for k in range(0,n):
            som=som+b[i][k]*c[k]
        p.append(som)
    return(p)
# ----- grafiek ruimtemeetkunde
def hoek(y,x):
    if y==0 and x>=0:c=0
    elif y==0 and x<0:c=pi
    elif x==0 and y>0:c=pi/2
    elif x==0 and y<0:c=3*pi/2
    else:
        c=atan(y/x)
        if x<0:c=c+pi
    return(c)
def omzet(x,y,z):
    c=hoek(y,x)
    r=sqrt(x*x+y*y);y1=sin(xyH+c)*r
    d=hoek(y1,z)
    r1=sqrt(y1*y1+z*z)
    X=transx(cos(xyH+c)*r);Y=transy(cos(yzH+d)*r1)
    return [X,Y]
def asteken(x,y,z):
    lis=[];ap(lis,-x,-y,-z);ap(lis,x,y,z)
    C.create_line(lis,fill=bl,arrow='last')
def rotmat(v):return [[1,0,0],[0,cos(v),sin(v)],[0,-sin(v),cos(v)]]
def tekenas():
    global xma,xmi,yma,ymi,zma,zmi,tmi,vmi,tma,vma,xyH,yzH
    C.delete(ALL);xma=float(E2.get());vlag=0
    yma=xma;zma=xma;xmi=-xma;ymi=xmi;zmi=xmi
    xyH=radians(draaixy);yzH=radians(draaiyz)
#assen
    asteken(xma,0,0);asteken(0,yma,0);asteken(0,0,zma)
#x,y,z letters
```

```
    di=0.1;totd=xma-di
    lis=[];ap(lis,totd,di,di);C.create_text(lis,text='x',fill=bl)
    lis=[];ap(lis,di,totd,di);C.create_text(lis,text='y',fill=bl)
    lis=[];ap(lis,di,di,totd);C.create_text(lis,text='z',fill=bl)
    if draaiyz==90:graftit='bovenaanzicht'
    else:graftit=' 3 dimensies'
    C.create_text(40,10,text=graftit,fill=bl)
def teken3d(a,b):
    global kl,vlag;tekenas();tmi=a;tma=b;vmi,vma=-pi,pi+0.5;vlag=0
#kromme
#niveaulijnen tekenen voor constante u
    stap=vma/roos;t=tmi-stap
    while t<tma:
        t=t+stap;ro=int(175+75*sin(t));kl=col((ro,0,0))
            lis=[];V=[t,f(t),0];v=vmi-stap
        while v<vma:v=v+stap;R=rotmat(v);x,y,z=prod(R,V);ap(lis,x,y,z)
        line = C.create_line(lis,fill=kl)
#niveaulijnen tekenen voor constante v
    v=vmi-stap
    while v<vma:
        v=v+stap;gr=int(175+75*sin(v));kl=col((0,gr,0))
        R=rotmat(v);lis=[];t=tmi-stap
        while t<tma:t=t+stap;V=[t,f(t),0];x,y,z=prod(R,V);ap(lis,x,y,z)
        line = C.create_line(lis,fill=kl)
    cirkelvlak(a,f(a)/10,f(a));cirkelvlak(b,f(b)/10,f(b))
def tekencil():
    global kl,vlag
    vlag=1;n=int(E5.get());a,b=mult(E3.get());tekenas()
    del_x=(b-a)/n;x=a
    somcil=0;
    for i in range(0,n):
        gem_fun=(f(x)+f(x+del_x))/2 # gemiddelde functiewaarde in interval
        somcil=somcil+del_x*pi*(gem_fun)**2
        vmi,vma=-pi,pi+0.5;stap=0.05;v=vmi-stap;t=x
        while v<vma:
            v=v+stap;bl=int(175+75*sin(v))
             kl=col((0,bl,bl));R=rotmat(v);lis=[]
            V=[x,gem_fun,0];x1,y1,z1=prod(R,V);ap(lis,x1,y1,z1)
            V=[x+del_x,gem_fun,0];x1,y1,z1=prod(R,V);ap(lis,x1,y1,z1)
            line = C.create_line(lis,fill=kl)
        cirkelvlak(x,gem_fun/10,gem_fun)
        x=x+del_x
    cirkelvlak(x,gem_fun/10,gem_fun)
    wis(E6);E6.insert(0,format(somcil,'.5f'))
    return
def cirkelvlak(d,r1,r2): # linker of rechter zijvlak bij inhoud
    vmi,vma=-pi,pi+0.5;stap=vma/roos;t=r1
    while t<r2:
        lis=[];V=[d,t,0];v=vmi-stap
        while v<vma:v=v+stap;R=rotmat(v);x,y,z=prod(R,V);ap(lis,x,y,z)
        line = C.create_line(lis,fill=kl);t=t+stap
```

```
    v=vmi-stap;lis=[]
    while v<vma:
        v=v+stap;V=[d,0,r1];R=rotmat(v);x,y,z=prod(R,V);ap(lis,x,y,z)
        V=[d,0,r2];R=rotmat(v);x,y,z=prod(R,V);ap(lis,x,y,z)
    line = C.create_line(lis,fill=kl)
def draai():
    global draaixy,draaiyz,vlag;a,b=mult(E3.get())
    draaixy=draaixy+draaihoek
    C.delete(ALL)
    if vlag==0:teken3d(a,b)
    else:tekencil()
def dim3():
    global draaixy,draaiyz,vlag;a,b=mult(E3.get());draaiyz=15
    C.delete(ALL)
    if vlag==0:teken3d(a,b)
    else:tekencil()
def boven():
    global draaixy,draaiyz,vlag;a,b=mult(E3.get());draaiyz=90;C.delete(ALL)
    if vlag==0:teken3d(a,b)
    else:tekencil()
def ap(li,x,y,z):X,Y=omzet(x,y,z);li.append(X);li.append(Y)
def df(x):return(f(x+dx)-f(x-dx))/dx/2
def inh(x):return pi*f(x)*f(x)
def but(xp,yp,t,co):Button(form,text=t,command=co). place(x=xp*sc,y=yp*sc,width=4*sc)
def ent(t,kl,xp,yp,w):lab(t,xp,yp);en=Entry(form,width=w,bg=kl); en.place(x=sc*(xp+3.2),y=sc*yp);return
en
def lab(t,xp,yp):Label(form,text=t).place(x=sc*xp,y=sc*yp)
def bereken():
    global a,b;a,b=mult(E3.get())
    if b<a:wissel=a;a=b;b=wissel;wis(E3);E3.insert(0,str(a)+','+str(b))
    n=100;h=(b-a)/n;x=a;som=0
    for j in range(1,n):
        x=x+h
        if j%2==0:t=2
        else:t=4
        som=som+t*inh(x)
    som=som+inh(a)+inh(b);integ=som*h/3;wis(E4);intst=format(integ,'.5f');wis(E4);E4.insert(0,intst)
    C.delete(ALL);teken3d(a,b);cirkelvlak(a,f(a)/10,f(a));cirkelvlak(b,f(b)/10,f(b))
def info():messagebox.showinfo(tit+': info',infstr)
def wis(E):E.delete(0,len(E.get()))
def nieuw():
    global xt,yt,Xl,Yl,costr,draaixy,draaiyz
    xt=[];yt=[];Xl=[];Yl=[];costr=E2.get()
    for en in entr:wis(en)
    C.delete(ALL)
    draaixy,draaiyz=15,15
#hoofdprogramma
yp=40;sc=15;tit='Integralen: formule inhoud';lg='lightgrey';wh='white';bl='black'
form=Tk();roos=20;vlag=0;draaihoek=7;draaixy,draaiyz=15,15
xp=yp;form.geometry(str(xp*sc+2)+'x'+str((yp+5)*sc));form.title(tit);afm=(yp-2)*sc
C=Canvas(form,bg=wh,height=afm,width=afm);C.place(x=sc,y=0)
```

```
E2=ent('XYZma:',wh,0,39,3);E1=ent('Functie:',wh,5,39,23);E3=ent('Interval:',wh,18,39,6)
E5=ent('#cilind:',wh,24,39,4);E4=ent('∫πy²dx:',lg,30,39,14);E6=ent('∑πy²Δx:',lg,30,42,14)
costr='4';E1.insert(0,'sin(x)');E2.insert(0,costr);ab='-1,2'; E3.insert(0,ab);E5.insert(0,'10')
entr=[E1,E2,E3,E4,E5,E6]
but(1,42,'Info',info);but(5,42,'Nieuw',nieuw);but(9,42,'3dim',dim3);but(13,42,'Boven',boven)
but(17,42,'Draai',draai);but(21,42,'∫ₐᵇ πf(x)²dx',bereken);but(25,42,'∑πf(xᵢ)²Δx',tekencil)
infstr='Demonstratie van de formule voor de inhoud\nvan het volume dat we bekomen door y=f(x)\n'
infstr=infstr+'te wentelen rond de x-as\n'
infstr=infstr+'∫ₐᵇ πf(x)²dx =lim(n→ ∞) ∑πf(xᵢ)²Δx\n'
form.mainloop()
```



116. Formule voor de zijdelingse oppervlakte van een omwentelingslichaam
formule_zijdelingse_oppervlakte

Het programma is bijna hetzelfde als het vorige, maar hier wordt de integraal $\int_a^b 2\pi\left|f(x)\right|\cdot\sqrt{1+f'(x)^2}\cdot dx$ met simpson berekend . Voor de sommatie 'Σ..' wordt het interval [a,b] verdeeld in '#..' afgeknotte kegels en dan wordt de som genomen van de zijdelingse oppervlaktes van deze kegels.

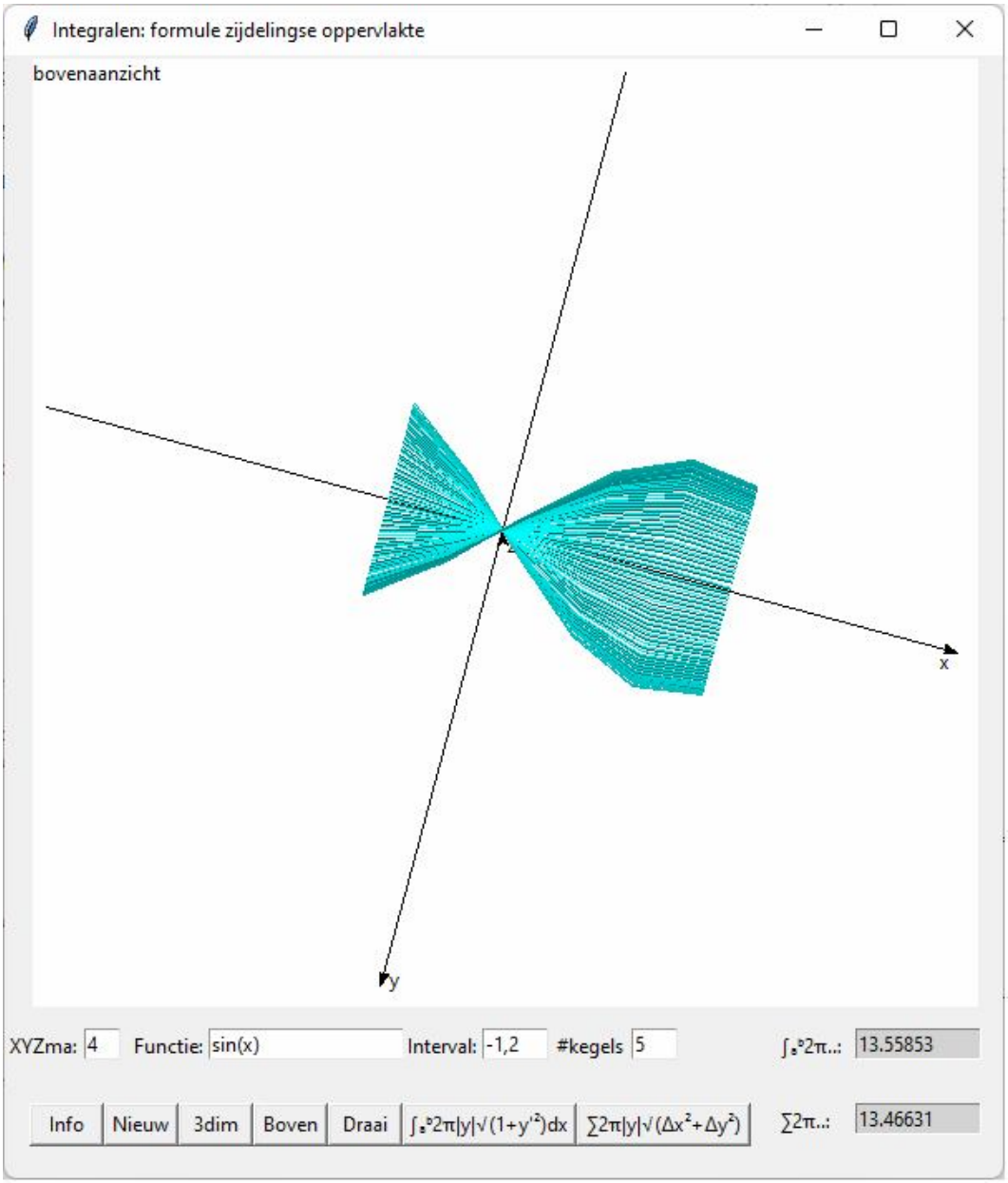
Programma:

```
# integraal_grafiekTK:formule zijdelingse oppervlakte
from math import *;from tkinter import *;from copy import *;from tkinter import messagebox
def col(rgb):return '#%02x%02x%02x' % rgb
def f(x):return eval(E1.get())
def mult(s):
    l=s.split(',');co=[]
    for i in l:co.append(float(i))
    return co
def transx(xp):return (xp-xmi)*afm/(xma-xmi)
def transy(yp):return afm-afm*(yp-ymi)/(yma-ymi)
def det(mat):
    le=len(mat)
    if le==2:d=mat[0][0]*mat[1][1]-mat[0][1]*mat[1][0]
    else:
        d=0
        for i in range(0,le):
            s=[]
            for k in range(0,le):
                if i!=k:s.append(mat[k][1:])
            d=d+mat[i][0]*(-1)**i*det(s)
    return d
def prod(b,c): # product 2 matrices
    n=len(b);p=[]
    for i in range(0,n):
        som=0
        for k in range(0,n):
            som=som+b[i][k]*c[k]
        p.append(som)
    return(p)
# ----- grafiek ruimtemeetkunde
def hoek(y,x):
    if y==0 and x>=0:c=0
    elif y==0 and x<0:c=pi
    elif x==0 and y>0:c=pi/2
    elif x==0 and y<0:c=3*pi/2
    else:
        c=atan(y/x)
        if x<0:c=c+pi
    return(c)
def omzet(x,y,z):
    c=hoek(y,x)
    r=sqrt(x*x+y*y);y1=sin(xyH+c)*r
    d=hoek(y1,z)
    r1=sqrt(y1*y1+z*z)
    X=transx(cos(xyH+c)*r);Y=transy(cos(yzH+d)*r1)
    return [X,Y]
def asteken(x,y,z):
    lis=[];ap(lis,-x,-y,-z);ap(lis,x,y,z)
    C.create_line(lis,fill='black',arrow='last')
```

```
def rotmat(v):return [[1,0,0],[0,cos(v),sin(v)],[0,-sin(v),cos(v)]]
def tekenas():
   global xma,xmi,yma,ymi,zma,zmi,tmi,vmi,tma,vma,xyH,yzH
   C.delete(ALL);xma=float(E2.get());vlag=0
   yma=xma;zma=xma;xmi=-xma;ymi=xmi;zmi=xmi
   xyH=radians(draaixy);yzH=radians(draaiyz)
#assen
   asteken(xma,0,0);asteken(0,yma,0);asteken(0,0,zma)
#x,y,z letters
   di=0.1;totd=xma-di
   lis=[];ap(lis,totd,di,di);C.create_text(lis,text='x',fill=bl)
   lis=[];ap(lis,di,totd,di);C.create_text(lis,text='y',fill=bl)
   lis=[];ap(lis,di,di,totd);C.create_text(lis,text='z',fill=bl)
   if draaiyz==90:graftit='bovenaanzicht'
   else:graftit=' 3 dimensies'
   C.create_text(40,10,text=graftit,fill=bl)
def teken3d(a,b):
   global kl,vlag;tekenas();tmi=a;tma=b;vmi,vma=-pi,pi+0.5;vlag=0
#kromme
#niveaulijnen tekenen voor constante u
   stap=vma/roos;t=tmi-stap
   while t<tma:
      t=t+stap;ro=int(175+75*sin(t));kl=col((ro,0,0));
           lis=[];V=[t,f(t),0];v=vmi-stap
      while v<vma:v=v+stap;R=rotmat(v);x,y,z=prod(R,V);ap(lis,x,y,z)
      C.create_line(lis,fill=kl)
#niveaulijnen tekenen voor constante v
   v=vmi-stap
   while v<vma:
      v=v+stap;gr=int(175+75*sin(v));kl=col((0,gr,0));
           R=rotmat(v);lis=[];t=tmi-stap
      while t<tma:t=t+stap;V=[t,f(t),0];x,y,z=prod(R,V);ap(lis,x,y,z)
      C.create_line(lis,fill=kl)
def tekenkegel():
   global kl,vlag
   vlag=1;n=int(E5.get());a,b=mult(E3.get());tekenas()
   del_x=(b-a)/n;x=a
   somkegel=0;
   for i in range(0,n):
      r0=abs(f(x));r1=abs(f(x+del_x))
      somkegel=somkegel+pi*(r0+r1)*sqrt((r0-r1)**2+del_x**2)
      vmi,vma=-pi,pi+0.5;stap=0.05;v=vmi-stap;t=x
      while v<vma:
         v=v+stap;bl=int(175+75*sin(v));kl=col((0,bl,bl));
          R=rotmat(v);lis=[]
         V=[x,f(x),0];x1,y1,z1=prod(R,V);ap(lis,x1,y1,z1)
         V=[x+del_x,f(x+del_x),0];x1,y1,z1=prod(R,V);ap(lis,x1,y1,z1)
         C.create_line(lis,fill=kl)
      x=x+del_x
   wis(E6);E6.insert(0,format(somkegel,'.5f'))
   return
```

```
def draai():
    global draaixy,draaiyz,vlag;a,b=mult(E3.get())
    draaixy=draaixy+draaihoek
    C.delete(ALL)
    if vlag==0:teken3d(a,b)
    else:tekenkegel()
def dim3():
    global draaixy,draaiyz,vlag;a,b=mult(E3.get());draaiyz=15
    C.delete(ALL)
    if vlag==0:teken3d(a,b)
    else:tekenkegel()
def boven():
    global draaixy,draaiyz,vlag;a,b=mult(E3.get());draaiyz=90
    C.delete(ALL)
    if vlag==0:teken3d(a,b)
    else:tekenkegel()
def ap(li,x,y,z):X,Y=omzet(x,y,z);li.append(X);li.append(Y)
def df(x):dx=1E-6;return(f(x+dx)-f(x-dx))/dx/2
def z_op(x):return 2*pi*abs(f(x))*sqrt(1+df(x)**2)
def but(xp,yp,w,t,co):Button(form,text=t,command=co).
    place(x=xp*sc,y=yp*sc,width=w*sc)
def ent(t,kl,xp,yp,w):lab(t,xp,yp);en=Entry(form,width=w,bg=kl);
    en.place(x=sc*(xp+3.2),y=sc*yp);return en
def lab(t,xp,yp):Label(form,text=t).place(x=sc*xp,y=sc*yp)
def bereken():
    global roos,a,b;a,b=mult(E3.get())
    if b<a:wissel=a;a=b;b=wissel;wis(E3);E3.insert(0,str(a)+','+str(b))
    n=100;h=(b-a)/n;x=a;som=0
    for j in range(1,n):
        x=x+h
        if j%2==0:t=2
        else:t=4
        som=som+t*z_op(x)
    som=som+z_op(a)+z_op(b)
    integ=som*h/3;wis(E4);intst=format(integ,'.5f');wis(E4);
    E4.insert(0,intst)
    C.delete(ALL);roos=20;teken3d(a,b)
def info():messagebox.showinfo(tit+': info',infstr)
def wis(E):E.delete(0,len(E.get()))
def nieuw():
    global xt,yt,Xl,Yl,costr,draaixy,draaiyz
    xt=[];yt=[];Xl=[];Yl=[];costr=E2.get()
    for en in entr:wis(en)
    C.delete(ALL)
    draaixy,draaiyz=15,15
#hoofdprogramma
yp=40;sc=15;tit='Integralen: formule zijdelingse oppervlakte';lg='lightgrey';wh='white';bl='black'
form=Tk();roos=40;vlag=0;draaihoek=7;draaixy,draaiyz=15,15
xp=yp;form.geometry(str(xp*sc+2)+'x'+str((yp+5)*sc));form.title(tit);afm=(yp-2)*sc
C=Canvas(form,bg='white',height=afm,width=afm);C.place(x=sc,y=0)
E1=ent('Functie:',wh,5,39,20);E2=ent('XYZma:',wh,0,39,3);E3=ent('Interval:',wh,16,39,6)
```

```
E4=ent('∫ₐᵇ2π..:',lg,31,39,12);E5=ent('#kegels',wh,22,39,4);
E6=ent('∑2π..:',lg,31,42,12)
costr='4';E1.insert(0,'sin(x)');E2.insert(0,costr);ab='-1,2';
E3.insert(0,ab);E5.insert(0,'3')
entr=[E1,E2,E3,E4,E5,E6]
but(1,42,3,'Info',info);but(4,42,3,'Nieuw',nieuw);but(7,42,3,'3dim',dim3);but(10,42,3,'Boven',boven)
but(13,42,3,'Draai',draai);but(16,42,7,"∫ₐᵇ 2π|y|√(1+y'²)dx",bereken);
but(23,42,7,'∑2π|y|√(Δx²+Δy²)',tekenkegel)
infstr='Demonstratie van de formule voor de zijdelingse oppervlakte\n'
infstr=infstr+'van het omwentelingsoppervlak dat we bekomen door y=f(x)\n'
infstr=infstr+'te wentelen rond de x-as: de lim(n→∞) van de som van\n'
infstr=infstr+'de zijdelingse oppervlaktes van afgeknotte kegels\n'
infstr=infstr+"lim(n→∞) ∑2π|y|√(Δx²+Δy²)= ∫ₐᵇ2π|y|√(1+y'²)dx\n"
infstr=infstr+'|f(x)|=gemiddelde van de stralen in grond-en bovenvlak'
form.mainloop()
```



117. Delen van een veelterm A(x) van de n-de graad door x^2+px+q euclidische_deling

Stel dat we een veelterm A(x) delen door x^2+px+q.
De rest is dan van de eerste graad, dus R(x)=rx+s
Dan geldt: deeltal is deler * quotiënt+rest: $A(x)=(x^2+px+q).B(x)+R(x)$
Als $A(x) = a_0x^n + a_1x^{n-1} + a_2x^{n-2} + a_3x^{n-3} + a_4x^{n-4} + ...a_{n-1}x^1 + a_nx^0$ en
$B(x)= b_0x^{n-2} + b_1x^{n-3} + b_2x^{n-4} + ...b_{n-3}x^1 + b_{n-2}x^0$ geldt:
$a_0x^n + a_1x^{n-1} + a_2x^{n-2} + a_3x^{n-3} + a_4x^{n-4} + ...a_{n-1}x^1 + a_nx^0 =$
$(x^2+px+q).(b_0x^{n-2} + b_1x^{n-3} + b_2x^{n-4} + ...b_{n-3}x^1 + b_{n-2}x^0) + rx^1 + sx^0$
We werken het rechterlid uit en stellen de coëfficiënten van de overeenkomstige machten gelijk.
$a_0 = b_0$
$a_1 = b_1 + p.b_0$
$a_2 = b_2 + p.b_1 + q.b_0$
$a_3 = b_3 + p.b_2 + q.b_1$
$a_4 = b_4 + p.b_3 + q.b_2$
$a_5 = b_5 + p.b_4 + q.b_3$
…….....
$a_{n-2} = b_{n-2} + p.b_{n-3} + q.b_{n-4}$
$a_{n-1} = 0 + p.b_{n-2} + q.b_{n-3} + r$
$a_n = 0 + 0 + q.b_{n-2} + s$
Uit de eerste n-1 vergelijkingen kunnen we de coëfficiënten b_i van het quotiënt berekenen.
$b_0 = a_0$
$b_1 = a_1 - p.b_0$
$b_2 = a_2 - p.b_1 - q.b_0$
$b_3 = a_3 - p.b_2 - q.b_1$
$b_4 = a_4 - p.b_3 - q.b_2$
$b_5 = a_5 - p.b_4 - q.b_3$
…….....
$b_{n-2} = a_{n-2} - p.b_{n-3} - q.b_{n-4}$
$r = a_{n-1} - p.b_{n-2} - q.b_{n-3}$
$s = a_n - q.b_{n-2}$

Dit geeft een methode om het quotiënt en de rest te bepalen. Het volgende pythonprogramma heeft een functie deling(a,p,q) die uitgaande van de list van de coëfficiënten a_i van het deeltal, de coëfficiënten b_i van het quotiënt berekent en opslaat in een list b.

Programma:

```
# euclidische deling
def mult(s):
        l=s.split(',')
        co=[]
        for i in l:co.append(float(i))
        return co
def deling(a,p,q):
        n=len(a)-1
        b=[a[0],a[1]-p*a[0]]
        i=1
        while i<n:
                i=i+1
                b.append(a[i]-p*b[i-1]-q*b[i-2])
```

```
        return(b)
a=mult(input('Geef de coëfficiënten:'))
m=len(a)-1
p,q=mult(input('Geef p en q:'))
b=deling(a,p,q)
r=b[m-1];s=a[m]-q*b[m-2]
print('coëff.quotiënt=',b[slice(len(b)-2)],'\ncoëff.rest=',r,s)
>>>
Geef de coëfficiënten:2,25,-18,-37,44,25
Geef p en q:3,-5
coëff.quotiënt= [2.0, 19.0, -65.0, 253.0]
coëff.rest= -1040.0 1290.0
>>>
```

118. Methode van Bairstow bairstow

Bij veeltermen is het niet steeds mogelijk om een nulpunt = reëel getal a te benaderen met de methode van Newton. Dan is het dus ook niet mogelijk om de veelterm te ontbinden als (x-a).B(x)
Maar er bestaan wel methodes om een deler te benaderen van de vorm x^2+px+q. Eén van deze methodes is de methode van Bairstow.
We kennen p en q niet, dus nemen we 'willekeurige' startwaarden voor p en q. En we berekenen het quotiënt B(x) en de rest rx+s zoals in het vorige puntje. Ook voor dit quotiënt zal de deling waarschijnlijk niet opgaan. Zodat we met dezelfde methode B(x) kunnen delen door x^2+px+q met op zijn beurt quotiënt C(x) en rest tx+u. We gieten dit in 1 vergelijking:
$A(x)=B(x)(x^2+px+q)+rx+s=[C(x)(x^2+px+q)+tx+u)](x^2+px+q)+rx+s$
Stel nu dat $x^2+(p+\Delta p)x+q+\Delta q$ een exacte deler is van A(x).
Dan is $[C(x)(x^2+(p+\Delta p)x+q+\Delta q)+tx+u](x^2+(p+\Delta p)x+q+\Delta q)+rx+s=0$
Is nu w een nulpunt van $x^2+(p+\Delta p)x+q+\Delta q$, dan is $w^2+(p+\Delta p)w+q+\Delta q=0$
Hieruit volgt $w^2=-pw-q-w\Delta p-\Delta q$ $w^2+pw+q=-w\Delta p-\Delta q$
Maar w is dan ook een nulpunt van A(x) zodat
$[C(w)(w^2+pw+q)+tw+u)](w^2+pw+q)+rw+s=0$
Invullen geeft: $[C(w)(-w\Delta p-\Delta q)+tw+u)](-w\Delta p-\Delta q)+rw+s=0$
Omdat Δp en Δq klein zijn, zijn 2de graadstermen van Δp en Δq zeer klein.
Als we deze weglaten, krijgen we niet de exacte waarde maar wel een benadering.
Dan blijft $(tw+u)](-w\Delta p-\Delta q)+rw+s \cong 0$
Haakjes uitwerken en w^2 vervangen door $-pw-q-w\Delta p-\Delta q$ geeft:
$-t(-pw-q-w\Delta p-\Delta q)\Delta p-tw\Delta q-uw\Delta p-u\Delta q+rw+s \cong 0$
Nogmaals laten we de 2de graadstermen van Δp en Δq weg:
$t(pw+q)\Delta p-tw\Delta q-uw\Delta p-u\Delta q+rw+s \cong 0$
$w(tp\Delta p-t\Delta q-u\Delta p+r)+tq\Delta p-u\Delta q+s \cong 0$
Een 2de graadsveelterm x^2+px+q heeft 2 oplossingen w_1 en w_2
Allebei voldoen ze aan de vorige lineaire vergelijking
Bijgevolg moet $tp\Delta p-t\Delta q-u\Delta p+r \cong 0$ en $tq\Delta p-u\Delta q+s \cong 0$

Of $(tp-u)\Delta p-t\Delta q \cong -r$
$tq\Delta p-u\Delta q \cong -s$

Als we dit stelsel oplossen naar Δp en Δq, en omdat het stelsel slechts benaderend juist is, zal $p+\Delta p$ en $q+\Delta q$ ook een benadering zijn van de exacte waarde p en q waarvoor x^2+px+q een deler is van A(x).
We vinden voor de oplossing:

$$\Delta p = \frac{ru - st}{t^2 q - u(tp - u)} \quad \text{en} \quad \Delta q = \frac{-(tp - u)s + rtq}{t^2 q - u(tp - u)}$$

r en s maar ook t en u vinden we als coëfficiënten van de rest van de euclidische deling van A(x), respectievelijk B(x) door x^2+px+q.
We gaan uit van een 'willekeurige' p en q en berekenen met de vorige formules Δp en Δq.
$x^2+(p+\Delta p)x+(q+\Delta q)$ is dan een eerste benadering van de gezochte deler.
Hiermee maken we terug de euclidische deling om een nieuwe r,s,t,u en Δp ,Δq te berekenen.
De opeenvolgende benaderingen komen steeds dichter bij de 'echte' p en q van de deler x^2+px+q. De methode convergeert als Δp en Δq zeer klein worden, dus als abs(Δp)+abs(Δq) zeer klein geworden is.

Programma:

```
# bairstow
def mult(s):
        l=s.split(',')
        co=[]
        for i in l:co.append(float(i))
        return co
def deling(a,p,q):
        n=len(a)-1
        b=[a[0],a[1]-p*a[0]]
        i=1
        while i<n:
                i=i+1
                b.append(a[i]-p*b[i-1]-q*b[i-2])
        return(b)
a=mult(input('Geef de coëfficiënten:'))
m=len(a)-1
p,q=mult(input('Geef p en q:'))
dif=1;dp=1;dq=1;tel=0
while dif>0.00001 and tel<20:
        b=deling(a,p,q);c=deling(b,p,q)
        r=b[m-1];s=a[m]-q*b[m-2]
        t=c[m-3];u=b[m-2]-q*b[m-4]
        no=t*t*q-u*t*p+u*u
        dp=(r*u-s*t)/no;dq=(-t*p*s+u*s+r*t*q)/no
        p=p+dp;q=q+dq
        dif=abs(dp)+abs(dq)
        tel=tel+1
if tel<20:print('x²+px+q   p,q=',p,q)
else:print('Geen convergentie')
>>>
Geef de coëfficiënten:2,20,48,-20,-51
Geef p en q:4,4
x^2+px+q   p,q= 5.866662244227896 4.926837437699567
```

Opmerking:
Het is niet onmogelijk dat de methode niet convergeert. In dat geval zal dif nooit kleiner worden dan 0.00001. Er komt dan geen antwoord: het programma blijft eindeloos in de while-loop hangen. Dit kan je oplossen met een teller: initialiseer hem vóór de while-loop :tel=0 en verhoog hem in de loop telkens met 1. Als tel=20 wordt, wordt de loop beëindigd zonder convergentie.

119. Ontbinden van een veelterm van een graad > 3 ontbinden_veelterm

Voor veeltermen van de derde graad kunnen we met de methode van Newton een nulpunt x_1 benaderen. We delen de veelterm door $x-x_1$ met Horner en vinden dan een quotiënt van de 2de graad die we verder kunnen ontbinden met discriminant en wortelformules.
Bairstow geeft ons de mogelijkheid om bij een veelterm A(x) van hogere graad een kwadratische vorm x^2+px+q af te zonderen, die we ontbinden met de wortelformules. Is het quotiënt B(x) nog van graad >3, dan kunnen we hierop terug Bairstow toepassen, enz..
Is de graad =3, dan kunnen we Newton toepassen om een deler $x-x_1$ af te zonderen.
Uiteindelijk blijft er steeds een quotiënt van de 2de graad over die we ook weer met de wortelformules kunnen ontbinden. Door de opeenvolgende benaderde delingen, mogen we niet verwachten dat de uiteindelijke ontbinding zeer nauwkeurig zal zijn.
In het volgende programma zal je programmaonderdelen uit vorige programma's herkennen.

Programma:

```
# Ontbinden in factoren van een veelterm
from math import sqrt;from copy import deepcopy
def mult(s):
   l=s.split(',');co=[]
   for i in l:co.append(float(i))
   return co
def v(x):
   if x>=0:sgn='+'
   else:sgn=''
   return sgn+format(x,'.3f')
def vc(x):
   return "-("+format(x,'.3f')+")"
def druk(p,q):
   global ans
   d=p*p-4*q
   if d>0:root=sqrt(d);x1,x2=(-p-root)/2,(-p+root)/2
   elif d==0:x1=x2=-p/2
   else:root=sqrt(-d);x1,x2=(-p-root*(1j))/2,(-p+root*(1j))/2
   if d>0:ans=ans+'(x'+v(-x2)+')(x'+v(-x1)+')'
   elif d==0:ans=ans+'(x'+v(-x2)+')²'
   else:ans=ans+'[x'+vc(x2)+'][x'+vc(x1)+']'
def deling(a,p,q):
        n=len(a)-1;b=[a[0],a[1]-p*a[0]];i=1
        while i<n:i=i+1;b.append(a[i]-p*b[i-1]-q*b[i-2])
        return(b)
def bairstow(c):
   global p,q,it,m,no
   a=c;m=len(a)-1
   p=2*a[1]/a[0]/m;q=2*a[2]/a[0]/m/(m-1)
   dif=1;it=0;dp=1;dq=1;no=1
   while it<50 and dif>0.00001 and no!=0:
      it=it+1;dif=abs(dp)+abs(dq)
      b=deling(a,p,q);c=deling(b,p,q)
      r=b[m-1];s=a[m]-q*b[m-2]
```

```
        t=c[m-3];u=b[m-2]-q*c[m-4];no=t*t*q-u*t*p+u*u
        if no!=0:
            dp=(r*u-s*t)/no;dq=(-t*p*s+u*s+r*t*q)/no
            p=p+dp;q=q+dq
    return p,q,it
def f(a,x):
    waarde=0;n=len(a)
    for i in range(0,n):waarde=waarde+a[i]*x**(n-i-1)
    return waarde
def newton(a):
    global it,start
    x=start;dx=0.0001;ox=1;it=0 #Newton
    while abs(x-ox)>dx and it<50:
        ox=x;it=it+1
        x=x-dx*f(a,x)/(f(a,x+dx)-f(a,x))
    return x
# druk de veelterm
def drukvlt(a):
    m=len(a);vlt=''
    for j in range(m,0,-1):vlt=vlt+v(a[m-j])+'x'+supdig[j-1]
    return vlt
# hoofdprogramma
a=[1]
while a[0]!=0:
    supdig=['\u2070','\u00B9','\u00B2','\u00B3','\u2074','\u2075','\u2076','\u2077','\u2078','\u2079']
    a=mult(input('Geef de coëfficiënten van de veelterm, 0 om te stoppen:'))
    m=len(a)-1;ans='';it=0;start=50;no=1
    print(drukvlt(a),'=\n',v(a[0]),end='')
    # afsplitsen van factoren x²+px+q
    while m>3 and it<50 and no!=0:
        bairstow(a)
        if it<50 and no!=0:
            na=[];na.append(a[0]);na.append(a[1]-na[0]*p)
            for i in range(2,m+1):na.append(a[i]-na[i-1]*p-na[i-2]*q)
            na.pop();na.pop();a=na;druk(p,q);m=len(a)-1
    it=0
    # afsplitsen van factoren x-x1
    while m>2 and it<50:
        start=20;x=newton(a)
        if it==50:start=-20;x=newton(a)
        if it<50:
            na=[a[0]]
            for i in range(1,m):na.append(a[i]+na[i-1]*x)
            a=na;m=len(a)-1
            ans=ans+'(x'+v(-x)+')'
    if m==2:p=a[1]/a[0];q=a[2]/a[0];druk(p,q);a=[1]
    if m==1:x=-a[1]/a[0];ans=ans+'(x'+v(-x)+')';a=[1]
    quot=drukvlt(a)
    if quot!='+1.000x\u2070':ans=ans+'('+quot+')'
    print(ans)
>>>
```

Geef de coëfficiënten van de veelterm, 0 om te stoppen:1,2,3,-7,15
$+1.000x^4+2.000x^3+3.000x^2-7.000x^1+15.000x^0$ =
$+1.000[x-(0.897+1.045j)][x-(0.897-1.045j)][x-(-1.897+2.077j)][x-(-1.897-2.077j)]$
Geef de coëfficiënten van de veelterm, 0 om te stoppen:

120. Eigenwaarden van een reguliere matrix. eigenwaarden

Een reguliere matrix is een vierkante matrix waarvan de determinant ≠ 0
Reguliere matrices hebben steeds eigenwaarden en eigenvectoren:
λ is een eigenwaarde en x is een eigenvector van een matrix A als geldt: A.x=λ.x
2x2-voorbeeld:

$$\begin{pmatrix}6 & 3\\ -1 & 2\end{pmatrix}\cdot\begin{pmatrix}x_1\\ x_2\end{pmatrix}=\lambda\cdot\begin{pmatrix}x_1\\ x_2\end{pmatrix}=\lambda.\begin{pmatrix}1 & 0\\ 0 & 1\end{pmatrix}\cdot\begin{pmatrix}x_1\\ x_2\end{pmatrix} \qquad \begin{pmatrix}6-\lambda & 3\\ -1 & 2-\lambda\end{pmatrix}\cdot\begin{pmatrix}x_1\\ x_2\end{pmatrix}=\begin{pmatrix}0\\ 0\end{pmatrix}$$

Deze vergelijking heeft enkel een van 0 verschillende oplossing als

$$\det\begin{pmatrix}6-\lambda & 3\\ -1 & 2-\lambda\end{pmatrix}=0 \quad \text{of} \quad (6-\lambda).(2-\lambda)+1.3=0 \quad \text{of} \quad 12-8\lambda+\lambda^2+3=0 \quad \lambda^2-8\lambda+15=0$$

Deze vergelijking wordt de karakteristieke vergelijking $f_A(\lambda)$ genoemd.
Er zijn 2 oplossingen (eigenwaarden) : $\lambda_1=3$ en $\lambda_2=5$
Vullen we 3 in, dan is

$$\begin{pmatrix}6-3 & 3\\ -1 & 2-3\end{pmatrix}\cdot\begin{pmatrix}x_1\\ x_2\end{pmatrix}=\begin{pmatrix}0\\ 0\end{pmatrix} \quad \text{of} \quad \begin{cases}3x_1+3x_2=0\\ -x_1-x_2=0\end{cases}$$

te herleiden tot
$x_1+x_2=0$ zodat $\begin{pmatrix}1\\ -1\end{pmatrix}$ een eigenvector is. Bepaal zelf de eigenvector die hoort bij $\lambda_2=5$.

Opmerkingen:
De som van de termen op de hoofddiagonaal noemen we de spoor S_1
$S_1=a_{11}+a_{22}=6+2=8$ maar ook $\lambda_1+\lambda_2=8$: blijkbaar is $\lambda_1+\lambda_2=S_1$
det A=6.2+1.3=15 maar ook $\lambda_1.\lambda_2=15$: dus $\lambda_1.\lambda_2=$ det A

We bewijzen deze 2 eigenschappen voor een 3x3-matrix

$$A=\begin{pmatrix}a_{11} & a_{12} & a_{13}\\ a_{21} & a_{22} & a_{23}\\ a_{31} & a_{32} & a_{33}\end{pmatrix}$$

De karakteristieke vergelijking vinden we door det (A-λ.I)=0 te stellen.

$$\det\begin{pmatrix}a_{11}-\lambda & a_{12} & a_{13}\\ a_{21} & a_{22}-\lambda & a_{23}\\ a_{31} & a_{32} & a_{33}-\lambda\end{pmatrix}=0$$

1. Ontwikkelen volgens de eerste rij geeft:
$f_A(\lambda)=(a_{11}-\lambda).[(a_{22}-\lambda).(a_{33}-\lambda)-a_{32}.a_{23}]-a_{12}.[\ a_{21}.(a_{33}-\lambda)-a_{31}.a_{23}]+a_{13}.[\ a_{21}.a_{32}-a_{31}.(a_{22}-\lambda)]=0$
We zonderen enkel de termen met λ^2 af: $(a_{11}+a_{22}+a_{33})\ \lambda^2$ of $S_1.\lambda^2$ (1)
Als we anderzijds $f_A(\lambda)$ ontbinden in factoren: $f_A(\lambda)=-(\lambda-\lambda_1)(\lambda-\lambda_2)(\lambda-\lambda_3)$
Ook hier zonderen we de termen met λ^2 af: $(\lambda_1+\lambda_2+\lambda_3).\ \lambda^2$ (2)
Vergelijk (1) met (2) : $\boldsymbol{\lambda_1+\lambda_2+\lambda_3=S_1}$
2. We hebben $f_A(\lambda)=\det(A-\lambda.I) = -(\lambda-\lambda_1)(\lambda-\lambda_2)(\lambda-\lambda_3)$: stel hierin λ=0, dan vinden we onmiddellijk : $\boldsymbol{\lambda_1.\lambda_2.\lambda_3 = \det A}$

Hoe kunnen we de karakteristieke vergelijking bepalen van een reguliere nxn- matrix ?

We maken terug de berekeningen voor een 3x3-matrix:
$f_A(\lambda)=(-1)^3.(c_0\lambda^3+c_1\lambda^2+c_2\lambda+c_3)= (-1)^3.(\lambda- \lambda_1).(\lambda- \lambda_2).(\lambda-\lambda_3)$ $(c_0=1)$
We laten het -teken weg en leiden beide vormen af naar λ:
$f'_A(\lambda)=3c_0\lambda^2+2\ c_1\lambda+c_2$ (1)

$$f'_A(\lambda)=\frac{f_A(\lambda)}{\lambda-\lambda_1}+\frac{f_A(\lambda)}{\lambda-\lambda_2}+\frac{f_A(\lambda)}{\lambda-\lambda_3}$$

We bepalen het eerste quotiënt met Horner:

$$\lambda_1\left|\begin{array}{cccc} c_0 & c_1 & c_2 & c_3 \\ & c_0\lambda_1 & c_1\lambda_1+c_0\lambda_1^2 & c_2\lambda_1+c_1\lambda_1^2+c_0\lambda_1^3 \\ c_0 & c_1+c_0\lambda_1 & c_2+c_1\lambda_1+c_0\lambda_1^2 & c_3+c_2\lambda_1+c_1\lambda_1^2+c_0\lambda_1^3 \end{array}\right|$$

Het quotiënt is dus $c_0\lambda^2+(c_1+c_0\lambda_1)\lambda+(c_2+c_1\lambda_1+c_0\lambda_1^2)$
De rest is $f_A(\lambda_1) = 0$
Op dezelfde manier vinden we de volgende quotiënten:
$c_0\lambda^2+(c_1+c_0\lambda_2)\lambda+(c_2+c_1\lambda_2+c_0\lambda_2^2)$ en $c_0\lambda^2+(c_1+c_0\lambda_3)\lambda+(c_2+c_1\lambda_3+c_0\lambda_3^2)$
Sommeren van de 3 quotiënten geeft:
$f'_A(\lambda)= 3c_0\lambda^2+[3c_1+c_0(\lambda_1+\lambda_2+\lambda_3)]\ \lambda+[3c_2+c_1(\lambda_1+\lambda_2+\lambda_3)+c_0(\lambda_1^2+\lambda_2^2+\lambda_3^2)]$ of
$f'_A(\lambda)= 3c_0\lambda^2+[3c_1+c_0.S_1]+[3c_2+c_1.S_1 +c_0.S_2]$ (2)
Stel de coëfficiënten van (1) en (2) gelijk:

$$\begin{cases} 2c_1 = 3c_1 + c_0.S_1 \\ c_2 = 3c_2 + c_1.S_1 + c_0.S_2 \end{cases}\left|\begin{array}{l} c_1 = -c_0.S_1 = -S_1 \\ c_2 = \dfrac{-1}{2}(c_1S_1 + c_0.S_2) \end{array}\right.$$

Opmerking: Als $A.x=\lambda.x$ is $A^2.x=A(A.x)=A(\lambda.x)=\lambda.A.x=\lambda^2.x$
Dwz. als λ_1 , λ_2 , λ_3 .... eigenwaarden zijn van A, dan zijn λ_1^2 , λ_2^2 , λ_3^2 eigenwaarden van A^2.
De eigenvectoren daarentegen van A^2 zijn dezelfde als deze van A.
We hebben $\lambda_1^2+\lambda_2^2+\lambda_3^2 = S_2$ gesteld: dit is inderdaad het spoor van de matrix A^2, dwz. de som van de hoofddiagonale elementen van A^2
We kunnen dit veralgemenen voor een nxn-matrix:
de coëfficiënten van de karakteristieke vergelijking $f_A(\lambda)$ worden gegeven door:

$c_0=1$ $\quad c_n = \dfrac{-1}{n}\sum_{i=1}^{n} c_{n-i}.S_i$: Hiervoor moeten we dus, behalve A ook A^2, A^3, .. A^n berekenen

en hiervan de sporen bepalen met $S_n = \sum_{i=1}^{n} a_{ii}$

Het volgende programma bepaalt van een willekeurige reguliere matrix eerst de karakteristieke vergelijking en berekent nadien met de methode van Bairstow de eigenwaarden.

Programma:

```
#Bepalen karakteristieke vergelijking en eigenwaarden van vierkante matix
from math import sqrt;from copy import deepcopy
def v(x):
   if x>0:sgn='+'
   else:sgn=''
   return sgn+format(x,'.3f')
def vc(x):return "-("+format(x,'.3f')+")"
def prin(p,q):
   global ans;d=p*p-4*q
   if d>0:root=sqrt(d);x1,x2=(-p-root)/2,(-p+root)/2
   elif d==0:x1=x2=-p/2
```

```
    else:root=sqrt(-d);x1,x2=(-p-root*(1j))/2,(-p+root*(1j))/2
    if d>0:ans=ans+'('+lamb+v(-x2)+')('+lamb+v(-x1)+')'
    elif d==0:ans=ans+'('+lamb+v(-x2)+')²'
    else:ans=ans+'['+lamb+vc(x2)+']['+lamb+vc(x1)+']'
    if d>=0:eig.append(x1);eig.append(x2)
def prinpln(a):
    m=len(a);pln=''
    for j in range(m,0,-1):pln=pln+v(a[m-j])+'x'+supdig[j-1]
    return pln
def division(a,p,q):
        n=len(a)-1;b=[a[0],a[1]-p*a[0]]
        i=1
        while i<n:
                i=i+1
                b.append(a[i]-p*b[i-1]-q*b[i-2])
        return(b)
#definities matrix
def id(n):
        I=[]
        for i in range(0,n):
                lin=[]
                for j in range(0,n):
                        if i==j:lin.append(1)
                        else:lin.append(0)
                I.append(lin)
        return(I)
def gt(x):
        if x<0:uitv=str(x)
        else: uitv='+'+str(x)
        return uitv
def prod(b,c):
        n=len(b);p=[]
        for i in range(0,n):
                lin=[]
                for j in range(0,n):
                        som=0
                        for k in range(0,n):
                                som=som+b[i][k]*c[k][j]
                        lin.append(som)
                p.append(lin)
        return(p)
def mult(s):
    l=s.split(',')
    co=[]
    for i in l:co.append(float(i))
    return co
def det(mat):
    le=len(mat)
    if le==1:d=mat[0][0]
    else:
       d=0
```

```
        for i in range(0,le):
            s=[]
            for k in range(0,le):
                if i!=k:s.append(mat[k][1:])
            d=d+mat[i][0]*(-1)**i*det(s)
    return d
def f(a,x):
    value=0;n=len(a)
    for i in range(0,n):value=value+a[i]*x**(n-i-1)
    return value
def karakverg(mat):
    global ans,c,lamb,chreq;n=len(mat)
    chreq='Ingevoerde matrix:\n'+str(mat);ans=''
    eig=[];p=id(n);tr=[];c=[]
    for i in range(0,n+1):
        p=prod(mat,p);sdia=0
        for j in range(0,n):sdia=sdia+p[j][j]
        tr.append(sdia)
    c.append(1);c.append(-tr[0])
    for k in range(2,n+1):
        som =tr[k-1]
        for j in range(1,k):
            som=som+c[j]*tr[k-j-1]
        c.append(-som/k)
    lamb=chr(955)
    chreq=chreq+'\nKarakteristieke vergelijking:'
    for i in range(0,n+1):
        chreq=chreq+gt(c[i])+lamb+supdig[n-i]
    chreq=chreq+'=0\nOntbinding:'
    ans=ans+chreq
    return ans,c
def newton(a):
    global it,start
    x=start;dx=0.0001;ox=1;it=0
    while abs(x-ox)>dx and it<50:
        ox=x;it=it+1
        x=x-dx*f(a,x)/(f(a,x+dx)-f(a,x))
    return x
def bairstow(c):
    global ans,eig,start,no;a=c;m=len(a)-1;eig=[];it=0;no=1
    while m>3 and it<50 and no!=0:
        p=2*a[1]/a[0]/m;q=2*a[2]/a[0]/m/(m-1)
        dif=1;it=0;dp=1;dq=1;no=1
        while it<50 and dif>0.00001:
            it=it+1;dif=abs(dp)+abs(dq)
            b=division(a,p,q);c=division(b,p,q)
            r=b[m-1];s=a[m]-q*b[m-2]
            t=c[m-3];u=b[m-2]-q*c[m-4];no=t*t*q-u*t*p+u*u
            if no!=0:
                dp=(r*u-s*t)/no;dq=(-t*p*s+u*s+r*t*q)/no
                p=p+dp;q=q+dq
```

```
        if it<50 and no!=0:
            prin(p,q) # ontbinding x²+px+q
            na=[];na.append(a[0]);na.append(a[1]-na[0]*p)
            for i in range(2,m+1):na.append(a[i]-na[i-1]*p-na[i-2]*q)
            na.pop();na.pop();a=na;m=len(a)-1
    it=0
    # afsplitsen factoren x-x1
    while m>2 and it<50:
        start=20;x=newton(a)
        if it==50:start=-20;x=newton(a)
        if it<50:
            eig.append(x)
            na=[a[0]]
            for i in range(1,m):na.append(a[i]+na[i-1]*x)
            a=na;m=len(a)-1
            ans=ans+'('+lamb+v(-x)+')'
    if m==2:p=a[1]/a[0];q=a[2]/a[0];prin(p,q);a=[1]
    if m==1:x=-a[1]/a[0];eig.append(x);ans=ans+'('+lamb+v(-x)+')';a=[1]
    quot=prinpln(a)
    if quot!='+1.000x\u2070':ans=ans+'('+quot+')'
    return ans,eig
# main
supdig=['\u2070','\u00B9','\u00B2','\u00B3','\u2074','\u2075','\u2076','\u2077','\u2078','\u2079']
if __name__ == '__main__':
    print('Berekenen van de eigenwaarden van een reguliere matrix')
    print('Geef de coëfficiënten van de nxn matrix gescheiden door kommas.')
    print('Nadat je de laatste rij hebt ingevoerd, tik <Enter>')
    determ=0
    while determ==0:
        mat=[];i=0;inp=' '
        while inp!='':
            print('Rij',i+1,'van coëff:',end=' ')
            inp=input(':')
            if inp!='':
                rowcoef=mult(inp);mat.append(rowcoef)
            m=len(rowcoef);n=len(mat);i=i+1;determ=det(mat)
        if determ==0:print('Determinant=0, opnieuw a.u.b.')
        else:
            c=[];karakverg(mat);bairstow(c)
            print(ans);print('Reële eigenwaarden =',eig)
>>>
Berekenen van de eigenwaarden van een reguliere matrix
Geef de coëfficiënten van de nxn matrix gescheiden door kommas.
Nadat je de laatste rij hebt ingevoerd, tik <Enter>
Rij 1 van coëff: :1,4,8
Rij 2 van coëff: :2,-3,5
Rij 3 van coëff: :7,6,-1
Rij 4 van coëff: :
Ingevoerde matrix:
[[1.0, 4.0, 8.0], [2.0, -3.0, 5.0], [7.0, 6.0, -1.0]]
Karakteristieke vergelijking:+1λ^3+3.0λ^2-95.0λ^1-385.0λ^0=0
```

Ontbinding:(λ-10.130)(λ+4.308)(λ+8.822)
Reële eigenwaarden = [10.129963830664233, -8.821733750620938,-4.308230080043295]
>>>

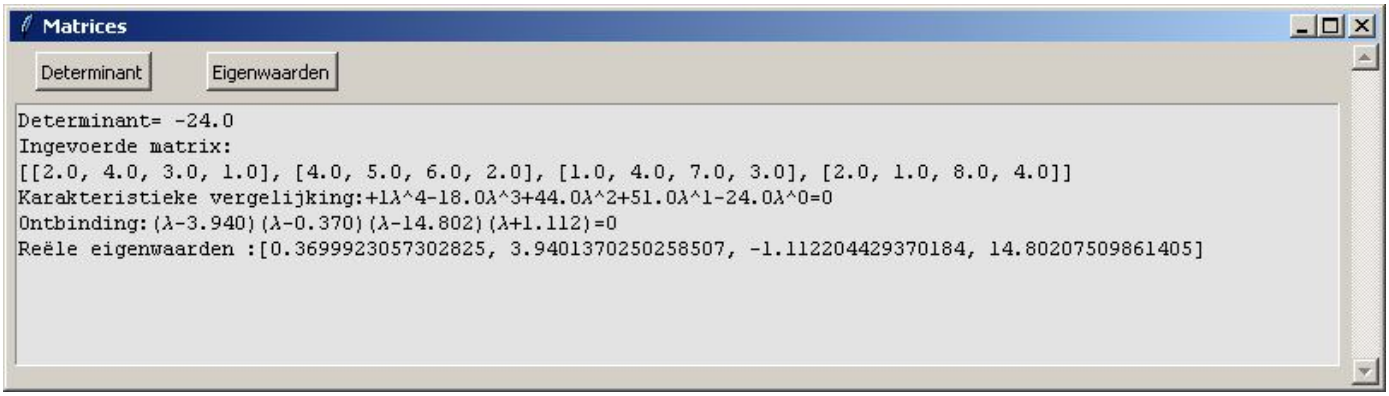
121. Eigenwaarden berekenen in een tkinter -jasje eigenwaarden_tkinter

Aan het begin van het hoofdprogramma zie je de if -instructie:
if __name__ == '__main__':
Al wat er volgt (ons programma), zal dus enkel en alleen uitgevoerd worden, als we effectief 'dit' programma hebben ingeladen. Dit heeft het voordeel dat we dit programma ook als module kunnen importeren in een ander programma om gebruik te kunnen maken van de vrij omvangrijke functies. In het volgende tkinter-programma doen we zo iets:
We gebruiken 3 vensters, het eerste 'form' om de dimensie n van de matrix in te voeren. Een tweede 'fw' dat afhankelijk van de waarde van n, de gepaste afmetingen krijgt om in n x n entries de matrix zelf in te voeren. (als gebruiker, tik gewoon 'getal' TAB 'getal' TAB... om van een entry naar de volgende te gaan). Een 3de venster 'fb' om de determinant en de eigenwaarden te berekenen.

Programma:

```
# eigenwaarden berekenen in tkinter-omgeving
from math import *;from tkinter import *;from eigenwaarden import *
def matrix():
    global mat_ent,fw,n;n=int(E2.get())
    afmx=45*n+5;afmy=22*n+40
    afmet=str(afmx)+'x'+str(afmy)
    fw=Toplevel(form);fw.geometry(afmet);fw.title('Invoeren')
    Button(fw,text='Ok',command=vulin).place(x=15,y=afmy-35)
    Button(fw,text='Nieuw',command=nieuw2).place(x=50,y=afmy-35)
    mat_ent=[]
    for i in range(0,n):
        lijn_ent=[]
        for j in range(0,n):
            ent=Entry(fw,width=5);ent.place(x=45*j+10,y=20*i+5)
            lijn_ent.append(ent)
        mat_ent.append(lijn_ent)
def determ():voegtoe('Determinant= '+str(det(mat)))
def eigenw():
    global mat;c=[]
    if det(mat)!=0:
        ant,c=karakverg(mat);ant,eig=bairstow(c);voegtoe(ant);voegtoe('Reële eigenwaarden :'+str(eig))
    else: voegtoe('De matrix is niet regulier, determinant= 0.0')
def voegtoe(lijn):tex.insert(INSERT,lijn+'\n')
def vulin():
    global mb,mat,tex;mat=[]
    for i in range(0,n):
        lijn=[]
        for j in range(0,n):
            lijn.append(float(mat_ent[i][j].get()))
        mat.append(lijn)
def nieuw2():
    for i in range(0,n):
        for j in range(0,n):l=len(mat_ent[i][j].get());mat_ent[i][j].delete(0,l)
def nieuw():nieuw2();fw.destroy()
```

```
form=Tk();form.geometry('130x70');form.title('Matrices')
Label(form,text='n=').place(x=2,y=10)
E2=Entry(form,width=3);E2.place(x=25,y=10)
Button(form,text='Matrix',command=matrix).place(x=70,y=5)
Button(form,text='Nieuw',command=nieuw).place(x=70,y=35)
fb=Toplevel(form);fb.geometry('920x210');
tex=Text(fb,bg='grey89',height=10,width=110,wrap=WORD);tex.place(x=5,y=35)
vsb=Scrollbar(fb,orient=VERTICAL,command=tex.yview,width=25)
tex['yscrollcommand']=vsb.set;vsb.pack(side=RIGHT,fill='y')
Button(fb,text='Determinant',command=determ).place(x=15,y=5)
Button(fb,text='Eigenwaarden',command=eigenw).place(x=115,y=5)
form.mainloop()
```



122. Oplossen van stelsels van 2 niet lineaire vergelijkingen met 2 onbekenden.

newton_niet_lin_stelsels

Zij het stelsel van niet lineaire vergelijkingen in x en y:
f(x,y)=0
g(x,y)=0
z=f(x,y) en z=g(x,y) zijn 2 oppervlakken in de ruimte Σ_0
Door z=0 te stellen in beide vergelijkingen, krijgen we de doorsneden van deze oppervlakken met het XY-vlak. M.a.w. 2 krommen in het XY-vlak, waarvan wij een snijpunt moeten zoeken.
Stel dat (x,y,0) een oplossing is van dit stelsel.
Dan is f(x,y)=0 en g(x,y)=0
Stel ook dat (xo,yo) een benadering is van (x,y)
x=xo+k
y=yo+l
We gebruiken de formule van Taylor, beperkt tot de lineaire termen:
f(xo+k,yo+l)~f(xo,yo)+k.f'x(xo,yo)+l.f'y(xo,yo)

Door al de volgende termen van de Taylorontwikkeling te laten vallen krijgen we slechts een benaderende gelijkheid. We passen deze formule toe op beide functies

0=f(x,y)=f(xo+k,yo+l)~f(xo,yo)+k.f'x(xo,yo)+l.f'y(xo,yo)
0=g(x,y)=g(xo+k,yo+l)~g(xo,yo)+k.g'x(xo,yo)+l.g'y(xo,yo)

Dit zijn eigenlijk de parametervergelijkingen van de raakvlakken aan de oppervlakken f(x,y) en g(x,y) in het punt van beide oppervlakken met xy- coördinaten (xo,yo). Het is niet verwonderlijk dat,dichtbij een oplossing ,deze vlakken bij benadering door het oplossingspunt (x,y,0) gaan.

We lossen dit stelsel op naar k en l:

k.f'x(xo,yo)+l.f'y(xo,yo) ~ - f(xo,yo)
k.g'x(xo,yo)+l.g'y(xo,yo) ~ - g(xo,yo)

Dit is een eenvoudig stelsel van 2 vergelijkingen met 2 onbekenden waaruit we k en l kunnen berekenen.
Om een goede startwaarde te vinden, bepalen we het koppel (xo,yo) voor x en y $\in$ (-10,9) waarvoor |f(xo,yo)|+|g(xo,yo)| minimaal is.
Uitgaande van een eerste benadering (xo,yo) vinden we achtereenvolgens (x1,y1)=(xo+k,yo+l) , (x2,y2)=(x1+k,y1+l) ...
Deze reeks convergeert als de opeenvolgende benaderingen steeds dichter bij elkaar komen te liggen. Het kan zijn dat er geen convergentie is. In de benaderingsloop wordt daarom een teller tel toegevoegd: als we na 20 benaderingen onvoldoende dicht bij een oplossing zitten, wordt er 'Geen oplossing' afgedrukt.

Programma

```
# oplossen van een niet lineair stelsel
from math import *
d=0.0001
def mult(s):
        l=s.split(',')
        co=[]
        for i in l:co.append(float(i))
        return co
def f(x,y):
        return eval(fs)
def g(x,y):
        return eval(gs)
def dfx(x,y):
        return( (f(x+d,y)-f(x,y))/d)
def dfy(x,y):
        return( (f(x,y+d)-f(x,y))/d)
def dgx(x,y):
        return( (g(x+d,y)-g(x,y))/d)
def dgy(x,y):
        return( (g(x,y+d)-g(x,y))/d)
def det(p,q,r,s):
        return(p*s-q*r)
print('oplossen van een niet lineair stelsel f(x,y)=0 , g(x,y)=0')
fs=input('f(x,y):')
gs=input('g(x,y):')
```

```
mi=100000
for x in range(-10,10):
      for y in range(-10,10):
            t=abs(f(x,y))+abs(g(x,y))
            if t<mi:
                  mi=t;xmi=x;ymi=y;fmi=f(x,y);gmi=g(x,y)
print ('Start:  x=%d y=%d f(x,y)=%d g(x,y)=%d' % (xmi,ymi,fmi,gmi ))
x=xmi;y=ymi
ox=x+1;oy=y+1;tel=0;D=1
while (abs(ox-x)>d or abs(oy-y)>d) and tel<20:
          ox=x;oy=y;tel=tel+1
          D=det(dfx(x,y),dfy(x,y),dgx(x,y),dgy(x,y))
          Dl=-det(f(x,y),dfy(x,y),g(x,y),dgy(x,y))
          Dk=-det(dfx(x,y),f(x,y),dgx(x,y),g(x,y))
          if D!=0:
                  l=Dl/D;k=Dk/D
          else:
                  break
          x=x+l;y=y+k
if D!=0 and tel<20:print('Oplossing=('+str(x)+','+str(y)+')')
else:print('Geen oplossing')
>>>
oplossen van een niet lineair stelsel f(x,y)=0 , g(x,y)=0
f(x,y):x**3+2*y*y-5
g(x,y):x*x+3*y**3-7
Start:  x=1 y=1 f(x,y)=-2 g(x,y)=-3
Oplossing=(1.2692008664658012,1.2156235714690613)
>>>
```

123. Niet lineaire stelsels. Een goede startwaarde kiezen met behulp van de 'grafiek'

niet_lin_stelsels_tkinter

Een alternatief is om het vorige programma uit te breiden met een programma die vooraf de grafiek van deze niet lineaire functies tekent. Of beter gezegd, niet echt de grafiek maar een gearceerd gebied waar de absolute waarde van de functiewaarde z=f(x,y) en z=g(x,y) kleiner is dan een voorop gegeven getal absZ. Voor de eerste functie een verticale arcering, voor de tweede een horizontale. In het gebied waar beide arceringen elkaar snijden, ligt waarschijnlijk een gemeenschappelijk nulpunt. De gebruiker voert een koppel coördinaten in die dit nulpunt benadert. Het kan natuurlijk ook gebeuren dat er zelfs dan geen gemeenschappelijk nulpunt is. Daarom wordt ook hier het aantal iteraties beperkt tot bijvoorbeeld 20.

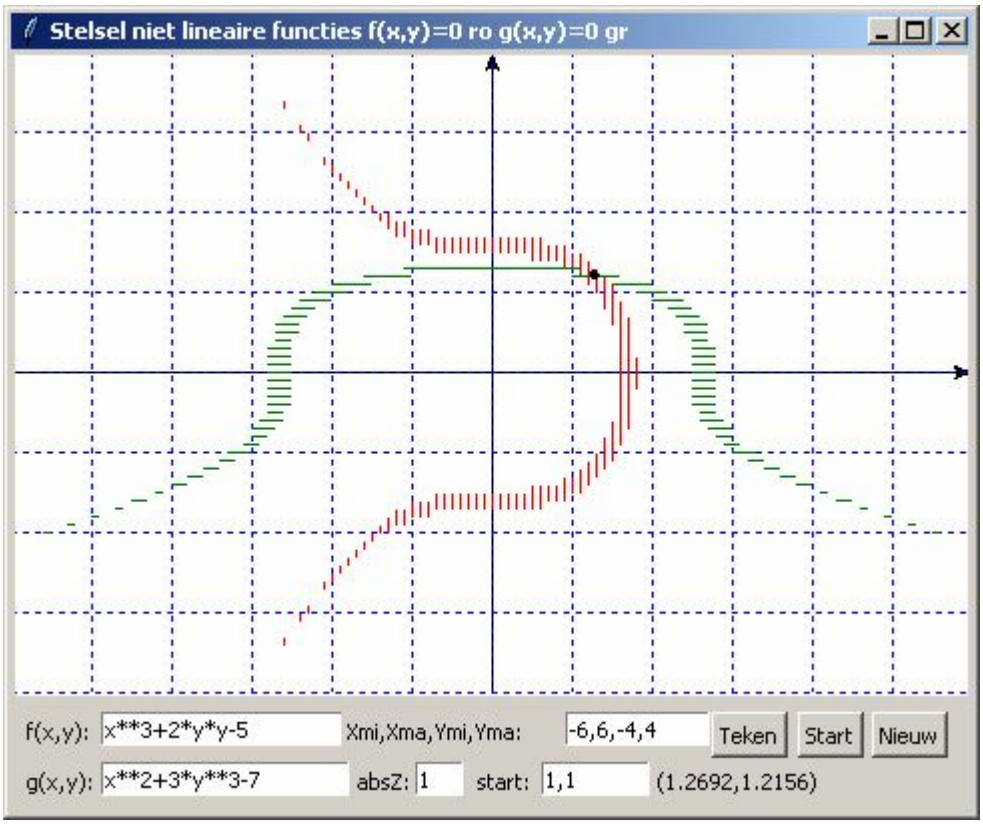
Programma:

```
# oplossen van een stelsel van niet lineaire vergelijkingen met tkinter
from math import *
from tkinter import *
def mult(s):
   l=s.split(',')
   co=[]
   for i in l:co.append(float(i))
   return co
def transx(x):
```

```
   return (x-xmi)*breedte/(xma-xmi)
def transy(y):
   return hoogteC-hoogteC*(y-ymi)/(yma-ymi)
def f(x,y):
   return eval(E1.get())
def g(x,y):
   return eval(E3.get())
d=0.00001
def dfx(x,y):
        return( (f(x+d,y)-f(x,y))/d)
def dfy(x,y):
        return( (f(x,y+d)-f(x,y))/d)
def dgx(x,y):
        return( (g(x+d,y)-g(x,y))/d)
def dgy(x,y):
        return( (g(x,y+d)-g(x,y))/d)
def det(p,q,r,s):
        return(p*s-q*r)
def teken():
   global xmi,xma,ymi,yma
   xmi,xma,ymi,yma=mult(E2.get());C.delete(ALL)
#assen
   X=transx(0);line = C.create_line(X,0,X,hoogteC,fill='black',arrow='first')
   Y=transy(0);line = C.create_line(0,Y,breedte,Y,fill='black',arrow='last')
# rooster
   for x in range(int(xmi),int(xma+1)):
      X=transx(x);line = C.create_line(X,0,X,hoogteC,fill='blue',dash=[1,2])
   for y in range(int(ymi),int(yma+1)):
      Y=transy(y);line = C.create_line(0,Y,breedte,Y,fill='blue',dash=[1,2] )
#kromme
   stap=0.1;az=float(E4.get())
   lis=[];x=xmi
   while x<xma:
      x=x+stap;y=ymi
      while y<yma:
         y=y+stap
         z=f(x,y)
         if abs(z)<az:
            X=transx(x);Y=transy(y)
            lis.append(X);lis.append(Y)
         else:
            if len(lis)==2:
               lis.append(lis[0]);lis.append(lis[1])
            if lis!=[]:
               line = C.create_line(lis,fill='red')
               lis=[]
   lis=[];y=ymi
   while y<yma:
      y=y+stap;x=xmi
      while x<xma:
         x=x+stap
```

```
            z=g(x,y)
            if abs(z)<az:
               X=transx(x);Y=transy(y)
               lis.append(X);lis.append(Y)
            else:
               if len(lis)==2:
                  lis.append(lis[0]);lis.append(lis[1])
               if lis!=[]:
                  line = C.create_line(lis,fill='green')
                  lis=[]
def start():
   x,y=mult(E5.get())
   ox=x+1;oy=y+1;tel=0
   while (abs(ox-x)>d or abs(oy-y)>d)and tel<20:
      ox=x;oy=y;tel=tel+1
      D=det(dfx(x,y),dfy(x,y),dgx(x,y),dgy(x,y))
      Dl=-det(f(x,y),dfy(x,y),g(x,y),dgy(x,y))
      Dk=-det(dfx(x,y),f(x,y),dgx(x,y),g(x,y))
      if D!=0:
         l=Dl/D;k=Dk/D
      else:
         opl='Geen oplossing'
         break
      x=x+l;y=y+k
   if D!=0 and tel<20: # oplossing gevonden
      opl='('+format(x,'.4f')+','+format(y,'.4f')+')'
      X,Y=transx(x),transy(y);C.create_oval(X-2,Y-2,X+2,Y+2,fill='black')
   else:
      opl='Geen oplossing'
   L6=Label(form,text=opl);L6.place(x=320,y=hoInv2)
def nieuw():
   E2.delete(0,len(E2.get()));E1.delete(0,len(E1.get()));C.delete(ALL)
   E3.delete(0,len(E3.get()));E4.delete(0,len(E4.get()));E5.delete(0,len(E5.get()));
   Label(form,text=(' ').ljust(40)).place(x=320,y=hoInv2)
#hoofdprogramma
breedte=480;hoogte=380;hoogteC=hoogte-60;hoInv=hoogte-50;hoInv2=hoogte-25
form=Tk();form.geometry(str(breedte)+'x'+str(hoogte))
form.title('Stelsel niet lineaire functies f(x,y)=0 ro g(x,y)=0 gr')
C = Canvas(form, bg="white", height=hoogteC, width=breedte);C.pack()
L1=Label(form,text='f(x,y):');L1.place(x=5,y=hoInv)
E1=Entry(form);E1.place(x=45,y=hoInv)
L2=Label(form,text='Xmi,Xma,Ymi,Yma:',);L2.place(x=165,y=hoInv)
E2=Entry(form);E2.place(x=277,y=hoInv,width=72);E2.insert(0,'-6,6,-4,4')
L3=Label(form,text='g(x,y):');L3.place(x=5,y=hoInv2)
E3=Entry(form);E3.place(x=45,y=hoInv2)
L4=Label(form,text='absZ:',);L4.place(x=170,y=hoInv2)
E4=Entry(form);E4.place(x=202,y=hoInv2,width=24);E4.insert(0,'1')
L5=Label(form,text='start:',);L5.place(x=230,y=hoInv2)
E5=Entry(form);E5.place(x=265,y=hoInv2,width=54)
B1=Button(form,text='Teken',command=teken);B1.place(x=350,y=hoInv)
B2=Button(form,text='Nieuw',command=nieuw);B2.place(x=430,y=hoInv)
```

```
B3=Button(form,text='Start',command=start);B3.place(x=393,y=hoInv)
form.mainloop()
```



124. Niet lineair stelsel van n vergelijkingen met n onbekenden -jacobiaan

nietlinstelsel_n_vergelijkingen

In een vorig onderdeel hebben we een stelsel van 2 niet lineaire vergelijkingen met 2 onbekenden opgelost.
Uitgaande van een startkoppel (x_o,y_o) werd het gemeenschappelijk nulpunt benaderd door het achtereenvolgens berekenen van $(x_1,y_1),(x_2,y_2),...$
waarbij $x_1 = x_o+k$, $y_1 = y_o+l$, ...
k en l zijn oplossingen van een stelsel
$k.f'_x+l.f'_y \cong - f$
$k.g'_x+l.g'_y \cong - g$
De matrix $\begin{pmatrix} f'_x & f'_y \\ g'_x & g'_y \end{pmatrix}$ wordt de Jacobiaan genoemd.
Om k te berekenen, vervangen we in de Jacobiaan, de coëfficiënten van k door het rechterlid: noem dit $-m_k$.
Om l te berekenen, vervangen we in de Jacobiaan, de coëfficiënten van l door het rechterlid:
noem dit $-m_l$.
$k=-\det(m_k)/\det(jacob)$ en $l=-\det(m_l)/\det(jacob)$

We kunnen dit veralgemenen naar een stelsel van n vergelijkingen met n onbekenden.
fs[i] voor i=0..n-1 zijn n strings die de n niet-lineaire vergelijkingen bevatten.
x[i] voor i=0..n-1 zijn de n onbekenden waarin deze vergelijkingen moeten uitgedrukt worden. x is dus de lijst van deze onbekenden.
Met f(i,x)= eval(fs[i]) kunnen de functiewaarden berekend worden.

df(i,j,x)= de partiële afgeleide van de i-de functie naar x[j] =
(f(i,[x[0],..x[j]+d,..x[n-1]]) - f(i,[x[0],..x[j],..x[n-1]]))/d
jaci= in de jacobiaan de i-de rij van de partiële afgeleiden.
jac is dan de lijst van al deze jaci's, m.a.w. de jacobiaan.
Uiteraard mag hier de functie det(mat) niet ontbreken.
In het hoofdprogramma worden eerst de n functies ingevoerd.
Nadien moeten n startwaarden voor x[0]..x[n-1] ingevoerd worden.
Functiewaarden berekenen voor x[0]..x[n-1] $\in$ [-10,9] zoals bij 2 vergelijkingen is hier voor grotere n bijna onmogelijk, een grafische voorstelling is ook niet mogelijk, vandaar een +/- 'blinde' start.
Een alternatief is om de gebruiker door middel van een menu, de mogelijkheid te geven om voor dezelfde functies steeds nieuwe startwaarden te kiezen.
De jacobiaan wordt dan berekend, alsook de n matrices m[k] die we bepalen door in de jacobiaan het i-de element van kolom k te vervangen door de kolom van de functiewaarden f(i,x).
Nu kunnen we een lijst l opvullen met -det(m[k])/D.
Deze getallen l[0]..l[n-1] kunnen we bijtellen bij de startwaarden x_0[0]..x_0[n-1] om een betere benadering x_1[0]..x_1[n-1] te vinden.
En dit blijven we herhalen tot als de som van de absolute waarden van de lijst l zeer klein geworden is.

Programma:

```
# oplossen van een niet lineair stelsel van n vergelijkingen met n onbekenden -jacobiaan
from math import *
from copy import deepcopy
d=0.0000001
def mult(s):
    l=s.split(',')
    co=[]
    for i in l:co.append(float(i))
    return co
def f(i,x):
    return eval (fs[i])
def df(i,j,x):
    fu1=f(i,x)
    x[j]=x[j]+d
    fu2=f(i,x)
    return (fu2-fu1)/d
def jacob():
    jac=[]
    for i in range(0,n):
        jaci=[]
        for j in range(0,n):
            jaci.append(df(i,j,x))
        jac.append(jaci)
    return jac
def det(mat):
    le=len(mat)
    if le==2:dt=mat[0][0]*mat[1][1]-mat[0][1]*mat[1][0]
    else:
        dt=0
        for i in range(0,le):
            s=[]
```

```
            for k in range(0,le):
                if i!=k:s.append(mat[k][1:])
            dt=dt+mat[i][0]*(-1)**i*det(s)
    return dt
# hoofdprogramma
inp=' ';i=0;fs=[]
print('Geef n functies van x[0],x[1]..x[n-1]\nNa de laatste functie,enkel <Enter>')
while inp!='':
    print('Functie ',i+1,end=' ')
    inp=input(':')
    fs.append(inp);i=i+1
n=i-1;start='1'
print('Aantal vergelijkingen=aantal onbekenden n =',n)
while start=='1':
    x=mult(input('Geef startwaarden voor x[0]..x['+str(n-1)+']:'));l=[]
    soml=1;tel=0
    while soml>0.0001 and tel<30:
        m=[];tel=tel+1
        for k in range(0,n):m.append([])
        for k in range(0,n):
            m[k]=deepcopy(jacob())
            for i in range(0,n):
                m[k][i].pop(k)
                m[k][i].insert(k,f(i,x))
        D=det(jacob());l=[];soml=0
        if D!=0:
            for k in range(0,n):
                l.append(-det(m[k])/D)
                x[k]=x[k]+l[k]
                soml=soml+abs(l[k])
        else:
            break
    if D!=0 and tel<30:print('Oplossing:',x)
    else:print('Geen oplossing')
    start=input('Kies: nieuwe startwaarden (1) of  einde (2)')

>>>
Geef n functies van x[0],x[1]..x[n-1]
Na de laatste functie,enkel <Enter>
Functie  1 :x[0]**3-x[1]**2*4+x[2]**4-15
Functie  2 :x[0]**2+x[1]**3+x[2]-11
Functie  3 :x[0]*4-x[2]**5-x[1]+13
Functie  4 :
Aantal vergelijkingen=aantal onbekenden n = 3
Geef startwaarden voor x[0]..x[2]:1,1,3
Oplossing: [2.3547266979994146, 1.5352958125691427, 1.836374178221822]
Kies: nieuwe startwaarden (1) of  einde (2)2
>>>
```

125. Regressie met een kwadratische functie regressie_kwadratischefunctie

Stel dat we de best passende kwadratische functie $y=a_0+a_1x+a_2x^2$ zoeken bij n puntenkoppels $[(x_0,y_0), (x_1,y_1),..., (x_{n-1},y_{n-1})]$

De som van de kwadraten van de residu's = $S_r= \sum_{i=0}^{n-1}\left(y_i - a_0 - a_1x_i - a_2x_i^2\right)^2$ moeten we minimaliseren.

Het minimum vinden we door de partiële afgeleiden naar a_0, a_1, $a_2 = 0$ te stellen:

we krijgen dan een stelsel van 3 vergelijkingen met 3 onbekenden:

$$\begin{cases} \frac{\delta S_r}{\delta a_0} = -2\sum\left(y_i - a_0 - a_1x_i - a_2x_i^2\right) = 0 \\ \frac{\delta S_r}{\delta a_1} = -2\sum x_i\left(y_i - a_0 - a_1x_i - a_2x_i^2\right) = 0 \\ \frac{\delta S_r}{\delta a_2} = -2\sum x_i^2\left(y_i - a_0 - a_1x_i - a_2x_i^2\right) = 0 \end{cases}$$

of na herschikking:

$$\begin{cases} n.a_0 + \sum x_i.a_1 + \sum x_i^2.a_2 = \sum y_i \\ \sum x_i.a_0 + \sum x_i^2.a_1 + \sum x_i^3.a_2 = \sum x_iy_i \\ \sum x_i^2.a_0 + \sum x_i^3.a_1 + \sum x_i^4.a_2 = \sum x_i^2y_i \end{cases} \text{ of } \begin{bmatrix} n & \sum x_i & \sum x_i^2 \\ \sum x_i & \sum x_i^2 & \sum x_i^3 \\ \sum x_i^2 & \sum x_i^3 & \sum x_i^4 \end{bmatrix} \bullet \begin{bmatrix} a_0 \\ a_1 \\ a_2 \end{bmatrix} = \begin{bmatrix} \sum y_i \\ \sum x_iy_i \\ \sum x_i^2y_i \end{bmatrix}$$

Stellen we :

$$\sum x_i = sx \quad \sum y_i = sy \quad \sum x_iy_i = sxy \quad \sum x_i^2 = sx2 \quad \sum x_i^3 = sx3 \quad \sum x_i^4 = sx4 \quad \sum x_i^2y_i = sx2y$$

dan wordt het stelsel:

$$\begin{cases} n.a_0 + sx.a_1 + sx2.a_2 = sy \\ sx.a_0 + sx2.a_1 + sx3.a_2 = sxy \\ sx2.a_0 + sx3.a_1 + sx4.a_2 = sx2y \end{cases}$$

Het komt er dus op neer om de coëfficiënten van dit stelsel te berekenen:

```
sx,sy,sxy,sx2,sx3,sx4,sx2y=0,0,0,0,0,0,0
for i range(0,n):
   sx= x+xi ; sy=sy+yi ; sxy=sxy+xi*yi ; sx2=sx2+xi²
   sx3= sx3+xi³; sx4=sx4+xi⁴; sx2y=sx2y+xi²yi
```

Deze getallen worden ingevuld in de matrix mat:

$$\text{mat} = \begin{pmatrix} n & sx & sx2 & sy \\ sx & sx2 & sx3 & sxy \\ sx2 & sx3 & sx4 & sx2y \end{pmatrix}$$

De procedure stelsel geeft ons de oplossing a_0, a_1, a_2 en aansluitend de kwadratische functie $y= a_0+a_1.x+a_2.x^2$ die het best fit bij de geven punten (x_i,y_i)

Programma:

```
#  regressie kwadratische functie
from copy import deepcopy
# algemene definities
def gt(g):
   if g>=0:sg='+'
   else:sg=''
   return sg+format(g,'.'+nwk+'f')
def det(mat):
   le=len(mat)
   if le==2:d=mat[0][0]*mat[1][1]-mat[0][1]*mat[1][0]
   else:
      d=0
      for i in range(0,le):
         s=[]
         for k in range(0,le):
            if i!=k:s.append(mat[k][1:])
         d=d+mat[i][0]*(-1)**i*det(s)
   return d
def stelsel(mat):
   acop=deepcopy(mat);m=[];b=[];n=3
   for i in range(0,n):b.append(mat[i][n])
   for i in range(0,n):acop[i].pop()
   for k in range(0,n):m.append([])
   for k in range(0,n):
      m[k]=deepcopy(acop)
      for i in range(0,n):
         m[k][i].pop(k)
         m[k][i].insert(k,b[i])
   d=det(acop);a=det(m[0])/d;b=det(m[1])/d;c=det(m[2])/d
   return(a,b,c)
#hoofdprogramma
xt=[];yt=[]
print('Regressie met een kwadratische functie:')
n=int(input("Geef het aantal koppels coördinaten:"))
for j in range(0,n):
   print("Punt",j+1, end=" :")
   coo=input("x,y=").split(',')
   x=float(coo[0]);y=float(coo[1])
   xt.append(x);yt.append(y)
sx,sy,sxy,sx2,sx3,sx4,sx2y=0,0,0,0,0,0,0
for j in range(0,n):
   sx=sx+xt[j];sy=sy+yt[j];sxy=sxy+xt[j]*yt[j];sx2=sx2+xt[j]**2
   sx3=sx3+xt[j]**3;sx4=sx4+xt[j]**4;sx2y=sx2y+xt[j]**2*yt[j]
vgl1=[n,sx,sx2,sy];vgl2=[sx,sx2,sx3,sxy];vgl3=[sx2,sx3,sx4,sx2y]
mat=[vgl1,vgl2,vgl3]
a0,a1,a2=stelsel(mat);nwk='3'
print('y='+gt(a0)+gt(a1)+'x'+gt(a2)+'x²')

>>>
Geef het aantal getallen:5
Punt 1 :x,y=-1,8
Punt 2 :x,y=0,1
Punt 3 :x,y=1,3
```

```
Punt 4 :x,y=2,12
Punt 5 :x,y=3,26
y=+1.800-2.300x+3.500x²
>>>
```

126. Regressie met veeltermfuncties van de n-de graad regressie_veeltermfunctie

Voor een veeltermfunctie van de 3-de graad vinden we de coëfficiënten a_0,a_1,a_2,a_3 van de best passende functie als oplossing van het stelsel:

$$\begin{cases} n.a_0 + \sum x_i.a_1 + \sum x_i^2.a_2 + \sum x_i^3.a_3 = \sum y_i \\ \sum x_i.a_0 + \sum x_i^2.a_1 + \sum x_i^3.a_2 + \sum x_i^4.a_3 = \sum x_i y_i \\ \sum x_i^2.a_0 + \sum x_i^3.a_1 + \sum x_i^4.a_2 + \sum x_i^5.a_3 = \sum x_i^2 y_i \\ \sum x_i^3.a_0 + \sum x_i^4.a_1 + \sum x_i^5.a_2 + \sum x_i^6.a_3 = \sum x_i^3 y_i \end{cases}$$

Je merkt dat we vooraf sommen moeten berekenen tot sx[6] = $\sum x_i^6$ en sy[3]= $\sum x_i^3 y_i$

Meer algemeen, voor een veeltermfunctie van de n-de graad berekenen we dus sommen tot sx[2n] = $\sum x_i^{2n}$ en sy[n]= $\sum x_i^n y_i$

Het programma is bijna hetzelfde als het vorige:

Programma:

```
#  regressie veeltermfunctie
from copy import deepcopy
# algemene definities
def gt(g):
   if g>=0:sg='+'
   else:sg=''
   return sg+format(g,'.'+nwk+'f')
def det(mat):
   le=len(mat)
   if le==2:d=mat[0][0]*mat[1][1]-mat[0][1]*mat[1][0]
   else:
      d=0
      for i in range(0,le):
         s=[]
         for k in range(0,le):
            if i!=k:s.append(mat[k][1:])
         d=d+mat[i][0]*(-1)**i*det(s)
   return d
def stelsel(mat):
   acop=deepcopy(mat);m=[];b=[];n=len(mat)
   for i in range(0,n):b.append(mat[i][n])
   for i in range(0,n):acop[i].pop()
   for k in range(0,n):m.append([])
   for k in range(0,n):
      m[k]=deepcopy(acop)
      for i in range(0,n):
         m[k][i].pop(k)
         m[k][i].insert(k,b[i])
   d=det(acop);a=[]
   for k in range(0,n):
      a.append(det(m[k])/d)
   return(a)
```

```
#hoofdprogramma
xt=[];yt=[];print('Regressie met een veeltermfunctie:')
g=int(input("Geef de graad g van de veelterm :"))
n=int(input('Geef het aantal koppels getallen n (n>g) :'))
for j in range(0,n):
   print("Punt",j+1, end=" :")
   coo=input("x,y=").split(',')
   x=float(coo[0]);y=float(coo[1])
   xt.append(x);yt.append(y)
sx=[n];sy=[]
for i in range(0,2*g+1):sx.append(0);sy.append(0)
for i in range(1,2*g+1):
   for j in range(0,n):sx[i]=sx[i]+xt[j]**i
for i in range(0,g+1):
   for j in range(0,n):sy[i]=sy[i]+xt[j]**i*yt[j]
mat=[]
for i in range(0,g+1):
   lijn=[]
   for j in range(0,g+1):
      lijn.append(sx[i+j])
   lijn.append(sy[i]);mat.append(lijn)
a=stelsel(mat);nwk='3';ant='y='
for i in range(0,g+1):ant=ant+gt(a[i])+'x^'+str(i)
print(ant)

>>>
Regressie met een veeltermfunctie:
Geef de graad g van de veelterm :3
Geef het aantal koppels getallen n (n>g) :5
Punt 1 :x,y=0,-6
Punt 2 :x,y=1,-7
Punt 3 :x,y=2,-5
Punt 4 :x,y=3,6
Punt 5 :x,y=4,30
y=-5.971x^0-1.190x^1-0.786x^2+0.833x^3
>>>
```

127. Werken met gegevensbestanden, toepassing: regressie

Bij statistiek wordt meestal gebruik gemaakt van grote hoeveelheden gegevens.

Het is niet handig om die nog allemaal te moeten invoeren op het ogenblik dat een statistisch programma wordt uitgevoerd. Het is veel beter dat die reeds klaar zitten in een afzonderlijk bestand. Een mogelijkheid is een tekstbestand, waarbij de afzonderlijke gegevens gescheiden worden door een bepaald teken, bvb. een **;**

Maar eigenlijk worden gegevens, i.h.b. numerieke gegevens het gemakkelijkst en het meest overzichtelijk ingevoerd in een Excel-bestand.

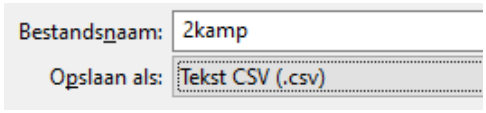
In python zijn er modules om deze gegevens rechtstreeks vanuit een Excel-bestand in te lezen, vb. **xlrd** en **openpyxl**.

Maar er is ook nog een alternatief: je kan het Excel-bestand ook opslaan als een csv-bestand (comma-separated values)

Dit moeten niet noodzakelijk kommas zijn.

Bij het volgende csv-bestand zijn de gegevens gescheiden door een newline (‘\n’)

	A	B	C
1	Loopnummers		
2	Naam	Verspringen	Hoogspringen
3	Karel	4,85	1,55
4	Paul	4,63	1,44
5	Piet	5,21	1,49
6	Marc	6,09	1,61
7	Victor	6,14	1,48
8	Ward	5,58	1,42
9	Joris	4,19	1,37
10	Willy	5,44	1,51
11	Tom	5,23	1,49
12	Stijn	5,18	1,56
13			



In het volgend programma wordt eerst de **cwd** getoond (current working directory). Nadien wordt **ld** de (python)-lijst van alle bestanden uit deze directory. Door de strings uit deze lijst te splitsen in deellijstjes, kunnen we enkel de csv-bestanden uitfilteren en tonen: de gebruiker kan dan kiezen welk bestand ingevoerd wordt met open(…). Tot slot wordt de **inh**oud gelezen en afgedrukt.

Programma:

```
# invoeren csv-lijst
import os
cwd=os.getcwd()
print('Map:',cwd,sep='')
ld=os.listdir()
for f in ld:
    fs=f.split('.')
    if len(fs)>1 and fs[1]=='csv':print(fs[0])
fn=input('Kies een bestand:')
fk=open(fn+'.csv')
inh=fk.read()
fk.close()
print(inh)
>>>
Map:C:\Python34\Dataprog
2kamp
Kies een bestand:2kamp
Loopnummers;;
Naam;Verspringen;Hoogspringen
Karel;4,85;1,55
Paul;4,63;1,44
Piet;5,21;1,49
Marc;6,09;1,61
Victor;6,14;1,48
Ward;5,58;1,42
Joris;4,19;1,37
Willy;5,44;1,51
Tom;5,23;1,49
Stijn;5,18;1,56
>>>
```

Eigenlijk is inh één grote string, namelijk:

'Loopnummers;;\nNaam;Verspringen;Hoogspringen\nKarel;4,85;1,55\nPaul;4,63;1,44\nPiet;5,21;1,49\nMarc;6,09;1,61\nVictor;6,14;1,48\nWard;5,58;1,42\nJoris;4,19;1,37\nWilly;5,44;1,51\nTom;5,23;1,49\nStijn;5,18;1,56\n'

De \n worden door de printopdracht vervangen door een nieuwe lijn te nemen. Om hiermee berekeningen te maken, moeten we er alle getallen uit afzonderen.

linh =de lijst van de rijen uit het csv-bestand, waarin de komma's werden vervangen door punten,zonder de newline-codes en waarin de lege cellen verwijderd werden:

Programma:

```
import os
from math import sqrt
cwd=os.getcwd()
print('Map:',cwd,sep='')
ld=os.listdir()
for f in ld:
    fs=f.split('.')
     if len(fs)>1 and fs[1]=='csv':print(fs[0])
fn=input('Kies een bestand:')
fk=open(fn+'.csv')
inh=fk.read()
linh=inh.replace(',','.').split('\n')
while '' in linh:linh.remove('')
print(linh)
fk.close()
>>>
Map:C:\Python34\Dataprog

2kamp
Kies een bestand:2kamp
['Loopnummers;;', 'Naam;Verspringen;Hoogspringen', 'Karel;4.85;1.55', 'Paul;4.63;1.44', 'Piet;5.21;1.49', 'Marc;6.09;1.61', 'Victor;6.14;1.48', 'Ward;5.58;1.42', 'Joris;4.19;1.37', 'Willy;5.44;1.51', 'Tom;5.23;1.49', 'Stijn;5.18;1.56']
>>>
```

Elk van de elementen van de lijst linh is een string waarin de elementen van de 3 kolommen voorkomen, gescheiden door een ;

Hieruit moeten we nu 1 lijst maken van namen en 2 lijsten a en b van de getallen: vanaf het 3de element kunnen we elk lijstelement van linh splitsen in een lijstje li van 1 woord-string en 2 'getal'-strings waarvan we de floats resp. toevoegen aan lijsten a en b.

Programma:

```
# gegevens halen uit csv-bestand
import os
from math import sqrt
cwd=os.getcwd();print('Map:',cwd,sep='');ld=os.listdir()
for f in ld:
    fs=f.split('.')
    if fs[1]=='csv':print(fs[0])
fn=input('Kies een bestand:');fk=open(fn+'.csv');inh=fk.read()
linh=inh.replace(',','.').split('\n')
while '' in linh:linh.remove('')
fk.close()
tit=[];t=linh[1].split(';')
```

```
tit.append(t[1]);tit.append(t[2]);
nm=[];a=[];b=[]
for i in  range (2,len(linh)):
    li=linh[i].split(';')
    nm.append(li[0])
    a.append(float(li[1]))
    b.append(float(li[2]))
print(tit,'\n',nm,'\n',a,'\n',b)
>>>
Map:C:\Python34\Dataprog
2kamp
Kies een bestand:2kamp
['Verspringen', 'Hoogspringen']
 ['Karel', 'Paul', 'Piet', 'Marc', 'Victor', 'Ward', 'Joris', 'Willy', 'Tom', 'Stijn']
 [4.85, 4.63, 5.21, 6.09, 6.14, 5.58, 4.19, 5.44, 5.23, 5.18]
 [1.55, 1.44, 1.49, 1.61, 1.48, 1.42, 1.37, 1.51, 1.49, 1.56]
>>>
```

We zouden deze gegevens bvb. kunnen gebruiken om curvefitting toe te passen. We gebruiken de programmacode uit eerdere programma's. regressie1_csv

Programma:

```
# lineaire regressie uitgaande van csv-bestand
import os
import math
def oploss(u1,u2,u3,v1,v2,v3):
    d=u1*v2-u2*v1;da=u3*v2-u2*v3;db=u1*v3-u3*v1
    return [da/d,db/d]
def correlatie(xco,yco,xsom,ysom):
    teller=0;x2noemer=0;y2noemer=0
    for j in range(0,n):
        teller=teller+(xco[j]-xsom/n)*(yco[j]-ysom/n)
        x2noemer=x2noemer+(xco[j]-xsom/n)*(xco[j]-xsom/n)
        y2noemer=y2noemer+(yco[j]-ysom/n)*(yco[j]-ysom/n)
    noemer=math.sqrt(x2noemer*y2noemer)
    correl=teller/noemer
    print("Correlatie=",round(correl,4))
def bestand():
    cwd=os.getcwd();print('Map:',cwd,sep='');ld=os.listdir()
    for f in ld:
        fs=f.split('.')
        if len(fs)>1 and fs[1]=='csv':print(fs[0])
    fn=input('Kies een bestand:');fk=open(fn+'.csv');inh=fk.read()
    linh=inh.replace(',','.').split('\n')
    while '' in linh:linh.remove('')
    fk.close()
    tit=[];t=linh[1].split(';')
    tit.append(t[1]);tit.append(t[2]);
    nm=[];a=[];b=[]
    for i in  range (2,len(linh)):
        li=linh[i].split(';')
        nm.append(li[0])
        a.append(float(li[1]))
        b.append(float(li[2]))
```

```
    return(a,b)
#hoofdprogramma
xt,yt=bestand();n=len(xt)
print('Xi=',xt,'\nYi=',yt)
# de noodzakelijke sommaties berekenen
sx=0;sy=0;sxy=0;sx2=0
for j in range(0,n):
    sx=sx+xt[j];sy=sy+yt[j];sxy=sxy+xt[j]*yt[j];sx2=sx2+xt[j]*xt[j]
a,b=oploss(sx,n,sy,sx2,sx,sxy)
# lineaire regressie
if b>=0:
    sg="+"
else:
    sg="-"
print("Regressierechte y=",round(a,4),"x"+sg,round(abs(b),4))
ao=a;bo=b
# lineaire correlatie
correlatie(xt,yt,sx,sy)
>>>
Map:C:\Python34\Dataprog3
2kamp
5kamp_lopen
prc1
prc2
prv1
prv2
Kies een bestand:2kamp
Xi= [4.85, 4.63, 5.21, 6.09, 6.14, 5.58, 4.19, 5.44, 5.23, 5.18]

Yi= [1.55, 1.44, 1.49, 1.61, 1.48, 1.42, 1.37, 1.51, 1.49, 1.56]
Regressierechte y= 0.0608 x+ 1.1724
Correlatie= 0.5214
>>>
```

128. Regressie met csv-bestanden en tkinter regressie2_csv

Het volgende programma werkt met een csv- bestand van 5 verschillende loopnummers: de gebruiker kan er 2 uit kiezen: hij heeft bovendien de keuze tussen lineaire functie, machtsfunctie, exponentiële functie en logaritmische functie.

Het bestand noemt **5kamp_lopen**:

Loopnummers					
Naam	100 m	400 m	800 m	1500 m	3000 m
Karel	12,45	52,29	125,14	4,45	9,7
Paul	13,29	57,4	133,5	4,85	10,3
Piet	12,15	53,2	122,36	4,19	9,24
Marc	11,55	51,2	117,76	4,22	9,45
Victor	14,24	61,24	144,19	5	10,25
Ward	13,74	55,68	125,5	4,11	8,85
Joris	14,05	59,01	133,44	4,67	10,57
Willy	11,44	47,44	107,26	3,92	8,64
Tom	15,05	63,21	147,52	5,11	11,25
Stijn	13,22	56,18	134,46	4,5	9,8

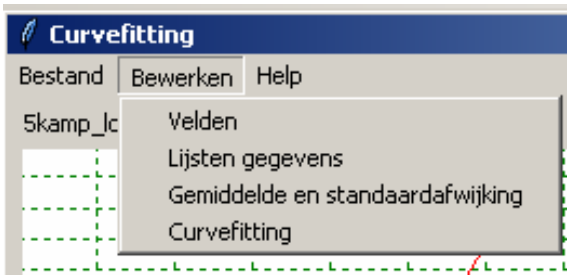
Programma:

```
# regressie uitgaande van csv-bestand
import os
from math import *
from tkinter import *
def grafbereik(x):
        mi=min(x);ma=max(x);e1=int(log10(mi));e2=int(log10(ma))
        ex=max(e1,e2);gr=10**(ex-1);mi=int(mi/10**ex)*10**ex
        ma=(int(ma/10**ex)+1)*10**ex
        return mi,ma,gr
def f(x):return eval(fun)
def mult(s):
   l=s.split(',');co=[]
   for i in l:co.append(float(i))
   return co
def transx(x):return (x-xmi)*bc/(xma-xmi)
def transy(y):return hc-hc*(y-ymi)/(yma-ymi)
def oploss(u1,u2,u3,v1,v2,v3):
   d=u1*v2-u2*v1;da=u3*v2-u2*v3;db=u1*v3-u3*v1
   return [da/d,db/d]
def correlatie(xco,yco,xsom,ysom):
   teller=0;x2noemer=0;y2noemer=0
   for j in range(0,n):
      teller=teller+(xco[j]-xsom/n)*(yco[j]-ysom/n)
      x2noemer=x2noemer+(xco[j]-xsom/n)*(xco[j]-xsom/n)
      y2noemer=y2noemer+(yco[j]-ysom/n)*(yco[j]-ysom/n)
   noemer=sqrt(x2noemer*y2noemer)
   correl=teller/noemer
   voegtoe("Correlatie="+str(round(correl,nk)))
def bestand(fn):
      global titel,x
      fk=open(fn+'.csv');inh=fk.read()
      linh=inh.replace(',','.').split('\n')
      while '' in linh:linh.remove('')
      fk.close();titel=linh[1].split(';');x=[]
      for j in range(0,len(titel)):x.append([])
      for i in  range (2,len(linh)):
         li=linh[i].split(';')
         for j in range(1,len(titel)):
            x[j].append(float(li[j]))
      xt=x[1];yt=x[2]
      return(xt,yt)
def velden():
            voegtoe('Velden')
            for j in range(1,len(titel)):voegtoe(str(j)+')'+titel[j])
def voegtoe(lijn):tex.insert(INSERT,lijn+'\n')
def lijst():
   global xt,yt
   ki,kj=mult(E2.get());ki,kj=int(ki),int(kj)
   xt=x[ki];yt=x[kj]
   voegtoe(titel[ki]+':\n'+str(xt)+'\n'+titel[kj]+':\n'+str(yt))
def invoer():
   global xt,yt,n,func
```

```
#invoer gegevens
   fw.withdraw()
   naambestand=Lb.get(Lb.curselection())
   Label(form,text=naambestand).place(x=5,y=2)
   xt,yt=bestand(naambestand);n=len(xt)
def regressie(menukeuze):
   global lxt,lyt,n,a,b,gx,gy,sx,sy,sx2,sxy,esxy,lsx,lsy,lsxy,losxy,lsx2,tit,fun,nk
   ki,kj=mult(E2.get());ki,kj=int(ki),int(kj)
   xt=x[ki];yt=x[kj]
   lxt=[];lyt=[]
   for i in range(0,n):
      lxt.append(log(xt[i]));lyt.append(log(yt[i]))
# de noodzakelijke sommaties berekenen
   sx=0;sy=0;sxy=0;sx2=0;lsx=0;lsy=0;lsxy=0;lsx2=0;esxy=0;esx2=0;losxy=0;sy2=0
   for j in range(0,n):
      sx=sx+xt[j];sy=sy+yt[j];sxy=sxy+xt[j]*yt[j]
      sx2=sx2+xt[j]*xt[j];esxy=esxy+xt[j]*lyt[j]
      sy2=sy2+yt[j]*yt[j]
      losxy=losxy+lxt[j]*yt[j]
      lsx=lsx+lxt[j];lsy=lsy+lyt[j];lsxy=lsxy+lxt[j]*lyt[j]
      lsx2=lsx2+lxt[j]*lxt[j]
   gx=sx/n;gy=sy/n
   cor=sqrt(n/(n-1))
   stafx=sqrt(sx2/n-gx**2)*cor
   stafy=sqrt(sy2/n-gy**2)*cor
   if menukeuze==1:
         voegtoe('Gemiddelde en standaardafwijking:')
         voegtoe(titel[ki]+'(x): '+format(gx,'.3f')+' , '+format(stafx,'.3f'))
         voegtoe(titel[kj]+'(y): '+format(gy,'.3f')+' , '+format(stafy,'.3f'))
   if menukeuze==2:
         func=vs.get();nk=3
         if func=='lineair':
            tit='Lineaire functie y=a.x+b';fun='a*x+b'
            linreg()
         elif func=='macht':
            tit='Machtsfunctie y=b.x^a';fun='b*x**a'
            machtreg()
         elif func=='expon':
            tit='Exponentiële functie y=b.a^x';fun='b*a**x'
            expreg()
         else:
            tit='Logaritmische functie y=a.ln(x)+b';fun='a*log(x)+b'
            logreg()
def linreg():
   global a,b;a,b=oploss(sx,n,sy,sx2,sx,sxy)
# lineaire regressie
   if b>=0:sg="+"
   else:sg="-"
   voegtoe(tit+'\ny='+str(round(a,nk))+"x"+sg+str(round(abs(b),nk)))
# lineaire correlatie
   correlatie(xt,yt,sx,sy)
def machtreg():
   ma,mb=oploss(lsx,n,lsy,lsx2,lsx,lsxy)
# machts- regressie
   global a,b;a=ma;b=e**mb
```

```
    voegtoe(tit+'\ny='+str(round(b,nk))+'*x^('+str(round(a,nk))+')')
# machts- correlatie
    correlatie(lxt,lyt,lsx,lsy)
def expreg():
    ea,eb=oploss(sx,n,lsy,sx2,sx,esxy)
# expon- regressie
    global a,b;a=e**ea;b=e**eb
    voegtoe(tit+'\ny='+str(round(b,nk))+'*('+str(round(a,nk))+')^x')
# expon- correlatie
    correlatie(xt,lyt,sx,lsy)
def logreg():
# logar- regressie
    global a,b;a,b=oploss(lsx,n,sy,lsx2,lsx,losxy)
    if b>=0:sg="+"
    else:sg="-"
    voegtoe(tit+'\ny='+str(round(a,nk))+"*log(x)"+sg+str(round(abs(b),nk)))
# logaritmische correlatie
    correlatie(lxt,yt,lsx,sy)
def bereken(mk):
    global xmi,xma,ymi,yma,xt,yt
    C.delete(ALL)
    xmi,xma,xgr=grafbereik(xt);ymi,yma,ygr=grafbereik(yt)
    grafinfox=str(xmi)+','+str(xma)+','+str(xgr)
    grafinfoy=str(ymi)+','+str(yma)+','+str(ygr)
    E1.delete(0,len(E1.get()));E1.insert(0,grafinfox)
    E3.delete(0,len(E3.get()));E3.insert(0,grafinfoy)
    regressie(mk)
#grafiek
#assen (indien binnen xmi_xma,ymi_yma )
    X=transx(0);line = C.create_line(X,0,X,hc,fill='blue',arrow='first')
    Y=transy(0);line = C.create_line(0,Y,bc,Y,fill='blue',arrow='last')
#rooster
    x =xmi
    while x<xma:
        X=transx(x);line = C.create_line(X,0,X,hc,fill='green',dash=[1,2])
        x=x+xgr
    y=ymi
    while y<yma:
            Y=transy(y);line = C.create_line(0,Y,bc,Y,fill='green',dash=[1,2] )
            y=y+ygr
#punten
    lis=[]
    for i in range(0,n):
        X,Y=transx(xt[i]),transy(yt[i])
        C.create_oval(X-3,Y-3,X+3,Y+3,fill='magenta')
    X,Y=transx(gx),transy(gy)
    C.create_oval(X-5,Y-5,X+5,Y+5,fill='darkred')
#kromme
    lis=[];stap=0.1;x=xmi
    while x<xma:
        x=x+stap;y=f(x);X=transx(x);Y=transy(y)
        lis.append(X);lis.append(Y)
    line = C.create_line(lis,fill='red')
def nieuw():
     E1.delete(0,len(E1.get()));C.delete(ALL);tex.delete('1.0',END)
```

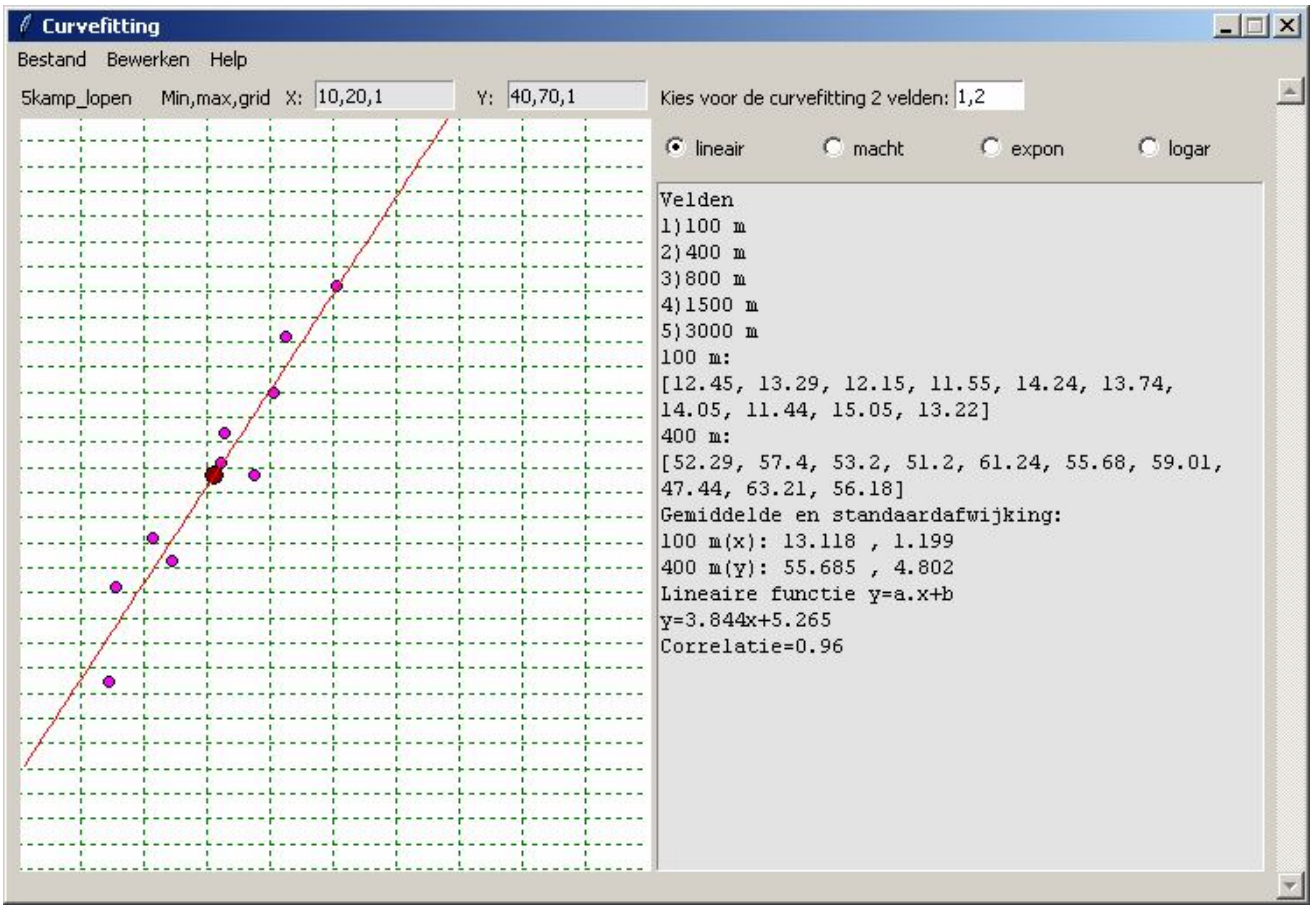
```
    Label(form,text='                    ').place(x=5,y=2)
def lijstbestanden():
   global Lb,fw
   fw=Toplevel(form);fw.geometry('170x200'); fw.title('Openen')
   bd=Button(fw,text='Openen',command=invoer);bd.place(x=50,y=165)
   cwd=os.getcwd();ld=os.listdir();mapcsv=str(cwd)
   L3=Label(fw,text=mapcsv,bg='grey89');L3.place(x=5,y=5)
   lijstcsv=[]
   for f in ld:
      fs=f.split('.')
      if len(fs)>1 and fs[1]=='csv':lijstcsv.append(fs[0])
   tupcsv=tuple(lijstcsv)
   Lb=Listbox(fw,listvariable=StringVar(value=tupcsv),selectmode='SINGLE',
    height=8,width=24,bg='grey89')
   Lb.place(x=5,y=25)
def sluithe():fh.withdraw()
def he():
      global fh
      fh=Toplevel(form);fh.geometry('400x250')
      bh=Button(fh,text='Ok',command=sluithe);bh.place(x=180,y=220)
      texhelp=Text(fh,bg='grey89',height=13,width=48);texhelp.place(x=2,y=2)
      hreg=open('helpreg.txt');helreg=hreg.read();hreg.close()
      texhelp.insert(INSERT,helreg+'\n')
def men():return Menu(mb,tearoff=0)
def ad(me,lab,co):me.add_command(label=lab,command=co)
#hoofdprogramma
global br,ho,hc,bc,bt,ht;sc=1
br=int(740);ho=int(470);hc=ho-40;bc=360;bt=370;bt2=85;hob=40
form=Tk();form.geometry(str(br)+'x'+str(ho));form.title('Curvefitting');form.resizable(False,False)
C = Canvas(form, bg="white", height=hc, width=bc);C.place(x=5,y=22)
Label(form,text='Min,max,grid   X:').place(x=bt2,y=2);Label(form,text=' Y:').place(x=bt2+177,y=2)
E1=Entry(form,bg='grey89');E1.place(x=bt2+100,y=2,width=70)
E3=Entry(form,bg='grey89');E3.place(x=bt2+210,y=2,width=70)
L2=Label(form,text='Kies voor de curvefitting 2 velden:',);L2.place(x=bt,y=2)
E2=Entry(form);E2.place(x=bt+190,y=2,width=40);E2.insert(0,'1,2');fun='a*x+b';a,b=0,0
tex=Text(form,bg='grey89',height=25,width=42,wrap=WORD);tex.place(x=bt,y=60)
vsb=Scrollbar(form,orient=VERTICAL,command=tex.yview,width=20)
tex['yscrollcommand']=vsb.set;vsb.pack(side=RIGHT,fill=Y)
ki=1;kj=2
def rb(func,xp,yp):
   Radiobutton(form,text=func,variable=vs,value=func).place(x=xp,y=yp)
vs=StringVar(form,'lineair')
rb('lineair',bt,30);rb('macht',bt+90,30)
rb('expon',bt+180,30);rb('logar',bt+270,30)
mb=Menu(form);fm=men();em=men();hm=men()
fm=Menu(mb,tearoff=0)
ad(fm,'Nieuw',nieuw);ad(fm,'Open',lijstbestanden)
ad(em,'Velden',velden);ad(em,'Lijsten gegevens',lijst)
ad(em,'Gemiddelde en standaardafwijking',lambda:bereken(1))
ad(em,'Curvefitting',lambda:bereken(2));ad(hm,'Help',he)
mb.add_cascade(label='Bestand',menu=fm)
mb.add_cascade(label='Bewerken',menu=em)
mb.add_cascade(label='Help',menu=hm)
form.config(menu=mb)
form.mainloop()
```



De prc- en prv-bestanden mag je niet openen: het zijn csv-bestanden die respectievelijk dienen voor de programma's 'Vlakke meetkunde' en 'Ruimtemeetkunde. (Zie verder)

Voor curvefitting kiezen we voor het programma 5kamp_lopen.
Om berekeningen en grafieken te maken, kiezen we het menu **Bewerken**
Velden geeft ons de titels van alle velden, in dit geval 1)100 m ..2)400 m..
Lijsten gegevens geeft de gegevens van de 2 velden die ingevuld zijn rechtsboven.
Met de keuze 1,2:
100 m:
[12.45,...
400 m:
[52.29,...
Gemiddelde en standaardafwijking: je krijgt van de gekozen velden bovendien de grafiek van de gegevens + gemiddelde van beide velden. De coördinaten worden automatisch aangepast zodat steeds alle gegevens te zien zijn. Om te zien welk deel van het 1ste kwadrant getoond wordt, zie je boven de grafiek de Min,max,grid van X en Y. Die moet je dus niet zelf invullen.

Curvefitting: Geeft de best passend grafiek, hierbij kan je kiezen voor een lineaire, machts-, exponentiële of logaritmische functie.



129. Meetkunde - werken met zelf gedefinieerde klassen

Zowel voor vlakke als ruimtemeetkunde zullen we een programma maken waarin zowat alle onderwerpen uit het leerplan SO aan bod komen.
Van het programma voor de vlakke meetkunde bestaan 2 versies: het eerste gaat enkel over punten en rechten, het tweede is een uitbreiding van het eerste, waarbij ook berekeningen en tekeningen met cirkels kunnen gemaakt worden. Enkel van het eerste is de listing opgenomen in de tekst

De drie programma's werken ongeveer op dezelfde manier, vandaar dat de wiskundige achtergrond simultaan wordt behandeld.. In de vlakke meetkunde werken we met punten, rechten (en cirkels), in de ruimtemeetkunde met punten, rechten en vlakken.
Om hiermee vlot te kunnen werken, gaan we in beide programma's klassen definiëren:

Een punt heeft een naam (hoofdletter) en 2 coördinaten in de vlakke meetkunde of 3 coördinaten in de ruimtemeetkunde:

```
class Punt:
    def __init__(self,naam,xco,yco,zco):
        self.naam=naam;self.x = xco;self.y = yco;self.z=zco
```

Een rechte heeft een naam (kleine letter) en 2 punten:

```
class Rechte:
        def __init__(self,naam,p1,p2):
            self.naam=naam;self.p1=p1;self.p2=p2
```

Een cirkel in de vlakke meetkunde heeft een naam (een gevolgd door een subscript-cijfer $_{0..9}$), een punt en een straal:

```
class Cirkel:
    def __init__(self,naam,p1,s):
        self.naam=naam;self.p1=p1;self.s=s
```

Een vlak in de ruimtemeetkunde heeft een naam (griekse letter) en 3 punten:

```
class Vlak:
    def __init__(self,naam,p1,p2,p3):
        self.naam=naam;self.p1=p1;self.p2=p2;self.p3=p3
```

Voorbeeld:

```
class Punt:
   def __init__(self,naam,xco,yco):
                self.naam=naam;self.x = xco;self.y = yco
class Rechte:
   def __init__(self,naam,p1,p2):
                self.naam=naam;self.p1=p1;self.p2=p2
punt1=Punt('P',2,3);punt2=Punt('Q',4,7)
rechte1=Rechte('r',punt1,punt2)
print(rechte1.naam+' is de rechte door '+rechte1.p1.naam+' en '+rechte1.p2.naam)
>>>
r is de rechte door P en Q
>>>
```

We zullen werken met lijsten:
in de vlakke meetkunde met punten, rechten en cirkels: lijsten p,r,c
in de ruimtemeetkunde met punten, rechten en vlakken: lijsten p,r,v

Om een eenvoudig verband te hebben tussen de verschillende kenmerken van een punt, rechte,cirkel of vlak, bvb. tussen de naam van een punt en zijn coördinaten zorgen we er voor dat p[0] naam ‘A’ , p[1] naam ‘B’ krijgt.enz..; Voor rechten krijgt r[0] naam ‘a’… voor vlakken krijgt v[0] naam ‘α’. Hiervoor gebruiken we de python- functies die een karakter omzetten naar zijn ascii-code (of algemener de unicode) en omgekeerd: **chr** en **ord**

Voor een cirkel heb ik gekozen voor 2 karakters, een ‘O’ gevolgd door een subscript- cijferindex van 0 tot 9. Deze keuzes geven wel wat beperkingen: omdat O, P punten zijn die ik nodig heb voor andere doeleinden, beperk ik het aantal punten tot 14 (A,B,C,…N) en 10 cirkels (0,1,2,..9)
Dit aantal is meer dan genoeg om alle mogelijke tekeningen en berekeningen te maken.

p=[p[0],p[1],..]
p[0].naam = ‘A’ = chr(0+65) p[1].naam = ‘B’ = chr(1+65) p[i].naam = ‘..’= chr(i+65)
r=[r[0],r[1],..]
r[0].naam = ‘a’ = chr(0+97) r[1].naam = ‘b’ = chr(1+97) r[i].naam = ‘..’= chr(i+97)
v=[v[0],v[1],..]
c =[c[0],c[1],..]
In het vlak
c =[c[0],c[1],..]
c[0].naam =’ O_0’ c[1].naam =’ O_1’ c[i].naam = ‘O’+sublis[i]
(sublis =lijst [$_{0,1,2,3,\ldots,9}$])
In de ruimte
v=[v[0],v[1],..]
v[0].naam = ‘α’ = chr(0+945) v[1].naam = ‘α’ = chr(1+945) v[i].naam =‘..’= chr(i+945)

Omgekeerd kan je van de naam terug naar de index:
Bijvoorbeeld voor punten: i= ord(p[i].naam)-65
Wil je bvb. de coördinaten van B: ind=ord('B')-65 ; print(p[ind].x,p[ind].y)

Je mag natuurlijk bij de definitie van de rechten, cirkels en vlakken enkel punten gebruiken die vooraf reeds gedefinieerd zijn.

Voorbeeld enkel met punten en rechten in het vlak

Programma:

```
# class-voorbeeld
class Punt:
        def __init__(self,naam,xco,yco):
            self.naam=naam;self.x = xco;self.y = yco
        def __repr__(self):
            return self.naam+'('+str(self.x)+','+str(self.y)+')'
class Rechte:
        def __init__(self,naam,p1,p2):
      self.naam=naam;self.p1=p1;self.p2=p2
        def __repr__(self):
            return self.naam+'('+str(self.p1)+','+str(self.p2)+')'
def gt(g):
   if g>=0:sg='+'
   else:sg=''
   return sg+str(g)
# punten en rechten initialiseren: er worden 2 arrays opgebouwd p[i] van punten en r[i] van rechten
p=[];copu=[[2,3],[1,-4],[-3,4],[2,5]];
print('\nPunten:')
for i in range(0,len(copu)):
   x,y=copu[i];p.append(Punt(chr(65+i),x,y));print(str(p[i]),end=' ')
r=[];rech=[[p[0],p[2]],[p[1],p[3]],[p[2],p[3]]]
print('\nRechten:')
for i in range(0,len(rech)):
   P,Q=rech[i];r.append(Rechte(chr(97+i),P,Q))
   print(str(r[i]),end=' ')
# nu worden vergelijkingen van de rechten afgedrukt:
print()
for i in range(0,len(rech)):
   print('Vergelijking van '+r[i].naam+':',end=' ')
   m=((r[i].p2.y-r[i].p1.y)/(r[i].p2.x-r[i].p1.x))
   print('y'+gt(-r[i].p1.y)+'='+str(m)+'*(x'+gt(-r[i].p1.x)+')')
>>>
Punten:
A(2,3) B(1,-4) C(-3,4) D(2,5)
Rechten:
a(A(2,3),C(-3,4)) b(B(1,-4),D(2,5)) c(C(-3,4),D(2,5))
Vergelijking van a: y-3=-0.2*(x-2)
Vergelijking van b: y+4=9.0*(x-1)
Vergelijking van c: y-4=0.2*(x+3)
>>>
```

130. Vlakke meetkunde - ruimtemeetkunde

Om alle punten, rechten, cirkels en vlakken tijdens het programma in te voeren, is nogal tijdrovend. Ook hier is het zeker interessant om ze op voorhand met Excel in te voeren en op te slaan als csv-bestand. Vanuit het programma wordt het bestand geopend en de gegevens automatisch aan de juiste klassen toegevoegd.

pr- bestanden : te gebruiken voor 'Vlakke_meetkunde1'

Vlakke meetkunde		
Naam P	Punten	Rechten
A	1,2	A,D
B	0,3	B,C
C	1,4	B,E
D	2,3	C,D
E	4,5	E,F
F	3,1	

prc- bestanden : te gebruiken voor 'Vlakke_meetkunde2'

Vlakke meetkunde			
Naam P	Punten	Rechten	Cirkels
A	1,2	A,B	A,2
B	0,1	A,C	B,2
C	-1,4	B,C	D,3
D	2,3	C,D	G,1
E	4,5	E,F	E,1
F	3,3		F,2
G	3,1		
H			

prv- bestanden : te gebruiken voor 'Ruimtemeetkunde'

Ruimtemeetkunde			
Naam P	Punten	Rechten	Vlakken
A	1,2,4	A,B	A,B,C
B	0,1,2	A,C	B,C,D
C	5,1,1	B,C	C,E,F
D	2,3,4	C,D	G,A,D
E	4,5,6	E,F	E,G,B
F	1,2,9		A,C,E
G	3,1,4		

Het is zeer gemakkelijk om dergelijke csv-bestandjes met excel te maken. Sla ze wel op met de juiste benaming. Omdat er ook reeds csv- bestanden in de 'cwd' zitten, met een andere structuur, is het aangewezen om de naam van deze bestanden gemakkelijk herkenbaar te maken: voor de vlakke meetkunde pr1, pr2 (**p**unten en **r**echten) of prc1, prc2 (**p**unten, **r**echten en **c**irkels),…
voor de ruimtemeetkunde prv1,prv2, (**p**unten, **r**echten en **v**lakken)….

Opgepast, de eerste kolom met de namen van de punten wordt niet gebruikt door het programma: ze is enkel nuttig in Excel ter informatie om de kolommen van rechten, vlakken of cirkels in te vullen. De namen van punten,rechten,…worden door het programma toegevoegd.

Vanuit Python is het dan niet zo moeilijk om deze bestanden in te laden en de gegevens op te slaan in de lijsten p, r ,c of v

In de 2 laatste programma's is ook de mogelijkheid voorzien om bijkomend punten, rechten en cirkels of vlakken in te voeren: het is immers soms interessant om bijkomend specifieke punten,rechten… te kunnen gebruiken

De programma's moeten alle interessante berekeningen en grafieken kunnen uitvoeren:
Parameter- en/of cartesiaanse vergelijkingen van rechten, cirkels en vlakken.
Bissectrices, snijpunten van rechten, rechte en cirkel, rechte en vlak
Loodrechte stand, raaklijnen

Afstanden tussen 2 punten, punt en rechte,punt en vlak
Hoeken tussen 2 punten, 2 rechten, 2 vlakken, rechte en vlak.
Bovendien moet van elk lijstje punten, rechten en vlakken een grafiek kunnen getekend worden.
Het assenstelsel moet kunnen gewijzigd worden in de vlakke meetkunde.
In de ruimte moet de grafiek willekeurig kunnen gedraaid worden. Hiervoor kunnen we +/- gebruik maken van de grafiek van de 3dim- programma's van in de cursus.

131. Berekeningen met rechten

Vergelijkingen

Om vergelijkingen op te stellen, kunnen we richtingsvectoren gebruiken:

Zijn p[jp1] en p[jp2] 2 gedefinieerde punten van een rechte, dan worden de coördinaten van een richtingsvector van deze rechte in het vlak gegeven door:

```
def ricvec(a,b):return[b.x-a.x,b.y-a.y]
rx,ry=ricvec(p[jp1],p[jp2])
```

In de ruimte voegen we gewoon een derde coördinaat toe:

```
def ricvec(a,b):return[b.x-a.x,b.y-a.y,b.z-a.z]
rx,ry,rz=ricvec(p[jp1],p[jp2])
```

(opgepast: a en b zijn hier geen namen van punten maar parameters van een functie, de functie moet aangeroepen worden met 2 gedefinieerde punten)

Op analoge wijze kunnen parametervergelijkingen bepaald worden
Als k en l parameters zijn, is de parametervergelijking van een rechte:
x,y= a.x+k*(b.x-a.x), a.y+k*(b.y-a.y)

In de ruimtemeetkunde moeten we een z-gedeelte toevoegen:
x,y,z= a.x+k*(b.x-a.x), a.y+k*(b.y-a.y), a.z+k*(b.z-a.z)

Voor de parametervergelijkingen van een vlak door 3 punten a,b,c in de ruimte hebben we trouwens 2 richtingsvectoren nodig:
x,y,z= a.x+k*(b.x-a.x)+l*(c.x-a.x), a.y+k*(b.y-a.y)+l*(c.y-a.y), a.z+k*(b.z-a.z)+l*(c.z-a.z)

Om de c.v. ux+vy+w=0 van een rechte in de vlakke meetkunde te bepalen

Als p1(x1,y1) en p2(x2,y2) punten zijn van de rechte, dan kunnen we nemen:
u= y2-y1 v=-x2+x1 w=-y2.x1+x2.y1 zodat:

```
def cv(a,b):u=b.y-a.y;v=-b.x+a.x;w=-b.y*a.x+b.x*a.y;return u,v,w
u,v,w=cv(r[jr].p1,r[jr].p2)
```

Om de c.v. $x^2+y^2+ux+vy+w=0$ van een cirkel in het vlak te bepalen:
Als p1(x1,y1) het middelpunt en s de straal is, dan nemen we:
u=-2x1 v=-2y1 $w=x1^2+y1^2-s^2$

```
def cc(p,r):u=-2*p.x;v=-2*p.y;w=p.x**2+p.y**2-r**2;return u,v,w

u,v,w=cc(c[i].p1,c[i].s)
```

Afstanden tussen 2 punten

De afstand tussen 2 punten a en b met de stelling van Pythagoras:
In het vlak: sqrt((a.x-b.x)**2+(a.y-b.y)**2)
In de ruimte: sqrt((a.x-b.x)**2+(a.y-b.y)**2+(a.z-b.z)**2)

Afstand tussen een punt en een rechte (vlakke meetkunde)

In de vlakke meetkunde kunnen we voor de afstand tussen een punt en een rechte, een gelijkaardige formule gebruiken:

Formule: $d(P,r) = \frac{|u.x_1 + v.y_1 + w|}{\sqrt{u^2 + v^2}}$

u,v,w=cv(r[jr].p1,r[jr].p2) (cv- functie geeft ons de u,v,w van de cartesiaanse vergelijking)
d_p_r=abs(u*p[jp].x+v*p[jp].y+w)/sqrt(u*u+v+v*v)

Hoek tussen 2 punten

Met de hoek tussen 2 punten P en Q wordt de scherpe hoek bedoeld, tussen de rechten OP en OQ :

$$\alpha = \angle(\overrightarrow{OP},\overrightarrow{OQ}) = \text{acos}(\frac{x_1.x_2 + y_1.y_2}{\sqrt{x_1^2 + y_1^2}.\sqrt{x_2^2 + y_2^2}})$$

Een algemene definitie kan dan gebruikt worden voor alle hoekberekeningen:

```
def bepaalhoek():
    br=(a1*a2+b1*b2)/(sqrt(a1**2+b1**2)*sqrt(a2**2+b2**2))
    hoekdeg=degrees(acos(br))
    if hoekdeg>90:hoekdeg=180-hoekdeg
    return (hoekdeg)

a1,b1=p[i1].x,p[i1].y ; a2,b2=p[i2].x,p[i2].y
hk=bepaalhoek()
```

Hoek tussen 2 rechten

De hoek tussen 2 rechten is de scherpe hoek tussen 2 richtingsvectoren van die rechten:
De hoek tussen $r_1(A(x_1,y_1),B(x_2,y_2))$ en $r_2(C(x_3,y_3),D(x_4,y_4))$ wordt dan gegeven door

$$\alpha = \angle(\overrightarrow{AB},\overrightarrow{CD}) = \text{acos}(\frac{(x_2 - x_1).(x_4 - x_3) + (y_2 - y_1).(y_4 - y_3)}{\sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}.\sqrt{(x_4 - x_3)^2 + (y_4 - y_3)^2}})$$

```
a1,b1=ricvec(r[jr1].p1,r[jr1].p2) ; a2,b2=ricvec(r[jr2].p1,r[jr2].p2)
hk=bepaalhoek()
```

132. Listing van het eerste programma 'Vlakke_meetkunde1'

vlakke_meetkunde1

```
# vlakke meetkunde 1 punten rechten
import os
import codecs
from tkinter import *
from tkinter import messagebox
from math import *
# algemene definities
def mult(s):
```

```
    li=s.split(',');co=[]
    for i in li:co.append(float(i))
    return co
def transx(x):return (x-xmi)*bc/(xma-xmi)
def transy(y):return hc-hc*(y-ymi)/(yma-ymi)
def gms(hoekdd):
    hoekstr=''
    gr=int(hoekdd);hoekstr=hoekstr+str(gr)+'g';decmi=60*(hoekdd-gr)
    mi=int(decmi);hoekstr=hoekstr+str(mi)+'m';decs=60*(decmi-mi)
    se=int(decs+0.5);hoekstr=hoekstr+str(se)+'s'
    return hoekstr
def stelsel2x2(u1,u2,u3,v1,v2,v3):d=u1*v2-u2*v1;da=u3*v2-u2*v3;db=u1*v3-u3*v1;return [da/d,db/d]
def foutm():h=messagebox.showinfo('Foutieve invoer','Je hebt niet de juiste gegevens ingevoerd\nInvoer wissen
en verbeteren a.u.b')
def foutm2():h=messagebox.showinfo('Lege lijst','Je hebt nog geen gegevens\nCsv-bestand openen')
#invoer class Punt,Rechte,Vlak
class Punt:
    def __init__(self,naam,xco,yco):self.naam=naam;self.x = xco;self.y = yco
    def __repr__(self):return self.naam+'('+str(self.x)+','+str(self.y)+')'
class Rechte:
        def __init__(self,naam,p1,p2):self.naam=naam;self.p1=p1;self.p2=p2
        def __repr__(self):return self.naam+'('+str(self.p1)+','+str(self.p2)+')'
# csv-bestanden tonen in cwd (current working directory)
def lijstbestanden(ke):
    global Lb,fw,E5;fw=Toplevel(form);fw.geometry('195x220');fw.title('Bestand')
    if ke=='openen':srt='csv';bd=Button(fw,text='Openen',command=invoerbestand)
    else:srt='txt';bd=Button(fw,text='Opslaan',command=opslaanbestand)
    bd.place(x=140,y=190);cwd=os.getcwd();ld=os.listdir();mapcsv=str(cwd)
    L3=Label(fw,text='Dir:'+mapcsv,bg='grey89');L3.place(x=5,y=5,width=185)
    L4=Label(fw,text='Bestand:');L4.place(x=5,y=190)
    E5=Entry(fw,bg='white');E5.place(x=55,y=190,width=80);lijstsrt=[]
    for f in ld:
        fs=f.split('.')
        if len(fs)>1 and fs[1]==srt:lijstsrt.append(fs[0])
    Lb=Listbox(fw,listvariable=StringVar(value=tuple(lijstsrt)),selectmode='SINGLE',
    height=9,width=30,bg='white');Lb.place(x=5,y=25)
def opslaanbestand():
    fw.withdraw();naambestand=E5.get();tekst=tex.get('1.0','end')
    if naambestand=='':naambestand=Lb.get(Lb.curselection())
    fo=open(naambestand+'.txt','w',encoding='utf-8');fo.write(tekst);fo.close()
#gebruiker maakt keuze uit csv-bestanden
def invoerbestand():
#invoer gegevens
    global xt,yt,n,func;fw.withdraw();naambestand=E5.get()
    if naambestand=='':naambestand=Lb.get(Lb.curselection())
    form.title('Vlakke meetkunde (punten en rechten): '+naambestand);bestand(naambestand)
def bestand(fn):
        global titel,x,lijstP,lijstR,fw,p,r,ip,ir
        fk=open(fn+'.csv');inh=fk.read();fk.close()
        linh=inh.split('\n') # linh =lijst rijen uit csv-bestand
        while '' in linh:linh.remove('')
        #lijsten opvullen met punten rechten li[1] = kolom punten...
        lijstP=[];lijstR=[];p=[];r=[];ip,ir=0,0 # ip,ir =teller aantal punten,rechten
        for i in  range (2,len(linh)):
            li=linh[i].split(';')
```

```
            if len(li[1])>0 and li[1][0]==chr(39):li[1]=li[1][1:len(li[1])] # eventuele ' verwijderen
            if li[1].count(',')==1:
               p_it=chr(ip+65)+'('+li[1]+')';lijstP.append(p_it) # lijstP met punt 'A(.,.)'....
               xi,yi=mult(li[1]);p.append(Punt(chr(ip+65),xi,yi));ip+=1 # p =lijst Punten (class Punt:p[0].naam='A')
        for i in  range (2,len(linh)):
            li=linh[i].split(';')
            if li[2].count(',')==1:
                r_it=chr(ir+97)+'('+li[2]+')';lijstR.append(r_it) # lijstR met rechte 'a(.,.)'....
                el=li[2].split(',');p1=p[indp(el[0])];p2=p[indp(el[1])]
                r.append(Rechte(chr(97+ir),p1,p2));ir+=1 # r =lijst Rechten (class Rechte:r[0].naam='a')
        fw=Toplevel(form);fw.geometry('30x330');fw.title('Lijst');toonlijst()
        Label(fw,text='Lijsten punten,\nrechten').place(x=10,y=5)
        bh=Button(fw,text='OK',command=lambda:fw.destroy());bh.place(x=40,y=40)
#def selec():global st;st=var.get()
# 2 lijsten, resp. van punten, rechten tonen in dialoogvenster
def toonlijst():
    global LP,LR,punten,rechten;punten='';rechten=''
    LP=Listbox(fw,listvariable=StringVar(value=tuple(lijstP)),selectmode=MULTIPLE,
     height=8,width=12,bg='grey89');LP.place(x=5,y=75)
    LR=Listbox(fw,listvariable=StringVar(value=tuple(lijstR)),selectmode=MULTIPLE,
    height=8,width=12,bg='grey89');LR.place(x=5,y=190)
    vsb3=Scrollbar(fw,orient=VERTICAL,command=LP.yview,width=25);
    LP['yscrollcommand']=vsb3.set;vsb3.place(x=85,y=75)
    vsb4=Scrollbar(fw,orient=VERTICAL,command=LR.yview,width=25);
    LR['yscrollcommand']=vsb4.set;vsb4.place(x=85,y=190)
    for punt in lijstP:punten=punten+str(punt)+' '
    for recht in lijstR:rechten=rechten+str(recht)+' '
def voegtoe(lijn):tex.insert(INSERT,lijn+'\n')
# helpbestand helprmtk.txt : kan met kladblok gewijzigd worden
def he():
     global fh;fh=Toplevel(form);fh.geometry('830x520');fh.title('Helptekst vlakke meetkunde')
     bh=Button(fh,text='OK',command=lambda:fh.destroy());bh.place(x=360,y=480)
     texhelp=Text(fh,bg='grey89',height=29,width=100);texhelp.place(x=5,y=2)
     hreg=open('helpvmtk1.txt');helreg=hreg.read();hreg.close();texhelp.insert(INSERT,helreg+'\n')
     vsb2=Scrollbar(fh,orient=VERTICAL,command=texhelp.yview,width=25)
     texhelp['yscrollcommand']=vsb2.set;vsb2.pack(side=RIGHT,fill='y')
# instellingen assenstelsel, parameters, lijst bij vorige grafiek, nauwkeurigheid
def inst():
    global fi,E1,E3,E4,E2,E7,E8,ci,cj,opstart,maxxyz,drxy,dryz,k,l,nwk,wkeu
    fi=Toplevel(form);fi.geometry('230x110');fi.title('Instellingen');w='white'
    Label(fi,text='minX').place(x=0,y=2);Label(fi,text='maxX').place(x=70,y=2)
    Label(fi,text='minY').place(x=0,y=25);Label(fi,text='maxY').place(x=70,y=25)
    E1=Entry(fi,bg=w);E1.place(x=35,y=2,width=30);E1.insert(0,xmi)
    E2=Entry(fi,bg=w);E2.place(x=105,y=2,width=30);E2.insert(0,xma)
    E3=Entry(fi,bg=w);E3.place(x=35,y=25,width=30);E3.insert(0,ymi)
    E4=Entry(fi,bg=w);E4.place(x=105,y=25,width=30);E4.insert(0,yma)
    Label(fi,text='param k').place(x=0,y=50)
    E7=Entry(fi,bg=w);E7.place(x=55,y=50,width=40);E7.insert(0,k)
    if opstart==0:ci=IntVar();cj=IntVar();opstart=1
    cb1=Checkbutton(fi,text='Rooster',variable=cj);cb1.place(x=135,y=0)
    cb2=Checkbutton(fi,text='Grafiek+lijst',variable=ci);cb2.place(x=135,y=25)
    L8=Label(fi,text='#decim.cijfers');L8.place(x=110,y=50)
    E8=Entry(fi,bg=w);E8.place(x=195,y=50,width=20);E8.insert(0,nwk)
    bi2=Button(fi,text='OK',command=confinst);bi2.place(x=90,y=75,width=50)
def confinst():fi.withdraw();checksel()
```

```
def checksel():
    global xmi,xma,ymi,yma,k,nwk,wkeu,rkeu
    xmi=float(E1.get());xma=float(E2.get());ymi=float(E3.get())
    yma=float(E4.get());k=float(E7.get());nwk=E8.get();wkeu=ci.get();rkeu=cj.get()
def info():
    global inf;inf=Toplevel(form);inf.geometry('140x60');inf.title('Info')
    Label(inf,text='2021 De Meester André').place(x=0,y=2)
    binf=Button(inf,text='OK',command=lambda:inf.destroy());binf.place(x=60,y=25)
# lr = lijsten parametervergelijkingen van rechte
def fr(k):return eval(lr[0])
def gr(k):return eval(lr[1])
def hr(k):return eval(lr[2])
# definities i.v.m. GRAFIEKEN
def omzet(x,y):return transx(x),transy(y)
def lijn(x1,y1,x2,y2):
    global X,Y;lis=[];X,Y=omzet(x1,y1);lis.append(X);lis.append(Y);X,Y=omzet(x2,y2);
    lis.append(X);lis.append(Y);return lis
def roosteken():
    x=xmi
    while x<xma:
        if x!=0:line = C.create_line(lijn(x,ymi,x,yma),fill='lightgrey',dash=[1,5])
        x=x+1
    y=ymi
    while y<yma:
        if y!=0:line = C.create_line(lijn(xmi,y,xma,y),fill='lightgrey',dash=[1,5])
        y=y+1
def tekeninit():
    global xmi,xma,ymi,yma,kmi,lmi,kma,lma,lijstfun,grafbas,ci,pos,k,wkeu,rkeu,nwk
    checksel();nieuwgrafbas=E1.get()+','+E2.get()+','+E3.get()+','+E4.get()
    nieuwgrafbas=str(xmi)+','+str(xma)+','+str(ymi)+','+str(yma)
    # indien er aan instellingen iets verandert, de grafiek wissen
    if nieuwgrafbas!=grafbas or wkeu==0:C.delete(ALL)
    grafbas=nieuwgrafbas;kmi=-4;kma=5;assen()
    if rkeu==1:roosteken()
def graf():dial('Grafiek lijst:',tekengraflijst)
def tekengraflijst():
    global i;lijstgr=E6.get().split(',');tekeninit()
    for j in range(0,len(lijstgr)):
        naam=lijstgr[j];soort=ord(naam[0])
        if soort<91:i=indp(naam);tekengrafp(p[i])              # als een punt: tekengrafp()
        if soort>96 and soort<123:i=indr(naam);tekengrafr(r[i]) # als een rechte: tekengrafr()
    voegtoe('Lijst: '+E6.get())
def tekengrafp(pun):
    X,Y=omzet(pun.x,pun.y);i=indp(pun.naam[0]);kl=colp[i%12]
    C.create_oval(X-2,Y-2,X+2,Y+2,fill=kl) # punt tekenen
    C.create_text(X+5,Y+5,text=pun.naam,fill=kl) # naam bijzetten
def tekengrafr(rech):
    global lr;lr=paramrech(rech.p1,rech.p2).split(',');i=indr(rech.naam[0])
    kl=colr[i%12];lis=[];X,Y=omzet(fr(kmi),gr(kmi));lis.append(X);lis.append(Y)
    X,Y=omzet(fr(kma),gr(kma));lis.append(X);lis.append(Y)
    line = C.create_line(lis,fill=kl) # rechte tekenen
    X,Y=omzet(fr(0.5),gr(0.5));C.create_text(X-5,Y+5,text=rech.naam,fill=kl) # naam bijzetten bij k=0.5
def assen():
    nm='X';lis=lijn(xmi,0,xma,0);X,Y=lis[2:4];line = C.create_line(lis,fill='black',arrow='last')
    C.create_text(X-8,Y+8,text=nm,fill='black')  # teken X-as
```

```
    nm='Y';lis=lijn(0,ymi,0,yma);X,Y=lis[2:4];line = C.create_line(lis,fill='black',arrow='last')
    C.create_text(X-8,Y+8,text=nm,fill='black')  # teken Y-as
    tekengrafp(oorsprong)
def grafr_vec():dial('Graf rechte+ricvec',tekengrafr_vec)
def tekengrafr_vec():
    global i,k,l,pos;naam=E6.get();nwk=E8.get()
    if (ord(naam) in rR):
        C.delete(ALL);tekeninit();k=float(E7.get());i =indr(naam);tekengrafr(r[i])
        rx,ry=ricvec(r[i].p1,r[i].p2);tekenvec(rx,ry);krx,kry=k*rx,k*ry
        bmina=r[i].p2.naam+'-'+r[i].p1.naam;C.create_text(X+10,Y-5,text=bmina,fill='grey')
        rx,ry=fr(k),gr(k);tekenvec(rx,ry);C.create_text(X+10,Y-5,text='P',fill='grey')
        tekenvec(r[i].p1.x,r[i].p1.y);tekengrafp(r[i].p1);tekengrafp(r[i].p2)
        tekenstip(0,0,krx,kry);tekenstip(krx,kry,rx,ry)
        ptek='P='+r[i].p1.naam+gt(k)+'*('+bmina+')';pos=1;voegtoe(ptek)
    else:foutm()
def tekenstip(x1,y1,x2,y2):line = C.create_line(lijn(x1,y1,x2,y2),fill='grey',dash=[1,2])
def tekenvec(x,y): line = C.create_line(lijn(0,0,x,y),fill='grey',arrow='last')
def indp(x):return ord(x)-65 # index van punt,rechte,bepalen uit naam
def indr(x):return ord(x)-97
def nieuw():
    global lijstP,lijstR,p,r,ip,ir;lijstP=[];lijstR=[];p=[];r=[];ip,ir,ic=0,0,0
    tex.delete('1.0',END);wisgraf();form.title('Vlakke meetkunde (punten en rechten): ____')
    #return
def wisgraf():global pos;C.delete(ALL);pos=1
def wis():E6.delete(0,len(E6.get()));LP.selection_clear(0,END);LR.selection_clear(0,END)
# AFSTANDEN
def afstand(a,b):return 'Afstand ('+a.naam+','+b.naam+') = '+format(sqrt((a.x-b.x)**2+(a.y-b.y)**2),'.'+nwk+'f')
def afstand2(a,b):return sqrt((a.x-b.x)**2+(a.y-b.y)**2)
def afst2p():dial('Afstand 2 punten',berekenafst2p)
def berekenafst2p():
    p1,p2=E6.get().split(',');checksel()
    if (ord(p1) in rP) and (ord(p2) in rP):
        i1=indp(p1);i2=indp(p2);voegtoe(afstand(p[i1],p[i2]))
    else:foutm()
def afstpr():dial('Afstand punt,rechte',berekenafstpr)
def berekenafstpr():
    lprv=E6.get().split(',');checksel();p1=lprv[0];r1=lprv[1]
    if (ord(p1) in rP) and (ord(r1) in rR):
        ir=indr(r1);ip=indp(p1);u,v,w=cv(r[ir].p1,r[ir].p2)
        d_p_r=abs(u*p[ip].x+v*p[ip].y+w)/sqrt(u*u+v+v*v)
        voegtoe('Afstand('+p[ip].naam+','+r[ir].naam+') = '+format(d_p_r,'.'+nwk+'f'))
    else:foutm()
# HOEKEN
def hoek2p():dial('Hoek 2 punten',berekenh2p)
def berekenh2p():
    global a1,b1,a2,b2;p1,p2=E6.get().split(',')
    if (ord(p1) in rP) and (ord(p2) in rP):
        i1=indp(p1);i2=indp(p2);a1,b1=p[i1].x,p[i1].y;a2,b2=p[i2].x,p[i2].y
        hk=bepaalhoek();voegtoe('Hoek ('+p[i1].naam+','+p[i2].naam+') = '+gms(hk))
    else:foutm()
def bepaalhoek():
    br=(a1*a2+b1*b2)/(sqrt(a1**2+b1**2)*sqrt(a2**2+b2**2));hoekdeg=degrees(acos(br))
    if hoekdeg>90:hoekdeg=180-hoekdeg
    return (hoekdeg)
def hoek2r():dial('Hoek 2 rechten',berekenh2r)
```

```
def berekenh2r():
    global a1,b1,a2,b2;r1,r2=E6.get().split(',')
    if (ord(r1) in rR) and (ord(r2) in rR):
        jr1=indr(r1);jr2=indr(r2);a1,b1=ricvec(r[jr1].p1,r[jr1].p2);a2,b2=ricvec(r[jr2].p1,r[jr2].p2)
        hk=bepaalhoek();voegtoe('Hoek ('+r[jr1].naam+','+r[jr2].naam+') = '+gms(hk))
    else:foutm()
# VERGELIJKINGEN
def ricvec(a,b):return[b.x-a.x,b.y-a.y]
def gt(g):
    if g>=0:sg='+'
    else:sg=''
    return sg+format(g,'.'+nwk+'f')
def pd(u,p):
    pdr=''
    if u!=0:pdr=gt(u)+'*'+p
    return pdr
def paramrech(a,b):rx,ry=ricvec(a,b);return gt(a.x)+pd(rx,'k')+','+gt(a.y)+pd(ry,'k')
def parr():dial('Param.vgl.rechte',berekenparr)
def berekenparr():
    checksel();r1=E6.get()
    if (ord(r1) in rR):j=indr(r1);voegtoe('Parametervergelijking van '+r[j].naam+' is\nx,y=
    '+paramrech(r[j].p1,r[j].p2))
    else:foutm()
def tekenevrech():dial('// door P aan rech',berekev)
def berekev():
    p1,r1=E6.get().split(',')
    if (ord(p1) in rP) and (ord(r1) in rR):
        tekeninit();jr=indr(r1);jp=indp(p1);u,v,w=cv(r[jr].p1,r[jr].p2)
        tekengrafp(p[jp]);tekengrafr(r[jr]);pt=p[jp];rt=r[jr]
        x4,y4=pt.x+rt.p1.x-rt.p2.x,pt.y+rt.p1.y-rt.p2.y; p4=Punt('L',x4,y4);ev=Rechte('l',pt,p4);tekengrafr(ev)
        voegtoe('C.v. // rechte l uit '+p[jp].naam+' aan '+r[jr].naam + ' ('+gt(u)+'*x'+gt(v)+'*y'+gt(w)+'=0) is')
        u,v,w=cv(pt,p4);voegtoe(gt(u)+'*x'+gt(v)+'*y'+gt(w)+'=0')
    else:foutm()
def loodlpr():dial('C.V. loodlijn P rech',berekenloodlpr)
def berekenloodlpr():
    p1,r1=E6.get().split(',')
    if (ord(p1) in rP) and (ord(r1) in rR):
        tekeninit();jr=indr(r1);jp=indp(p1);u,v,w=cv(r[jr].p1,r[jr].p2);wll=u*p[jp].y-v*p[jp].x
        voegtoe('C.v. loodlijn l uit '+p[jp].naam+' op '+r[jr].naam + ' ('+gt(u)+'*x'+gt(v)+'*y'+gt(w)+'=0) is')
        x,y=stelsel2x2(u,v,-w,v,-u,-wll);voegtoe(gt(v)+'*x'+gt(-u)+'*y'+gt(wll)+'=0  Snijpunt = S'+snijp(x,y))
        vp=Punt('S',x,y);ll=Rechte('n',p[jp],vp);tekengrafp(p[jp]);tekengrafr(r[jr]);tekengrafp(vp);tekengrafr(ll)
    else:foutm()
def bissrech():dial('Bissectrice 2 rechten',berekenbissrech)
def berekenbissrech():
    r1,r2=E6.get().split(',');tekeninit()
    if (ord(r1) in rR) and (ord(r2) in rR):
        ir1=indr(r1);ir2=indr(r2);u1,v1,w1=cv(r[ir1].p1,r[ir1].p2);u2,v2,w2=cv(r[ir2].p1,r[ir2].p2)
        voegtoe('Bissectrices van '+r[ir1].naam+' en '+r[ir2].naam+' :');tekengrafr(r[ir1]);tekengrafr(r[ir2])
        if u1*v2-u2*v1!=0:
            s2=sqrt(u2*u2+v2*v2);s1=sqrt(u1*u1+v1*v1)
            u3=s2*u1-s1*u2;v3=s2*v1-s1*v2;w3=s2*w1-
            s1*w2;u4=s2*u1+s1*u2;v4=s2*v1+s1*v2;w4=s2*w1+s1*w2
            voegtoe(gt(u3)+'*x'+gt(v3)+'*y'+gt(w3)+'=0');voegtoe(gt(u4)+'*x'+gt(v4)+'*y'+gt(w4)+'=0')
            x,y=stelsel2x2(u1,v1,-w1,u2,v2,-w2);voegtoe('Snijpunt = S'+snijp(x,y))
            xsp,ysp=x,y;sp=Punt('S',xsp,ysp);tekengrafp(sp)
```

```
            if v3!=0:x=xsp+1;y=(-w3-u3*x)/v3
            else:y=ysp+1;x=-w3/u3
            vb1=Punt('L1',x,y);b1=Rechte('l1',sp,vb1);tekengrafr(b1)
            if v4!=0:x=xsp+1;y=(-w4-u4*x)/v4
            else:y=ysp+1;x=-w4/u4
            vb2=Punt('L2',x,y);b2=Rechte('l2',sp,vb2);tekengrafr(b2)
        else:voegtoe('Evenwijdige rechten')
    else:foutm()
def cv(a,b):u=b.y-a.y;v=-b.x+a.x;w=-b.y*a.x+b.x*a.y;return u,v,w
def carr():dial('C.v.van rechte',drukcv)
def drukcv():
    checksel();r1=E6.get()
    if (ord(r1) in rR):
        ir=indr(r1);u,v,w=cv(r[ir].p1,r[ir].p2)
        voegtoe('Cartesische vergelijking van de rechte '+r[ir].naam+':\n'+gt(u)+'*x'+gt(v)+'*y'+gt(w)+'=0')
    else:foutm()
def cc(p,r):u=-2*p.x;v=-2*p.y;w=p.x**2+p.y**2-r**2;return u,v,w
def snijp(x,y):return '('+format(x,'.'+nwk+'f')+','+format(y,'.'+nwk+'f')+')'
def gegtek():voegtoe('Punten, rechten:');voegtoe(punten);voegtoe(rechten)
def seltek():dial('Lijst:info->tekst',druklijst)
def druklijst():
    lijstgr=E6.get().split(',');lijstinfo=''
    for j in range(0,len(lijstgr)):
        naam=lijstgr[j];soort=ord(naam[0])
        if soort<91:lijstinfo=lijstinfo+str(p[indp(naam)])+'  '
        if soort>96 and soort<123:lijstinfo=lijstinfo+str(r[indr(naam)])+'  '
    voegtoe(lijstinfo)
# SNIJPUNTEN
def snijprr():dial('Snijp. 2 rechten:',berekensnijprr)
def berekensnijprr():
    r1,r2=E6.get().split(',');tekeninit()
    if (ord(r1) in rR) and (ord(r2) in rR):
        ir1=indr(r1);ir2=indr(r2);u1,v1,w1=cv(r[ir1].p1,r[ir1].p2);u2,v2,w2=cv(r[ir2].p1,r[ir2].p2)
        if u1*v2-u2*v1!=0:
            x,y=stelsel2x2(u1,v1,-w1,u2,v2,-w2);tekengrafp(Punt('S',x,y))
            voegtoe('Het snijpunt van '+r[ir1].naam+' en '+r[ir2].naam+' is S('+gt(x)+','+gt(y)+')')
        else:voegtoe('De rechten '+r[ir1].naam+' en '+r[ir2].naam+' zijn evenwijdig')
        tekengrafp(r[ir1].p1);tekengrafp(r[ir1].p2);tekengrafp(r[ir2].p1)
        tekengrafp(r[ir2].p2);tekengrafr(r[ir1]);tekengrafr(r[ir2])
    else:foutm()
# DIALOOG
def dial(tek,com):
    global Lb,fb,E6,fw,rP,rR,grafbas,params,nwk,ci,wkeu,tekvlag
    if ip==0 and ir==0 and ic==0:foutm2()
    else:
        rP=range(65,65+ip);rR=range(97,97+ir)
        fw=Toplevel(form);fw.geometry('30x330');fw.title('Dial');toonlijst()
        Label(fw,text=tek).place(x=0,y=0,width=110)
        E6=Entry(fw,bg='white');E6.place(x=5,y=20,width=95)
        bf=Button(fw,text='Vul in',command=vulin);bf.place(x=5,y=40,height=15,width=40);
        bw=Button(fw,text='Wis',command=wis);bw.place(x=60,y=40,height=15,width=40)
        bd=Button(fw,text='Calc',command=com);bd.place(x=5,y=56,height=15,width=40)
        bh=Button(fw,text='OK',command=lambda:fw.destroy());bh.place(x=60,y=56,height=15,width=40)
    return
def vulin():
```

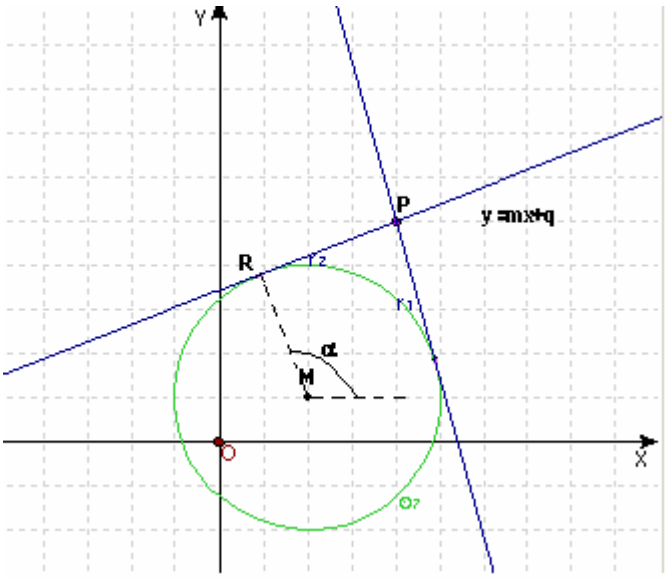
```
    global liP,liR;liP=LP.curselection();liR=LR.curselection()
    le=len(E6.get())
    for i in liP:
       pn=p[i].naam
       if not pn in E6.get():E6.insert(END,','+pn)
    for i in liR:
       rn=r[i].naam
       if not rn in E6.get():E6.insert(END,','+rn)
    if le==0:E6.delete(0,1)
#menu-functies
def men():return Menu(mb,tearoff=0)
def ad(me,lab,co):me.add_command(label=lab,command=co)
def cas(l,m):mb.add_cascade(label=l,menu=m)
# HOOFDPROGRAMMA
colp=['red','salmon','coral','indianred','magenta','violet','lightcoral','tomato','orangered','firebrick','darkorange','darkr
ed']
colr=['blue','cadetblue','powderblue','royalblue','purple','cornflowerblue','cyan','steelblue','skyblue','dodgerblue','lig
htsteelblue','darkblue']
xmi=-10;xma=10;ymi=-10;yma=10;k=1.6;nwk='3';wkeu=0;rkeu=0;pos=1;opstart=0
grafbas=str(xmi)+','+str(xma)+','+str(ymi)+','+str(yma)
rP=range(65,91);rR=range(97,123)
sc=1;p=[];r=[];lijstP=[];lijstR=[];ip=0;ir=0;tekstf=''
oorsprong=Punt('O',0,0);br=1050;ho=525;hc=ho-10;bc=hc;bt=525
form=Tk();form.geometry(str(br)+'x'+str(ho));form.title('Vlakke meetkunde (punten en rechten): ____')
form.resizable(False,False)
C = Canvas(form, bg="white", height=hc, width=bc);C.place(x=2,y=2)
tex=Text(form,bg='grey89',height=29,width=60,wrap=WORD);tex.place(x=bt,y=3)
vsb=Scrollbar(form,orient=VERTICAL,command=tex.yview,width=35)
tex['yscrollcommand']=vsb.set;vsb.pack(side=RIGHT,fill='y')
#menu
mb=Menu(form);fm=men();am=men();vm=men();ex=men();hem=men();gm=men();wm=men()
ad(fm,'pr-bestand openen',lambda:lijstbestanden('openen'))
ad(fm,'Tekst opslaan ',lambda:lijstbestanden('opslaan'))
ad(gm,'Rechte + vectoren',grafr_vec);ad(gm,'Snijpunt 2 rechten',snijprr);ad(gm,'C.v. bissectrices 2
rechten',bissrech)
ad(gm,'C.v. // door punt aan rechte',tekenevrech);ad(gm,'C.v. loodlijn punt op rechte',loodlpr);ad(gm,'Lijst
punten,rechten',graf)
ad(am,'Afstand 2 punten',afst2p);ad(am,'Afstand punt rechte',afstpr)
ad(am,'Hoek 2 punten',hoek2p);ad(am,'Hoek 2 rechten',hoek2r)
ad(vm,'Parametervergelijkingen rechte',parr);ad(vm,'Cart. vergelijking rechte',carr)
ad(wm,'Wis grafiek',wisgraf);ad(wm,'Wis tekst',lambda:tex.delete('1.0',END));ad(wm,'Nieuw',nieuw)
ad(ex,'Instellingen',inst);ad(ex,'Geselecteerde lijst gegevens -->tekst',seltek)
ad(ex,'Volledige lijst gegevens -->tekst',gegtek);ad(hem,'Help',he);ad(hem,'Info',info)
cas('Bestand',fm);cas('Grafiek',gm);cas('Vergelijkingen',vm)
cas('Afstand-hoek',am);cas('Wissen',wm);cas('Extra',ex);cas('Help',hem)
form.config(menu=mb)
inst();fi.withdraw()
form.mainloop()
```

Op blz. 222,223 zie je een schermafdruk van de 3 programma's

133. Berekeningen met cirkels vlakke_meetkunde2

Raaklijn uit een punt aan een cirkel:

In de vlakke meetkunde is de analytische berekening van de raaklijnen aan een cirkel op te lossen met een goniometrische vergelijking $A\cdot\sin\alpha+B\cdot\cos\alpha+C=0$:



Als $M(x_1,y_1)$ het middelpunt is van een cirkel met straal s en we trekken een raaklijn y=mx+q uit een punt $P(x_2,y_2)$ aan de cirkel, dan zijn de coördinaten van het raakpunt:
$R(x_1+s.\cos\alpha,\ y_1+s.\sin\alpha)$.
Omdat de raaklijn RP loodrecht staat op de straal RM, is het scalair product=0.
$[x_2-(x_1+s.\cos\alpha)].s.\cos\alpha+[y_2-(y_1+s.\sin\alpha)].s.\sin\alpha=0$
Stel $y_2-y_1=A$ $x_2-x_1=B$ en deel door s:
$(B-s.\cos\alpha).\cos\alpha+(A-s.\sin\alpha).\sin\alpha=0$

Stel nog s=- C, dan hebben we de goniometrische vergelijking: $A.\sin\alpha+B.\cos\alpha+C=0$ (*)

Is $\alpha\neq\pi$ dan mogen we sin en cos vervangen in functie van $\tan\frac{\alpha}{2}$ geeft:

$$A.\frac{2\tan\frac{\alpha}{2}}{1+\tan^2\frac{\alpha}{2}}+B.\frac{1-\tan^2\frac{\alpha}{2}}{1+\tan^2\frac{\alpha}{2}}+C=0 \quad \text{Stel } \tan\frac{\alpha}{2}=x, \text{ dan is } \alpha=2.\text{atan}(x)$$

$$A.\frac{2x}{1+x^2}+B.\frac{1-x^2}{1+x^2}+C=0 \qquad 2Ax+B(1-x^2)+C(1+x^2)=0$$

$(C-B).x^2+2Ax+C+B=0$

1. Is C-B verschillend 0 dan hebben we:

De discriminant $D=b^2-4ac=4A^2-4(C-B)(C+B)=4A^2-4C^2+4B^2=4\cdot(A^2+B^2-C^2)=4D'$
Is $A^2+B^2>C^2$, dan is D' >0 en hebben we 2 raaklijnen, nl.

$x_1 = \frac{-A+\sqrt{D'}}{C-B}$ en $x_2 = \frac{-A-\sqrt{D'}}{C-B}$

Dit geeft ons 2 α's: $\alpha_1 = 2.\text{atan}(x_1)$, $\alpha_2 = 2.\text{atan}(x_2)$

Daarmee krijgen we de coördinaten van de raakpunten met de cirkel:

Voor de 1ste raaklijn: $(x_1+s.\cos\ \alpha_1, y_1+s.\sin\alpha_1)$

Voor de 2de raaklijn: $(x_1+s.\cos\ \alpha_2, y_1+s.\sin\alpha_2)$

En de vergelijkingen van beide raaklijnen:

$\sin\alpha_1.(y-y_1)+\cos\alpha_1.(x-x_1)-s=0$ en $\sin\alpha_2.(y-y_1)+\cos\alpha_2.(x-x_1)-s=0$

Is D=0, dan hebben we enkel $x_1 = \frac{-A}{C-B}$ $\alpha_1 = 2\text{atan}(x_1)$,

dus 1 raaklijn: het punt P ligt op de cirkelomtrek.

Is D<0, dan zijn er geen raaklijnen: het punt P ligt in de cirkel

Is B=C en $A \neq 0$ dan wordt de vergelijking (*): $A.\sin\alpha+B.\cos\alpha+B=0$

Als B=C=0: $x_2=x_1$ en s=0 : straal=0, geen cirkel: dit mogen we uitsluiten.

$\sin\alpha=(\cos\alpha+1).-B/A$: dit geeft 2 oplossingen

Is $\cos\alpha+1=0$, dan is $\alpha_1=\pi$, $\sin\alpha_1=0$: eerste oplossing

Is $\cos\alpha+1 \neq 0$, dan is $\frac{\sin\alpha}{\cos\alpha+1} = \frac{-B}{A}$ of $\tan(\frac{\alpha}{2}) = \frac{-B}{A}$ waaruit $\alpha_2 = 2.\text{atan}(\frac{-B}{A})$

Is B=C en A=0 dan is $x_2 = x_1 - s$ en $y_2 = y_1$. Het punt P ligt dan op de cirkel voor $\alpha = \pi$

We hebben dan slechts 1 raaklijn.

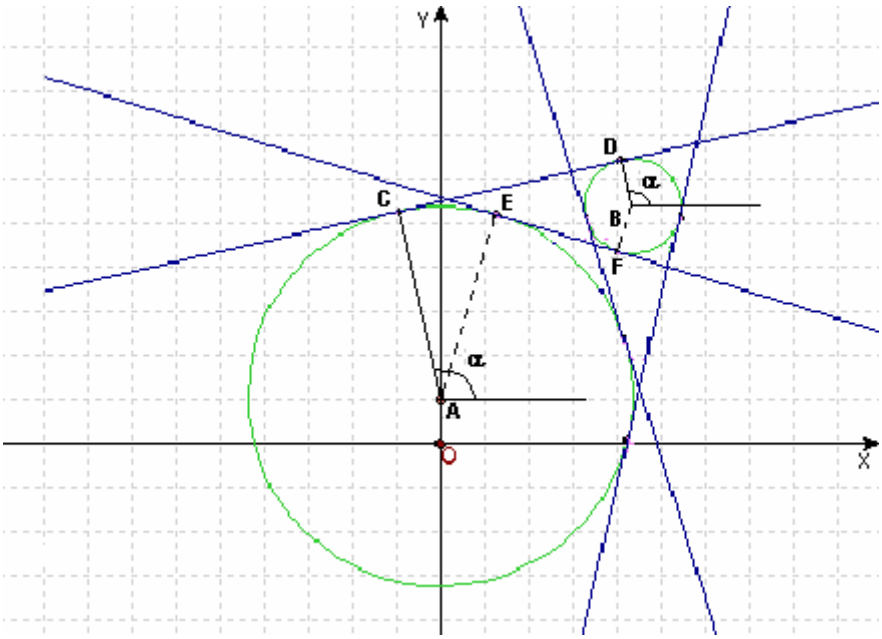
Gemeenschappelijke raaklijnen van 2 cirkels

Stel 2 cirkels:

1: met middelpunt $A(x_1,y_1)$ en straal s_1

2: met middelpunt $B(x_2,y_2)$ en straal s_2

$C(x_1+s_1.\cos\ \alpha, y_1+s_1.\sin\alpha)$ en $D(x_2+s_2.\cos\ \alpha, y_2+s_2.\sin\alpha)$



Uitwendige raaklijnen

De rechte door $C(x_1+s_1.\cos\ \alpha, y_1+s_1.\sin\alpha)$ en $D(x_2+s_2.\cos\ \alpha, y_2+s_2.\sin\alpha)$

staat loodrecht op de stralen in het raakpunt als:
$[(x_2+s_2.\cos\alpha) - (x_1+s_1.\cos\alpha)].\cos\alpha + [(y_2+s_2.\sin\alpha) - (y_1+s_1.\sin\alpha)].\sin\alpha=0$
Vereenvoudiging geeft $(x_2 - x_1).\cos\alpha+(y_2 - y_1).\sin\alpha+(s_2 - s_1)=0$ (1)

Inwendige raaklijnen
Voor raaklijnen die inwendig raken, zijn de hoeken in beide cirkels antisupplementair, dus
α en $\alpha+\pi$ zodat als $E(x_1+s_1.\cos\alpha, y_1+s_1.\sin\alpha)$ (zie in de tekening welke α bedoeld wordt)
dan is $F(x_2+s_2.\cos(\alpha+\pi), y_2+s_2.\sin(\alpha+\pi)) = F(x_2- s_2.\cos\alpha, y_2 - s_2.\sin\alpha)$
Loodrechte stand hebben we nu als
$[(x_2 - s_2.\cos\alpha) - (x_1+s_1.\cos\alpha)].\cos\alpha + [(y_2 - s_2.\sin\alpha) - (y_1+s_1.\sin\alpha)].\sin\alpha=0$
of na vereenvoudiging: $(x_2-x_1).\cos\alpha+(y_2-y_1).\sin\alpha + (-s_2-s_1)=0$ (2)

Zowel voor uitwendige als inwendige raaklijnen:
Als we voor (1): $y_2-y_1=A$ $x_2-x_1=B$ $s_2-s_1=C$
en voor (2): $y_2-y_1=A$ $x_2-x_1=B$ $-s_2-s_1=C$ stellen,
vinden we terug de goniometrische vergelijking: $A.\sin\alpha+B.\cos\alpha+C=0$
Ook nu kunnen we met dezelfde voorwaarden herleiden tot $(C-B).x^2+2Ax+C+B=0$
met $D= 4A^2-4C^2+4B^2$
Is D>0, dan vinden we ook hier: $\alpha_1 = 2.\text{atan}(x_1)$, $\alpha_2 = 2.\text{atan}(x_2)$
Daarmee krijgen we de coördinaten van de raakpunten met beide cirkels:
Voor de 1ste raaklijn:
$(x_1+s_1.\cos\alpha_1, y_1+s_1.\sin\alpha_1)$ en $(x_2\pm s_2.\cos\alpha_1, y_2 \pm s_2.\sin\alpha_1)$
Voor de 2de raaklijn:
$(x_1+s_1.\cos\alpha_2, y_1+s_1.\sin\alpha_2)$ en $(x_2\pm s_2.\cos\alpha_2, y_2 \pm s_2.\sin\alpha_2)$
En de vergelijkingen van beide raaklijnen:
$\cos\alpha_1.(x-x_1)+\sin\alpha_1.(y-y_1)-s_1=0$ en $\cos\alpha_2.(x-x_1)+\sin\alpha_2.(y-y_1)-s_1=0$
(Voor inwendige en uitwendige raaklijnen heb je uiteraard niet dezelfde α_1 en α_2
De rico's (als $\alpha \neq 0$ en $\neq \pi$) van de raaklijnen m_1 en m_2 zijn dan $\frac{-1}{\tan\alpha_1}$ resp. $\frac{-1}{\tan\alpha_2}$
Dan kunnen we de raaklijnen ook schrijven:

$$y=m_1.(x- x_1-s_1.\cos\alpha_1) + y_1+s_1.\sin\alpha_1$$
$$y=m_2.(x- x_1-s_1.\cos\alpha_2) + y_1+s_1.\sin\alpha_2$$

Is D=0, dan hebben we enkel $x_1 = \frac{-A}{C-B}$ $\alpha_1 = 2\text{atan}(x_1)$, dus 1 raaklijn: de cirkels raken elkaar uitwendig of inwendig
Is D<0, dan zijn er geen raaklijnen. De kleinste cirkel lig volledig binnen de grootste.

Deze redenering geldt zowel voor uitwendige als inwendige raaklijnen maar voor uitwendige raaklijnen nemen we C=s1-s2, bij inwendige C= -s1-s2

Enkel als C-B = 0 en $A \neq 0$
(C = B of $x_2-x_1 = s_2-s_1$ bij uitwendige, $x_2-x_1 = -s_2-s_1$ bij inwendige) hebben we:
$A.\sin\alpha+B.\cos\alpha+B=0$ waaruit: $\sin\alpha=(\cos\alpha+1).(-B/A)$: dit geeft 2 oplossingen
Is $\cos\alpha+1 =0$, dan is $\alpha_1 = \pi$, $\sin\alpha_1 =0$: eerste oplossing
Is $\cos\alpha+1 \neq 0$, dan is $\frac{\sin\alpha}{\cos\alpha+1} = \frac{-B}{A}$ of $\tan(\frac{\alpha}{2}) = \frac{-B}{A}$ waaruit $\alpha_2 = 2.\text{atan}(\frac{-B}{A})$
Is B=C en A=0, dan raken de cirkels elkaar uitwendig of inwendig met een verticale raaklijn.

134. Berekeningen voor ruimtemeetkunde

Cart.verg. van een vlak

Om de c.v. ax+by+cz+d=0 van een vlak in de ruimte te bepalen, zullen we met de functies cop en lijn alle coördinaten van de 3 punten van het vlak overbrengen naar een 3x3-matrix (copun). Lossen we het overeenkomstige stelsel op met de functie stelsel(), het rechterlid gelijkgesteld aan 1, dan vinden we de a,b en c van een cartesiaanse vergelijking

ax+by+cz-1=0. Beter nog is de vergelijking ax+by+cz+d=0 met:
d=-det(acop);a=det(m[0]);b=det(m[1]);c=det(m[2])
(dan moeten we voor a, b en c niet delen door de determinant)

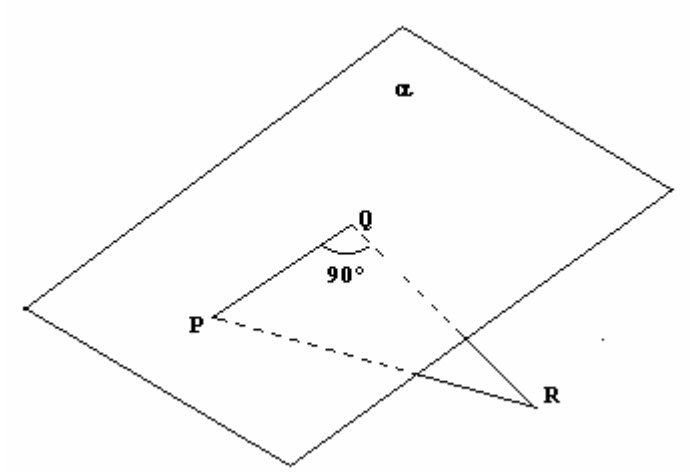
Afstand tussen een punt en een vlak

Formule: $d(P,\alpha) = \dfrac{|a.x_1 + b.y_1 + cz_1 + d|}{\sqrt{a^2 + b^2 + c^2}}$

Voor de afstand tussen een punt en een vlak moeten we in de c.v. van het vlak, x,y en z vervangen door de coördinaten van het punt, de abs nemen, en delen door de vkw.van $a^2+b^2+c^2$:
copun=cop(v[i]);a,b,c,d=stelsel(copun)
d_p_v=(abs(a*p[jp].x+b*p[jp].y+c*p[jp].z+d))/sqrt(a*a+b*b+c*c)

Afstand tussen een punt en een rechte (ruimtemeetkunde)

Voor de afstand van een punt tot een rechte zullen we eerst het loodvlak bepalen vanuit dit punt op de gegeven rechte:



Is P het gegeven punt en R een punt van de gegeven rechte, dan nemen we het loodvlak α die de rechte snijdt in Q. We hoeven de coördinaten van Q niet te berekenen want
$d(P,Q)^2=d(P,R)^2-d(R,\alpha)^2$.
Voor R kunnen we 1 van de gegeven punten van de rechte nemen, bvb. r[jr].p1
We hebben wel de cartesiaanse vergelijking ax+by+cz-d=0 van het vlak α nodig: dit is vrij eenvoudig: voor de a,b,c van de vergelijking nemen we een richtingsvector van de rechte, d vinden we door de coördinaten van P in te vullen.
In het programma ziet dit er als volgt uit:
a,b,c=ricvec(r[jr].p1,r[jr].p2) ; d=a*p[jp].x+b*p[jp].y+c*p[jp].z

De afstand van p1 van de rechte tot het loodvlak is dan:

d_rp1_lv=(abs(a*r[jr].p1.x+b*r[jr].p1.y+c*r[jr].p1.z-d))/sqrt(a*a+b*b+c*c)
De afstand van r[jr].p1 tot het gegeven punt buiten de rechte is
    d_rp1_p=afstand2(r[jr].p1,p[jp])
met def afstand2(a,b):
    return sqrt((a.x-b.x)**2+(a.y-b.y)**2+(a.z-b.z)**2)
De afstand van het punt p tot de rechte is dan
    d_r_p=sqrt(abs(d_rp1_p**2-d_rp1_lv**2)) (Pythagoras)

Hoeken

Hoek tussen 2 punten

Met de hoek tussen 2 punten P en Q wordt de scherpe hoek bedoeld, tussen de rechten OP en OQ :

$$\alpha = \angle(\overrightarrow{OP},\overrightarrow{OQ}) = \text{acos}(\frac{x_1.x_2 + y_1.y_2 + z_1.z_2}{\sqrt{x_1^2 + y_1^2 + z_1^2}.\sqrt{x_2^2 + y_2^2 + z_2^2}})$$

Een algemene definitie kan dan gebruikt worden voor alle hoekberekeningen:
def bepaalhoek():
    br=(a1*a2+b1*b2+c1*c2)/(sqrt(a1**2+b1**2+c1**2)*sqrt(a2**2+b2**2+c2**2))
    hoekdeg=degrees(acos(br))
    if hoekdeg>90:hoekdeg=180-hoekdeg
    return (hoekdeg)

a1,b1,c1=p[i1].x,p[i1].y,p[i1].z ; a2,b2,c2=p[i2].x,p[i2].y,p[i2].z
hk=bepaalhoek()

Hoek tussen 2 rechten

De hoek tussen 2 rechten is de scherpe hoek tussen 2 richtingsvectoren van die rechten:
De hoek tussen $r_1(A(x_1,y_1),B(x_2,y_2))$ en $r_2(C(x_3,y_3),D(x_4,y_4))$ wordt dan gegeven door

$$\alpha = \angle(\overrightarrow{AB},\overrightarrow{CD}) = \text{acos}(\frac{(x_2 - x_1).(x_4 - x_3) + (y_2 - y_1).(y_4 - y_3) + (z_2 - z_1).(z_4 - z_3)}{\sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2 + (z_2 - z_1)^2}.\sqrt{(x_4 - x_3)^2 + (y_4 - y_3)^2 + (z_4 - z_3)^2}})$$

a1,b1,c1=ricvec(r[jr1].p1,r[jr1].p2) ; a2,b2,c2=ricvec(r[jr2].p1,r[jr2].p2)
hk=bepaalhoek()

Hoek tussen rechte en vlak

We nemen de hoek tussen een richtingsvector (a1,b1,c1) van de rechte en de normaalvector (a2,b2,c2) van het vlak. En hiervan nemen we de complementaire hoek

a1,b1,c1=ricvec(r[jr].p1,r[jr].p2) ; a2,b2,c2,d2=stelsel(cop(v[jv]))
hk=90-bepaalhoek()

Hoek tussen 2 vlakken

We nemen de hoek tussen de 2 normaalvectoren (a1,b1,c1) en (a2,b2,c2) van beide vlakken.

a1,b1,c1,d1=stelsel(cop(v[iv]));a2,b2,c2,d2=stelsel(cop(v[jv]))
hk=bepaalhoek()

Loodlijn uit punt op vlak

We bepalen eerst de parametervergelijking van de loodlijn uit $P(x_1,y_1)$ op α met vergelijking $ax+by+cz+d=0$:
$x, y, z=x_1+ka, y_1+kb, z_1+kc$. Om de coördinaten van het snijpunt met α te vinden, vullen we x,y,z in de vergelijking van het vlak in: $a(x_1+ka)+b(y_1+kb)+c(z_1+kc)+d=0$
door herschikking: $ax_1+by_1+cz_1+d_1+k(a^2+b^2+c^2)=0$
zodat $k=-(ax_1+by_1+cz_1+d_1)/(a^2+b^2+c^2)$
Als we k invullen in de parametervergelijking van de loodlijn, krijgen we de coördinaten van het snijpunt.

```
copun=cop(v[jv])
a,b,c,d=stelsel(copun)
k=-(a*p[jp].x+b*p[jp].y+c*p[jp].z+d)/(a*a+b*b+c*c)
x=p[jp].x+k*a;y=p[jp].y+k*b;z=p[jp].z+k*c
```

C.V. van de bissectricevlakken van 2 gegeven vlakken

We bepalen eerst de c.v. van beide vlakken v[jv1] en v[jv2]:

```
a1,b1,c1,d1=stelsel(cop(v[jv1]));a2,b2,c2,d2=stelsel(cop(v[jv2]))
```

Een punt P(x,y) ligt in een bissectricevlak als z'n afstand tot beide vlakken α en β gelijk is:

$$d(P,\alpha)=\frac{|a_1x+b_1y+c_1z+d_1|}{\sqrt{a_1^{\,2}+b_1^{\,2}+c_1^{\,2}}}=d(P,\beta)=\frac{|a_2x+b_2y+c_2z+d_2|}{\sqrt{a_2^{\,2}+b_2^{\,2}+c_2^{\,2}}}$$

Als we de noemers $=s_1$, resp s_2 stellen, dan geldt:

$$|a_1x+b_1y+c_1z+d_1|.s_2=|a_2x+b_2y+c_2z+d_1|.s_1$$

waaruit de vergelijkingen van de 2 bissectrices volgen

$$(a_1s_2+a_2s_1)x+(b_1s_2+b_2s_1)y+(c_1s_2+c_2s_1)z+(d_1s_2+d_2s_1)=0$$
$$(a_1s_2-a_2s_1)x+(b_1s_2-b_2s_1)y+(c_1s_2-c_2s_1)z+(d_1s_2-d_2s_1)=0$$

of

$$a_3x+b_3y+c_3z+d_3=0$$
$$a_4x+b_4y+c_4z+d_4=0$$

```
s1=sqrt(a1*a1+b1*b1+c1*c1) ; s2=sqrt(a2*a2+b2*b2+c2*c2)
a3=a1*s2+a2*s1 ; b3=b1*s2+b2*s1 ; c3=c1*s2+c2*s1 ; d3=d1*s2+d2*s1;
a4=a1*s2-a2*s1;b4=b1*s2-b2*s1;c4=c1*s2-c2*s1;d4=d1*s2-d2*s1
```

Snijlijn van 2 vlakken

Om de snijlijn van 2 vlakken α: $a_1x+b_1y+c_1z+d_1=0$ en β : $a_2x+b_2y+c_2z+d_2=0$ te bepalen, zoeken we 2 snijpunten, dus oplossingen van het stelsel van deze vergelijkingen.
Is $a_1b_2-a_2b_1\neq 0$ dan kiezen we 2 waarden voor z, bvb. 1 en -1 en lossen we het stelsel op naar x en y:

$$\begin{cases}a_1x+b_1y=-c_1-d_1\\a_2x+b_2y=-c_2-d_2\end{cases}\text{ en }\begin{cases}a_1x+b_1y=c_1-d_1\\a_2x+b_2y=c_2-d_2\end{cases}.$$

We vinden dan 2 punten $U(x_1,y_1,1)$ en $V(x_2,y_2,-1)$ waarmee we parametervergelijkingen van de snijlijn bepalen (en dus ook de grafiek kunnen tekenen)
Is $a_1b_2-a_2b_1=0$ dan zijn er 2 andere mogelijkheden die we op een gelijkaardige manier oplossen, namelijk als $a_1c_2-a_2c_1\neq 0$ of ten slotte als $b_1c_2-b_2c_1\neq 0$

Een rechte $r(P_1,P_2)$ doorlopen met een (voortbrengende) vector

$$\vec{P}=\vec{P}_1+k.(\vec{P}_2-\vec{P}_1)$$

Dit menu-item toont de grafiek van de rechte r, de vector $\vec{P}$, de richtingsvector $\vec{P}_2-\vec{P}_1$ alsook de rol van de parameter k bij het doorlopen van de rechte.

Een vlak v(P_1,P_2,P_3) doorlopen met een (voortbrengende) vector

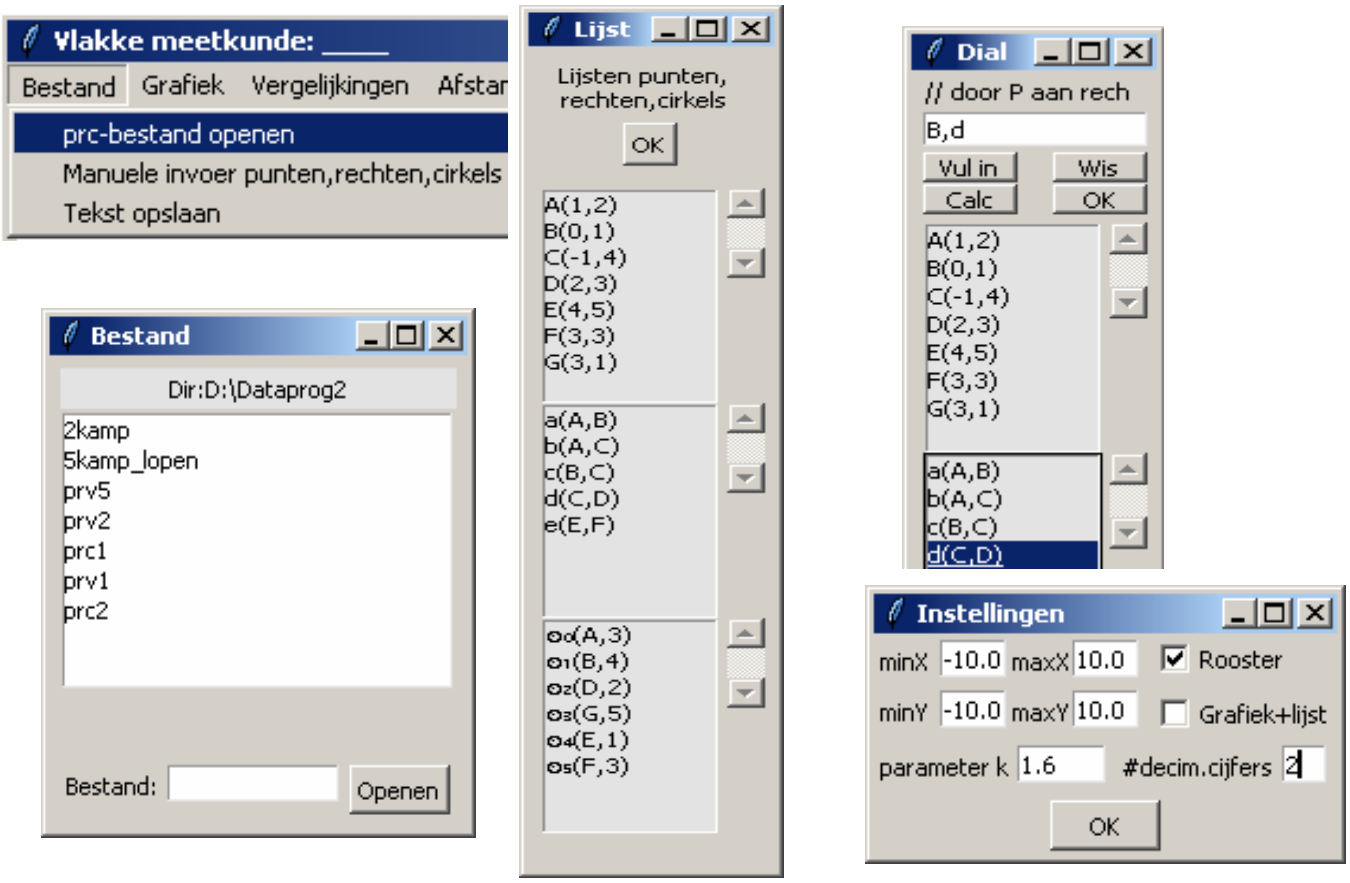
$$\vec{P} = \vec{P}_1 + k.(\vec{P}_2 - \vec{P}_1) + l(\vec{P}_3 - \vec{P}_1)$$

Dit menu-item toont de grafiek van het vlak v, de vector $\vec{P}$, de richtingsvectoren $\vec{P}_2 - \vec{P}_1$ en $\vec{P}_3 - \vec{P}_1$ alsook de rol van de parameters k en l bij het doorlopen van het vlak.

Deze 2 menu–items zijn toegevoegd om een beter inzicht te krijgen hoe de vergelijkingen van rechte of vlak bepaald worden. (Ook in de vlakke meetkunde kunnen parametervergelijkingen van de rechte op deze manier bijdragen tot inzicht)

Aan de slag met deze programma's:

Om aan de slag te gaan met één van de drie programma's kies je eerst voor Bestand – pr (of prc of prv) openen: in het nieuwe venster zie je dan een lijstje van bestanden, waar je een keuze doet: kies het juiste soort bestand (pr.. of prc.. voor vlakke meetkunde, prv.. voor ruimtemeetkunde). Deze bestanden heb je vooraf gemaakt met Excel en opgeslaan als csv-bestand (comma separated values) Je krijgt onmiddellijk 2 of 3 lijsten met de gegevens van punten, rechten (en cirkels of vlakken) te zien, zoals je die hebt ingevoerd in Excel, maar nu wel ook voorzien van een naam.



Deze lijsten zijn hier enkel een bevestiging dat je de gegevens hebt ingeladen: tik dus OK, en je kan starten met de menu's. Telkens als je hierin een keuze maakt, wordt de lijst met gegevens getoond in een **dial**oog-venster en krijg je gelegenheid om de gevraagde gegevens in te vullen: je klikt op een gegeven en de knop **Vul in**. Vul in de juiste volgorde de gevraagde gegevens in: wordt er bijvoorbeeld gevraagd om een punt en een rechte in te vullen, doe dit niet omgekeerd, dwz. eerst een punt en dan een rechte. Met **Calc** wordt het menu-item uitgevoerd, soms enkel in het tekstvenster, soms enkel in het grafisch venster, soms gelijktijdig in beide. **Wis**, wist enkel het invoervenster. Om de grafiek of de tekst te wissen, heb je de menu-items **Wissen - Wis grafiek** of **Wissen - Wis Tekst**

Je kan aan de ingevoerde gegevens ook nog **manueel gegevens toevoegen**. Dit is de 2de optie van het menu **Bestand**. Met de 3de optie **Bestand Tekst opslaan** kun je de tekst uit het rechterdeel opslaan.

Met het menu **Extra – instellingen** kan je de uitvoer naar grafisch en tekstvenster aanpassen aan je eigen keuze. Deze instellingen worden doorgegeven aan een checksel() definitie, die vóór elke berekening wordt opgeroepen.

De betekenis van de meeste instellingen zijn vanzelfsprekend.

De parameter **k** wordt gebruikt bij de grafische voorstelling van een rechte door middel van een voortbrengende vector $\vec{P} = \vec{A} + k.(\vec{B} - \vec{A})$

De parameters **k** en **l** worden gebruikt bij de grafische voorstelling van een vlak door middel van een voortbrengende vector $\vec{P} = \vec{A} + k.(\vec{B} - \vec{A}) + l.(\vec{C} - \vec{A})$

Grafiek + lijst : als er reeds een grafiek getekend is, kan je nog een lijst van elementen toevoegen.
Met het menu **Extra** kan je ook **geselecteerde gegevens** of **alle gegevens** overbrengen naar het tekstvenster.
Je zou dit bvb. vooraf kunnen doen, zodat je steeds ziet met welke gegevens de berekeningen gemaakt zijn.
Zeker ook als je de berekeningen in een tekstbestand wil opslaan.
De menu-opties **Wissen – Wis grafiek** , **Wissen – Wis tekst** en **Wis – Nieuw** zijn vanzelfsprekend.
Met de optie **Help** is er ook nog een Helptekst voorzien.

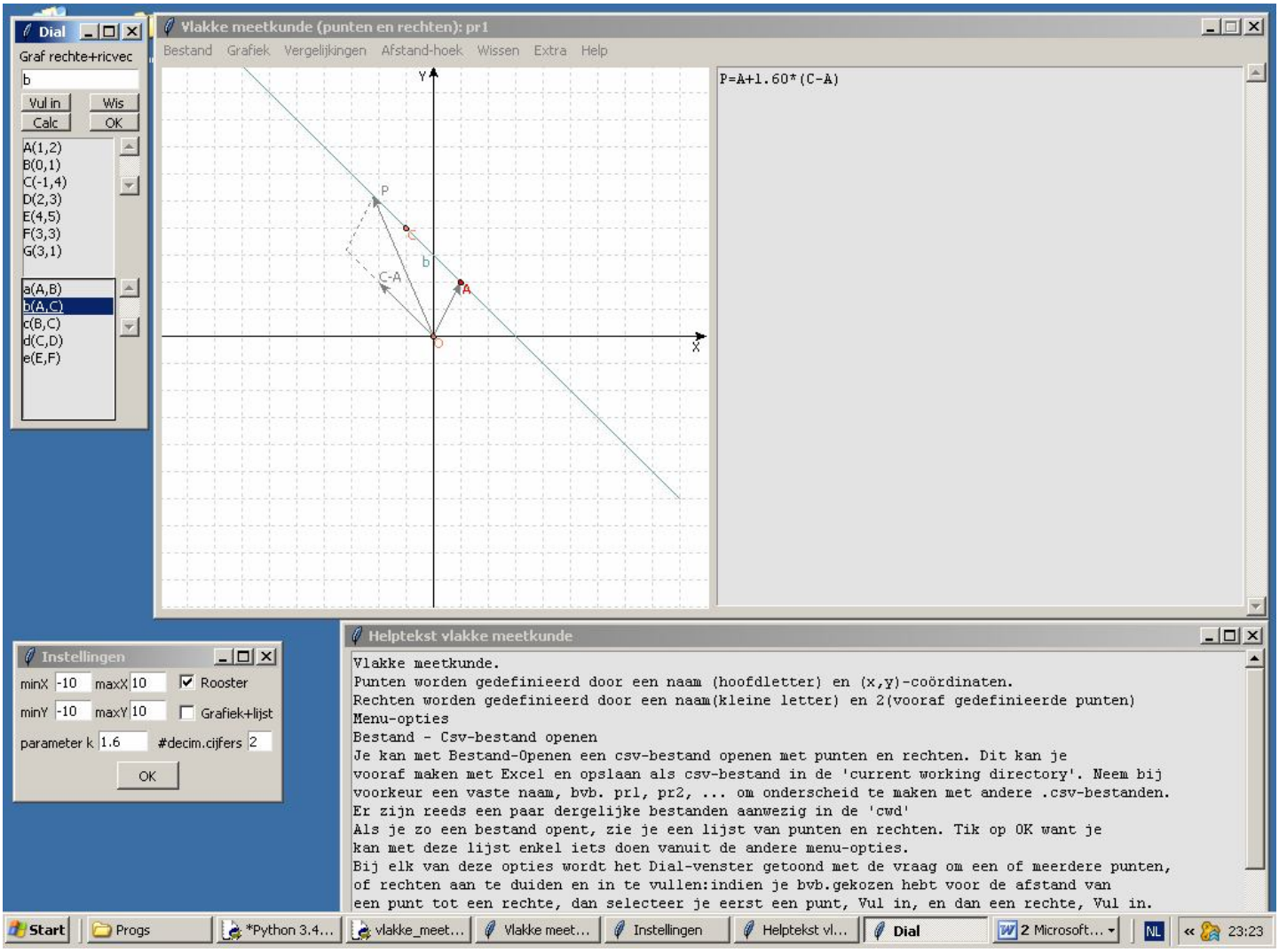
Het programma ruimtemeetkunde zit ongeveer op dezelfde manier in elkaar
Maar het menu **Instellingen** heeft een paar speciale mogelijkheden:

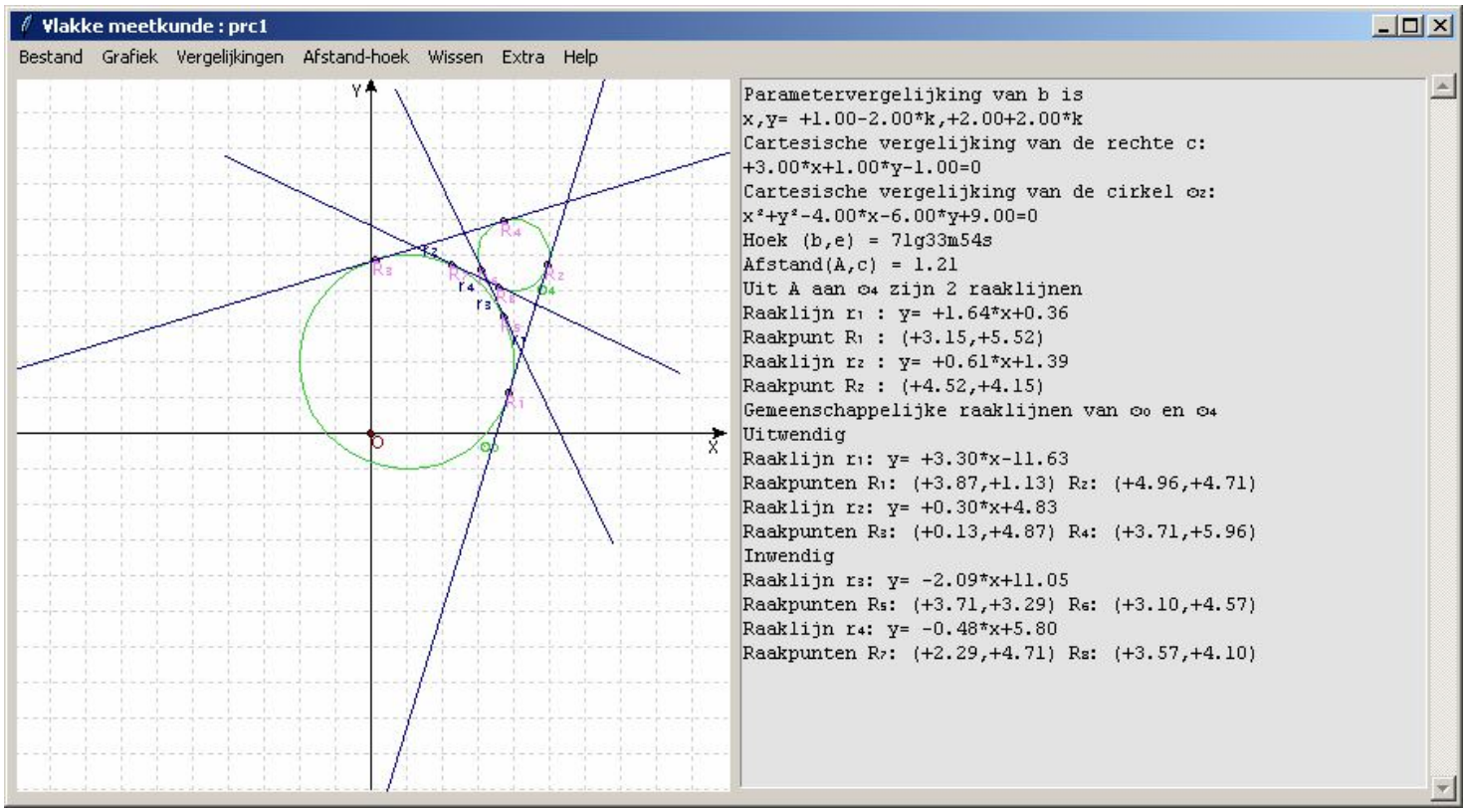
Op de volgende bladzijden worden een paar schermafdrukken getoond van beide programma's.
In het programma ruimtemeetkunde is de draaihoek van XY 135° en de draaihoek YZ=20°
X,Y en Z variëren van -10 tot 10. Rechten en vlakken worden getoond met afmeting 8.
Het raster van het vlak telt 15 onderverdelingen.
⇐ is aangekruist , dus zal elke klik op **Calc** in het **dial**-venster de grafiek 5° naar links doen draaien.
Vlakken worden voorgesteld door gerasterde vierhoeken. Om de grafiek een ruimtelijk gevoel te geven, zijn de opties ⇐ en ⇓ zeker interessant. Je kan de grafiek helemaal laten ronddraaien.

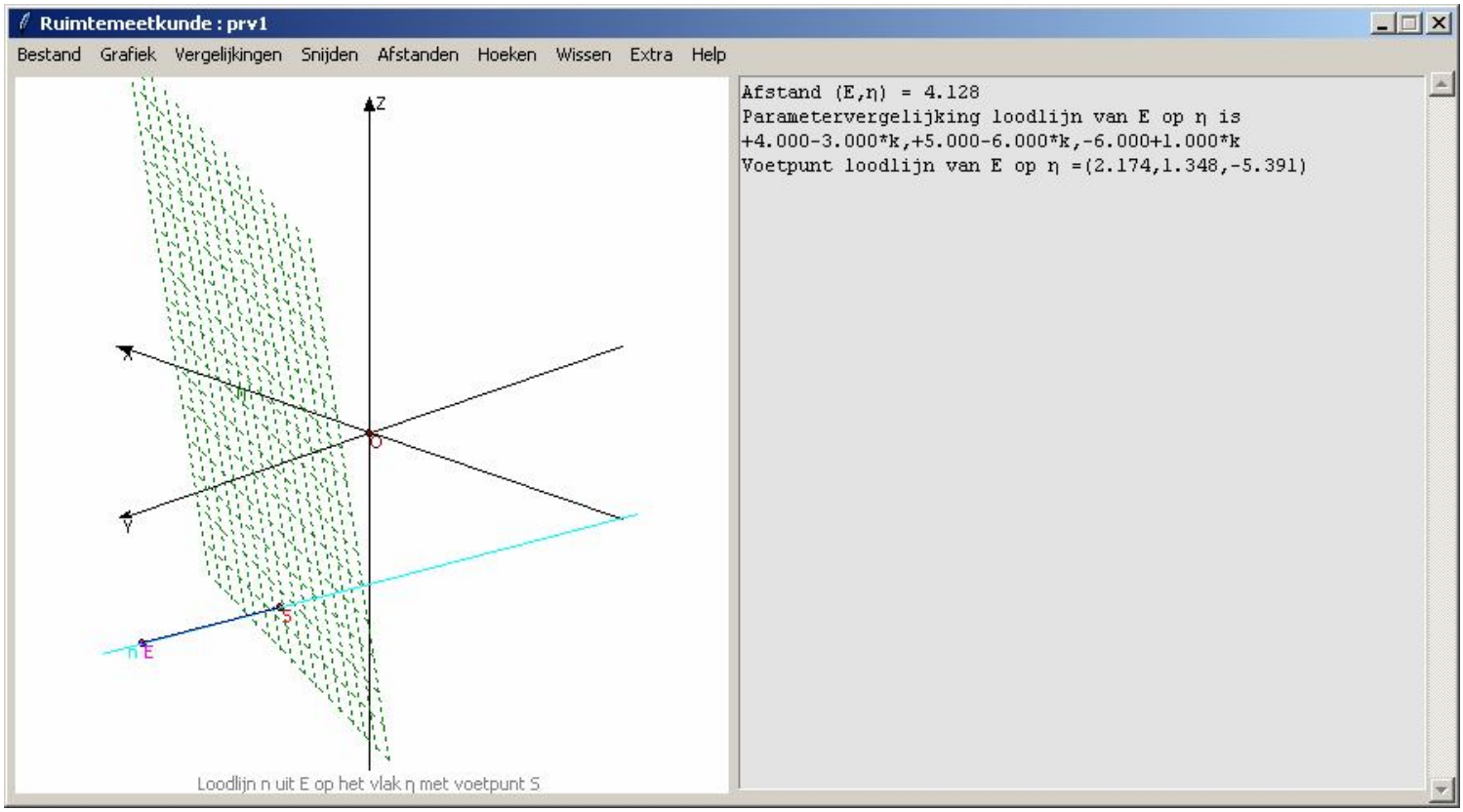
Opmerkingen:
-Ik heb de listings van de 2 laatste programma's niet opgenomen in deze tekst, en ik heb ook niet alles uitgelegd. De listings vind je terug in de map **programmas**. Ik denk dat de programma's vrij goed leesbaar zijn.
-In de lijst met programma's zitten 2 regressieprogramma's: regressie1_csv werkt in de shell, is heel eenvoudig en werkt met het bijgevoegde csv-'2kamp'-bestand.
Het tweede regressie2_csv werkt met het csv-'5kamp_lopen' bestand.
Het programma vlakke_meetkunde werkt met de csv-bestanden prc1 en prc2 (**p**unt **r**echte **c**irkel)
Het programma ruimte_meetkunde werkt met de csv-bestanden prv1 en prv2 (**p**unt **r**echte **v**lak)
Je kan met Excel oneindig veel van deze bestanden maken, als je dezelfde structuur behoudt.
Denk er aan: als je (bij vlakke of ruimtemeetkunde) in Excel een cel laat beginnen met een -teken, bvb. -3,2,-5 , dan laat je het -teken voorafgaan, door een ' , dus '-3,2,-5.
-Je kan beide programma's met nieuwe opties uitbreiden:
Bvb. bij vlakke meetkunde: driehoeken, en hiervan dan ingeschreven of omgeschreven cirkel.

Bij ruimtemeetkunde: bollen (beschreven door een parametervergelijking) , grafiek van bissectricevlakken...

135. Schermafdruk van de 3 programma's







Opmerking: aantonen dat rechten en vlakken kunnen opgebouwd worden met steunvectoren, richtingsvectoren en parameters kan ook eenvoudig met de volgende 2 programma's

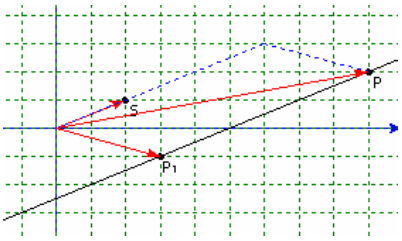
136 Vectoriële, parameter en cartesische vergelijkingen van een rechte in het vlak

ricvecrechTK

In plaats van de grafiek te tekenen van een lineaire functie, kunnen we ook uitgaan van de vectoriële vergelijking van een rechte: $\vec{P} = \vec{P}_1 + k.\vec{S}$. Hierin is $\vec{P}_1(x_1, y_1)$ een steunvector en $\vec{S}(a, b)$ een richtingsvector van de rechte.

Als de parameter k varieert van $-\infty$ tot $+\infty$ zal de vector $\vec{P}$ de rechte doorlopen.

Vb. met $\vec{P}_1(3,-1)$, $\vec{S}(2,1)$ en $k = 3$



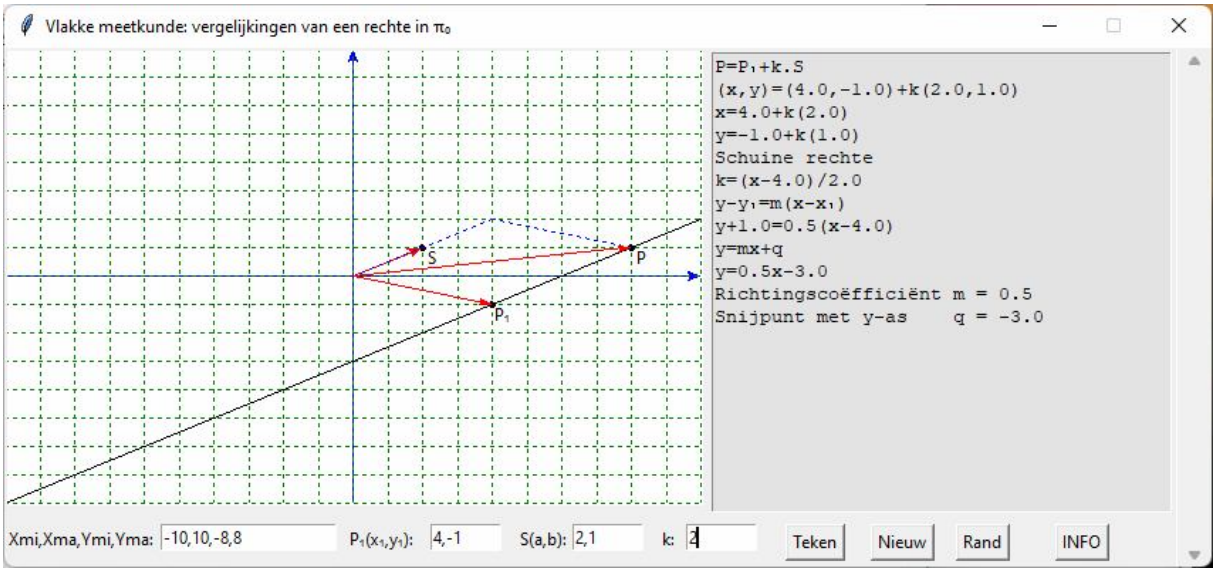
Het volgende programma laat toe om elk van deze basisgegevens te wijzigen: de grafiek wordt dan getekend zoals hierboven, en in het tekstveld worden de vergelijkingen berekend.

Programma:

```
# Vergelijkingen van een rechte in het vlak
from math import sqrt;from tkinter import *
from tkinter import messagebox;from random import randint as ri
def mult(s):
    l=s.split(',');co=[]
    for i in l:co.append(float(i))
    return co
def transx(x):return (x-xmi)*wid/(xma-xmi)
def transy(y):return heigC-heigC*(y-ymi)/(yma-ymi)
def ad(lin):tex.insert(INSERT,lin+'\n')
def sg(a):
    if a>0 or abs(a)<0.0001:t='+'
    else:t=''
    return t
def recfun(x):return b*(x-x1)/a+y1
def appen(x,y):X=transx(x);Y=transy(y);lis.append(X);lis.append(Y)
def punt(x,y,nm):
    col='black';X=transx(x);Y=transy(y);C.create_oval(X-2,Y-2,X+2,Y+2,fill=col)
    C.create_text(X+7,Y+7,text=nm,fill=col)
def teken():
    global xmi,xma,ymi,yma,x1,y1,a,b,k,lis
    xmi,xma,ymi,yma=mult(E1.get());x1,y1=mult(E2.get())
    a,b=mult(E3.get());k=float(E4.get());C.delete(ALL)
#assen
    X=transx(0);line = C.create_line(X,0,X,heigC,fill='blue',arrow='first')
    Y=transy(0);line = C.create_line(0,Y,wid,Y,fill='blue',arrow='last')
# rooster
    for x in range(int(xmi),int(xma+1)):
        X=transx(x);line = C.create_line(X,0,X,heigC,fill='green',dash=[1,2])
    for y in range(int(ymi),int(yma+1)):
        Y=transy(y);line = C.create_line(0,Y,wid,Y,fill='green',dash=[1,2])
# steunvector P1(x1,y1),richtingsvector S(a,b) en algemene vector P(x1+k.a,y1+k.b)
    punt(x1,y1,'P\u2081');punt(a,b,'S');punt(x1+k*a,y1+k*b,'P')
# rechte
    lis=[]
    if not (a==0 and b==0):
        if a!=0:appen(xmi,recfun(xmi));appen(xma,recfun(xma))
        else:appen(x1,ymi);appen(x1,yma)
```

```
        line = C.create_line(lis)
# vectoren
        lis=[];appen(0,0);appen(x1,y1);line = C.create_line(lis,arrow='last',fill='red')
        lis=[];appen(0,0);appen(a,b);line = C.create_line(lis,arrow='last',fill='red')
        lis=[];appen(0,0);appen(k*a,k*b);line = C.create_line(lis,dash=[1,2],fill='blue')
        lis=[];appen(x1+k*a,y1+k*b);appen(k*a,k*b);line = C.create_line(lis,dash=[1,2],fill='blue')
        lis=[];appen(0,0);appen(x1+k*a,y1+k*b);line = C.create_line(lis,arrow='last',fill='red')
# omzetten naar cartesische vergelijking
    tex.delete('1.0',END)
    ad('P=P\u2081+k.S');ad('(x,y)=('+str(x1)+','+str(y1)+')+k('+str(a)+','+str(b)+')')
    ad('x='+str(x1)+'+k('+str(a)+')'); ad('y='+str(y1)+'+k('+str(b)+')');
    if a!=0:
        if b!=0:
            ad('Schuine rechte')
            ad('k=(x'+sg(-x1)+str(-x1)+')/'+str(a))
            m=b/a;q=y1-b*x1/a
            ad('y-y\u2081=m(x-x\u2081)')
            ad('y'+sg(-y1)+str(-y1)+'='+str(round(m,3))+'(x'+sg(-x1)+str(-x1)+')')
            ad('y=mx+q')
            ad('y='+str(round(m,3))+'x'+sg(q)+str(round(q,3)))

        else:ad('Horizontale rechte');ad('y='+str(y1));m=0;q=y1
        ad('Richtingscoëfficiënt m = '+str(round(m,3)))
        ad('Snijpunt met y-as    q = '+str(round(q,3)))
    else:
        if b!=0:ad('Verticale rechte');ad('x='+str(x1))
        else:ad('Geen rechte.S(0,0)=geen richtingsvector')
def help():
    helstr='Grafiek, vectoriële, parameter- en cartesische vergelijkingen'
    helstr=helstr+' van een rechte. m=richtingscoëfficiënt,q=snijpunt Y-as\n P\u2081=steunvector,
S=richtingsvector'
    helstr=helstr+', P=willekeurige vector die de rechte doorloopt als'
    helstr=helstr+' k varieert van -\u221E naar +\u221E.'
    h=messagebox.showinfo('INFO:',helstr)
def nieuw():
    for en in [E1,E2,E3,E4]:en.delete(0,len(en.get()))
    C.delete(ALL);tex.delete('1.0',END)
def ran():
    nieuw()
    xmi,xma=-ri(5,15),ri(5,15);ymi,yma=-ri(4,10),ri(4,10)
E1.insert(0,str(xmi)+','+str(xma)+','+str(ymi)+','+str(yma))
    x1,y1=ri(-3,3),ri(-3,3);E2.insert(0,str(x1)+','+str(y1));a=0;b=0
    while a==0 and b==0:a,b=ri(-3,3),ri(-3,3)
    E3.insert(0,str(a)+','+str(b));k=ri(-3,3);E4.insert(0,str(k))
    return
# hoofdprogramma
wid=490;heig=365;heigC=heig-45;hoInv=heig-30
form=Tk();form.geometry(str(850)+'x'+str(heig));form.title('Vlakke meetkunde: vergelijkingen van een rechte in
\u03C0\u2080')
form.resizable(False,False)
C = Canvas(form, bg="white", height=heigC, width=wid);C.place(x=0,y=0)
L1=Label(form,text='Xmi,Xma,Ymi,Yma:',);L1.place(x=0,y=hoInv)
E1=Entry(form);E1.place(x=110,y=hoInv);E1.insert(0,'-10,10,-8,8')
L2=Label(form,text='P\u2081(x\u2081,y\u2081):');L2.place(x=240,y=hoInv)
E2=Entry(form);E2.place(x=300,y=hoInv,width=50);E2.insert(0,'3,-1')
```

```
L3=Label(form,text='S(a,b):',);L3.place(x=360,y=hoInv)
E3=Entry(form);E3.place(x=400,y=hoInv,width=50);E3.insert(0,'2,1')
L4=Label(form,text='k:',);L4.place(x=460,y=hoInv)
E4=Entry(form);E4.place(x=480,y=hoInv,width=50);E4.insert(0,'3')
tex=Text(form,bg='grey89',height=20,width=40,wrap=WORD);tex.place(x=wid+8,y=3)
vsb=Scrollbar(form,orient=VERTICAL,command=tex.yview,width=25)
tex['yscrollcommand']=vsb.set;vsb.pack(side=RIGHT,fill='y')
B1=Button(form,text='Teken',command=teken);B1.place(x=550,y=hoInv)
B2=Button(form,text='Nieuw',command=nieuw);B2.place(x=610,y=hoInv)
B3=Button(form,text='Rand',command=ran);B3.place(x=670,y=hoInv)
B4=Button(form,text='INFO',command=help);B4.place(x=740,y=hoInv)
form.mainloop()
```



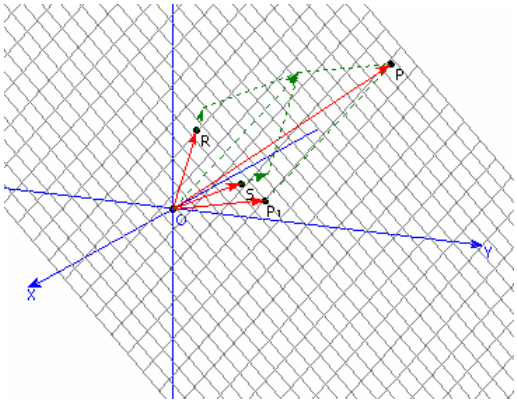
137. Vectoriële, parameter en cartesische vergelijkingen van rechten en vlakken in de ruimte.

ricvecrechvlakTK

Het voordeel van de aanpak uit het vorige programma, is dat we dit kunnen veralgemenen naar vergelijkingen van rechten en vlakken in de ruimte.

De vectoriële vergelijking van een rechte blijft nog steeds $\vec{P} = \vec{P}_1 + k.\vec{R}$, de vectoriële vergelijking van een vlak is $\vec{P} = \vec{P}_1 + k.\vec{R} + l.\vec{S}$ met $\vec{P}_1(x_1, y_1, z_1)$ een steunvector van rechte of vlak, $\vec{R}(a,b,c)$ een richtingsvector van een rechte terwijl $\vec{R}(a,b,c)$ en $\vec{S}(d,e,f)$ richtingsvectoren van een vlak zijn. Als k en l parameters zijn die variëren van $-\infty$ tot $+\infty$,zal het punt $\vec{P}(x,y,z)$ de rechte, respectievelijk het vlak doorlopen.

Vb. met $\vec{P}_1(2,3,1)$, $\vec{R}(1,1,2), \vec{S}(1,2,1)$ $k = 1.3$ en $l = 1.4$



Het volgende programma laat toe om elk van deze basisgegevens te wijzigen. Het is bovendien ook mogelijk om de grafiek rond de draaien zodat we ons een beter ruimtelijk inzicht kunnen vormen.

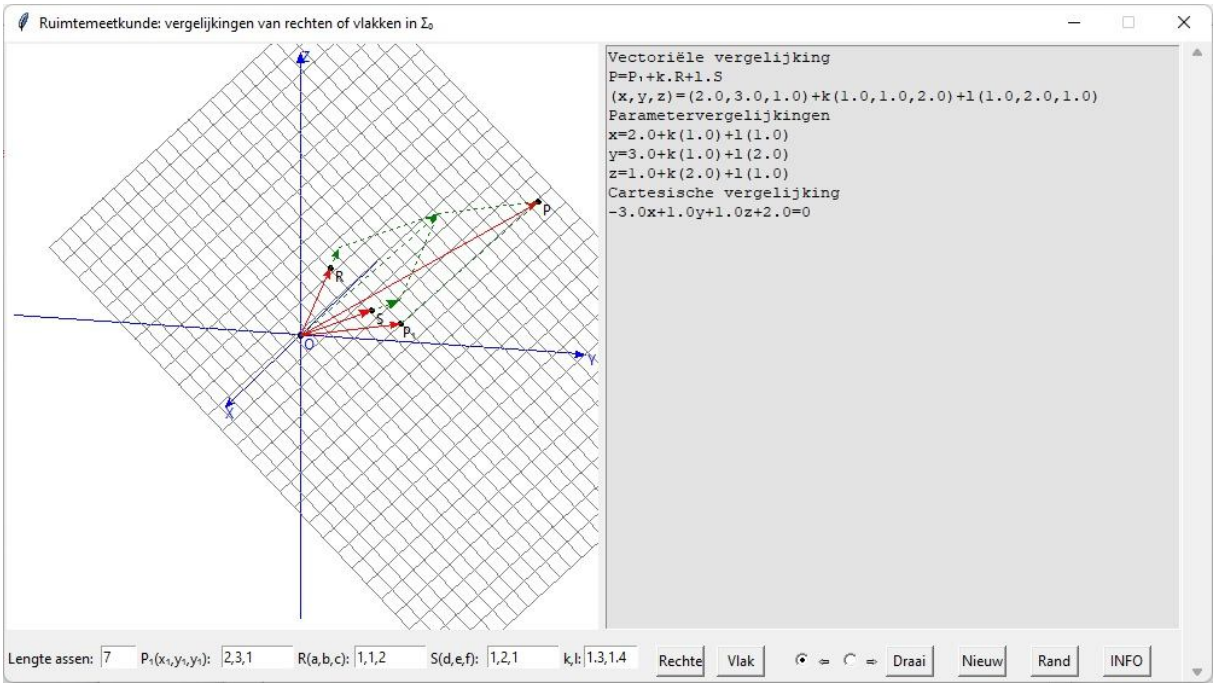
Programma:

```
# Vergelijkingen van rechte en vlak in de ruimte
from math import *;from tkinter import *;from tkinter import messagebox;from random import randint as ri
def mult(s):
    l=s.split(',');co=[]
    for i in l:co.append(float(i))
    return co
def transx(x):return (x-xmi)*wid/(xma-xmi)
def transy(y):return heigC-heigC*(y-ymi)/(yma-ymi)
def det(mat):
    le=len(mat)
    if le==2:d=mat[0][0]*mat[1][1]-mat[0][1]*mat[1][0]
    else:
        d=0
        for i in range(0,le):
            s=[]
            for k in range(0,le):
                if i!=k:s.append(mat[k][1:])
            d=d+mat[i][0]*(-1)**i*det(s)
    return d
def ad(lin):tex.insert(INSERT,lin+'\n')
def v(x):
    if x>=0:sgn='+'
    else:sgn=''
    return sgn+format(x,'.2f')
def sg(a):
    if a>0 or abs(a)<0.0001:t='+'
    else:t=''
    return t+str(a)
def recfun(x):return b*(x-x1)/a+y1
def appen(x,y,z):X,Y=omzet(x,y,z);lis.append(X);lis.append(Y)
def punt(x,y,z,nm):
    col='black';X,Y=omzet(x,y,z);C.create_oval(X-2,Y-2,X+2,Y+2,fill=col)
    C.create_text(X+7,Y+7,text=nm,fill=col)
# definities i.v.m. GRAFIEKEN
def hoek(x,y):
    if x==0 and y>=0:c=0
    elif x==0 and y<0:c=pi
    elif y==0 and x>0:c=pi/2
    elif y==0 and x<0:c=3*pi/2
    else:
        c=atan(x/y)
        if y<0:c=c+pi
    return(c)
def omzet(x,y,z):
    c=hoek(x,y)
    r=sqrt(x*x+y*y);x1=sin(ah+c)*r
    d=hoek(x1,z)
    r1=sqrt(x1*x1+z*z)
    X=transx(cos(ah+c)*r);Y=transy(cos(bh+d)*r1)
```

```
    return [X,Y]
def asteken(nm,x,y,z):
    lis=[]
    X,Y=omzet(-x,-y,-z);lis.append(X);lis.append(Y)
    X,Y=omzet(x,y,z);lis.append(X);lis.append(Y)
    line = C.create_line(lis,fill='blue',arrow='last')
    C.create_text(X+4,Y+4,text=nm,fill='blue')  # x,y,z bij assen schrijven
def grafinit():
    global axlen,xmi,ymi,zmi,xma,yma,zma,ah,bh,x1,y1,z1,a,b,c,d,e,f,k,l,lis
    axlen=float(E1.get());x1,y1,z1=mult(E2.get());a,b,c=mult(E3.get());d,e,f=mult(E4.get());k,l=mult(E5.get())
    xma,yma,zma=axlen,axlen,axlen;xmi,ymi,zmi=-axlen,-axlen,-axlen
    ah=radians(drxy);bh=radians(dryz)
    C.delete(ALL)
#assen + oorsprong
    asteken('X',axlen,0,0);asteken('Y',0,axlen,0);asteken('Z',0,0,axlen)
    X,Y=omzet(0,0,0);
    C.create_oval(X-2,Y-2,X+2,Y+2,fill='blue') # punt tekenen
    C.create_text(X+7,Y+7,text='O',fill='blue') # naam bijzetten
def grafrechte():
    global lis,vlak;grafinit();vlak=0
# steunvector P1(x1,y1,z1),richtingsvector R(a,b,c) en algemene vector P(x1+k.a,y1+k.b,z1+k.c)
    punt(x1,y1,z1,'P\u2081');punt(a,b,c,'R');punt(x1+k*a,y1+k*b,z1+k*c,'P')
# teken grafiek rechte + vectoren + stippellijnen
    t=-5;lis=[];appen(x1+t*a,y1+t*b,z1+t*c)
    t=5;appen(x1+t*a,y1+t*b,z1+t*c)
    tek(lis)
    lis=[];appen(0,0,0);appen(x1,y1,z1);tekvec(lis)
    lis=[];appen(0,0,0);appen(x1+k*a,y1+k*b,z1+k*c);tekvec(lis)
    lis=[];appen(0,0,0);appen(a,b,c);tekvec(lis)
    lis=[];appen(a,b,c);appen(k*a,k*b,k*c);tekstipvec(lis)
    lis=[];appen(k*a,k*b,k*c);appen(x1+k*a,y1+k*b,z1+k*c);tekstip(lis)
# tekst
    tex.delete('1.0',END)
    ad('Vectoriële vergelijking');
    ad('P=P\u2081+k.R');ad('(x,y,z)=('+str(x1)+','+str(y1)+','+str(z1)+')+k('+str(a)+','+str(b)+','+str(c)+')')
    ad('Parametervergelijkingen');ad('x='+str(x1)+'+k('+str(a)+')');
    ad('y='+str(y1)+'+k('+str(b)+')');ad('z='+str(z1)+'+k('+str(c)+')');
    ad('Cartesische vergelijking');ad('(x'+sg(-x1)+')/'+str(a)+'='+'(y'+sg(-y1)+')/'+str(b)+'='+'(z'+sg(-z1)+')/'+str(c))
    return
def leng(u,v,w):return sqrt(u*u+v*v+w*w)
def grafvlak():
    global lis,vlak,a,b,c,d,e,f;vlak=2;grafinit()
# grafiek van het vlak
# constructie van 2 richtingsvectoren met lengte 1 die loodrecht op elkaar staan
    noem=((a+d)*d+(b+e)*e+(c+f)*f)
    if noem==0:
        d=2*d;e=2*e;f=2*f
        noem=((a+d)*d+(b+e)*e+(c+f)*f)
    n=((a+d)*a+(b+e)*b+(c+f)*c)/noem
    x2,y2,z2=a+d,b+e,c+f;x3,y3,z3=-a+n*d,-b+n*e,-c+n*f
    leng2=leng(x2,y2,z2);leng3=leng(x3,y3,z3)
    x2,y2,z2=x2/leng2,y2/leng2,z2/leng2
    x3,y3,z3=x3/leng3,y3/leng3,z3/leng3
    tma=7;stap=0.5;t=-tma-stap
#niveaulijnen x2,...
```

```
    while t<tma:
       lis=[];t=t+stap
       appen(x1-tma*x2+t*x3,y1-tma*y2+t*y3,z1-tma*z2+t*z3)
       appen(x1+tma*x2+t*x3,y1+tma*y2+t*y3,z1+tma*z2+t*z3)
       tek(lis)
    t=-tma-stap
#niveaulijnen x3,...
    while t<tma:
      lis=[];t=t+stap
      appen(x1-tma*x3+t*x2,y1-tma*y3+t*y2,z1-tma*z3+t*z2)
      appen(x1+tma*x3+t*x2,y1+tma*y3+t*y2,z1+tma*z3+t*z2)
      tek(lis)
# steunvector P1(x1,y1,z1),richtingsvectoren R(a,b,c), S(d,e,f) en algemene vector P(x1+k.a,y1+k.b,z1+k.c)
    punt(x1,y1,z1,'P\u2081');punt(a,b,c,'R');punt(d,e,f,'S');punt(x1+k*a+l*d,y1+k*b+l*e,z1+k*c+l*f,'P')
# teken grafiek vlak +vectoren +stippellijnen
    lis=[];appen(0,0,0);appen(x1,y1,z1);tekvec(lis)
    lis=[];appen(0,0,0);appen(x1+k*a+l*d,y1+k*b+l*e,z1+k*c+l*f);tekvec(lis)
    lis=[];appen(0,0,0);appen(a,b,c);tekvec(lis)
    lis=[];appen(0,0,0);appen(d,e,f);tekvec(lis)
    lis=[];appen(a,b,c);appen(k*a,k*b,k*c);tekstipvec(lis)
    lis=[];appen(d,e,f);appen(l*d,l*e,l*f);tekstipvec(lis)
    lis=[];appen(k*a,k*b,k*c);appen(k*a+l*d,k*b+l*e,k*c+l*f);tekstip(lis)
    lis=[];appen(l*d,l*e,l*f);appen(k*a+l*d,k*b+l*e,k*c+l*f);tekstip(lis)
    lis=[];appen(0,0,0);appen(k*a+l*d,k*b+l*e,k*c+l*f);tekstipvec(lis)
    lis=[];appen(k*a+l*d,k*b+l*e,k*c+l*f);
appen(x1+k*a+l*d,y1+k*b+l*e,z1+k*c+l*f);tekstip(lis)
    lis=[];appen(x1,y1,z1);appen(x1+k*a+l*d,y1+k*b+l*e,z1+k*c+l*f);
tekstip(lis)
# tekst
    tex.delete('1.0',END)
    ad('Vectoriële vergelijking');ad('P=P\u2081+k.R+l.S')
    ad('(x,y,z)=('+str(x1)+','+str(y1)+','+str(z1)+')+k('+str(a) +','+str(b)+','+str(c)+')+l('+str(d)+','+str(e)+','+str(f)+')')
    ad('Parametervergelijkingen');ad('x='+str(x1)+'+k('+str(a)+') +l('+str(d)+')')
    ad('y='+str(y1)+'+k('+str(b)+')+l('+str(e)+')'); ad('z='+str(z1)+'+k('+str(c)+')+l('+str(f)+')');
    ad('Cartesische vergelijking');u=b*f-c*e;v=-a*f+c*d;w=a*e-b*d;t=-det([[x1,y1,z1],[a,b,c],[d,e,f]])
    ad(str(u)+'x'+sg(v)+'y'+sg(w)+'z'+sg(t)+'=0')
    #else:er=messagebox.showinfo('Fout','Deling door 0')
    return
def draai():
    global drxy;draaihoek=10
    if rich==0:drxy=drxy+draaihoek
    if rich==1:drxy=drxy-draaihoek
    C.delete(ALL)
    if vlak==0:grafrechte()
    else:grafvlak()
    return
def appen(x,y,z):X,Y=omzet(x,y,z);lis.append(X);lis.append(Y)
def tek(li):line = C.create_line(li,fill='grey')
def tekvec(li):line = C.create_line(li,arrow='last',fill='red')
def tekstip(li):line = C.create_line(li,dash=[1,2],fill='green')
def tekstipvec(li):line = C.create_line(li,dash=[1,2],fill='green',arrow='last')
def selec():global rich;rich=var.get()
def info():
    helstr='Grafiek, vectoriële, parameter- en cartesische vergelijkingen'
    helstr=helstr+' van rechte of vlak in de ruimte.\n P\u2081=steunvector,'
```

```
    helstr=helstr+' R is een richtingsvector voor de rechte\n'
    helstr=helstr+' R en S zijn richtingsvectoren voor het vlak.\n'
    helstr=helstr+' P=willekeurige vector die de rechte of het vlak'
    helstr=helstr+' doorloopt als k of/en l variëren van -\u221E naar +\u221E.'
    helstr=helstr+' Vul de Entries in, of klik op Random.'
    helstr=helstr+' Kies voor Rechte of Vlak,nadien eventueel Draai'
    h=messagebox.showinfo('INFO:',helstr)
def nieuw():
    for en in [E1,E2,E3,E4,E5]:en.delete(0,len(en.get()))
    C.delete(ALL);tex.delete('1.0',END);drxy=15
def ran():
    nieuw()
    axlen=ri(4,10);E1.insert(0,str(axlen))
    x1,y1,z1=ri(-3,3),ri(-3,3),ri(-3,3);E2.insert(0,str(x1)+','+str(y1)+','+str(z1))
    a,b,c,d,e,f=0,0,0,0,0,0
    while a==0 and b==0 and c==0:a,b,c=ri(-3,3),ri(-3,3),ri(-3,3)
    while d==0 and e==0 and f==0:d,e,f=ri(-3,3),ri(-3,3),ri(-3,3)
    E3.insert(0,str(a)+','+str(b)+','+str(c));
E4.insert(0,str(d)+','+str(e)+','+str(f))
    k=ri(-15,15)/10;l=ri(-15,15)/10;E5.insert(0,str(k)+','+str(l));drxy=15
    return
def but(xp,yp,t,co):Button(form,text=t,command=co).place(x=xp,y=yp,width=40)
#hoofdprogramma
wid=490;heig=530;heigC=heig-45;hoInv=heig-30;drxy=15;dryz=15
form=Tk();form.geometry(str(1000)+'x'+str(heig));form.resizable(False,False)
form.title('Ruimtemeetkunde: vergelijkingen van rechten of vlakken in Σ\u2080')
C = Canvas(form, bg="white", height=heigC, width=wid);C.place(x=0,y=0)
L1=Label(form,text='Lengte assen:',);L1.place(x=0,y=hoInv)
E1=Entry(form);E1.place(x=80,y=hoInv,width=30);E1.insert(0,'7')
L2=Label(form,text='P\u2081(x\u2081,y\u2081,y\u2081):');L2.place(x=110,y=hoInv)
E2=Entry(form);E2.place(x=180,y=hoInv,width=60);E2.insert(0,'2,3,1')
L3=Label(form,text='R(a,b,c):',);L3.place(x=240,y=hoInv)
E3=Entry(form);E3.place(x=290,y=hoInv,width=60);E3.insert(0,'1,1,2')
L4=Label(form,text='S(d,e,f):',);L4.place(x=350,y=hoInv)
E4=Entry(form);E4.place(x=400,y=hoInv,width=60);E4.insert(0,'1,2,1')
L5=Label(form,text='k,l:',);L5.place(x=460,y=hoInv)
E5=Entry(form);E5.place(x=480,y=hoInv,width=45);E5.insert(0,'1.3,1.4')
tex=Text(form,bg='grey89',height=30,width=59,wrap=WORD); tex.place(x=wid+8,y=3)
vsb=Scrollbar(form,orient=VERTICAL,command=tex.yview,width=25)
tex['yscrollcommand']=vsb.set;vsb.pack(side=RIGHT,fill='y')
rich=0;var = IntVar();pijl=['\u21E6','\u21E8'] #unicode pijlen
for i in range(0,2):
    rb=Radiobutton(form,text=pijl[i],variable=var,value=i,command=selec)
    rb.place(x=650+i*40,y=hoInv)
but(540,hoInv,'Rechte',grafrechte);but(590,hoInv,'Vlak',grafvlak)
but(730,hoInv,'Draai',draai);but(790,hoInv,'Nieuw',nieuw)
but(850,hoInv,'Rand',ran);but(910,hoInv,'INFO',info)
form.mainloop()
```



138. Dubbelklikken in het TK canvas 1 basisdubbelklikTK1

Het is perfect mogelijk om een figuur (veelhoek of kromme) in het canvas te tekenen door met de muis punten te dubbelklikken in het canvas C: de kern hiervan zit in de volgende code:
def cursorco(e):X=e.x;Y=e.y
def punten():C.bind('<Double-Button-1>',cursorco)

X en Y bevatten dan de Canvas- coördinaten van het geklikte punt.
In een eerste voorbeeldje worden deze coördinaten in een tekstveld afgedrukt:

Programma:

```
# Dubbelklikken in het TK canvas
from tkinter import *
def cursorco(e):
    X=e.x;Y=e.y
    if X<afm and Y<afm:
        C.create_oval(X-2,Y-2,X+2,Y+2,fill='black')
        tex.insert(INSERT,'('+str(X)+','+str(Y)+')\n')
def points():C.bind('<Double-Button-1>',cursorco)
# ------ main
sc=10;form=Tk()
xp=40;yp=40;form.geometry(str((xp+15)*sc)+'x'+str((yp+2)*sc))
form.title('Dubbelklikken in het TK canvas 1');afm=(yp-2)*sc
C=Canvas(form,bg='white',height=afm,width=afm);C.place(x=sc,y=0)
tex=Text(form,bg='grey89',height=int(afm*0.06333),width=int(afm/20))
tex.place(x=(yp-1)*sc,y=0);points();form.mainloop()
```

139 Dubbelklikken in het TK canvas 2 basisdubbelklikTK2

Om wiskundig te kunnen werken met deze punten, moeten we ze omzetten naar wiskundige coördinaten, d.w.z. een assenstelsel definiëren, en tekenen. En omzettingsformules maken van Canvas-coördinaten (Xp,Yp) naar wiskundige coördinaten (xp,yp):

```
def transX(Xp):return xmi+Xp*(xma-xmi)/afm
def transY(Yp):return ymi-(Yp-afm)*(yma-ymi)/afm
xp= transX(Xp); yp= transY(Yp)
```

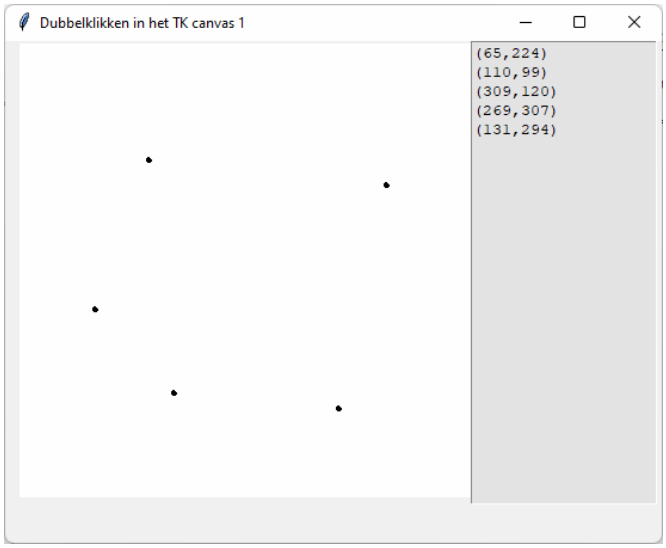
Deze formules zijn gemakkelijk af te leiden uit:
Xp=(xp-xmi)*afm/(xma-xmi) en Yp=afm-afm*(yp-ymi)/(yma-ymi)

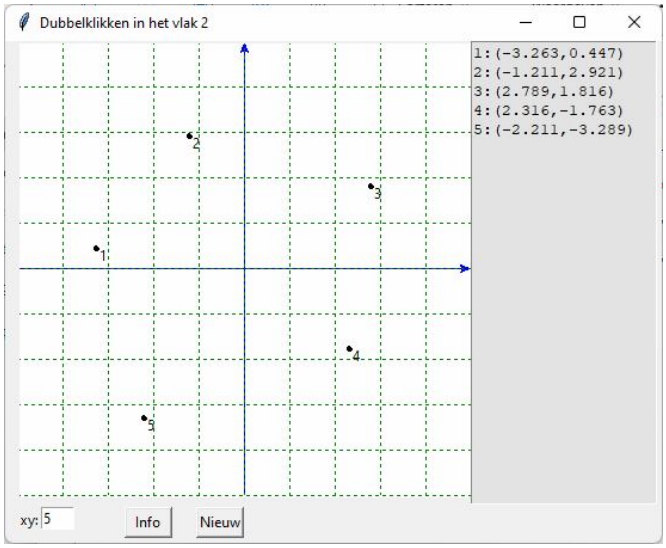
In het volgende programma worden de punten genummerd en rechts -in het tekstveld- krijg je de wiskundige coördinaten

Programma:

```
# Dubbelklikken in het canvas TK 2
from tkinter import *;from tkinter import messagebox
def mult(s):
    l=s.split(',');co=[]
    for i in l:co.append(float(i))
    return co
def transx(xp):return (xp-xmi)*afm/(xma-xmi)
def transy(yp):return afm-afm*(yp-ymi)/(yma-ymi)
def transX(Xp):return xmi+Xp*(xma-xmi)/afm
def transY(Yp):return ymi-(Yp-afm)*(yma-ymi)/afm
#------- grafiek vlakke meetkunde
def assen():
# assen
    X=transx(0);C.create_line(X,0,X,afm,fill='blue',arrow='first')
    Y=transy(0);C.create_line(0,Y,afm,Y,fill='blue',arrow='last')
# rooster
    for x in range(int(xmi),int(xma+1)):
        X=transx(x);C.create_line(X,0,X,afm,fill='green',dash=[1,2])
    for y in range(int(ymi),int(yma+1)):
        Y=transy(y);C.create_line(0,Y,afm,Y,fill='green',dash=[1,2])
# punten vastleggen na dubbelklik
def cursorco(e):
    global X1,Y1,i;X=e.x;Y=e.y
    X1,Y1=[],[]
    if X<afm and Y<afm:
        i=i+1;Xl.append(X);Yl.append(Y)
        C.create_oval(X-2,Y-2,X+2,Y+2,fill='black')
        C.create_text(X+6,Y+6,text=str(i))
        ad(str(i)+':('+format(transX(X),'.3f')+','+format(transY(Y),'.3f')+')\n')
def punten():
    global xmi,xma,ymi,yma,i
    afm=float(E1.get());xmi,ymi=-afm,-afm;xma,yma=afm,afm;i=0
    assen();C.bind('<Double-Button-1>',cursorco)
def help():
    helstr='Als je dubbelklikt in het vlak, kan je rechts de coördinaten aflezen'
    h=messagebox.showinfo('INFO:',helstr)
def nieuw():global Xl,Yl,costr;Xl=[];Yl=[];tex.delete('1.0',END);C.delete(ALL);punten()
# ----- definities formulier
def ad(lin):tex.insert(INSERT,lin)
def lab(t,xp,yp):Label(form,text=t).place(x=sc*xp,y=sc*yp)
def ent(t,xp,yp):lab(t,xp,yp);en=Entry(form,width=4);en.place(x=sc*(xp+1.6),y=sc*yp);return en
# ------ main
sc=15;form=Tk()
xp=40;yp=40;form.geometry(str((xp+15)*sc)+'x'+str((yp+2)*sc))
```

```
form.title('Dubbelklikken in het vlak 2');afm=(yp-2)*sc
C=Canvas(form,bg='white',height=afm,width=afm);C.place(x=sc,y=0)
tex=Text(form,bg='grey89',height=int(afm*0.06333),width=int(afm/20),wrap=WORD)
tex.place(x=(yp-1)*sc,y=0);E1=ent('xy:',1,39);E1.insert(0,'5')
Button(form,text='Info',command=help).place(x=10*sc,y=39*sc,width=3*sc)
Button(form,text='Nieuw',command=nieuw).place(x=14*sc,y=39*sc,width=3*sc)
nieuw();form.mainloop()
```





140 Transformaties in het vlak vshdTK

Gegeven is een centrum C(0,0) en een rechte r door C.
xy= maximum op x- en y-as, α = hoek tussen rechte r en positieve x-as. Assen kan je aan of uit zetten. Dit kan je allemaal wijzigen en met 'Nieuw' krijg je dan deze instelling.
Het is de bedoeling van dit programma om eerst enkele punten te dubbelklikken in het Canvas. Dit zijn dan de 'originele' hoekpunten van een veelhoek. Met de knop 'Orig' wordt de veelhoek door deze punten getekend. Je kan dan kiezen voor een verschuiving, spiegeling, homothetie of draaiing.
v= afstand verschuiving, β is de hoek van de verschuiving. h is de verhouding van de homothetie.
γ is de hoek van de draaiing.
Om op een veelhoek een transformatie toe te passen, volstaat het om een formule te vinden om één hoekpunt te transformeren. Stel een punt P(a,b).
1. Voor een verschuiving over een afstand v met hoek β is het getransformeerde punt $P_1(a+v*\cos(\beta),b+v*\sin(\beta))$

2. Voor een spiegeling t.o.v. de rechte r (met vergelijking $y=m.x=\tan(\alpha).x$)

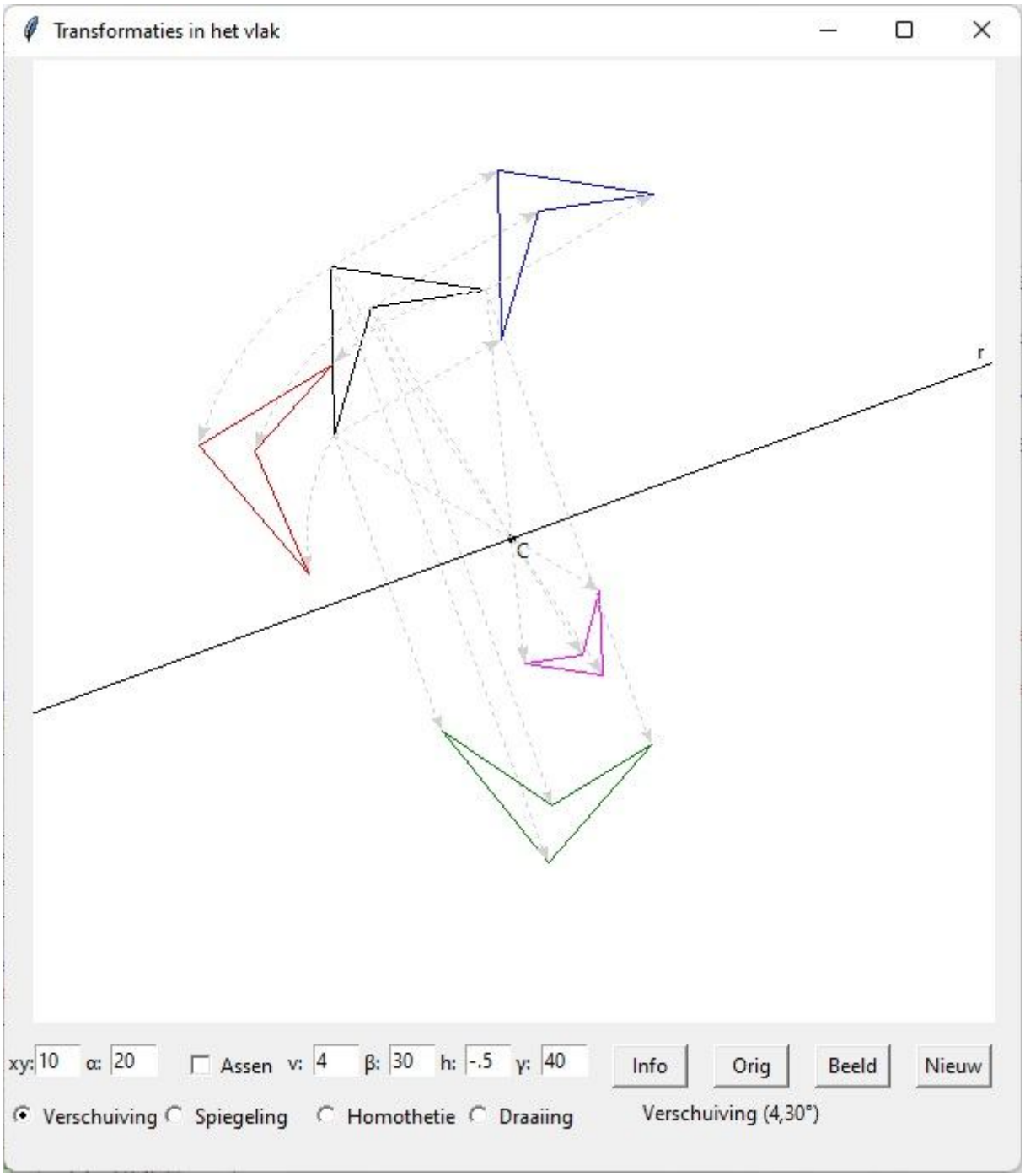
Veronderstel dat het gespiegelde punt $P_2(x,y)$ is. Voor de loodlijn geldt: $y-b=\frac{-1}{m}(x-a)$

waaruit volgt: $x=a+m(b-y)$. (1)
Bovendien is $x^2+y^2=a^2+b^2$ zodat $[a+m(b-y)]^2+y^2=a^2+b^2$
Na herschikking: $(m^2+1)y^2-2m(mb+a)y+m^2b^2+2mba-b^2=0$
$A=(m^2+1)$ $B=-2m(mb+a)$ $C=m^2b^2+2mba-b^2$ som $S=-B/A$
Omdat y-coördinaat origineel $=y_1=b$ zal de y-coördinaat beeld $= y_2 = S-y_1 = S-b$
En met (1) berekenen we $x_2=a+m(b-y_2)$.
(x_2,y_2) zijn dan de coördinaten van het gespiegelde punt P_2.

3. Voor een homothetie met verhouding h is het getransformeerde punt $P_3(h.a,h.b)$

4. Voor een draaiing rond C over een hoek γ nemen we het product van een origineel punt $P\begin{pmatrix} a \\ b \end{pmatrix}$ met de rotatiematrix $\text{rotmat} = \begin{pmatrix} \cos\gamma & -\sin\gamma \\ \sin\gamma & \cos\gamma \end{pmatrix}$

rotmat=[[cos(gamma),-sin(gamma)],[sin(gamma),cos(gamma)]]
x,y=prod(rotmat,[a,b]) (met de productfunctie van 2 matrices)
In xt en yt worden 'wiskundige' x-en y-coördinaten van het origineel opgeslagen; afhankelijk van de keuze k worden met behulp van de vorige formules de 'wiskundige' x-en y-coördinaten van het beeld berekend en opgeslagen in xb en yb.
De functie 'lijnst' zal alle punten met elkaar verbinden en aan het einde nog een lijnst toevoegen naar het eerste punt. XT en YT zijn dan de overeenkomstige 'canvas' coördinaten.
De functie 'streep' zorgt voor een grijze streepjeslijn of streepjeskromme tussen het originele punt en het beeldpunt.



Programma:

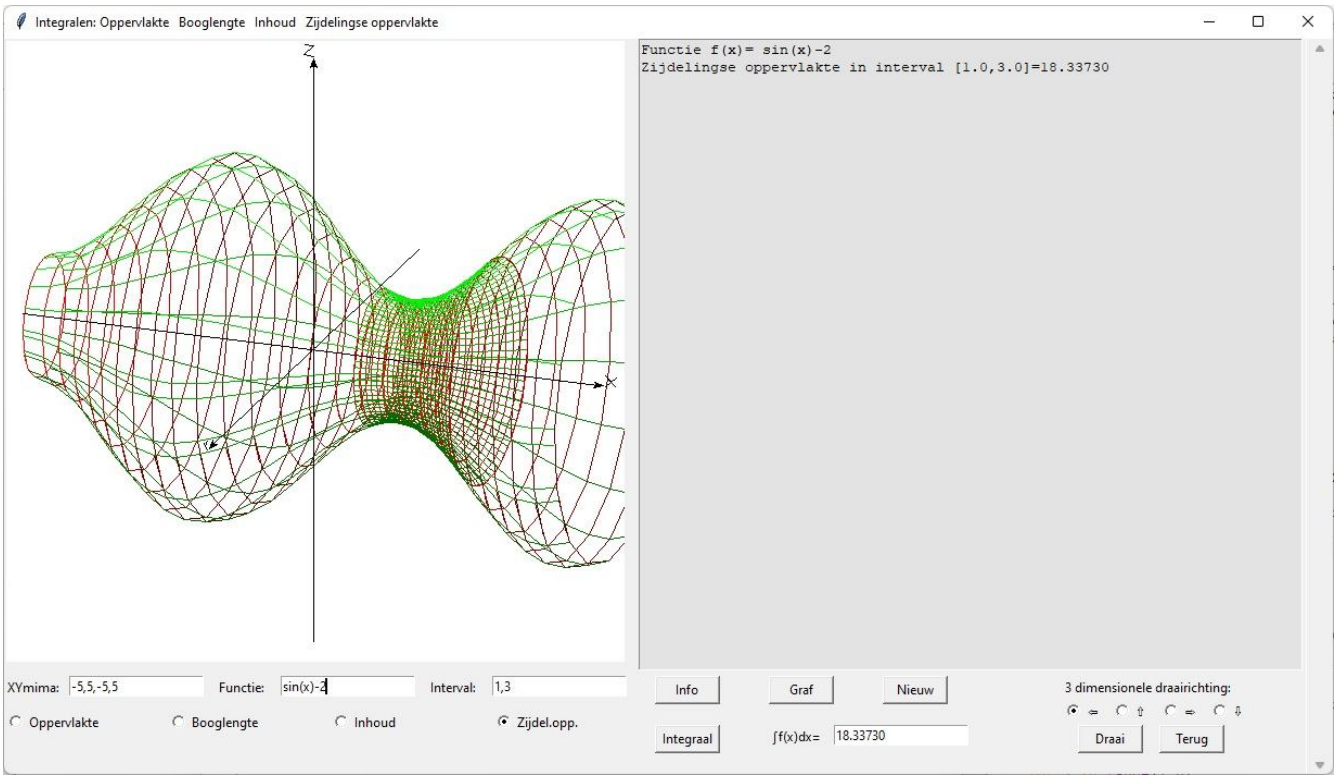
```
# Veelhoek: verschuiving,spiegeling,homothetie,draaiing TK
from math import *;from tkinter import *;from copy import *;from tkinter import messagebox
def col(rgb):return '#%02x%02x%02x' % rgb
def transx(xp):return (xp-xmi)*afm/(xma-xmi)
def transy(yp):return afm-afm*(yp-ymi)/(yma-ymi)
def transX(Xp):return xmi+Xp*(xma-xmi)/afm
def transY(Yp):return ymi-(Yp-afm)*(yma-ymi)/afm
def prod(b,c): # product 2 matrices
    n=len(b);p=[]
    for i in range(0,n):
        som=0
        for k in range(0,n):
            som=som+b[i][k]*c[k]
        p.append(som)
    return(p)
#------- grafiek vlakke meetkunde
def assen():
#assen
    X=transx(0);C.create_line(X,0,X,afm,fill='blue',arrow='first')
    Y=transy(0);C.create_line(0,Y,afm,Y,fill='blue',arrow='last')
# rooster
    for x in range(int(xmi),int(xma+1)):
        X=transx(x);C.create_line(X,0,X,afm,fill='green',dash=[1,2])
    for y in range(int(ymi),int(yma+1)):
        Y=transy(y);C.create_line(0,Y,afm,Y,fill='green',dash=[1,2] )
def basis():
    if k2==1:assen()
    lis=[];X=transx(0);Y=transy(0);lis.append(X);lis.append(Y)
    C.create_oval(X-2,Y-2,X+2,Y+2,fill='black')
    C.create_text(X+7,Y+7,text='C')
    lis=[];m=tan(radians(float(E2.get())));
    X=transx(xmi);Y=transy(m*xmi);lis.append(X);lis.append(Y)
    X=transx(xma);Y=transy(m*xma);lis.append(X);lis.append(Y)
    line = C.create_line(lis,fill='black')
    C.create_text(X-7,Y-7,text='r')
def teken():
#veelhoek
    lis=[];stap=0.2;x=xmi
    C.delete(ALL);assen();le=len(XT)
    for i in range(0,le):C.create_oval(XT[i]-2,YT[i]-2,XT[i]+2,YT[i]+2,fill='black')
    while x<xma:
        x=x+stap;y=f(x);X=transx(x);Y=transy(y)
        lis.append(X);lis.append(Y)
    line = C.create_line(lis,fill='red')
def orig():
    global X1,Y1,xt,yt;cl='black';C.delete(ALL);basis()
    xt,yt=[],[]
    for X in Xl:xt.append(transX(X))
    for Y in Yl:yt.append(transY(Y))
    lijnst(xt,yt,cl)
def lijnst(x1,y1,cl):
    XT,YT=[],[]
```

```
    for x in x1:XT.append(transx(x))
    for y in y1:YT.append(transy(y))
    lis=[];le=len(XT)
    for i in range(0,le):
       lis.append(XT[i]);lis.append(YT[i])
    lis.append(XT[0]);lis.append(YT[0])
    line = C.create_line(lis,fill=cl)
def hoek(x,y):
    hk=atan(y/x)
    if  x<0  :hk=hk+pi
    return hk
def streep(x1,y1,x2,y2):
    le=len(x1)
    for i in range(0,le):
       lis=[]
       if k!=3:
          lis.append(transx(x1[i]));lis.append(transy(y1[i]))
          lis.append(transx(x2[i]));lis.append(transy(y2[i]))
       else:
          r=sqrt(x1[i]*x1[i]+y1[i]*y1[i]);hk1=hoek(x1[i],y1[i]);hk2=hk1+gam;
          nrstap=int(45*gam/pi);stap=gam/nrstap;h=hk1;tel=0
          while tel<nrstap+1:lis.append(transx(r*cos(h)));
           lis.append(transy(r*sin(h)));h=h+stap;tel=tel+1
       line = C.create_line(lis,fill='lightgrey',dash=[1,2],arrow='last')
    return
def beeld():
    global xb,yb,gam; xb,yb=[],[]
    m=tan(radians(float(E2.get())));v=float(E3.get());beta=radians(float(E4.get()))
    h=float(E5.get());gam=radians(float(E6.get()))
    rotmat=[[cos(gam),-sin(gam)],[sin(gam),cos(gam)]];le=len(xt)
    dat=[E3.get()+','+E4.get()+'°','t.o.v. r','C,'+E5.get(),'C,'+E6.get()+'°']
    for i in range (0,le):
       if k==0:xb.append(xt[i]+v*cos(beta));yb.append(yt[i]+v*sin(beta));cl='blue'
       if k==1:
          a,b=xt[i],yt[i];cl='green'
          A=m*m+1;B=-2*m*(m*b+a);C=m*m*b*b+2*m*b*a-b*b;D=B*B-4*A*C
          S=-B/A;y=S-yt[i]
          x=-m*y+m*b+a
          yb.append(y);xb.append(x)
       if k==2:xb.append(h*xt[i]);yb.append(h*yt[i]);cl='magenta'
       if k==3:x,y=prod(rotmat,[xt[i],yt[i]]);xb.append(x);yb.append(y);cl='red'
    lijnst(xb,yb,cl);streep(xt,yt,xb,yb)
    Label(form,text='                              ').place(x=25*sc,y=41*sc)
    Label(form,text=keuze[k]+' ('+dat[k]+')').place(x=25*sc,y=41*sc)
    return
def selec():global k;k=var.get()
def selec2():global k2;k2=var2.get();print(k2)
# punten vastleggen na dubbelklik
def cursorco(e):
    global X1,Y1;X=e.x;Y=e.y
    X1,Y1=[],[]
    if X<afm and Y<afm:
       Xl.append(X);Yl.append(Y)
       C.create_oval(X-2,Y-2,X+2,Y+2,fill='black')
def punten():
```

```
    global xmi,xma,ymi,yma
    afm=float(E1.get());xmi,ymi=-afm,-afm;xma,yma=afm,afm;
    if k2==1:assen()
    C.bind('<Double-Button-1>',cursorco)
def but(xp,yp,t,co):Button(form,text=t,command=co).place(x=xp*sc,y=yp*sc,width=3*sc)
# ----- definities formulier
def formulier(yp,t):
    global C,afm,tex
    xp=yp;form.geometry(str(xp*sc+2)+'x'+str((yp+4)*sc));form.title(t);afm=(yp-2)*sc
    C=Canvas(form,bg='white',height=afm,width=afm);C.place(x=sc,y=0)
def ad(lin):tex.insert(INSERT,lin)
def lab(t,xp,yp):Label(form,text=t).place(x=sc*xp,y=sc*yp)
def ent(t,xp,yp):lab(t,xp,yp);en=Entry(form,width=4);en.place(x=sc*(xp+1.2),y=sc*yp);return en
def de(en):en.delete(0,len(en.get()))
def inp(en):return en.get()
def fval(en):return(eval(inp(en)))
def val(en):return mult(inp(en))
def help():
    helstr='xy: -min en max van coördinaten, α hoek spiegelas r met positieve x-as. '
    helstr=helstr+'Assen en rooster kunnen op of af gezet worden. '
    helstr=helstr+'Eventueel eerst  xy, α en Assen aanpassen en met ';
    helstr=helstr+'Nieuw wijzigen.\nNadien op enkele punten in het canvas dubbelklikken.\n'
    helstr=helstr+'Met Orig wordt de veelhoek getekend.\nNadien kies je voor één van de '
    helstr=helstr+'4 opties:\nVerschuiving, Spiegeling, Homothetie, Draaiing\n'
    helstr=helstr+'v en β dienen voor de verschuiving,\nh voor de homothetie, γ voor de draaiing'
    h=messagebox.showinfo('INFO:',helstr)
def nieuw():
    global xt,yt,Xl,Yl;xt=[];yt=[];Xl=[];Yl=[]
    C.delete(ALL);punten();basis()
# ------ main
sc=15;form=Tk();formulier(40,'Transformaties in het vlak');
E1=ent('xy:',0,39);E2=ent('α:',3,39)
E3=ent('v:',11,39);E4=ent('β:',14,39)
E5=ent('h:',17,39);E6=ent('γ:',20,39)
E1.insert(0,'10');E2.insert(0,'20');E3.insert(0,'4')
E4.insert(0,'30');E5.insert(0,'.5');E6.insert(0,'40')
keuze=['Verschuiving','Spiegeling','Homothetie','Draaiing']
var=IntVar();k=0;dx=10**-8;var2=IntVar();k2=0
for i in range(0,4):
    rb=Radiobutton(form,text=keuze[i],variable=var,value=i,command=selec)
    rb.place(x=i*6*sc,y=41*sc)
Checkbutton(form,text='Assen',variable=var2,command=selec2).place(x=7*sc,y=39*sc)
but(24,39,'Info',help);but(28,39,'Orig',orig)
but(32,39,'Beeld',beeld);but(36,39,'Nieuw',nieuw)
nieuw()
form.mainloop()
```

141 Integralen integ_grafTK

Nogmaals integralen: je hebt de keuze; ofwel vul je een functie in, ofwel dubbelklik je op enkele punten in het vlak: in dat geval wordt de veeltermfunctie berekend die door deze punten gaat. Heb je beide gedaan, dan wordt voorrang gegeven aan de ingevulde functie.
Je hebt dan de keuze tussen oppervlakte,booglengte,inhoud of zijdelingse oppervlakte van het omwentelingslichaam: de integraal wordt berekend, maar in het canvas krijg je ook een meetkundige voorstelling, zowel in het vlak als in de ruimte.



Programma:

```
# integraal_grafiekTK
from math import *;from tkinter import *;from copy import *;from tkinter import messagebox
def col(rgb):return '#%02x%02x%02x' % rgb
def fx(x):return eval(xst)
def fy(x):return eval(yst)
def mult(s):
    l=s.split(',');co=[]
    for i in l:co.append(float(i))
    return co
def transx(xp):return (xp-xmi)*afm/(xma-xmi)
def transy(yp):return afm-afm*(yp-ymi)/(yma-ymi)
def transX(Xp):return xmi+Xp*(xma-xmi)/afm
def transY(Yp):return ymi-(Yp-afm)*(yma-ymi)/afm
def det(mat):
    le=len(mat)
    if le==2:d=mat[0][0]*mat[1][1]-mat[0][1]*mat[1][0]
    else:
        d=0
        for i in range(0,le):
            s=[]
            for k in range(0,le):
                if i!=k:s.append(mat[k][1:])
            d=d+mat[i][0]*(-1)**i*det(s)
    return d
def prod(b,c): # product 2 matrices
    n=len(b);p=[]
    for i in range(0,n):
        som=0
```

```
        for k in range(0,n):
            som=som+b[i][k]*c[k]
        p.append(som)
    return(p)
# ----- grafiek ruimtemeetkunde
def hoek(y,x):
    if y==0 and x>=0:c=0
    elif y==0 and x<0:c=pi
    elif x==0 and y>0:c=pi/2
    elif x==0 and y<0:c=3*pi/2
    else:
        c=atan(y/x)
        if x<0:c=c+pi
    return(c)
def omzet(x,y,z):
    c=hoek(y,x)
    r=sqrt(x*x+y*y);y1=sin(xyH+c)*r
    d=hoek(y1,z)
    r1=sqrt(y1*y1+z*z)
    X=transx(cos(xyH+c)*r);Y=transy(cos(yzH+d)*r1)
    return [X,Y]
def asteken(x,y,z):
    lis=[]
    X,Y=omzet(-x,-y,-z);lis.append(X);lis.append(Y)
    X,Y=omzet(x,y,z);lis.append(X);lis.append(Y)
    line = C.create_line(lis,fill='black',arrow='last')
def rotmat(v):return [[1,0,0],[0,cos(v),sin(v)],[0,-sin(v),cos(v)]]
def teken3d(link,rech):
    # draaihoeken xyH= hoek xy  yzH = hoek yz
    # t van tmi =>tma parameter waarover grafiek in x-richting
    # v van vmi =>vma parameter van wenteling
    global xmi,xma,ymi,yma,zmi,zma,tmi,vmi,tma,vma,a,b,xyH,yzH
    yma=xma;zma=xma;xmi=-xma;ymi=xmi;zmi=xmi
    tmi,tma=link,rech
    vmi,vma=-pi,pi+0.5
    xyH=radians(draaixy);yzH=radians(draaiyz);x=0
#kromme
#niveaulijnen tekenen voor constante u
    stap=vma/roos;t=tmi
    while t<tma:
        ro=int(175+75*sin(t));kl=col((ro,0,0));t=t+stap; lis=[];V=[t,f(t),0];v=vmi
        while v<vma:
            v=v+stap;R=rotmat(v);x,y,z=prod(R,V);X,Y=omzet(x,y,z)
            lis.append(X);lis.append(Y)
        line = C.create_line(lis,fill=kl)
#niveaulijnen tekenen voor constante v
    v=vmi
    while v<vma:
        gr=int(175+75*sin(v));kl=col((0,gr,0));
R=rotmat(v);v=v+stap;lis=[];t=tmi
        while t<tma:
            t=t+stap;V=[t,f(t),0];x,y,z=prod(R,V);X,Y=omzet(x,y,z)
            lis.append(X);lis.append(Y)
        line = C.create_line(lis,fill=kl)
#assen
```

```
    asteken(xma,0,0);asteken(0,yma,0);asteken(0,0,zma)
#x,y,z letters
    hl=0.17;st=xma+0.03
    lis=[omzet(st,0,0),omzet(st+hl,0,hl),omzet(st+hl/2,0,hl/2), omzet(st,0,hl),omzet(st+hl,0,0)]
    line = C.create_line(lis,fill='black') #x
    lis=[omzet(0,st,0),omzet(0,st+hl,hl), omzet(0,st+hl/2,hl/2),omzet(0,st,hl)]
    line = C.create_line(lis,fill='black') #y
    lis=[omzet(0,0,st),omzet(-hl,0,st),omzet(0,0,st+hl),omzet(-hl,0,st+hl)]
    line = C.create_line(lis,fill='black') #z
def cirkelvlak(d,r1,r2): # linker of rechter zijvlak bij inhoud
    vmi,vma=-pi,pi+0.5;stap=3*vma/roos
    if k==2:
        t=r1
        while t<r2:
            lis=[];V=[d,t,0];v=vmi
            while v<vma:
                v=v+stap;R=rotmat(v);x,y,z=prod(R,V);X,Y=omzet(x,y,z)
                lis.append(X);lis.append(Y)
            line = C.create_line(lis,fill='green');t=t+stap
        v=vmi;lis=[]
        while v<vma:
            V=[d,0,r1];R=rotmat(v);x,y,z=prod(R,V); X,Y=omzet(x,y,z);lis.append(X);lis.append(Y)
            V=[d,0,r2];R=rotmat(v);x,y,z=prod(R,V);X,Y=omzet(x,y,z); lis.append(X);lis.append(Y);v=v+stap
        line = C.create_line(lis,fill='green')
def draai():
    global draaixy,draaiyz,roos;draaihoek=15;a,b=mult(E3.get())
    if k>=2:
        if rich==0:draaixy=draaixy+draaihoek
        if rich==1:draaiyz=draaiyz-draaihoek
        if rich==2:draaixy=draaixy-draaihoek
        if rich==3:draaiyz=draaiyz+draaihoek
        C.delete(ALL);roos=10;teken3d(xmi,xma);roos=40;teken3d(a,b)
        roos=40;cirkelvlak(a,0,f(a));cirkelvlak(b,0,f(b))
def terug():
    global draaixy,draaiyz,roos;draaihoek=15;a,b=mult(E3.get());draaixy,draaiyz=20,20
    if k>=2:
        C.delete(ALL);roos=10;teken3d(xmi,xma);roos=40;teken3d(a,b);cirkelvlak(a,0,f(a));cirkelvlak(b,0,f(b))
#------- grafiek vlakke meetkunde
def assen():
#assen
    X=transx(0);C.create_line(X,0,X,afm,fill='blue',arrow='first')
    Y=transy(0);C.create_line(0,Y,afm,Y,fill='blue',arrow='last')
# rooster
    for x in range(int(xmi),int(xma+1)):
        X=transx(x);C.create_line(X,0,X,afm,fill='green',dash=[1,2])
    for y in range(int(ymi),int(yma+1)):
        Y=transy(y);C.create_line(0,Y,afm,Y,fill='green',dash=[1,2] )
def graf():
    global costr,func,X1,Y1,xt,yt,XT,YT
    xt,yt=[],[];XT,YT=[],[]
    if E2.get()==costr and E1.get()=='':
        for X in Xl:xt.append(transX(X))
        for Y in Yl:yt.append(transY(Y))
        stels=[];n=len(xt)
        for x in xt:XT.append(transx(x))
```

```
        for y in yt:YT.append(transy(y))
        for i in range(0,n):
            xm=[]
            for j in range(0,n):
                xm.append(xt[i]**j)
            xm.reverse();stels.append(xm)
        acop=deepcopy(stels);m=[];rechlid=yt
        for k in range(0,n):m.append([])
        for k in range(0,n):
            m[k]=deepcopy(acop)
            for i in range(0,n):
                m[k][i].pop(k)
                m[k][i].insert(k,rechlid[i])
        func=''
        for k in range(0,n):
            co=det(m[k])/det(acop);cost=format(co,'5f')
            if co>=0:cost='+'+cost
            func=func+cost+'*x**'+str(n-k-1)
        teken();ad('Functie f(x)= '+func+'\n')
    elif E1.get()!='' and  E2.get()==costr:func=E1.get();teken();ad('Functie f(x)= '+func+'\n')
    else:nieuw()
    costr=E2.get()
    return
def teken():
#kromme
    lis=[];stap=0.2;x=xmi
    C.delete(ALL);assen();le=len(XT)
    for i in range(0,le):C.create_oval(XT[i]-2,YT[i]-2,XT[i]+2,YT[i]+2,fill='black')
    while x<xma:
        x=x+stap;y=f(x);X=transx(x);Y=transy(y)
        lis.append(X);lis.append(Y)
    line = C.create_line(lis,fill='red')
#--------- integraalberekening
def df(x):return(f(x+dx)-f(x))/dx
def inh(x):return pi*f(x)*f(x)
def bl(x):return sqrt(1+df(x)**2)
def zo(x):return 2*pi*abs(f(x))*sqrt(1+df(x)**2)
def selec():global k;k=var.get()
def selec2():global rich;rich=var2.get()
def bereken():
    # k= 0 : oppervlakte  k=1 : booglengte  k=2 : inhoud  k=3 : zijdelingse oppervlakte
    global roos,a,b;a,b=mult(E3.get())
    if b<a:wissel=a;a=b;b=wissel;de(E3);E3.insert(0,str(a)+','+str(b))
    n=100;h=(b-a)/n;x=a;som=0
    for j in range(1,n):
            x=x+h
            if j%2==0:t=2
            else:t=4
            if k==0:som=som+t*f(x)
            if k==1:som=som+t*bl(x)
            if k==2:som=som+t*inh(x)
            if k==3:som=som+t*zo(x)
    if k==0:som=som+f(a)+f(b)
    if k==1:som=som+bl(a)+bl(b)
    if k==2:som=som+inh(a)+inh(b)
```

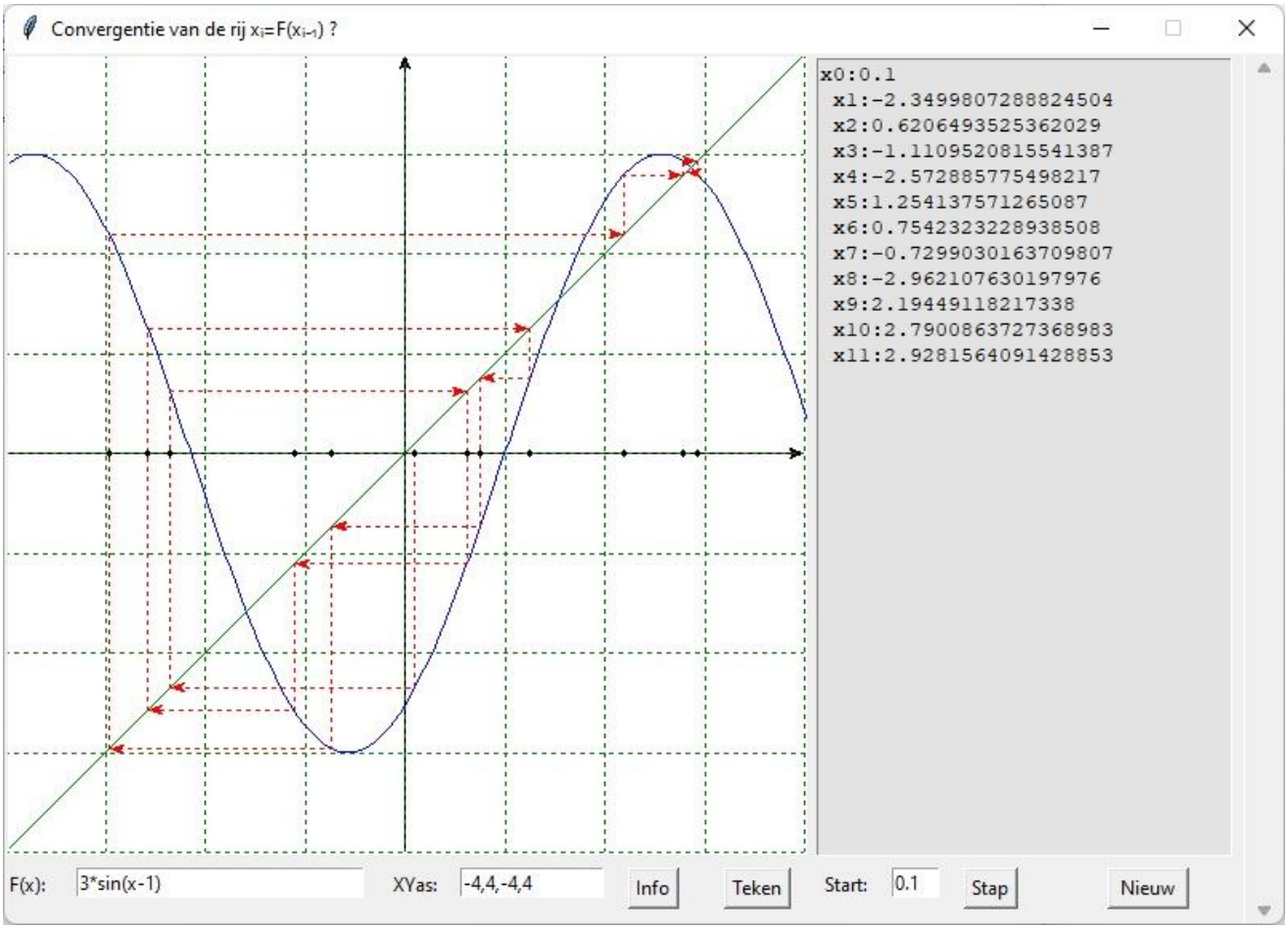
```
    if k==3:som=som+zo(a)+zo(b)
    integ=som*h/3;de(E4);intst=format(integ,'.5f');E4.insert(0,intst)
    antw=' in interval ['+str(a)+','+str(b)+']='+intst+'\n'
    if k==0:
       x=a;stap=0.05;teken()
       while x<b:
          lis=[];y=0;X=transx(x);Y=transy(y); lis.append(X);lis.append(Y);y=f(x)
          X=transx(x);Y=transy(y);lis.append(X); lis.append(Y);x=x+stap;C.create_line(lis,fill='green' )
       ad('Oppervlakte'+antw)
    if k==1:
       teken()
       for dik in[0.02,0.01,0,-0.01,0.02]:
          x=a;stap=0.05;lis=[]
          while x<=b:
             y=f(x);X=transx(x);Y=transy(y+dik)
             lis.append(X);lis.append(Y);x=x+stap
          line = C.create_line(lis,fill='darkblue')
       ad('Booglengte'+antw)
    if k==2:
       C.delete(ALL);roos=10;teken3d(xmi,xma); roos=40;teken3d(a,b);cirkelvlak(a,0,f(a));cirkelvlak(b,0,f(b))
       ad('Inhoud'+antw)
    if k==3:
       C.delete(ALL);roos=10;teken3d(xmi,xma);roos=40;teken3d(a,b)
       ad('Zijdelingse oppervlakte'+antw)
# punten vastleggen na dubbelklik
def cursorco(e):
    global X1,Y1;X=e.x;Y=e.y
    X1,Y1=[],[]
    if X<afm and Y<afm:
       Xl.append(X);Yl.append(Y)
       C.create_oval(X-2,Y-2,X+2,Y+2,fill='black')
def punten():
    global xmi,xma,ymi,yma
    xmi,xma,ymi,yma=mult(E2.get());assen()
    C.bind('<Double-Button-1>',cursorco)
def but(xp,yp,t,co):Button(form,text=t,command=co).place(x=xp*sc,y=yp*sc,width=4*sc)
# ----- definities formulier
def formulier(yp,t):
    global C,afm,tex
    xp=2*yp;form.geometry(str(xp*sc+25)+'x'+str((yp+5)*sc));form.title(t);afm=(yp-2)*sc
    C=Canvas(form,bg='white',height=afm,width=afm);C.place(x=0,y=0)
    tex=Text(form,bg='grey89',height=int(afm*0.06333),width=int(afm*4/30),wrap=WORD);
    tex.place(x=(yp-1)*sc,y=0)
    vsb=Scrollbar(form,orient=VERTICAL,command=tex.yview,width=25)
    tex['yscrollcommand']=vsb.set;vsb.pack(side=RIGHT,fill='y')
def f(x):return eval(func)
def ad(lin):tex.insert(INSERT,lin)
def lab(t,xp,yp):Label(form,text=t).place(x=sc*xp,y=sc*yp)
def ent(t,xp,yp):lab(t,xp,yp);en=Entry(form);en.place(x=sc*(xp+4),y=sc*yp);return en
def de(en):en.delete(0,len(en.get()))
def inp(en):return en.get()
def fval(en):return(eval(inp(en)))
def val(en):return mult(inp(en))
def help():
    helstr='Integraal van een functie op een interval [a,b].\n
```

```
   Ofwel een functie invullen, ofwel dubbelklikken op punten in het vlak: '
    helstr=helstr+' dan wordt de veeltermfunctie bepaald die door deze punten gaat.
    Kies één van de 4 opties,vul a,b in'
    helstr=helstr+' en klik op\n Graf , Integraal. Met Draai worden 3dim grafieken gewenteld.'
    h=messagebox.showinfo('INFO:',helstr)
def nieuw():
    global xt,yt,Xl,Yl,costr;xt=[];yt=[];Xl=[];Yl=[];costr=E2.get()
    for en in entr:en.delete(0,len(en.get()))
    C.delete(ALL);tex.delete('1.0',END)
    draaixy,draaiyz=20,20;punten()
# ------ main
sc=15;form=Tk();formulier(40,'Integralen: Oppervlakte  Booglengte  Inhoud  Zijdelingse oppervlakte');
E1=ent('Functie:',13,39);E2=ent('XYmima:',0,39);E3=ent('Interval:',26,39);E4=ent('\u222bf(x)dx=',47,42);entr=[E
1,E4]
costr='-5,5,-5,5';E2.insert(0,costr);ab='1,3';E3.insert(0,ab)
keuze=['Oppervlakte','Booglengte','Inhoud','Zijdel.opp.']
var=IntVar();k=0;dx=10**-8
for i in range(0,4):
    rb=Radiobutton(form,text=keuze[i],variable=var,value=i,command=selec)
    rb.place(x=i*10*sc,y=41*sc)
pijl=['\u21E6','\u21E7','\u21E8','\u21E9'] #unicode pijlen
rich=0;var2 = IntVar()
Label(form,text='3 dimensionele draairichting:').place(x=65*sc,y=39*sc)
for j in range(0,4):
    rb=Radiobutton(form,text=pijl[j],variable=var2,value=j,command=selec2)
    rb.place(x=(65+j*3)*sc,y=40*sc+5)
but(40,39,'Info',help);but(47,39,'Graf',graf);but(54,39,'Nieuw',nieuw)
but(40,42,'Integraal',bereken);but(66,42,'Draai',draai);but(71,42,'Terug',terug)
draaixy,draaiyz=20,20;nieuw()
form.mainloop()
```

142. Convergentie van de rij $x_i=F(x_{i-1})$ x_iterTk

```
# Convergentie van de rij x_i=F(x_i-1)
from math import *;from tkinter import *;from tkinter import messagebox
def mult(s):
    l=s.split(',');co=[]
    for i in l:co.append(float(i))
    return co
def transx(x):return (x-xmi)*wid/(xma-xmi)
def transy(y):return heigC-heigC*(y-ymi)/(yma-ymi)
def F(x):return eval(E1.get())
def G(x):return x
def ad(lin):tex.insert(INSERT,lin+' ')
def teken():
    global xmi,xma,ymi,yma,stapnr
    xmi,xma,ymi,yma=mult(E2.get());C.delete(ALL);stapnr=0;tex.delete('1.0',END)
#assen
    X=transx(0);line = C.create_line(X,0,X,heigC,fill='black',arrow='first')
    Y=transy(0);line = C.create_line(0,Y,wid,Y,fill='black',arrow='last')
# rooster
    for x in range(int(xmi),int(xma+1)):
        X=transx(x);line = C.create_line(X,0,X,heigC,fill='green',dash=[1,2])
    for y in range(int(ymi),int(yma+1)):
        Y=transy(y);line = C.create_line(0,Y,wid,Y,fill='green',dash=[1,2] )
```

```
#kromme
   krom(F,'blue');krom(G,'green')
def krom(fu,col):
   lis=[];stap=0.05;x=xmi
   while x<xma:
      x=x+stap;y=fu(x);X=transx(x);Y=transy(y); lis.append(X);lis.append(Y)
   C.create_line(lis,fill=col)
def stap():
   global stapnr,xp,lis
   if stapnr==0:xp=float(E3.get())
   xo=xp;xp=F(xp)
   ad('x'+str(stapnr)+':'+str(xo)+'\n')
   if stapnr==0:y1=0
   else: y1=xo
   stapnr=stapnr+1
   lis=[];X=transx(xo);Y=transy(y1);lis.append(X);lis.append(Y)
   Y=transy(xp);lis.append(X);lis.append(Y)
   X=transx(xp);lis.append(X);lis.append(Y)
   C.create_line(lis,fill='red',dash=[1,2],arrow='last')
   X=transx(xo);Y=transy(0)
   C.create_line(X,Y-2,X,Y+2,fill='black')
def lab(t,xp,yp):Label(form,text=t).place(x=sc*xp,y=sc*yp)
def ent(t,xp,yp,w):lab(t,xp,yp);en=Entry(form);
en.place(x=sc*(xp+3),y=sc*yp,width=w*sc);return en
def info():
   helstr='Een oplossing benaderen voor x=F(x)\n'
   helstr=helstr+' Vul een functie in voor F(x)'
   helstr=helstr+', eventueel ander assenstelsel en'
   helstr=helstr+' Teken de grafiek.\n'
   helstr=helstr+' Als y=F(x) de grafiek y=x snijdt,'
   helstr=helstr+' kan het zijn dat de rij x_i=F(x_i-1) convergeert naar dit snijpunt.'
   helstr=helstr+' Vul een startwaarde in, en klik herhaaldelijk op Stap.'
   h=messagebox.showinfo('INFO:',helstr)
def nieuw():
   global stapnr,xp
   for E in [E1,E2,E3]:E.delete(0,len(E.get()))
   C.delete(ALL);tex.delete('1.0',END);stapnr=0
#hoofdprogramma
global stapnr
wid=500;heig=545;heigC=heig-45;sc=15;stapnr=0
form=Tk();form.geometry(str(800)+'x'+str(heig));form.title('Convergentie van de rij x_i=F(x_i-1) ?')
form.resizable(False,False)
C = Canvas(form, bg="white", height=heigC, width=wid);C.place(x=0,y=0)
E1=ent('F(x):',0,34,12);E2=ent('XYas:',16,34,6);E3=ent('Start:',34,34,2)
E1.insert(0,'3*sin(x-1)');E2.insert(0,'-4,4,-4,4');E3.insert(0,'0.1')
tex=Text(form,bg='grey89',height=31,width=32,wrap=WORD);tex.place(x=wid+8,y=3)
vsb=Scrollbar(form,orient=VERTICAL,command=tex.yview,width=25)
tex['yscrollcommand']=vsb.set;vsb.pack(side=RIGHT,fill='y')
B1=Button(form,text='Teken',command=teken);B1.place(x=30*sc,y=34*sc)
B2=Button(form,text=' Nieuw ',command=nieuw);B2.place(x=46*sc,y=34*sc)
B3=Button(form,text='Stap',command=stap);B3.place(x=40*sc,y=34*sc)
B4=Button(form,text='Info',command=info);B4.place(x=26*sc,y=34*sc)
form.mainloop()
```



143. Integralen van rationale functies (primitieve functies) ration1integ

Onbepaalde integralen van veeltermbreuken vragen heel wat rekenwerk.

Stel $\int \frac{A(x)}{B(x)} \cdot dx$ waarbij A(x) en B(x) veeltermen zijn. Om dergelijke integralen op te lossen, maken we eerst de euclidische deling en herleiden de breuk tot $Q(x) + \frac{R(x)}{B(x)}$

Om het rekenwerk te vereenvoudigen kunnen we voor B(x) een veelterm nemen van de 2de graad. Het volgende programmaatje geeft ons 'random' oefeningen van deze soort:

Programma:

```
#integraal rationale functie met noemer Ax²+Bx+c
from random import randint as ri;from sympy import *
def vc(x):
    if x==1:s='+1'
    elif x==-1:s='-1'
    elif x>=0:s='+'+str(x)
    else:s=str(x)
    return s
x,f=symbols('x f')
# random teller
n=ri(4,4);T='(';a=[]
for i in range(0,n+1):a.append(ri(-3,5))
```

```
for i in range (n,-1,-1):
   T=T+vc(a[i])+'*x**'+str(i)
# random noemer
n=2;N='(';b=[]
for i in range(0,n+1):b.append(ri(-3,5))
for i in range (n,-1,-1):
   N=N+vc(b[i])+'*x**'+str(i)
T=T+')';N=N+')';f=T+'/'+N
init_printing()
pprint(Integral(f,x))
pprint(integrate(f,x))
```

```
⌠
⎮     4       3
⎮ 2.x  - 3.x  + x - 1
⎮ ──────────────────── dx
⎮          2
⎮     - 3.x  + x - 2
⌡
```

```
                                    2    x    2                  6.√23.x     √23
     3       2           25.log(x  - ─ + ─)    116.√23.atan(──────── - ─── )
  2.x     7.x     19.x              3    3                      23          23
- ──── + ──── + ──── - ──────────────────── - ──────────────────────────────
   9       18      27             81                      1863
```

Heel wat leerlingen maken rekenfouten bij de euclidische deling, zodat de rest van deze 'lange' oefening ook verkeerd is. En eigenlijk wordt er hier getest op het berekenen van integralen en niet op het maken van een euclidische deling (leerstof van 2 jaar voordien).
De noemer is een kwadratische functie van de vorm Ax^2+Bx+C
We kunnen het rekenen met breuken bij de euclidische deling vermijden door A=1 te stellen.
En we kunnen het rekenen met wortelvormen ook vermijden met de 3 basisvormen:
$x^2+2px+p^2$ $x^2+2px+p^2-c^2$ $x^2+2px+p^2+c^2$
Rekenfouten zijn altijd mogelijk maar nu is de kans al veel kleiner.

Programma: ration2integ

```
#integraal rationale functie met noemer x²+2px +p²+/-c²
from random import randint as ri
def vc(x):
   if x==1:s='+1'
   elif x==-1:s='-1'
   elif x>=0:s='+'+str(x)
   else:s=str(x)
   return s
from sympy import *
x,f=symbols('x f')
for k in range(1,4):
   # random teller
   n=ri(2,4);T='(';a=[]
   for i in range(0,n+1):a.append(ri(-3,5))
   for i in range (n,-1,-1):
      T=T+vc(a[i])+'*x**'+str(i)
   # random noemer
   p=ri(1,5);c=ri(1,5)
   if k==1:q=p*p
   elif k==2:q=p*p+c*c
```

```
    else:q=p*p-c*c
    init_printing()
    T=T+')';N='(x**2'+vc(2*p)+'*x'+vc(q)+')';f=T+'/'+N
    pprint(Integral(f,x))
    pprint(integrate(f,x))
```

```
⌠
⎮    3      2
⎮ 3.x  + 3.x  - x
⎮ ──────────────── dx
⎮   2
⎮  x  + 10.x + 25
⌡
   2
3.x                                  295
─── - 27.x + 194.log(x + 5) + ─────
 2                               x + 5
⌠
⎮    3      2
⎮ 3.x  - 3.x  - 3.x - 3
⎮ ──────────────────── dx
⎮       2
⎮      x  + 8.x + 20
⌡
                                          x
   2                   2              75.atan(─ + 2)
3.x           153.log(x  + 8.x + 20)      2
─── - 27.x + ──────────────────────── - ──────────
 2                      2                    2

 2
⌠
⎮    3    2
⎮ 3.x  - x  - x + 4
⎮ ──────────────── dx
⎮        2
⎮    x + 6.x
⌡
  2
3.x              2.log(x)          337.log(x + 6)
─── - 19.x + ──────────── + ──────────────────
 2                3                     3
```

144. Nauwkeurige berekening van een vierkantswortel. (en derdemachtswortel)

Er bestaat een methode om een vierkantswortel cijfer na cijfer te berekenen.
Stel een string a = '120535.431' : het deeltal ; string b =voorlopig " : de deler.
We nemen de cijfers van a voor en na de komma samen in groepjes van 2: in dit geval: 12 05 35 43 10
Staan er een oneven aantal voor de komma, dan vormt het voorste cijfer het 1ste groepje.
Staan er een oneven aantal achter de komma, voeg een '0' toe
Is het een geheel getal, voeg '.00' toe
In het programma worden deze getallen opgeslagen in een list A, dus A[0]='12',A[1]='5',...
Omdat de positie van de decimale punt 6 is (beginnen tellen vanaf 0), is de positie van de decimale punt in de vierkantswortel 3: steeds geldt: pospuntroot=(pospunt-1) // 2 + 1 , hier 3=(6-1) // 2 + 1 . (// =div in Python)
We zullen nu een soort staartberekening uitvoeren:
Om deze beter te begrijpen, de volgende redenering

Voor de eenvoud laten we voorlopig bij b de decimalen weg
a='120535.431' =12*10000+05*100+35*1+43*0.01+10*0.0001
b= 'pqr', dwz 100p+10q+r
Omdat b^2 ~< a is $(100p+10q+r)^2$ ~< 12*10000+05*100+35*1+...
Of p^2.10000+[(2p*10+q)*q]*100+{[2(10p+q)*10+r]*r} ~< 12*10000+05*100+35*1+..
Als je beide leden vergelijkt, moet p het grootste getal zijn waarvan het kwadraat<=12
Dus p=3: er blijft: [(2p*10+q)*q]*100+{[2(10p+q)*10+r]*r} ~< 3*10000+5*100+35+..
of ook nog: [(60+q)*q]*100+{[2(10p+q)*10+r]*r} ~< 305*100+35+..
Dus moet q het grootste getal zijn zodat (60+q)*q<=305, dus q=4
Als we dit invullen, 256*100+[2*34*10+r]*r ~< 305*100+35
Er blijft [680+r]*r=4935+..
r= grootste getal zodat [680+r]*r<=4935
Dus r=7, enz.
Hieruit kunnen we de volgend staartberekening afleiden:

```
1205354310          | 34718.....
 9                  | 6'4x4=256          6=  3*2
 305                | 68'7x7=4809        68=34*2
 256                | 694'1x1=6941       694=347*2
  4935              | 6942'8x8=555424    6942=3471*2
 4809                |
  12643
   6941
   570210
   555424
    1478600
     .....
```

Programma: vkwortel

```
# nauwkeurige berekening van een vierkantswortel
a =input('Geef een positief getal:')
print('Met **:\n'+str((float(a)**0.5)))
n=int(input('# cijfers:'))
pospunt_a=a.find('.')
if pospunt_a==-1:pospunt_a=len(a);a=a+'.00'
k= (len(a)-pospunt_a-1)% 2
if k==1:a=a+'0'
print('Start string:',a)
pospunt_b=(pospunt_a-1) // 2+1
A=[];j= pospunt_a % 2
if j>0:A.append(int(a[0:j]))
for i in range(j,pospunt_a,2):A.append(int(a[i:i+2]))
for i in range(pospunt_a+1,len(a),2):A.append(int(a[i:i+2]))
print('A=',A)
#staart berekening
for i in range(0,n):A.append(0)
b='';d=0
while d*d<=A[0]:d=d+1
d=d-1;b=b+str(d);A[0]=A[0]-d*d
for i in range(1,n):
    d=0;A[i]=A[i-1]*100+A[i];ib=int(b)
    while(2*ib*10+d)*d<=A[i]:d=d+1
    d=d-1;minus=(2*ib*10+d)*d
    A[i]=A[i]-minus;b=b+str(d)
```

```
lb=b[0:pospunt_b]+'.'+b[pospunt_b:len(b)]
print('Met cijfer na cijfer berekening:\n'+lb)
```

```
Geef een positief getal:13725.012
Met **:
117.1537963533406
# cijfers:100
Start string: 13725.0120
A= [1, 37, 25, 1, 20]
Met cijfer na cijfer berekening:
117.153796353340594593923461993551732393626130160598097698790611339922343047735343506805481
7346219956
```

A[i] wordt extreem groot, als i groter wordt: je kan dit controleren door ergens een lijntje print(A[i]) toe te voegen
In Python 3 is dit blijkbaar geen probleem. Integers mogen enorm groot worden.
De berekende vierkantswortel is een string, dus geen Python float.
Je kan er in Python niet mee rekenen, het is enkel een voorstelling van de vierkantswortel door een opeenvolging van cijfers.
Neem je de float van deze string, dan rond hij af tot een 15-tal cijfers.

Een vergelijkbare methode kan ook gebruikt worden voor de berekening van 3de machtswortels:
neem terug **a**='120535.431' and **b**= 'pqr'
b³=a betekent $(100p+10q+r)^3=a$ of $1000000p^3+300000p^2q+30000pq^2+1000q^3$+termen met r.

Programma: derdewortel

```
#nauwkeurige berekening van een derde wortel
a =input('Geef een positief getal:')
print('Met ** :\n'+str((float(a)**(1/3))))
n=int(input('# cijfers:'))
pospunt_a=a.find('.')
if pospunt_a==-1:pospunt_a=len(a);a=a+'.000'
k= (len(a)-pospunt_a-1)% 3
if k==1:a=a+'00'
if k==2:a=a+'0'
print('Start string:',a)
pospunt_b=(pospunt_a-1) //3 +1
#verdeling in groepjes van 3 cijfers
A=[];j= pospunt_a % 3
if j>0:A.append(int(a[0:j]))
for i in range(j,pospunt_a,3):A.append(int(a[i:i+3]))
for i in range(pospunt_a+1,len(a),3):A.append(int(a[i:i+3]))
print('A=',A)
#staart berekening
for i in range(0,n):A.append(0)
b='';d=0
while d*d*d<=A[0]:d=d+1
d=d-1;b=b+str(d);A[0]=A[0]-d*d*d
for i in range(1,n):
    d=0;A[i]=A[i-1]*1000+A[i];ib=int(b)
    while d*(300*ib**2+30*ib*d+d*d)<=A[i]:d=d+1
    d=d-1; minus=d*(300*ib**2+30*ib*d+d*d)
    A[i]=A[i]-minus; b=b+str(d)
lb=b[0:pospunt_b]+'.'+b[pospunt_b:len(b)]
print('Met cijfer na cijfer berekening:\n'+lb)
```

```
Geef een positief getal:1247.26
Met ** :
10.764296824097231
# cijfers:50
Start string: 1247.260
A= [1, 247, 260]
Met cijfer na cijfer berekening:
10.764296824097233157632717527519255183917039809730
```

145. n-de machtswortels benaderen met breuken n_de_wortel_breuk

Een interessante oefening lijkt me om n-de machtswortels te berekenen uit een rationaal getal, binnen de verzameling *Q* van de rationale getallen. Een n-de machtswortel van een rationaal getal a/b is een nulpunt van de functie $f(x) = x^n - \frac{a}{b}$.

De benaderingsfunctie is $x - \frac{f(x)}{f'(x)} = x - \frac{x^n - \frac{a}{b}}{n.x^{n-1}} = \frac{(n-1)\cdot x^n + \frac{a}{b}}{n.x^{n-1}}$

Is *x* zelf een breuk $\frac{p}{q}$ dan wordt deze functie: $\frac{(n-1)\cdot \frac{p^n}{q^n} + \frac{a}{b}}{n\cdot \frac{p^{n-1}}{q^{n-1}}} = \frac{(n-1)\cdot p^n.b + q^n\cdot a}{n\cdot p^{n-1}.q.b}$

Voor de nieuwe breuk $\frac{p_N}{q_N}$ geldt: $p_N = (n-1)\cdot p^n.b + q^n\cdot a$ en $q_N = n\cdot p^{n-1}.q.b$

Deze p_N en q_N noemen we in het programma terug p en q om de benadering verder te zetten
Hoever moeten we deze benadering voortzetten ? Stel een groot getal G.

Tot als $\left|\frac{p^n}{q^n} - \frac{a}{b}\right| < \frac{1}{G}$ of $\left|G\cdot(p^n.b - q^n.a)\right| < q^n.b$

Programma:

```
# n-de machtswortels benaderen met breuken
def mult(s):
    l=s.split(',');co=[]
    for i in l:co.append(int(i))
    return co
def ggd(r,s):
    t=1
    while t!=0:t=r%s;r=s;s=t
    return r
def nwort(p,q):
    p,q=(n-1)*p**n*b+a*q**n,n*p**(n-1)*q*b
    d=ggd(p,q)
    return(p//d,q//d)
# hoofdprogramma
print('Benaderen n-wortel uit a/b  met een breuk p/q:')
a,b,n=mult(input('a,b,n:'));x=0;G=1000
```

```
print(str(n)+'-de wortel uit a/b='+str(a)+'/'+str(b))
while x**n<a:x=x+1
x=x-1;p=x;q=1
while abs(G*(p**n*b-a*q**n))>q**n*b:p,q=nwort(p,q)
wortst=str(p)+'  /  '+str(q);print('p/q=\n'+wortst);decwort=eval(wortst)
print('Decimale vorm a/b=',a/b)
print('Decimale vorm p/q=',decwort)
print('Proef:\n',n,'-de wortel uit',a/b,'=',(a/b)**(1/n))

>>>
Benaderen n-wortel uit a/b  met een breuk p/q:
a,b,n:89,7,3
3-de wortel uit a/b=89/7
p/q=
22215858039020422517833437340725302927391627627586177618213563570826751250100959721  /
95183930059652065779958910290324553339024861399030357742308915804546329345400911 50
Decimale vorm a/b= 12.714285714285714
Decimale vorm p/q= 2.3339925158687684
Proef:
 3 -de wortel uit 12.714285714285714 = 2.333981031724222
>>>
```

Wat blijkt is dat we zeer snel breuken krijgen, waarvan teller en noemer enorm grote gehele getallen zijn. In python kan dit.

146. Rekentoestel hoofdbewerkingen met rationale getallen. bewerkingen_met_breuken

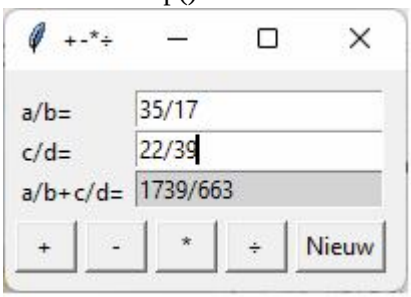
We maken een formulier (tkinter) met 2 invoervelden: één voor a/b, één voor c/d
Ook 4 knoppen ('buttons') voor de 4 hoofdbewerkingen. En een uitvoerveld voor het resultaat e/f
Voor + , - , . , ÷ geldt er:

$$\frac{e}{f}=\frac{a}{b}+\frac{c}{d}=\frac{a\cdot d+b\cdot c}{b\cdot d} \qquad \frac{e}{f}=\frac{a}{b}-\frac{c}{d}=\frac{a\cdot d-b\cdot c}{b\cdot d} \qquad \frac{e}{f}=\frac{a}{b}\cdot\frac{c}{d}=\frac{a\cdot c}{b\cdot d} \qquad \frac{e}{f}=\frac{a}{b}\div\frac{c}{d}=\frac{a\cdot d}{b\cdot c}$$

Programma:

```
# Bewerkingen met breuken
from tkinter import *
def mult(s):
    l=s.split('/');co=[]
    for i in l:co.append(int(i))
    return co
def ggd(a,b):
    c=1
    while c!=0:c=a%b;a=b;b=c
    return a
def bereken(i):
    a,b=mult(E1.get());c,d= mult(E2.get())
    E3.delete(0,len(E3.get()))
    Label(form,text='a/b'+bew[i]+'c/d=',).place(x=5,y=50)
    # i=0: +  i=1: -  i=2: *  i=3: ÷
    if i==0:e=a*d+ b*c;f=b*d
```

```
    if i==1:e=a*d- b*c;f=b*d
    if i==2:e=a*c;f=b*d
    if i==3:e=a*d;f=b*c
    gd=ggd(e,f);e,f=e//gd,f//gd
    E3.insert(0,str(e)+'/'+str(f))
def nieuw():
    for ent in [E1,E2,E3]:
        ent.delete(0,len(ent.get()))
    Label(form,text='a/b..c/d=',).place(x=5,y=50)
def but(i):Button(form,text=bew[i],width=3,command=lambda:bereken(i)).place(x=5+35*i,y=75)
# hoofdprogramma
bew=['+','-','*','÷']
form=Tk();form.geometry('200x110');form.title('+-*÷')
Label(form,text='a/b=',).place(x=5,y=10);E1=Entry(form);E1.place(x=65,y=10)
Label(form,text='c/d=',).place(x=5,y=30);E2=Entry(form);E2.place(x=65,y=30)
E3=Entry(form,bg='lightgrey');E3.place(x=65,y=50)
but(0);but(1);but(2);but(3)
Button(form,text='Nieuw',command=nieuw).place(x=145,y=75);nieuw()
form.mainloop()
```



147. Bijzondere krommen: parametervergelijkingen en grafiek

paramvgl_paramcsv

Hierna volgt een lijstje van een aantal bijzondere krommen en hun parametervergelijking.
Het is gemaakt in Excel en gesaved als **'param.csv'**

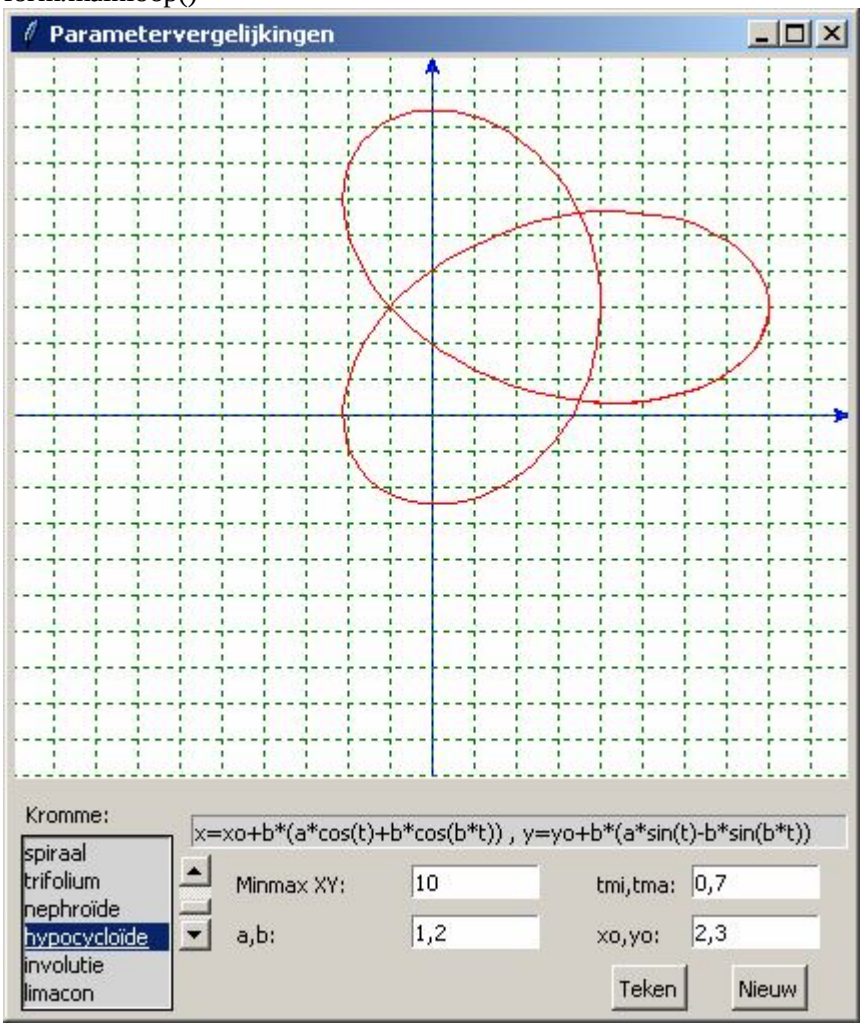
Naam	Vergelijking X	Vergelijking Y
ellips	a*cos(t)	b*sin(t)
parabool	a*t*t	t
cardioïde	2*a*cos(t)*(1-cos(t))	2*a*sin(t)*(1-cos(t))
cissoïde	2*a*t*t/(1+t*t)	2*a*t*t*t/(1+t*t)
lemniscaat	a*cos(2*t)*cos(t)	a*cos(2*t)*sin(t)
astroïde	a*cos(t)**3	a*sin(t)**3
cochleoïde	a*sin(t)*cos(t)/t	a*sin(t)**2/t
epicicloïde	a*cos(b*t)-b*cos(a*t)	a*sin(b*t)-b*sin(a*t)
spiraal	a*t*cos(t)	a*t*sin(t)
trifolium	a*cos(t)**2*(4*sin(t)**2-1)	a*cos(t)*sin(t)*(4*sin(t)**2-1)
nephroïde	a*cos(2*t)*(1+2*sin(t))	a*sin(2*t)*(1+2*sin(t))
hypocycloïde	b*(a*cos(t)+b*cos(b*t))	b*(a*sin(t)-b*sin(b*t))
involutie	a*(cos(t)+t*sin(t))	a*(sin(t)-t*cos(t))
limacon	(b+2*a*cos(t))*cos(t)	(b+2*a*cos(t))*sin(t)
lissajous	a*sin(b*t+1)	sin(t)
rhodenea	a*sin(b*t)*cos(t)	a*sin(b*t)*sin(t)
tricuspoïde	a*(2*cos(t)+cos(2*t))	a*(2*sin(t)-sin(2*t))

Een eerste programma ' Bijzondere krommen' laadt automatisch dit bestand in, en laat de gebruiker toe om de grafieken te tekenen: de parameters t en de variabelen a en b kunnen gewijzigd worden.

Programma:

```
#parametervergelijkingen met param.csv
from tkinter import *;from math import *
def mult(s):
   l=s.split(',')
   co=[]
   for i in l:co.append(float(i))
   return co
def transx(x):
   return (x-xmi)*breedte/(xma-xmi)
def transy(y):
   return hoogteC-hoogteC*(y-ymi)/(yma-ymi)
def fx(t):return eval(xst)
def fy(t):return eval(yst)
class Kromme:
   def __init__(self,naam,xs,ys):self.naam=naam;self.xs=xs;self.ys=ys
def teken():
   global xmi,xma,ymi,yma
   xma=float(E2.get());C.delete(ALL)
   yma=xma;zma=xma;xmi=-xma;ymi=xmi;zmi=xmi
#assen
   X=transx(0);line = C.create_line(X,0,X,hoogteC,fill='blue',arrow='first')
   Y=transy(0);line = C.create_line(0,Y,breedte,Y,fill='blue',arrow='last')
# rooster
   for x in range(int(xmi),int(xma+1)):
      X=transx(x);line = C.create_line(X,0,X,hoogteC,fill='green',dash=[1,2])
   for y in range(int(ymi),int(yma+1)):
      Y=transy(y);line = C.create_line(0,Y,breedte,Y,fill='green',dash=[1,2] )
#kromme
   global a,b,xst,yst
   xst=k[Lb.curselection()[0]].xs;yst=k[Lb.curselection()[0]].ys
   tmi,tma=mult(E4.get());a,b=mult(E3.get());xo,yo=mult(E5.get())
   lis=[];stap=0.05;t=tmi;E1.delete(0,len(E1.get()))
   while t<tma:
      t=t+stap;x=xo+fx(t);y=yo+fy(t);X=transx(x);Y=transy(y)
      lis.append(X);lis.append(Y)
   line = C.create_line(lis,fill='red')
   voorschrift='xo+'+k[Lb.curselection()[0]].xs+', yo+'+k[Lb.curselection()[0]].ys
   E1.insert(0,voorschrift)
def nieuw():
   for en in [E1,E2,E3,E4,E5]:en.delete(0,len(en.get()))
   C.delete(ALL)
# hoofdprogramma
fk=open('param.csv');inhoud=fk.read();fk.close()
linh=inhoud.split('\n');k=[]
while '' in linh:linh.remove('')
for i in range(2,len(linh)):li=linh[i].split(';');nm,x,y=li;k.append(Kromme(nm,x,y))
breedte=420;hoogte=560;hoogteC=hoogte-140;hoInv=hoogte-120
form=Tk();form.geometry(str(breedte)+'x'+str(hoogte));form.title('Parametervergelijkingen')
nml,xsl,ysl=[],[],[]
for kr in k:nml.append(kr.naam);xsl.append(kr.xs);ysl.append(kr.ys)
```

```
tupfun=tuple(nml)
Lb=Listbox(form,listvariable=StringVar(value=tupfun),selectmode='Single',height=6,width=12,bg='lightgrey')
Lb.place(x=5,y=hoInv+15)
vsb=Scrollbar(form,orient=VERTICAL,command=Lb.yview,width=25);Lb['yscrollcommand']=vsb.set;vsb.place(
x=82,y=hoInv+20)
C = Canvas(form, bg="white", height=hoogteC, width=breedte);C.pack()
L1=Label(form,text='Kromme:',);L1.place(x=5,y=hoInv-10)
Label(form,text='Minmax
XY:',).place(x=110,y=hoInv+25);E2=Entry(form,width=10);E2.place(x=200,y=hoInv+25)
Label(form,text='a,b:',).place(x=110,y=hoInv+50);E3=Entry(form,width=10);E3.place(x=200,y=hoInv+50)
Label(form,text='tmi,tma:',).place(x=290,y=hoInv+25);E4=Entry(form,width=10);E4.place(x=340,y=hoInv+25)
Label(form,text='xo,yo:',).place(x=290,y=hoInv+50);E5=Entry(form,width=10);E5.place(x=340,y=hoInv+50)
E2.insert(0,'10');E3.insert(0,'1,2');E4.insert(0,'0,7');E5.insert(0,'2,3')
Label(form,text='x,y=').place(x=100,y=hoInv)
E1=Entry(form,width=46,bg='lightgrey');E1.place(x=130,y=hoInv)
B1=Button(form,text='Teken',command=teken);B1.place(x=300,y=hoInv+75)
B2=Button(form,text='Nieuw',command=nieuw);B2.place(x=360,y=hoInv+75)
form.mainloop()
```



148: Omwentelingslichamen van bijzondere krommmen

Een tweede programma gebruikt hetzelfde bestand en laat deze krommen om de x-as wentelen. Je merkt dat voor de berekening van x,y,z de coordinaten [x,y,0] vermenigvuldigd worden met de rotatiematrix:

$$\begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos(v) & \sin(v) \\ 0 & -\sin(v) & \cos(v) \end{pmatrix}$$

In pythontaal:

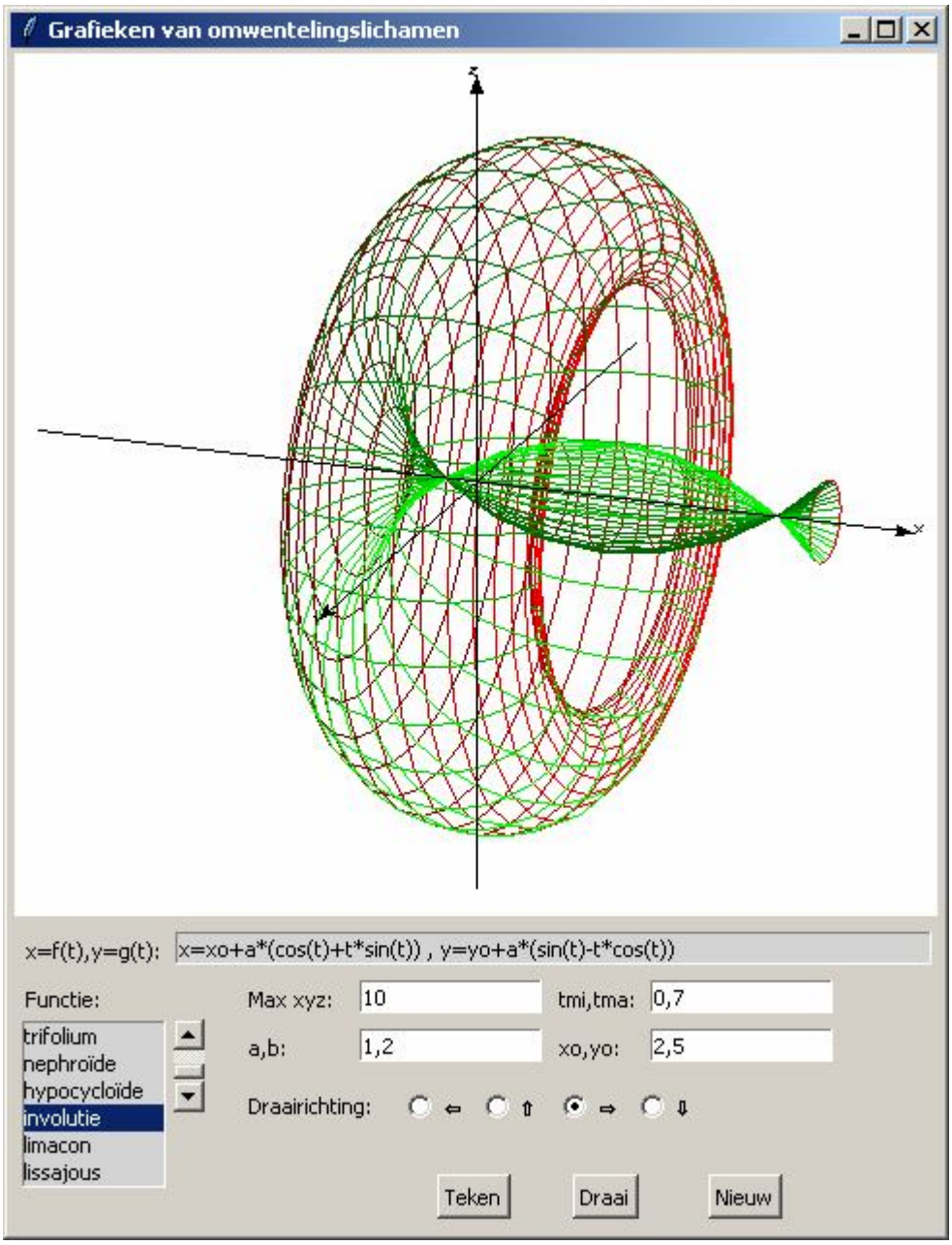
```
R=[[1,0,0],[0,cos(v),sin(v)],[0,-sin(v),cos(v)]]
V=[xo+fx(t),yo+fy(t),0]
x,y,z=prod(R,V)
X,Y=omzet(x,y,z)
```

Programma: omwentelingslichaam_paramcsv

```
# omwentelingslichamen van parametervergelijkingen
from math import *
from tkinter import *
def col(rgb):return '#%02x%02x%02x' % rgb
def mult(s):
    l=s.split(',');co=[]
    for i in l:co.append(float(i))
    return co
def prod(b,c):
    n=len(b);p=[]
    for i in range(0,n):
        som=0
        for k in range(0,n):
            som=som+b[i][k]*c[k]
        p.append(som)
    return(p)
def transx(x):
    return (x-xmi)*breedte/(xma-xmi)
def transy(y):
    return hoogteC-hoogteC*(y-ymi)/(yma-ymi)
def hoek(y,x):
    if y==0 and x>=0:c=0
    elif y==0 and x<0:c=pi
    elif x==0 and y>0:c=pi/2
    elif x==0 and y<0:c=3*pi/2
    else:
        c=atan(y/x)
        if x<0:c=c+pi
    return(c)
def omzet(x,y,z):
    c=hoek(y,x)
    r=sqrt(x*x+y*y);y1=sin(ah+c)*r
    d=hoek(y1,z)
    r1=sqrt(y1*y1+z*z)
    X=transx(cos(ah+c)*r);Y=transy(cos(bh+d)*r1)
    return [X,Y]
def asteken(x,y,z):
    lis=[]
    X,Y=omzet(-x,-y,-z);lis.append(X);lis.append(Y)
    X,Y=omzet(x,y,z);lis.append(X);lis.append(Y)
    line = C.create_line(lis,fill='black',arrow='last')
def sel():
```

```
    global rich;rich=var.get()
def teken():
    global xmi,xma,ymi,yma,zmi,zma,tmi,vmi,tma,vma,a,b,ah,bh,xst,yst
    xma=float(E2.get());C.delete(ALL)
    yma=xma;zma=xma;xmi=-xma;ymi=xmi;zmi=xmi
    tmi,tma=mult(E4.get());a,b=mult(E3.get())
    xo,yo=mult(E5.get());vmi,vma=-pi,pi
    ah=radians(draaixy);bh=radians(draaiyz)
    voorschrift='x=xo+'+k[Lb.curselection()[0]].xs+' , y=yo+'+k[Lb.curselection()[0]].ys
    E1.delete(0,len(E1.get()));E1.insert(0,voorschrift)
    xst=k[Lb.curselection()[0]].xs;yst=k[Lb.curselection()[0]].ys
#kromme
    stap=vma/15
#niveaulijnen tekenen voor constante u
    t=tmi
    while t<tma:
        ro=int(175+75*sin(t));kl=col((ro,0,0))
        t=t+stap;lis=[]
        V=[xo+fx(t),yo+fy(t),0]
        v=vmi
        while v<vma:
            v=v+stap
            R=[[1,0,0],[0,cos(v),sin(v)],[0,-sin(v),cos(v)]]
            x,y,z=prod(R,V)
            X,Y=omzet(x,y,z)
            lis.append(X);lis.append(Y)
        line = C.create_line(lis,fill=kl)
#niveaulijnen tekenen voor constante v
    v=vmi
    while v<vma:
        gr=int(175+75*sin(v));kl=col((0,gr,0))
        R=[[1,0,0],[0,cos(v),sin(v)],[0,-sin(v),cos(v)]]
        v=v+stap;lis=[]
        t=tmi
        while t<tma:
            t=t+stap
            V=[xo+fx(t),yo+fy(t),0];x,y,z=prod(R,V);X,Y=omzet(x,y,z)
            lis.append(X);lis.append(Y)
        line = C.create_line(lis,fill=kl)
#assen
    asteken(xma,0,0);asteken(0,yma,0);asteken(0,0,zma)
#x,y,z letters
    hl=0.17;st=xma+0.03
    lis=[omzet(st,0,0),omzet(st+hl,0,hl),omzet(st+hl/2,0,hl/2),omzet(st,0,hl),omzet(st+hl,0,0)]
    line = C.create_line(lis,fill='black') #x
    lis=[omzet(0,st,0),omzet(0,st+hl,hl),omzet(0,st+hl/2,hl/2),omzet(0,st,hl)]
    line = C.create_line(lis,fill='black') #y
    lis=[omzet(0,0,st),omzet(-hl,0,st),omzet(0,0,st+hl),omzet(-hl,0,st+hl)]
    line = C.create_line(lis,fill='black') #z
def draai():
    global draaixy,draaiyz
    if rich==0:draaixy=draaixy+30
    if rich==1:draaiyz=draaiyz-30
    if rich==2:draaixy=draaixy-30
    if rich==3:draaiyz=draaiyz+30
```

```
    teken()
def nieuw():
    for en in [E1,E2,E3,E4,E5]:en.delete(0,len(en.get()))
    global draaixy,draaiyz;draaixy=20;draaiyz=20;C.delete(ALL)
def fx(t):return eval(xst)
def fy(t):return eval(yst)
class Kromme:
    def __init__(self,naam,xs,ys):self.naam=naam;self.xs=xs;self.ys=ys
# hoofdprogramma
fk=open('param.csv');inhoud=fk.read();fk.close()
linh=inhoud.split('\n');k=[]
while '' in linh:linh.remove('')
for i in range(2,len(linh)):li=linh[i].split(';');nm,x,y=li;k.append(Kromme(nm,x,y))
breedte=480;hoogte=610;hoogteC=hoogte-165;hoInv=hoogte-155;hoInv2=hoogte-40
form=Tk();form.geometry(str(breedte)+'x'+str(hoogte))
form.title('Grafieken van omwentelingslichamen')
nml,xsl,ysl=[],[],[]
for kr in k:nml.append(kr.naam);xsl.append(kr.xs);ysl.append(kr.ys)
Label(form,text='Functie:').place(x=5,y=hoogte-130)
tupfun=tuple(nml)
Lb=Listbox(form,listvariable=StringVar(value=tupfun),selectmode='Single',height=6,width=12,bg='lightgrey')
Lb.place(x=5,y=hoogte-110)
vsb=Scrollbar(form,orient=VERTICAL,command=Lb.yview,width=25);Lb['yscrollcommand']=vsb.set;vsb.place(
x=80,y=500)
C = Canvas(form, bg="white", height=hoogteC, width=breedte)
C.pack();draaixy=20;draaiyz=20
L1=Label(form,text='x=f(t),y=g(t):',);L1.place(x=5,y=hoInv)
E1=Entry(form,bg=('lightgrey'));E1.place(x=85,y=hoInv,width=390)
L2=Label(form,text='Max xyz:',);L2.place(x=120,y=hoInv+25)
E2=Entry(form,width=15);E2.place(x=180,y=hoInv+25)
Label(form,text='a,b:',).place(x=120,y=hoInv+50);E3=Entry(form,width=15);E3.place(x=180,y=hoInv+50)
Label(form,text='tmi,tma:',).place(x=280,y=hoInv+25);E4=Entry(form,width=15);E4.place(x=330,y=hoInv+25)
Label(form,text='xo,yo:',).place(x=280,y=hoInv+50);E5=Entry(form,width=15);E5.place(x=330,y=hoInv+50)
E2.insert(0,'10');E3.insert(0,'1,2');E4.insert(0,'0,7');E5.insert(0,'2,3')
B1=Button(form,text='Teken',command=teken);B1.place(x=220,y=hoInv2)
B2=Button(form,text='Draai',command=draai);B2.place(x=290,y=hoInv2)
B4=Button(form,text='Nieuw',command=nieuw);B4.place(x=360,y=hoInv2)
pijl=['\u21E6','\u21E7','\u21E8','\u21E9'] #unicode pijlen
rich=0;var = IntVar()
L4=Label(form,text='Draairichting:');L4.place(x=120,y=hoInv+80)
for i in range(0,4):
    rb=Radiobutton(form,text=pijl[i],variable=var,value=i,command=sel)
    rb.place(x=200+40*i,y=hoInv+80)
form.mainloop()
```



149. Orbitalen van Neon neon

Een voorbeeld van 3-dimensionale omwentelingslichamen zijn de s en p-orbitalen van Neon.
De kern, alsook de 1s en 2s-orbitalen zijn omwentelingslichamen van de cirkel tussen -pi en pi. Om de p-orbitalen te simuleren kunnen we een stukje van een lemniscaat wentelen rond x-, y- en z-as. De afmetingen van de 5 orbitalen kunnen gewijzigd worden met andere getallen in te voeren in het 2p,1s,2s-venster.

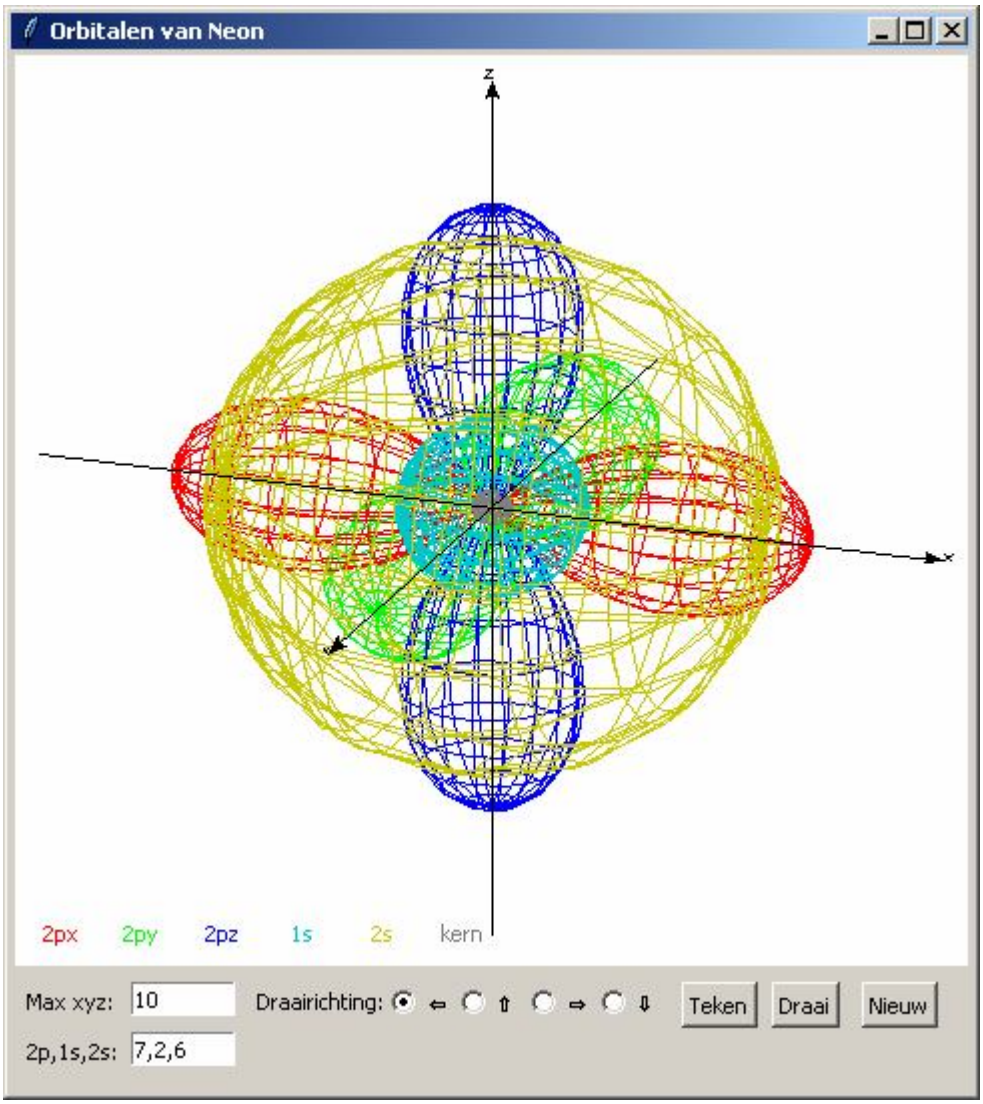
Programma:

```
# orbitalen van Neon
from math import *
from tkinter import *
def col(rgb):return '#%02x%02x%02x' % rgb
def mult(s):
    l=s.split(',');co=[]
    for i in l:co.append(float(i))
```

```
    return co
def prod(b,c):
    n=len(b);p=[]
    for i in range(0,n):
        som=0
        for k in range(0,n):
            som=som+b[i][k]*c[k]
        p.append(som)
    return(p)
def transx(x):return (x-xmi)*breedte/(xma-xmi)
def transy(y):return hoogteC-hoogteC*(y-ymi)/(yma-ymi)
def hoek(y,x):
    if y==0 and x>=0:c=0
    elif y==0 and x<0:c=pi
    elif x==0 and y>0:c=pi/2
    elif x==0 and y<0:c=3*pi/2
    else:
        c=atan(y/x)
        if x<0:c=c+pi
    return(c)
def omzet(x,y,z):
    c=hoek(y,x);r=sqrt(x*x+y*y);y1=sin(ah+c)*r
    d=hoek(y1,z);r1=sqrt(y1*y1+z*z)
    X=transx(cos(ah+c)*r);Y=transy(cos(bh+d)*r1)
    return [X,Y]
def asteken(x,y,z):
    lis=[];X,Y=omzet(-x,-y,-z);lis.append(X);lis.append(Y)
    X,Y=omzet(x,y,z);lis.append(X);lis.append(Y)
    line = C.create_line(lis,fill='black',arrow='last')
def sel():
    global rich;rich=var.get()
def vulin():
    X,Y=omzet(x,y,z);lis1.append(X);lis1.append(Y)
    X,Y=omzet(z,x,y);lis2.append(X);lis2.append(Y)
    X,Y=omzet(y,z,x);lis3.append(X);lis3.append(Y)
def tekenlijn():
    line = C.create_line(lis1,fill=kl1)
    line = C.create_line(lis2,fill=kl2)
    line = C.create_line(lis3,fill=kl3)
def vulin2():X,Y=omzet(x,y,z);lis.append(X);lis.append(Y)
def tekenlijn2(kl):line = C.create_line(lis,fill=kl)
def teken():
    global
x,y,z,xmi,xma,ymi,yma,zmi,zma,tmi,vmi,tma,vma,a,b,c,f,xo,yo,ah,bh,xst,yst,lis,lis1,lis2,lis3,kl1,kl2,kl3,kl4,kl5,kl
6,stapv,stapt
    global orb,kleur
    xma=float(E2.get());C.delete(ALL)
    yma=xma;zma=xma;xmi=-xma;ymi=xmi;zmi=xmi
    a,b,c=mult(E3.get())
    xo,yo=0,0;vmi,vma=-pi,11*pi/10
    ah=radians(draaixy);bh=radians(draaiyz)
#p-orbitalen
    xst='a*cos(2*t)*cos(t)';yst='a*cos(2*t)*sin(t)'
    f=10;stapv=vma/f
    kl1=col((254,0,0));kl2=col((0,254,0));kl3=col((0,0,254))
```

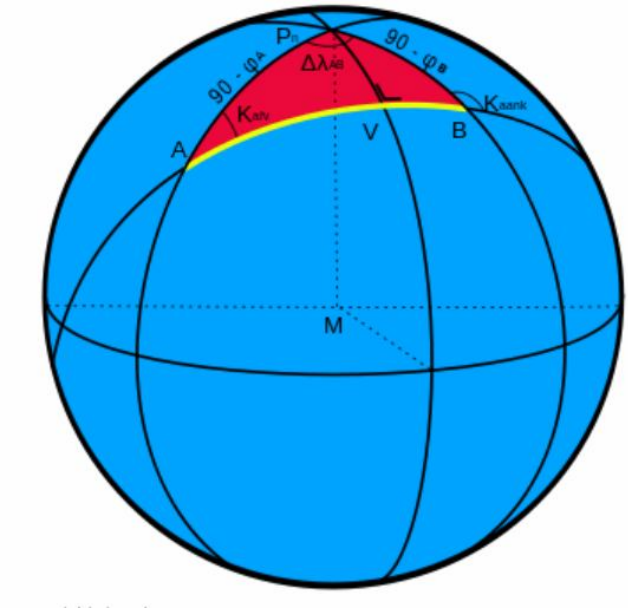
```
    kl4=col((0,200,200));kl5=col((200,200,0));kl6='grey'
#niveaulijnen tekenen voor constante t
    tmi=-0.1;tma=0.7;nivt()
    tmi=3.05;tma=3.9;nivt()
#niveaulijnen tekenen voor constante v
    tmi=-0.1;tma=0.7;nivv()
    tmi=3.05;tma=3.88;nivv()
#s-orbitalen
    tmi=-pi;tmax=11*pi/10
    xst='b*cos(t)';yst='b*sin(t)';nivt2(kl4);nivv2(kl4)
    xst='c*cos(t)';yst='c*sin(t)';nivt2(kl5);nivv2(kl5)
#neonkern
    xst='0.4*cos(t)';yst='0.4*sin(t)';nivt2(kl6)
#assen
    asteken(xma,0,0);asteken(0,yma,0);asteken(0,0,zma)
#x,y,z letters
    hl=0.17;st=xma+0.03
    lis=[omzet(st,0,0),omzet(st+hl,0,hl),omzet(st+hl/2,0,hl/2),omzet(st,0,hl),omzet(st+hl,0,0)]
    line = C.create_line(lis,fill='black') #x
    lis=[omzet(0,st,0),omzet(0,st+hl,hl),omzet(0,st+hl/2,hl/2), omzet(0,st,hl)]
    line = C.create_line(lis,fill='black') #y
    lis=[omzet(0,0,st),omzet(-hl,0,st),omzet(0,0,st+hl),omzet(-hl,0,st+hl)]
    line = C.create_line(lis,fill='black') #z
#verklaring orbitalen
    kleur=[kl1,kl2,kl3,kl4,kl5,kl6]; orb=['2px','2py','2pz','1s','2s','kern'];tekst()
def tekst():
    for i in range(0,6):
        C.create_text(25+40*i,440,text=orb[i],fill=kleur[i])
def nivt():
    global lis,lis1,lis2,lis3,x,y,z
    stapt=(tma-tmi)/f;t=tmi
    while t<tma:
        t=t+stapt;lis1=[];lis2=[];lis3=[];V=[xo+fx(t),yo+fy(t),0];v=vmi
        while v<vma:v=v+stapv;R=[[1,0,0],[0,cos(v),sin(v)],[0,-sin(v),cos(v)]];x,y,z=prod(R,V);vulin()
        tekenlijn()
def nivv():
    global lis1,lis2,lis3,x,y,z
    stapt=(tma-tmi)/f;v=vmi
    while v<vma:
        R=[[1,0,0],[0,cos(v),sin(v)],[0,-sin(v),cos(v)]]
        v=v+stapv;lis1=[];lis2=[];lis3=[];t=tmi
        while t<tma:t=t+stapt;V=[xo+fx(t),yo+fy(t),0];x,y,z=prod(R,V);vulin()
        tekenlijn()
def nivt2(kl):
    global lis,x,y,z
    stapt=tma/f;t=tmi
    while t<tma:
        t=t+stapt;lis=[];V=[xo+fx(t),yo+fy(t),0];v=vmi
        while v<vma:v=v+stapv;R=[[1,0,0],[0,cos(v),sin(v)],[0,-sin(v),cos(v)]];x,y,z=prod(R,V);vulin2()
        tekenlijn2(kl)
def nivv2(kl):
    global lis1,x,y,z
    stapt=tma/f;v=vmi
    while v<vma:
        R=[[1,0,0],[0,cos(v),sin(v)],[0,-sin(v),cos(v)]]
```

```
        v=v+stapv;lis=[];t=tmi
        while t<tma:t=t+stapt;V=[xo+fx(t),yo+fy(t),0];x,y,z=prod(R,V);vulin2()
        tekenlijn2(kl)
def draai():
    global draaixy,draaiyz
    if rich==0:draaixy=draaixy+30
    if rich==1:draaiyz=draaiyz-30
    if rich==2:draaixy=draaixy-30
    if rich==3:draaiyz=draaiyz+30
    teken()
def nieuw():
    global draaixy,draaiyz;draaixy=20;draaiyz=20;C.delete(ALL)
def fx(t):return eval(xst)
def fy(t):return eval(yst)
# hoofdprogramma
breedte=480;hoogte=520;hoogteC=hoogte-65;hoInv=hoogte-55
form=Tk();form.geometry(str(breedte)+'x'+str(hoogte))
form.title('Orbitalen van Neon')
C = Canvas(form, bg="white", height=hoogteC, width=breedte)
C.pack();draaixy=20;draaiyz=20
L2=Label(form,text='Max xyz:',);L2.place(x=5,y=hoInv)
E2=Entry(form,width=8);E2.place(x=60,y=hoInv)
Label(form,text='2p,1s,2s:',).place(x=5,y=hoInv+25);E3=Entry(form,width=8);E3.place(x=60,y=hoInv+25)
E2.insert(0,'10');E3.insert(0,'7,2,6')
B1=Button(form,text='Teken',command=teken);B1.place(x=335,y=hoInv)
B2=Button(form,text='Draai',command=draai);B2.place(x=380,y=hoInv)
B4=Button(form,text='Nieuw',command=nieuw);B4.place(x=425,y=hoInv)
pijl=['\u21E6','\u21E7','\u21E8','\u21E9'] #unicode pijlen
rich=0;var = IntVar()
L4=Label(form,text='Draairichting:');L4.place(x=120,y=hoInv)
for i in range(0,4):
rb=Radiobutton(form,text=pijl[i],variable=var,value=i,command=sel)
    rb.place(x=185+35*i,y=hoInv)
form.mainloop()
```



150. Afstand en kompaskoers navigatieTk

Gegeven de geografische coördinaten (breedtecirkel en meridiaan) van 2 plaatsen A en B op de aarde. Om van A naar B te gaan, moeten we de kompaskoers berekenen.
Noord (N) = 0° Oost (E) = 90° Zuid (S) =180° West (W) = 270°
Indien we deze koers over de ganse reis aanhouden, dan volgen we niet de kortste afstand maar wel een loxodroom, dit is een 3 dimensionale kromme die elke meridiaan met de zelfde hoek snijdt en als een spiraal naar Noord- of Zuidpool gaat. D.w.z. dat als we over een grote afstand steeds dezelfde koers volgen, we helemaal niet in onze bestemming aankomen.
Wensen we de kortste afstand af te leggen, dan moeten we navigeren langs een grootcirkel (orthodroom) van de aardbol. Voor korte afstanden is er weinig verschil tussen beide routes.
In het volgende programma wordt de afstand en de kompaskoers berekend tussen 2 steden A en B. Je merkt dat de kompaskoers bij aankomst verschillend is van deze bij vertrek.
Om dus uiteindelijk +/- in B uit te komen, moet de koers, gelijkmatig verdeeld over de ganse reis gewijzigd worden van de koers in A tot de koers in B



(Wikipedia)
We stellen:
$\varphi_A : FA,\ \varphi_B : FB,\ \lambda_A : LA,\ \lambda_B : LB,$ dan is
$\Delta\lambda_{AB} = \lambda_B - \lambda_A = LB - LA$
De afstand tussen punt A met breedte FA en meridiaan LA en punt B met breedte FB en meridiaan LB kan bepaald worden aan de hand van boldriehoek AP_nB met behulp van de eerste cosinusregel. Daarbij is de zijde AP_n het complement van de breedte FA, de zijde P_nB het complement van de breedte FB en de hoek P_n het meridiaanverschil LB-LA tussen vertrek A en bestemming B.
De afstand AB wordt dan gegeven door AB= acos[sinFA.sinFB+ cosFA.cosFB.cos(LB-LA)]
De kompaskoers vinden we met de formule: K=

$$\text{acos}\left(\frac{\text{sinFB} - \text{sinFA.cosAB}}{\text{cosFA.sinAB}}\right)$$

Indien LB-LA<0, dan nemen we 360-K

Opmerkingen:
-0mdat een grootcirkel een andere kromme is dan een loxodroom, zal bij grootcirkelnavigatie de te volgen kompaskoers continu wijzigen van vertrekpunt tot aankomst. Van Amsterdam naar Dublin bijvoorbeeld, verandert deze kompashoek van 282° naar 273° over een afstand van 750 km. Om concreet zo goed mogelijk de juiste koers te volgen, kan bijvoorbeeld deze afstand verdeeld worden in 9 gelijke stukken, waarin de koers telkens één graad wordt verminderd.
-De Noordpool valt niet samen met het magnetische Noorden. Om de echte 'grondkoers' te bepalen, moeten we een correctie (variatie) toepassen op de kompaskoers. Hiervoor heeft men variatietabellen nodig.
In het programma definiëren we een klasse Point.
Elk 'Point' dat we definiëren krijgt een naam (stad), een breedtegraad en een meridiaan in ° en ' . Deze Points worden ingelezen uit een csv-bestand 'geo'. Om de afstand en de kompaskoers te berekenen, moeten in de listbox 2 steden A en B gemarkeerd worden. Met de knop 'Bereken' wordt zowel de kompaskoers van A naar B als van B naar A berekend.

Programma:

```
#Afstand en kompaskoers
from math import *;from tkinter import *
class Point:
    def __init__(self,name,lat,mer):self.name=name;self.lat=lat;self.mer=mer
def val(s):
    co=1;ls=len(s)-1
    if s[ls]=='W' or s[ls]=='S':co=-1
    s=s[0:ls];gm=s[0:].split('.')
    if gm[1]=='':dec=0
    else:dec=float(gm[1])/60
    co=co*(float(gm[0])+dec)
    return co*pi/180
def mult(geo):g=geo.split(',');return val(g[0]),val(g[1])
def deg(x):
    gdec=180*x/pi;gint=int(gdec)
    gmin=round((gdec-gint)*60)
    return(str(gint)+'.'+str(gmin)+'deg')
def komp(FX,FY,LX,LY,AB):
```

```
    angle=acos((sin(FY)-sin(FX)*cos(AB))/cos(FX)/sin(AB))
    if LY<LX:angle=2*pi-angle
    return angle
def nieuw():E2.delete(0,len(E2.get()));E3.delete(0,len(E3.get()));wis()
def wis():tex.delete('1.0',END)
def ad(lijn):tex.insert(INSERT,lijn+' ')
def cont(a):
    c=pi+a
    if c>2*pi:c=c-2*pi
    return c
def bereken():
    global geoA,geoB,bv,LB,LA,FB,FA
    lico=[];plaats=[]
    for p in Lb.curselection():lico.append(k[p].lat);lico.append(k[p].mer);plaats.append(k[p].name)
    FA,LA,FB,LB=lico #F=breedtecirkel L=meridiaan
    nieuw();E2.insert(0,FA+','+LA);E3.insert(0,FB+','+LB);
    wis();geoA=E2.get();geoB=E3.get()
    FA,LA=mult(geoA);FB,LB=mult(geoB);bv=LB-LA
    AB=acos(sin(FB)*sin(FA)+cos(FB)*cos(FA)*cos(bv))
    am=format(60*180*AB/pi,'.0f');ak=format(60*180*AB*1.852/pi,'.0f')
    Acomp=komp(FA,FB,LA,LB,AB);Bcomp=komp(FB,FA,LB,LA,AB)
    answ='A= '+plaats[0]+'  B= '+plaats[1]
    answ=answ+'\nAfstand AB= '+ak+' km ='+am+' ml\nKoers A ==> B'
    answ=answ+'\nIn A :'+deg(Acomp)+'\nIn B :'+deg(cont(Bcomp))
    answ=answ+'\nKoers B ==> A'
    answ=answ+'\nIn B :'+deg(Bcomp)+'\nIn A :'+deg(cont(Acomp))
    ad(answ)
#main
gr='lightgrey';wid=420;high=160;highC=high-140;hoInv=high-120
form=Tk();form.geometry(str(wid)+'x'+str(high));form.title('Afstand en kompaskoers')
fk=open('geo.csv');inhoud=fk.read();fk.close();linh=inhoud.split('\n');k=[]
while '' in linh:linh.remove('')
for i in range(2,len(linh)):li=linh[i].split(';');name,lat,mer=li; k.append(Point(name,lat,mer))
nml,latl,merl=[],[],[]
for kr in k:nml.append(kr.name);latl.append(kr.lat);merl.append(kr.mer)
Lb=Listbox(form,listvariable=StringVar(value=tuple(nml)),selectmode=MULTIPLE,height=5,width=12,bg=gr)
Lb.place(x=5,y=25)
vsb=Scrollbar(form,orient=VERTICAL,command=Lb.yview,width=25);Lb['yscrollcommand']=vsb.set;vsb.place(
x=78,y=30)
L1=Label(form,text='Kies 2 steden A en B:',);L1.place(x=5,y=3)
L2=Label(form,text='brc A,mer A:');L2.place(x=5,y=100)
E2=Entry(form,width=15,bg=gr);E2.place(x=80,y=100)
L3=Label(form,text='brc B,mer B:');L3.place(x=5,y=130)
E3=Entry(form,width=15,bg=gr);E3.place(x=80,y=130)
tex=Text(form,bg=gr,height=9,width=30,wrap=WORD);tex.place(x=180,y=8)
vsb=Scrollbar(form,orient=VERTICAL,command=tex.yview,width=25)
tex['yscrollcommand']=vsb.set;vsb.pack(side=RIGHT,fill='y')
B1=Button(form,text='Bereken',command=bereken);B1.place(x=120,y=30)
B2=Button(form,text=' Nieuw ',command=nieuw);B2.place(x=120,y=60)
form.mainloop()
```

151. De lineaire functie linfunctie

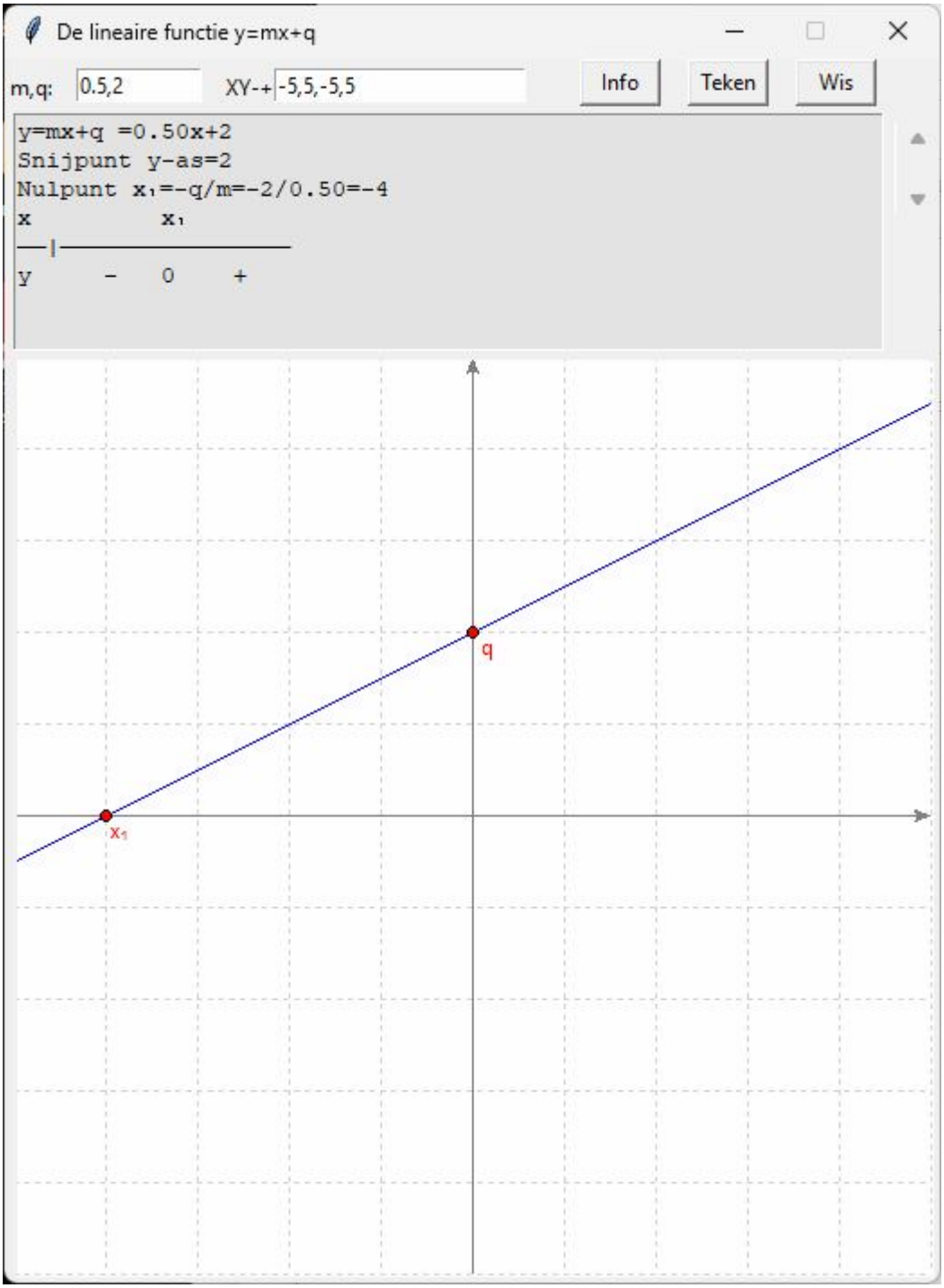
Dit programma kan gebruikt worden bij het bespreken van nulpunten, tekenverloop en grafiek van de lineaire functie.

Programma

```
# Lineaire functie met tkinter
from math import *;from tkinter import *;from tkinter import messagebox
def mult(s):
    l=s.split(',');co=[]
    for i in l:co.append(float(i))
    return co
def transx(x):return (x-xmi)*breedteC/(xma-xmi)
def transy(y):return hoogteC-hoogteC*(y-ymi)/(yma-ymi)
def ad(lin):
    tex.insert(INSERT,lin)
def sg(x):
    if x==int(x):x=int(x);sx=str(x)
    else:sx=format(x,'.2f')
    if x>=0:return '+'+sx
    else: return sx
def sg2(x):
    if x==int(x):x=int(x);sx=str(x)
    else:sx=format(x,'.2f')
    return sx
def ap(li,x,y):X=transx(x);Y=transy(y);li.append(X);li.append(Y)
def punt(x,y,te,kl):
    diamx=(xma-xmi)/200;diamy=(yma-ymi)/200;d=3*diamx
    diam=0.1;lis=[];ap(lis,x-diamx,y-diamy);ap(lis,x+diamx,y+diamy);C.create_oval(lis,fill=kl)
    lis=[];ap(lis,x+d,y-d);C.create_text(lis,text=te,fill=kl)
def bereken():
#tekst
    tex.delete('1.0',END)
    m,q=mult(E1.get())
    ad('y=mx+q ='+sg2(m)+'x'+sg(q))
    ad('\nSnijpunt y-as='+sg2(q))
    if m!=0:
        x1=-q/m
        ad('\nNulpunt x1=-q/m='+sg2(-q)+'/'+sg2(m)+'='+sg2(x1))
```

```
        ad('\nx         x1\n')
        ad('———|————————————————————————————\n')
        if m>0:ad('y     -   0    +')
        if m<0:ad('y     +   0    -')
    elif m==0:
        ad('\nGeen nulpunten')
        ad('x\n')
        ad('———|————————————————————————————\n')
        if q>0:ad('y       +')
        if q<0:ad('y       -')
#grafiek
#rooster
    C.delete(ALL)
    global xmi,xma,ymi,yma;xmi,xma,ymi,yma=mult(E2.get())
    for x in range(int(xmi),int(xma+1)):
        X=transx(x);C.create_line(X,0,X,hoogteC,fill='lightgrey',dash=[1,2])
    for y in range(int(ymi),int(yma+1)):
        Y=transy(y);C.create_line(0,Y,breedteC,Y,fill='lightgrey',dash=[1,2] )
#assen
    X=transx(0);C.create_line(X,0,X,hoogteC,fill='grey',arrow='first')
    Y=transy(0);C.create_line(0,Y,breedteC,Y,fill='grey',arrow='last')
#rechte
    lis=[];ap(lis,xmi,m*xmi+q);ap(lis,xma,m*xma+q)
    C.create_line(lis,fill=col[0])
#punten
    if m!=0:punt(x1,0,'x1',col[1])
    punt(0,q,'q',col[1])
    return
def wis(E):E.delete(0,len(E.get()))
def nieuw():
     for E in (E1,E2):wis(E)
     C.delete(ALL);tex.delete('1.0',END)
def info():h=messagebox.showinfo(tit,infstr)
#hoofdprogramma
col=['blue','red','green']
breedte=520;hoogte=680;tit='De lineaire functie y=mx+q'
hoogteC=510;breedteC=510
form=Tk();form.geometry(str(breedte)+'x'+str(hoogte));form.title(tit)
form.resizable(False,False)
C = Canvas(form, bg="white", height=hoogteC, width=breedteC);C.place(x=5,y=165)
L1=Label(form,text='m,q:',);L1.place(x=0,y=5)
E1=Entry(form);E1.place(x=40,y=5,width=70);E1.insert(0,'0.5,2')
Label(form,text='XY-+:').place(x=120,y=5)
E2=Entry(form);E2.place(x=150,y=5,width=140);E2.insert(0,'-5,5,-5,5')
tex=Text(form,bg='grey89',height=8,width=60,wrap=WORD);tex.place(x=5,y=30)
vsb=Scrollbar(form,orient=VERTICAL,command=tex.yview,width=25)
tex['yscrollcommand']=vsb.set;vsb.place(x=495,y=35)
Button(form,text='Info',width=5,command=info).place(x=320,y=0)
Button(form,text='Teken',width=5,command=bereken).place(x=380,y=0)
Button(form,text='Wis',width=5,command=nieuw).place(x=440,y=0)
infstr='Geef waarden voor m en q\nen klik op "Teken"'
```

form.mainloop()



152. De kwadratische functie kwadfunctie

Dit programma kan gebruikt worden bij het bespreken van nulpunten, ontbinding in factoren, tekenverloop en grafiek van de kwadratische functie.

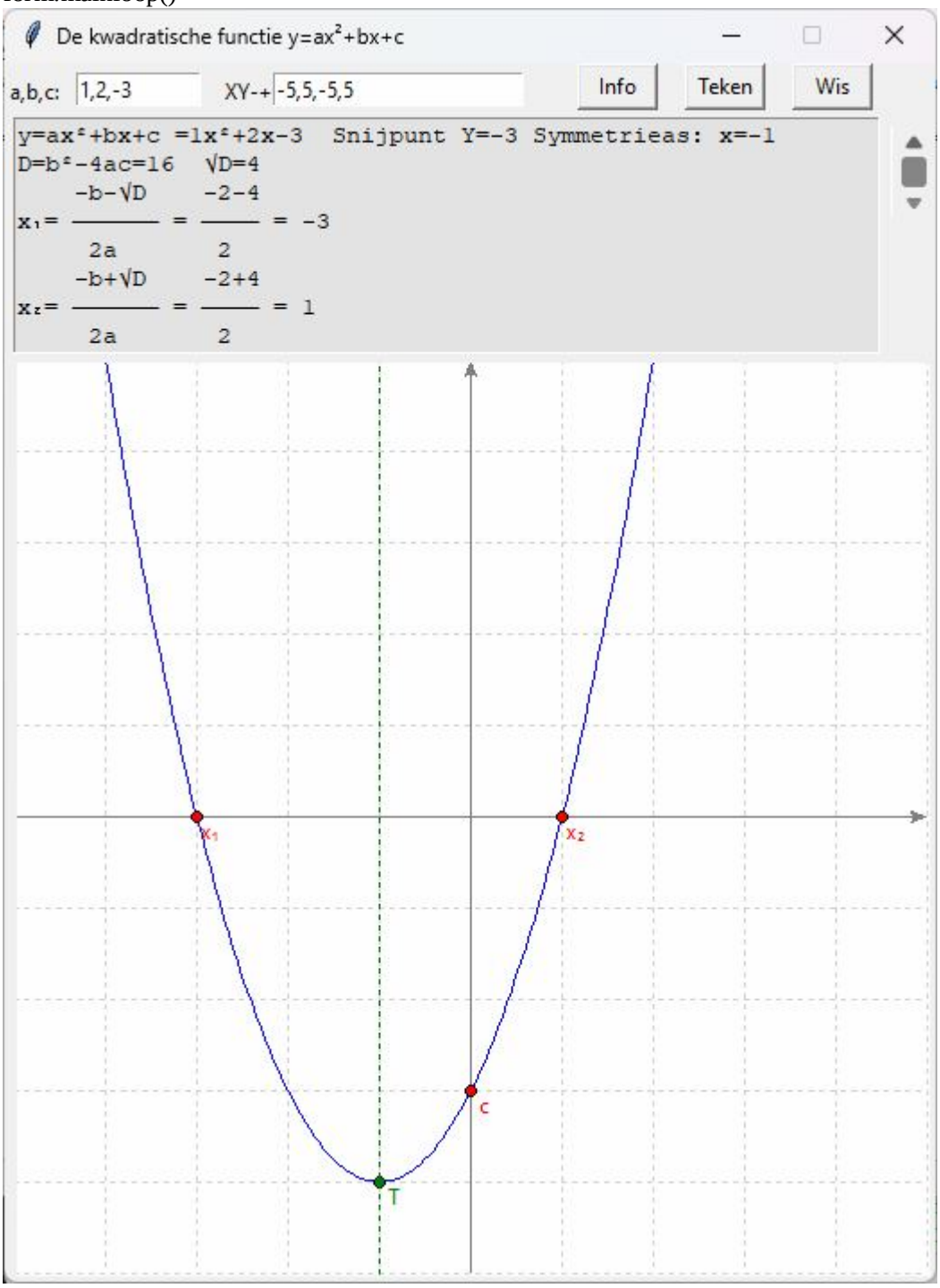
Programma

Kwadratische functie met tkinter

```
from math import *;from tkinter import *;from tkinter import messagebox
def mult(s):
   l=s.split(',');co=[]
   for i in l:co.append(float(i))
   return co
def transx(x):return (x-xmi)*breedteC/(xma-xmi)
def transy(y):return hoogteC-hoogteC*(y-ymi)/(yma-ymi)
def ad(lin):
   tex.insert(INSERT,lin)
def sg(x):
   if x==int(x):x=int(x);sx=str(x)
   else:sx=format(x,'.2f')
   if x>=0:return '+'+sx
   else: return sx
def sg2(x):
   if x==int(x):x=int(x);sx=str(x)
   else:sx=format(x,'.2f')
   return sx
def ap(li,x,y):X=transx(x);Y=transy(y);li.append(X);li.append(Y)
def punt(x,y,te,kl):
   diamx=(xma-xmi)/200;diamy=(yma-ymi)/200;d=3*diamx
   diam=0.1;lis=[];ap(lis,x-diamx,y-diamy);ap(lis,x+diamx,y+diamy);C.create_oval(lis,fill=kl)
   lis=[];ap(lis,x+d,y-d);C.create_text(lis,text=te,fill=kl)
def bereken():
#tekst
   tex.delete('1.0',END)
   a,b,c=mult(E1.get());D=b*b-4*a*c
   if a==0:messagebox.showinfo(tit+':fout','Dit is geen kwadratische functie');return
   ad('y=ax²+bx+c ='+sg2(a)+'x²'+sg(b)+'x'+sg(c))
   ad('  Snijpunt Y='+sg2(c)+' Symmetrieas: x='+sg2(-b/a/2))
   ad('\nD=b²-4ac='+sg2(D))
   if D<0:
      ad('\nGeen oplossingen\nGeen ontbinding\n')
      ad('x\n')
      ad('——|————————————————————————\n')
      if a>0:ad('y       +')
      if a<0:ad('y       -')
   elif D==0:
      ad('\nx₁=x₂= -b/2a ='+sg2(-b/2/a))
      ad('\ny=a(x-x₁)²='+sg2(a)+'(x'+sg(b/2/a)+')²')
      ad('x         x₁\n')
      ad('——|————————————|————————————\n')
      if a>0:ad('y     +   0    +')
      if a<0:ad('y     -   0    -')
   else:
      sqD=sqrt(D);x1=(-b-sqD)/2/a;x2=(-b+sqD)/2/a
      if x1>x2:wissel=x1;x1=x2;x2=wissel
      ad('  √D='+sg2(sqD))
      tel1=sg(-b)+sg(-sqD);tel2=sg(-b)+sg(sqD)
      breuk=''.ljust(len(tel1),'—')
      ad('\n    -b-√D    '+tel1)
```

```
        ad('\nx1= ————————— = '+breuk+' = '+sg2(x1))
        ad('\n     2a       '+sg2(2*a))
        ad('\n    -b+√D    '+tel2)
        ad('\nx2= ————————— = '+breuk+' = '+sg2(x2))
        ad('\n     2a       '+sg2(2*a))
        ad('\ny=a(x-x1)(x-x2)='+sg2(a)+'(x'+sg(-x1)+')(x'+sg(-x2)+')\n')
        ad('x         x1        x2\n')
        ad('——|————————————|————————————|————————\n')
        if a>0:ad('y     +   0    -    0   +')
        if a<0:ad('y     -   0    +    0   -')
#grafiek
    #rooster
    C.delete(ALL)
    global xmi,xma,ymi,yma;xmi,xma,ymi,yma=mult(E2.get())
    for x in range(int(xmi),int(xma+1)):
        X=transx(x);C.create_line(X,0,X,hoogteC,fill='lightgrey',dash=[1,2])
    for y in range(int(ymi),int(yma+1)):
        Y=transy(y);C.create_line(0,Y,breedteC,Y,fill='lightgrey',dash=[1,2] )
#assen
    X=transx(0);C.create_line(X,0,X,hoogteC,fill='grey',arrow='first')
    Y=transy(0);C.create_line(0,Y,breedteC,Y,fill='grey',arrow='last')
#parabool
    stap=0.05;x=xmi-stap;lis=[]
    while x<xma:x=x+stap;ap(lis,x,a*x*x+b*x+c)
    C.create_line(lis,fill=col[0])
#punten
    if D>0:punt(x1,0,'x1',col[1]);punt(x2,0,'x2',col[1])
    elif D==0:punt(x1,0,'x1',col[1])
    punt(0,c,'c',col[1]);punt(-b/2/a,-D/4/a,'T',col[2])
#symmetrieas
    lis=[];ap(lis,-b/2/a,ymi);ap(lis,-b/2/a,xma)
    C.create_line(lis,fill=col[2],dash=[1,2])
    return
def wis(E):E.delete(0,len(E.get()))
def nieuw():
     for E in (E1,E2):wis(E)
     C.delete(ALL);tex.delete('1.0',END)
def info():h=messagebox.showinfo(tit,infstr)
#hoofdprogramma
col=['blue','red','green']
breedte=520;hoogte=680;tit='De kwadratische functie y=ax²+bx+c'
hoogteC=510;breedteC=510
form=Tk();form.geometry(str(breedte)+'x'+str(hoogte));form.title(tit)
form.resizable(False,False)
C = Canvas(form, bg="white", height=hoogteC, width=breedteC);C.place(x=5,y=165)
L1=Label(form,text='a,b,c:',);L1.place(x=0,y=5)
E1=Entry(form);E1.place(x=40,y=5,width=70);E1.insert(0,'1,2,-3')
Label(form,text='XY-+:').place(x=120,y=5)
E2=Entry(form);E2.place(x=150,y=5,width=140);E2.insert(0,'-5,5,-5,5')
tex=Text(form,bg='grey89',height=8,width=60,wrap=WORD);tex.place(x=5,y=30)
vsb=Scrollbar(form,orient=VERTICAL,command=tex.yview,width=25)
```

```
tex['yscrollcommand']=vsb.set;vsb.place(x=495,y=35)
Button(form,text='Info',width=5,command=info).place(x=320,y=0)
Button(form,text='Teken',width=5,command=bereken).place(x=380,y=0)
Button(form,text='Wis',width=5,command=nieuw).place(x=440,y=0)
infstr='Geef waarden voor a,b,c\nen klik op "Teken"'
form.mainloop()
```



Oefenprogramma's

In de volgende 4 programma's worden bij het klikken op de button 'Teken' een paar gegevens van een functie getekend: punten of raaklijnen. De gebruiker krijgt hiermee net genoeg informatie om het voorschrift van de functie te berekenen. Het antwoord bestaat steeds uit gehele getallen.

Is het fout, dan krijgt hij een foutmelding. Hij krijgt dan telkens de grafiek te zien van de functie die overeenstemt met de gegevens die hij ingevuld heeft. (in rode stippellijn)
Is het goed, dan wordt de correcte grafiek in volle blauwe lijn overtekend.
Als hij zonder berekening het juiste antwoord wil zien, tikt de gebruiker op 'Antw'

Let hier ook op de betekenis van de instructies:
B1["state"] = DISABLED
B2["state"]=NORMAL

153. Bepalen voorschrift $y=ax^2+bx+c$ (1) bepalen2degraad_1

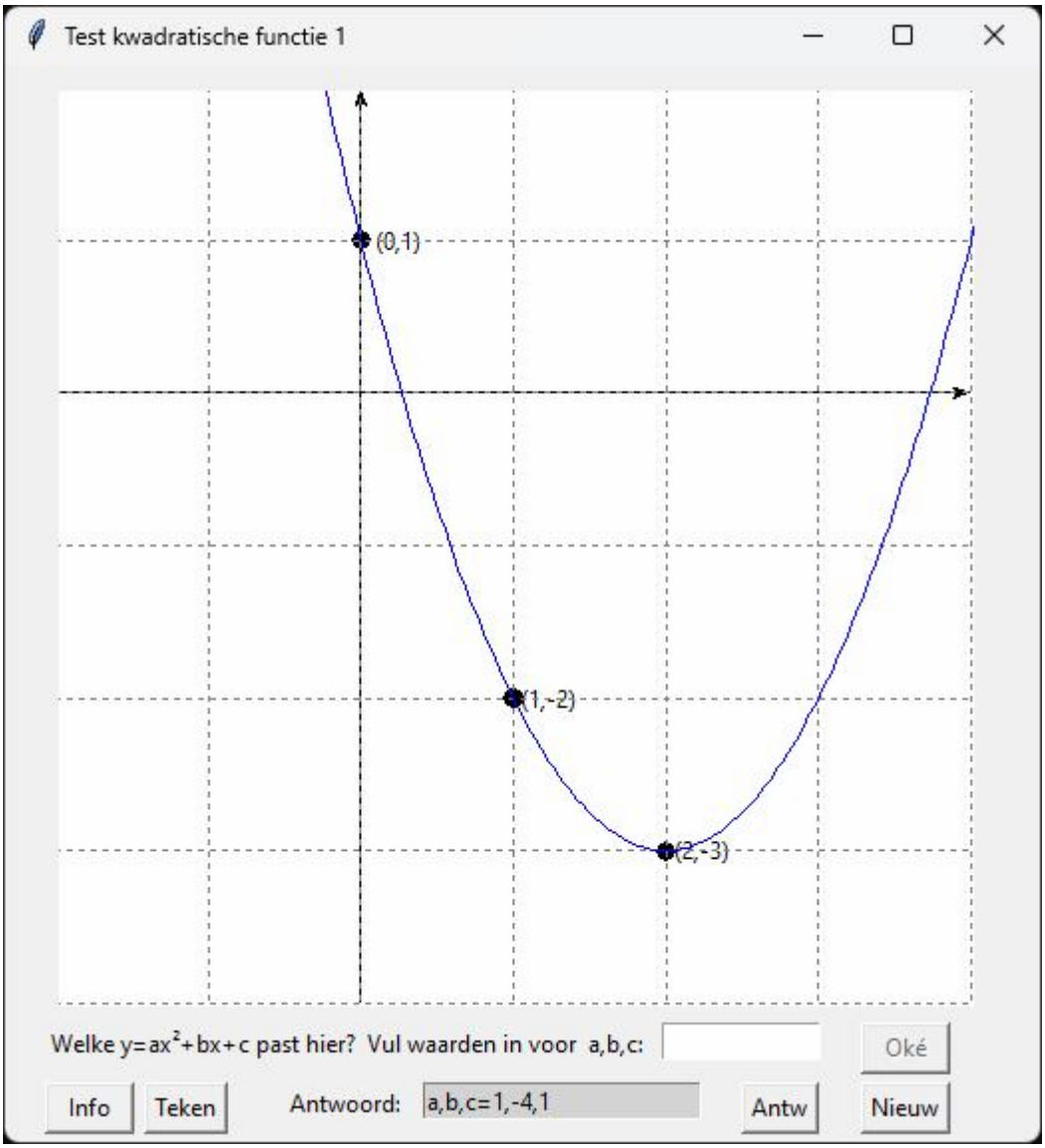
Telkens de gebruiker op 'Teken' tikt, kiest het programma random 3 punten met gegeven coördinaten.
De gebruiker moet a,b,c invullen zodat de grafiek van $y=ax^2+bx+c$ door deze punten gaat.

Programma

```
# Bepalen voorschrift y=ax²+bx+c
from math import *;from tkinter import *;from tkinter import messagebox;from random import randint as ri
def mult(s):
    l=s.split(',');co=[]
    for i in l:co.append(float(i))
    return co
def transx(x):return (x-xmi)*breedteC/(xma-xmi)
def transy(y):return hoogteC-hoogteC*(y-ymi)/(yma-ymi)
def f(x):return eval('a*x**2+b*x+c')
def aanpas(x):
    global xmi,xma,ymi,yma
    f1=f(x)
    if f1>=yma:yma=f1+1
    if f1<=ymi:ymi=f1-1
def teken():
    B1["state"] = DISABLED;B2["state"]=NORMAL
    global xmi,xma,ymi,yma,a,b,c,d
    C.delete(ALL);wis(E1);wis(E2);a=0;x2=0;x3=0
    while a==0:a=ri(-2,2)
    b=ri(-4,4);c=ri(-4,4)
    x1=ri(-2,2)
    while x2==x1:x2=ri(-2,2);
    while x3==x1 or x3==x2:x3=ri(-2,2)
#aanpassen assenstelsel
    xmi,ymi=-3,-3;xma,yma=3,3;aanpas(x1);aanpas(x2);aanpas(x3)
    axma=xmi+yma-ymi
    if axma>xma:xma=axma
#tekenen grafiek
#rooster
    for x in range(int(xmi),int(xma+1)):
        X=transx(x);C.create_line(X,0,X,hoogteC,fill='lightgrey',dash=[1,2])
    for y in range(int(ymi),int(yma+1)):
        Y=transy(y);C.create_line(0,Y,breedteC,Y,fill='lightgrey',dash=[1,2] )
#assen
    X=transx(0);C.create_line(X,0,X,hoogteC,fill='grey',arrow='first')
```

```
   Y=transy(0);C.create_line(0,Y,breedteC,Y,fill='grey',arrow='last')
#2 punten tekenen +coord
   punt(x1,f(x1));punt(x2,f(x2));punt(x3,f(x3))
#assen
   X=transx(0);C.create_line(X,0,X,hoogteC,fill='black',arrow='first')
   Y=transy(0);C.create_line(0,Y,breedteC,Y,fill='black',arrow='last')
def test():
   a1,b1,c1=mult(E1.get())
   stap=0.1;x=xmi-stap;li=[];y=ymi-1
   while x<xma:
      x=x+stap
      ap(li,x,a1*x**2+b1*x+c1)
   C.create_line(li,fill='red',dash=[1,2])
   if a!=a1 or b!=b1 or c!=c1:
      C.create_line(li,fill='red',dash=[1,2])
      messagebox.showinfo('Test:','fout')
   else:
      messagebox.showinfo('Test:','goed');B1["state"] = NORMAL;B2["state"]=DISABLED
      C.create_line(li,fill='blue');antw()
   return
def antw():
   wis(E2);E2.insert(0,'a,b,c='+str(a)+','+str(b)+','+str(c))
   B1["state"] = NORMAL;B2["state"]=DISABLED
   stap=(xma-xmi)/200;x=xmi-stap;li=[];y=ymi-1
   while x<xma:x=x+stap;ap(li,x,f(x))
   C.create_line(li,fill='blue')
   return
def punt(a,b):
   rx=(xma-xmi)/200;ry=(yma-ymi)/200
   lis=[];ap(lis,a-rx,b-ry);ap(lis,a+rx,b+ry);C.create_oval(lis,fill='black')
   lis=[];ap(lis,a+6*rx,b+4*ry);C.create_text(lis,text='('+format(a,'.0f')+','+format(b,'.0f')+')')
def ap(li,x,y):X=transx(x);Y=transy(y);li.append(X);li.append(Y)
def info():h=messagebox.showinfo(tit,infstr)
def wis(E):E.delete(0,len(E.get()))
def nieuw():wis(E1);wis(E2);C.delete(ALL)
#hoofdprogramma
tit='Test kwadratische functie 1'
lg='lightgrey';nwk='.4f';dx=1E-6;d2x=1E-4
breedte=520;hoogte=540;breedteC=460;hoogteC=460;hoInv1=hoogte-60;hoInv2=hoogte-30
form=Tk();form.geometry('520x540');form.title(tit)
C = Canvas(form, bg="white", height=hoogteC, width=breedteC);C.place(x=25,y=10)
Label(form,text='Welke y=ax²+bx+c past hier?  Vul waarden in voor').place(x=20,y=hoInv1)
Label(form,text='a,b,c:').place(x=290,y=hoInv1)
E1=Entry(form);E1.place(x=330,y=hoInv1,width=80)
Label(form,text='Antwoord:').place(x=140,y=hoInv2)
E2=Entry(form,bg='lightgrey');E2.place(x=210,y=hoInv2,width=140)
Button(form,text='Info',command=info).place(x=20,y=hoInv2,width=45)
B1=Button(form,text='Teken',command=teken);B1.place(x=70,y=hoInv2);B1["state"]=NORMAL
B2=Button(form,text='Oké',command=test,width=5);B2.place(x=430,y=hoInv1);B2["state"]=DISABLED
B3=Button(form,text='Antw',command=antw);B3.place(x=370,y=hoInv2)
Button(form,text='Nieuw',command=nieuw,width=5).place(x=430,y=hoInv2)
```

```
infstr="Tik op 'Teken': je krijgt 3 punten\n"
infstr=infstr+'Bepaal a,b,c zodat y=ax²+bx+c\n'
infstr=infstr+'past aan deze kromme\n'
infstr=infstr+"Tik dan op 'Oké'\n"
infstr=infstr+"Vind je het niet, tik op 'Antw'"
form.mainloop()
```



154. Bepalen voorschrift y=ax²+bx+c (2) bepalen2degraad_2

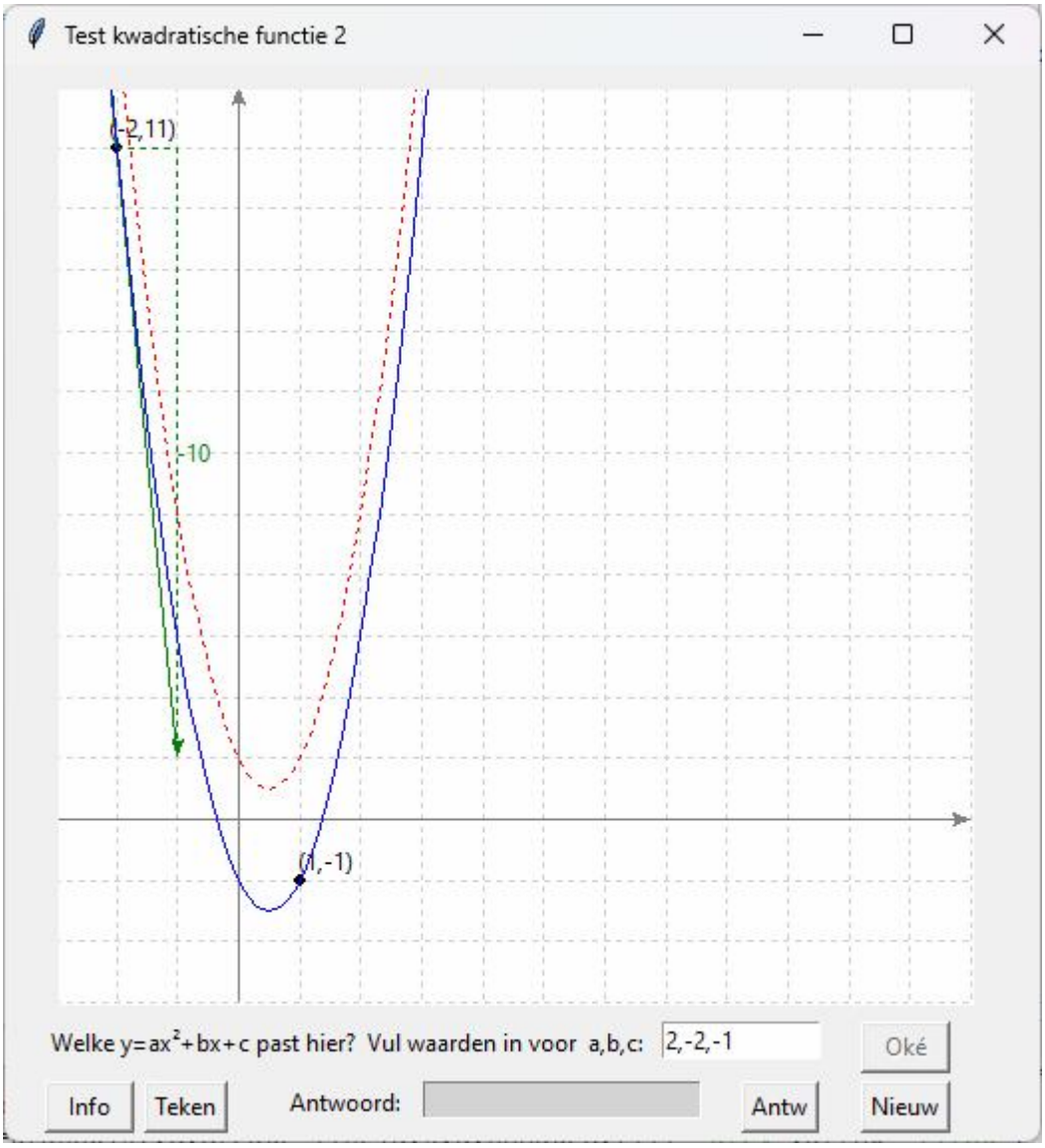
Dit programma is bijna analoog met het vorige maar nu krijgt de gebruiker slechts 2 punten, alsook de raaklijn (+rico) in één van beide punten. Elementaire kennis van afgeleide functies is hier vereist.

Programma

```
# Bepalen voorschrift y=ax²+bx+c
from math import *;from tkinter import *;from tkinter import messagebox;from random import randint as ri
def mult(s):
    l=s.split(',');co=[]
    for i in l:co.append(float(i))
```

```
    return co
def transx(x):return (x-xmi)*breedteC/(xma-xmi)
def transy(y):return hoogteC-hoogteC*(y-ymi)/(yma-ymi)
def f(x):return eval('a*x**2+b*x+c')
def df(x):return (2*a*x+b)
def aanpas(x):
    global xmi,xma,ymi,yma
    f1=f(x);f1d=f(x)+df(x)
    if f1>=yma:yma=f1+1
    if f1<=ymi:ymi=f1-1
    if x==x1 and f1d>=yma:yma=f1d+1
    if x==x1 and f1d<=ymi:ymi=f1d-1
def teken():
    B1["state"] = DISABLED;B2["state"]=NORMAL
    global xmi,xma,ymi,yma,x1,a,b,c,d
    C.delete(ALL);wis(E1);wis(E2)
    b=ri(-4,4);c=ri(-4,4);d=ri(-4,4);a=ri(-1,2)
    while a==0:a=ri(-1,2)
    x1=ri(-2,2);x2=ri(x1+1,3)
#aanpassen assenstelsel
    xmi,xma=-3,4;ymi,yma=-3,4;aanpas(x1);aanpas(x2)
    axma=xmi+yma-ymi
    if axma>xma:xma=axma
#tekenen grafiek
#rooster
    for x in range(int(xmi),int(xma+1)):
        X=transx(x);C.create_line(X,0,X,hoogteC,fill='lightgrey',dash=[1,2])
    for y in range(int(ymi),int(yma+1)):
        Y=transy(y);C.create_line(0,Y,breedteC,Y,fill='lightgrey',dash=[1,2] )
#assen
    X=transx(0);C.create_line(X,0,X,hoogteC,fill='grey',arrow='first')
    Y=transy(0);C.create_line(0,Y,breedteC,Y,fill='grey',arrow='last')
#2 punten tekenen +coord
    punt(x1,f(x1));punt(x2,f(x2))
#raaklijn in x1,f(x1)
    li=[];ap(li,x1-1,f(x1)-df(x1));ap(li,x1+1,f(x1)+df(x1));C.create_line(li,fill='green')
#rico raaklijn
    li=[];ap(li,x1+1,f(x1));ap(li,x1+1,f(x1)+df(x1));C.create_line(li,fill='green',arrow='last',dash=[1,2])
    li=[];ap(li,x1,f(x1));ap(li,x1+1,f(x1));C.create_line(li,fill='green',dash=[1,2])
    li=[];ap(li,x1+1.25,f(x1)+df(x1)/2);C.create_text(li,fill='green',text=format(df(x1),'.0f'))
def test():
    a1,b1,c1=mult(E1.get())
    stap=0.1;x=xmi-stap;li=[];y=ymi-1
    while x<xma:
        x=x+stap
        ap(li,x,a1*x**2+b1*x+c1)
    C.create_line(li,fill='red',dash=[1,2])
    if a!=a1 or b!=b1 or c!=c1:
        C.create_line(li,fill='red',dash=[1,2])
        messagebox.showinfo('Test:','fout')
    else:
```

```
        messagebox.showinfo('Test:','goed');B1["state"] = NORMAL;B2["state"]=DISABLED
        C.create_line(li,fill='blue');antw()
    return
def antw():
    wis(E2);E2.insert(0,'a,b,c='+str(a)+','+str(b)+','+str(c))
    B1["state"] = NORMAL;B2["state"]=DISABLED
    stap=(xma-xmi)/200;x=xmi-stap;li=[];y=ymi-1
    while x<xma:x=x+stap;ap(li,x,f(x))
    C.create_line(li,fill='blue')
    return
def punt(a,b):
    rx=(xma-xmi)/200;ry=(yma-ymi)/200
    lis=[];ap(lis,a-rx,b-ry);ap(lis,a+rx,b+ry);C.create_oval(lis,fill='black')
    lis=[];ap(lis,a+6*rx,b+4*ry);C.create_text(lis,text='('+format(a,'.0f')+','+format(b,'.0f')+')')
def ap(li,x,y):X=transx(x);Y=transy(y);li.append(X);li.append(Y)
def info():h=messagebox.showinfo(tit,infstr)
def wis(E):E.delete(0,len(E.get()))
def nieuw():wis(E1);wis(E2);C.delete(ALL)
#hoofdprogramma
tit='Test kwadratische functie 2'
lg='lightgrey';nwk='.4f';dx=1E-6;d2x=1E-4
breedte=520;hoogte=540;breedteC=460;hoogteC=460;hoInv1=hoogte-60;hoInv2=hoogte-30
form=Tk();form.geometry('520x540');form.title(tit)
C = Canvas(form, bg="white", height=hoogteC, width=breedteC);C.place(x=25,y=10)
Label(form,text='Welke y=ax²+bx+c past hier?  Vul waarden in voor').place(x=20,y=hoInv1)
Label(form,text='a,b,c:').place(x=290,y=hoInv1)
E1=Entry(form);E1.place(x=330,y=hoInv1,width=80)
Label(form,text='Antwoord:').place(x=140,y=hoInv2)
E2=Entry(form,bg='lightgrey');E2.place(x=210,y=hoInv2,width=140)
Button(form,text='Info',command=info).place(x=20,y=hoInv2,width=45)
B1=Button(form,text='Teken',command=teken);B1.place(x=70,y=hoInv2);B1["state"]=NORMAL
B2=Button(form,text='Oké',command=test,width=5);B2.place(x=430,y=hoInv1);B2["state"]=DISABLED
B3=Button(form,text='Antw',command=antw);B3.place(x=370,y=hoInv2)
Button(form,text='Nieuw',command=nieuw,width=5).place(x=430,y=hoInv2)
infstr="'Tik op 'Teken': je krijgt 2 punten\n"
infstr=infstr+'en de raaklijn(+rico) in een punt\n'
infstr=infstr+'Bepaal a,b,c zodat y=ax²+bx+c\n'
infstr=infstr+'past aan deze kromme\n'
infstr=infstr+"'Tik dan op 'Oké'\n"
infstr=infstr+"'Vind je het niet, tik op 'Antw'"
form.mainloop()
```



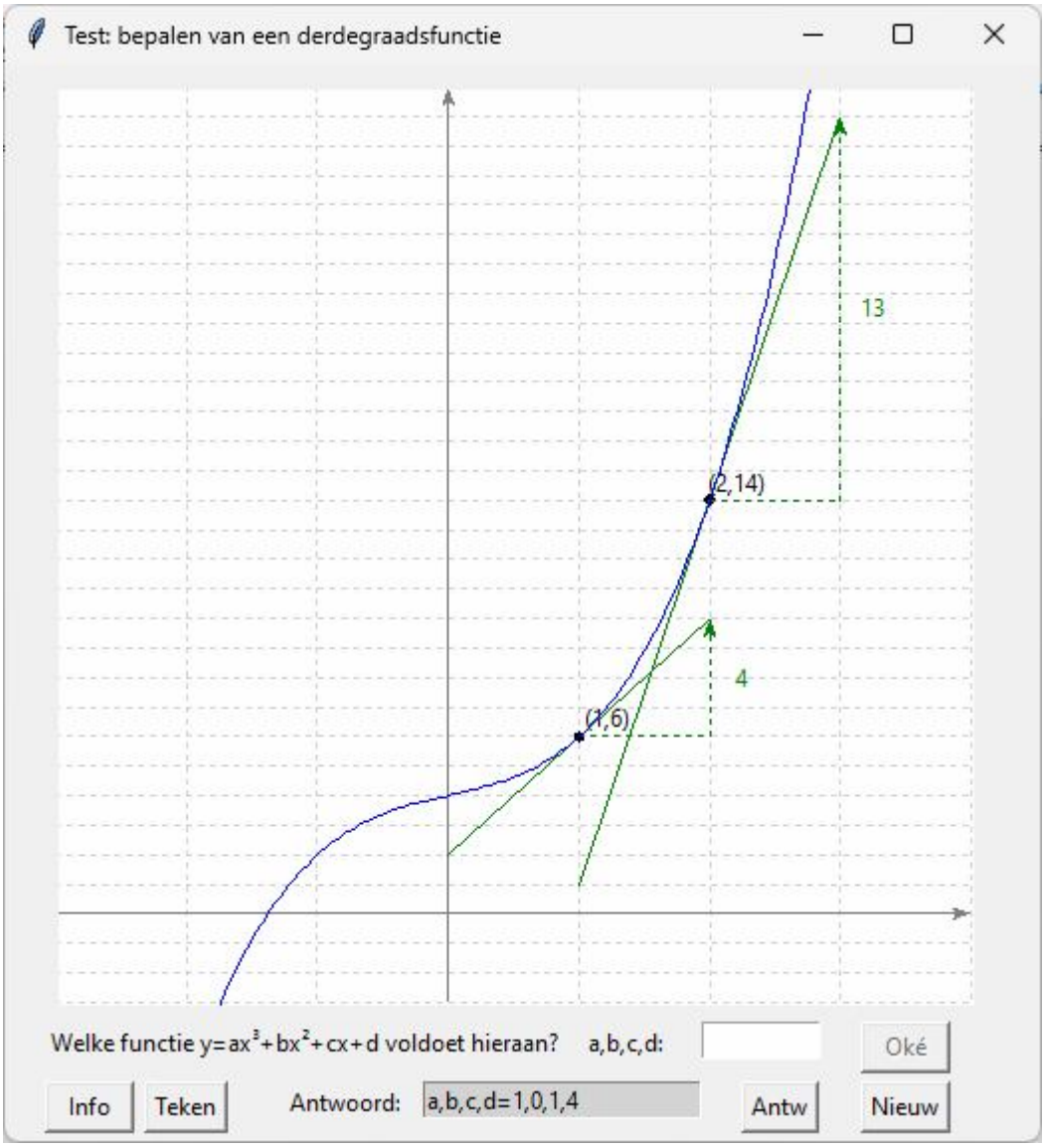
155. Bepalen voorschrift $y=ax^3+bx^2+cx+d$ (3) bepalen3degraad

Van een derde graadsfunctie ziet de gebruiker telkens 2 punten en de raaklijnen (+ rico) in deze punten. Hiermee heeft hij voldoende informatie om het voorschrift te bepalen. Omdat de y-waarden bij een derde graadsfunctie meestal veel groter zijn dan de x-waarden, zal dan de grafiek in de y-richting samengedrukt worden.

```
# Bepalen voorschtrift y=ax³+bx²+cx+d
from math import *;from tkinter import *;from tkinter import messagebox;from random import randint as ri
def mult(s):
    l=s.split(',');co=[]
    for i in l:co.append(float(i))
    return co
def transx(x):return (x-xmi)*breedteC/(xma-xmi)
def transy(y):return hoogteC-hoogteC*(y-ymi)/(yma-ymi)
def f(x):return eval('a*x**3+b*x**2+c*x+d')
def df(x):return (f(x+dx)-f(x-dx))/dx/2
def aanpas(x):
    global xmi,xma,ymi,yma
```

```
    f1=f(x);f1d=f(x)+df(x)
    if f1>=yma:yma=f1+1
    if f1<=ymi:ymi=f1-1
    if f1d>=yma:yma=f1d+1
    if f1d<=ymi:ymi=f1d-1
def teken():
    B1["state"] = DISABLED;B2["state"]=NORMAL
    global xmi,xma,ymi,yma,a,b,c,d
    C.delete(ALL);wis(E1);wis(E2)
    b=ri(-1,1);c=ri(-3,3);d=ri(-4,4);a=ri(-1,1)
    while a==0:a=ri(-1,1)
    x1=ri(-2,1);x2=x1+1
#aanpassen assenstelsel
    xmi,xma=-3,4;ymi,yma=-3,4
    aanpas(x1);aanpas(x2)
#tekenen grafiek
#rooster
    for x in range(int(xmi),int(xma+1)):
        X=transx(x);C.create_line(X,0,X,hoogteC,fill='lightgrey',dash=[1,2])
    for y in range(int(ymi),int(yma+1)):
        Y=transy(y);C.create_line(0,Y,breedteC,Y,fill='lightgrey',dash=[1,2] )
#assen
    X=transx(0);C.create_line(X,0,X,hoogteC,fill='grey',arrow='first')
    Y=transy(0);C.create_line(0,Y,breedteC,Y,fill='grey',arrow='last')
#2 punten tekenen
    punt(x1,f(x1));punt(x2,f(x2))
#raaklijnen in deze punten
    li=[];ap(li,x1-1,f(x1)-df(x1));ap(li,x1+1,f(x1)+df(x1));C.create_line(li,fill='green')
    li=[];ap(li,x2-1,f(x2)-df(x2));ap(li,x2+1,f(x2)+df(x2));C.create_line(li,fill='green')
#rico raaklijn 1
    li=[];ap(li,x1+1,f(x1));ap(li,x1+1,f(x1)+df(x1));C.create_line(li,fill='green',arrow='last',dash=[1,2])
    li=[];ap(li,x1,f(x1));ap(li,x1+1,f(x1));C.create_line(li,fill='green',dash=[1,2])
    li=[];ap(li,x1+1.25,f(x1)+df(x1)/2);C.create_text(li,fill='green',text=format(df(x1),'.0f'))
#rico raaklijn 2
    li=[];ap(li,x2+1,f(x2));ap(li,x2+1,f(x2)+df(x2));C.create_line(li,fill='green',arrow='last',dash=[1,2])
    li=[];ap(li,x2,f(x2));ap(li,x2+1,f(x2));C.create_line(li,fill='green',dash=[1,2])
    li=[];ap(li,x2+1.25,f(x2)+df(x2)/2);C.create_text(li,fill='green',text=format(df(x2),'.0f'))
def test():
    a1,b1,c1,d1=mult(E1.get())
    stap=(xma-xmi)/200;x=xmi-stap;li=[];y=ymi-1
    while x<xma:
        x=x+stap
        ap(li,x,a1*x**3+b1*x**2+c1*x+d1)
    C.create_line(li,fill='red',dash=[1,2])
    if a!=a1 or b!=b1 or c!=c1 or d!=d1:messagebox.showinfo('Test:','fout')
    else:messagebox.showinfo('Test:','goed');antw()
    return
def antw():
    wis(E2);E2.insert(0,'a,b,c,d='+str(a)+','+str(b)+','+str(c)+','+str(d))
    B1["state"] = NORMAL;B2["state"]=DISABLED
    stap=(xma-xmi)/200;x=xmi-stap;li=[];y=ymi-1
```

```
    while x<xma:x=x+stap;ap(li,x,f(x))
    C.create_line(li,fill='blue')
    return
def punt(a,b):
    rx=(xma-xmi)/200;ry=(yma-ymi)/200
    lis=[];ap(lis,a-rx,b-ry);ap(lis,a+rx,b+ry);C.create_oval(lis,fill='black')
    lis=[];ap(lis,a+6*rx,b+4*ry);C.create_text(lis,text='('+format(a,'.0f')+','+format(b,'.0f')+')')
def ap(li,x,y):X=transx(x);Y=transy(y);li.append(X);li.append(Y)
def info():h=messagebox.showinfo(tit,infstr)
def wis(E):E.delete(0,len(E.get()))
def nieuw():wis(E1);wis(E2);C.delete(ALL)
#hoofdprogramma
tit='Test: bepalen van een derdegraadsfunctie'
lg='lightgrey';nwk='.4f';dx=1E-6;d2x=1E-4
breedte=520;hoogte=540;breedteC=460;hoogteC=460;hoInv1=hoogte-60;hoInv2=hoogte-30
form=Tk();form.geometry('520x540');form.title(tit)
C = Canvas(form, bg="white", height=hoogteC, width=breedteC);C.place(x=25,y=10)
Label(form,text='Welke functie y=ax³+bx²+cx+d voldoet hieraan? ').place(x=20,y=hoInv1)
Label(form,text='a,b,c,d:').place(x=290,y=hoInv1)
E1=Entry(form);E1.place(x=350,y=hoInv1,width=60)
Label(form,text='Antwoord:').place(x=140,y=hoInv2)
E2=Entry(form,bg=lg);E2.place(x=210,y=hoInv2,width=140)
Button(form,text='Info',command=info).place(x=20,y=hoInv2,width=45)
B1=Button(form,text='Teken',command=teken);B1.place(x=70,y=hoInv2);B1["state"]=NORMAL
B2=Button(form,text='Oké',command=test,width=5);B2.place(x=430,y=hoInv1);B2["state"]=DISABLED
B3=Button(form,text='Antw',command=antw);B3.place(x=370,y=hoInv2)
Button(form,text='Nieuw',command=nieuw,width=5).place(x=430,y=hoInv2)
infstr="Tik op 'Teken': je krijgt 2 punten\n"
infstr=infstr+'en de raaklijn van  een\n3de graadsfunctie te zien:\n'
infstr=infstr+"Bepaal a,b,c,d en tik dan op 'Oké'\n"
infstr=infstr+"Vind je het niet, tik op 'Antw'"
form.mainloop()
```



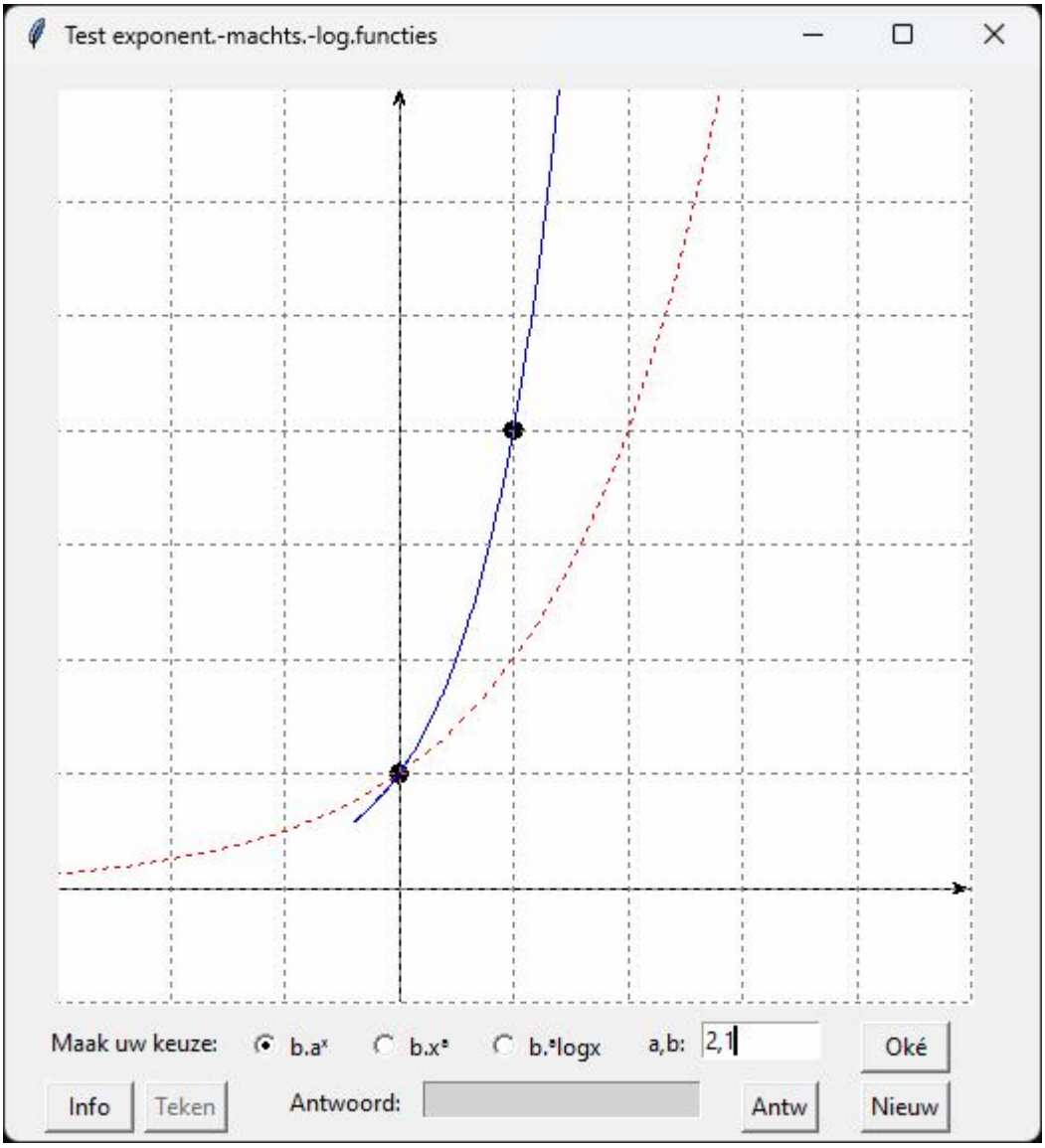
156. Bepalen voorschrift $y=b.a^x$ of $y=b.x^a$ of $y=b.{}^a\log x$ (4) exmalog

Nog een oefening op hetzelfde stramien: nu krijgt de gebruiker 2 punten met daartussen een stukje kromme. Hij moet kiezen (radiobutton) tussen een exponentiële, een machts- of een logaritmische functie, en de waarde van a en b bepalen.

```
# exmalog met tkinter
from math import *;from tkinter import *;from tkinter import messagebox;from random import randint as ri
def mult(s):
    l=s.split(',');co=[]
    for i in l:co.append(float(i))
    return co
def transx(x):return (x-xmi)*breedteC/(xma-xmi)
def transy(y):return hoogteC-hoogteC*(y-ymi)/(yma-ymi)
def f(x):return eval(E1.get())
def df(x):return (f(x+dx)-f(x-dx))/dx/2
def teken():
    B1["state"] = DISABLED;B2["state"]=NORMAL
```

```
    global xmi,xma,ymi,yma,keu,a,b
    C.delete(ALL);wis(E1);wis(E2)
    keu=ri(0,2);a=ri(2,5);b=ri(1,4)
    while b*a>10:b=ri(1,4)
#aanpassen assenstelsel aan keu,a,b
    if keu==0 and b*a>0:yma=b*a+3;ymi=-1
    if keu==0 and b*a<0:ymi=b*a-3;yma=1
    if keu==1 and b>0:yma=b+3;ymi=-1
    if keu==1 and b<0:ymi=b-3;yma=1
    if keu==0 or keu==1:xmi=-3;xma=xmi+yma-ymi
    if keu==2: xma=a+3;yma=b+3;xmi=-3;ymi=xmi+yma-xma
#2 punten tekenen + kromme ertussen
    if keu==0:punt(1,b*a);punt(0,b);kromme(a,b,-0.4,1.4)
    if keu==1:punt(1,b);punt(0,0);kromme(a,b,-0.4,1.4)
    if keu==2:punt(a,b);punt(1,0);kromme(a,b,0.6,a+0.4)
#assen
    X=transx(0);line = C.create_line(X,0,X,hoogteC,fill='black',arrow='first')
    Y=transy(0);line = C.create_line(0,Y,breedteC,Y,fill='black',arrow='last')
#rooster
    for x in range(int(xmi),int(xma+1)):
        X=transx(x);line = C.create_line(X,0,X,hoogteC,fill='gray',dash=[1,2])
    for y in range(int(ymi),int(yma+1)):
        Y=transy(y);line = C.create_line(0,Y,breedteC,Y,fill='gray',dash=[1,2] )
def test():
    a1,b1=mult(E1.get())
    stap=0.1;x=xmi-stap;li=[];y=ymi-1
    if k==2 and a1<=1:messagebox.showinfo('Foutieve invoer:','Bij log-functie moet a >0 en ≠ 0');return
    while x<xma:
        x=x+stap
        if k==0:y=b1*a1**x
        if k==1:y=b1*x**a1
        if k==2 and x>0:y=b1*log(x)/log(a1)
        if y!=ymi-1:ap(li,x,y)
    C.create_line(li,fill='red',dash=[1,2])
    if (keu==k and a==a1 and b==b1) or (keu==k and keu==2 and abs(a1**b-a**b1)<1E-4):
        messagebox.showinfo('Test:','goed');B1["state"] =
NORMAL;B2["state"]=DISABLED;kromme(a,b,xmi,xma)
    else:messagebox.showinfo('Test:','fout')
    return
def kromme(a,b,van,tot):
#tekenen grafiek
    stap=(tot-van)/200;x=van-stap;li=[];y=ymi-1
    while x<tot:
        x=x+stap
        if keu==0:y=b*a**x
        if keu==1:y=b*x**a
        if keu==2 and x>0:y=b*log(x)/log(a)
        if y>10*ymi and y<10*yma:ap(li,x,y)
    C.create_line(li,fill='blue')
def antw():
    wis(E2);E2.insert(0,keuze[keu]+'  a,b='+str(a)+','+str(b))
```

```
   B1["state"] = NORMAL;B2["state"]=DISABLED;kromme(a,b,xmi,xma)
   return
def punt(a,b):
   r=(xma-xmi)/100
   lis=[];ap(lis,a-r,b-r);ap(lis,a+r,b+r);C.create_oval(lis,fill='black')
def selec():global k;k=var.get()
def ap(li,x,y):X=transx(x);Y=transy(y);li.append(X);li.append(Y)
def info():h=messagebox.showinfo(tit,infstr)
def wis(E):E.delete(0,len(E.get()))
def nieuw():wis(E1);wis(E2);C.delete(ALL)
#hoofdprogramma
tit='Test exponent.-machts.-log.functies'
lg='lightgrey';nwk='.4f';dx=1E-6;d2x=1E-4
breedte=520;hoogte=540;breedteC=460;hoogteC=460;hoInv1=hoogte-60;hoInv2=hoogte-30
form=Tk();form.geometry('520x540');form.title(tit)
C = Canvas(form, bg="white", height=hoogteC, width=breedteC);C.place(x=25,y=10)
Label(form,text='Maak uw keuze:').place(x=20,y=hoInv1)
Label(form,text='a,b:').place(x=320,y=hoInv1)
E1=Entry(form);E1.place(x=350,y=hoInv1,width=60)
Label(form,text='Antwoord:').place(x=140,y=hoInv2)
E2=Entry(form,bg='lightgrey');E2.place(x=210,y=hoInv2,width=140)
var=IntVar();k=0;keuze=['b.a^x','b.x^a','b.^alogx']
for i in range(0,3):
   rb=Radiobutton(form,text=keuze[i],variable=var,value=i,command=selec)
   rb.place(x=120+i*60,y=hoInv1)
Button(form,text='Info',command=info).place(x=20,y=hoInv2,width=45)
B1=Button(form,text='Teken',command=teken);B1.place(x=70,y=hoInv2);B1["state"]=NORMAL
B2=Button(form,text='Oké',command=test,width=5);B2.place(x=430,y=hoInv1);B2["state"]=DISABLED
B3=Button(form,text='Antw',command=antw);B3.place(x=370,y=hoInv2)
Button(form,text='Nieuw',command=nieuw,width=5).place(x=430,y=hoInv2)
infstr="Tik op 'Teken': je krijgt de grafiek\n"
infstr=infstr+'van  een functie te zien: ofwel\n'
infstr=infstr+'y=b.a^x of y=b.x^a of y=b.^alogx\n'
infstr=infstr+'Kies de juiste functie en vul\n'
infstr=infstr+"a en b in. Tik dan op 'Oké'\n"
infstr=infstr+"Vind je het niet, tik op 'Antw'"
form.mainloop()
```



157. Periodiek sparen + grafiek periodiek sparen

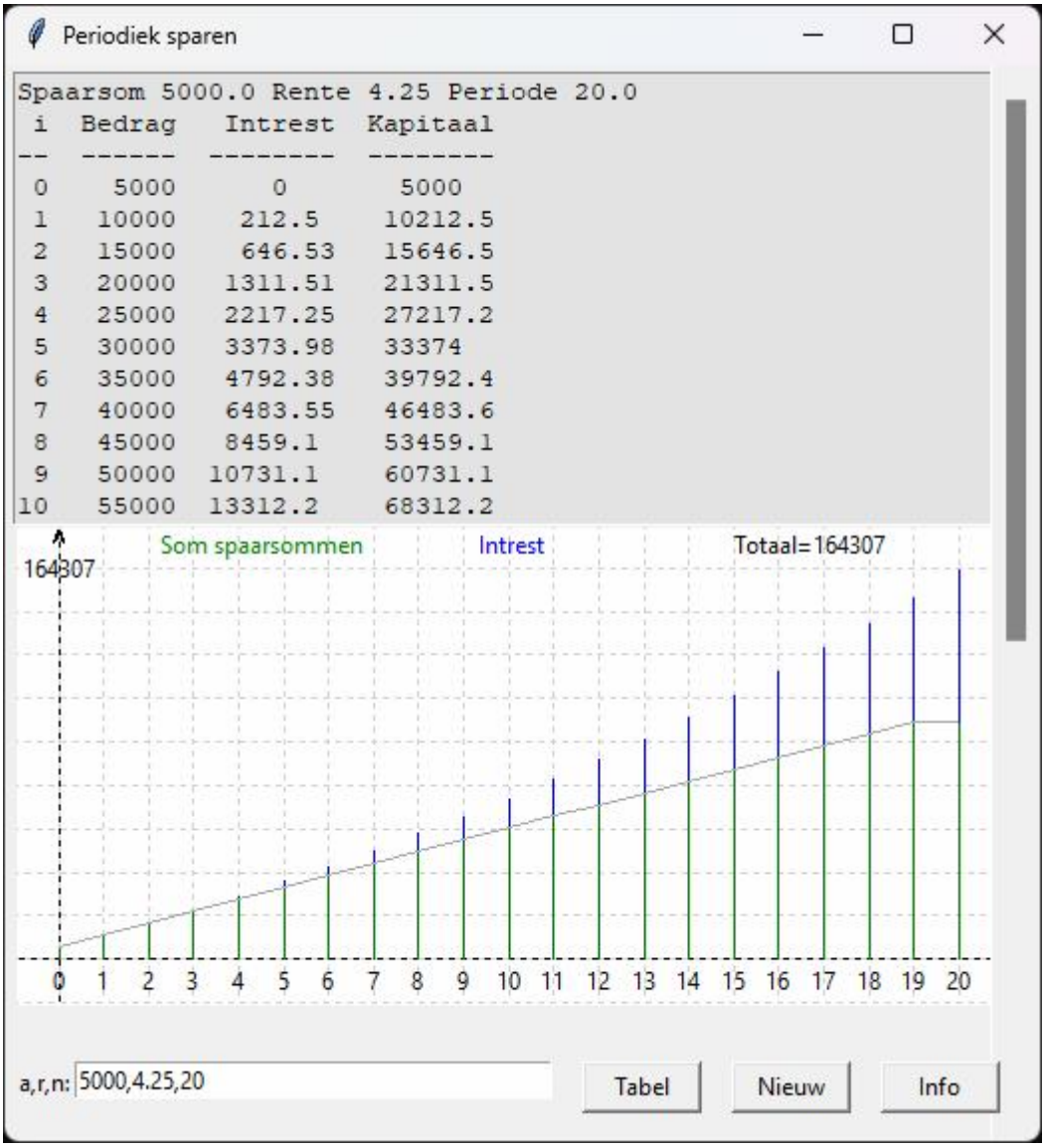
Gedurende een periode n wordt telkens een spaarsom a belegd aan een rentevoet r.
kapN(j)=som gespaarde sommen waarbij j = 0..n-1
kapI(j)= som intresten voor j = 0..n
kapT(j)=som gespaarde sommen en intrest waarbij j = 0..n
Het programma toont een tabel alsook een grafiek van deze 3 sommen

Programma

```
# sparen
from tkinter import *;from tabulate import tabulate;from tkinter import messagebox
def mult(s):
    l=s.split(','); co=[]
    for i in l:co.append(float(i))
    return co
def transx(x):return (x-xmi)*breedteC/(xma-xmi)
def transy(y):return hoogteC-hoogteC*(y-ymi)/(yma-ymi)
```

```
def ad(lin):tex.insert(INSERT,lin+'')
def ap(li,x,y):X=transx(x);Y=transy(y);li.append(X);li.append(Y)
def lijn(j):
        li=[]
        li.append(str(j))
        li.append(format(kapN[j],'.2f'))
        li.append(format(kapI[j],'.2f'))
        li.append(format(kapT[j],'.2f'))
        return li
def bereken():
        global kapN,kapT,kapI,xmi,ymi,xma,yma
        C.delete(ALL);tex.delete('1.0',END)
        a,r,n=mult(E1.get())
        ad('Spaarsom '+str(a)+' Rente '+str(r)+' Periode '+str(n)+'\n')
        ad(' i  Bedrag   Intrest  Kapitaal\n')
        kapN=[a];kapT=[a];kapI=[0]
        i=r/100;u=1+i
        tabel=[];tabel.append(lijn(0));n= int(n)
        for j in range(1,n+1):
                if j<n:kapN.append(kapN[j-1]+a)
                else:kapN.append(kapN[j-1])
                kapI.append(kapT[j-1]*u-kapN[j-1])
                kapT.append(kapT[j-1]*u+a)
                tabel.append(lijn(j))
        ad(tabulate(tabel))
        pa=kapT[n]
        ad('\n'+'Eindkapitaal='+format(pa,'.2f'))
        xmi=-1;xma=n+2;ymi=-1;yma=10
        schaal=(yma-1)/pa
        kl='black'
        # assen
        X=transx(0);C.create_line(X,0,X,hoogteC,fill=kl,arrow='first')
        Y=transy(0);C.create_line(0,Y,breedteC,Y,fill=kl,arrow='last')
        # rooster
        for x in range(int(xmi),int(xma+1)):
             X=transx(x);C.create_line(X,0,X,hoogteC,fill='lightgrey',dash=[1,2])
        for y in range(int(ymi),int(yma+1)):
                Y=transy(y);C.create_line(0,Y,breedteC,Y,fill='lightgrey',dash=[1,2] );lisgr=[]
        for j in range(0,n+1):
                kN=kapN[j]*schaal;kT=kapT[j]*schaal
                lisk=[];lisp=[];lisco=[];ap(lisk,j,0);ap(lisk,j,kN)
                ap(lisp,j,kN);ap(lisp,j,kT);ap(lisco,j,-0.5)
                ap(lisgr,j,kN)
                C.create_line(lisk,fill='green')
                C.create_line(lisp,fill='blue')
                C.create_text(lisco,fill='black',text=str(j))
        lipa=[];ap(lipa,0,yma-1);C.create_text(lipa,text=format(pa,'.0f'))
        C.create_line(lisgr,fill='darkgrey')
        C.create_text(125,10,fill='green',text='Som spaarsommen')
        C.create_text(250,10,fill='blue',text='Intrest')
        C.create_text(400,10,text='Totaal='+format(pa,'.0f'))
```

```
def info():messagebox.showinfo(tit+': info',infstr);return
def nieuw():
        C.delete(ALL);tex.delete('1.0',END)
        E1.delete(0,len(E1.get()))
# hoofdprogramma
breedteC=520;hoogte=540;hoogteC=240;hoInv1=hoogte-40
form=Tk();form.geometry('520x540')
tit='Periodiek sparen';form.title(tit)
C = Canvas(form, bg="white", height=hoogteC, width=breedteC)
C.place(x=4,y=230)
tex=Text(form,bg='grey89',height=14,width=61,wrap=WORD);tex.place(x=4,y=3)
vsb=Scrollbar(form,orient=VERTICAL,command=tex.yview,width=25)
tex['yscrollcommand']=vsb.set;vsb.pack(side=RIGHT,fill='y')
Label(form,text='a,r,n:').place(x=4,y=hoInv1)
E1=Entry(form);E1.place(x=35,y=hoInv1,width=240);E1.insert(0,'5000,4.25,20')
Button(form,text='Tabel',command=bereken).place(x=290,y=hoInv1,width=60)
Button(form,text='Nieuw',command=nieuw).place(x=365,y=hoInv1,width=60)
Button(form,text='Info',command=info).place(x=440,y=hoInv1,width=60)
infstr='a=spaarsom\n'
infstr=infstr+'r=rentevoet\n'
infstr=infstr+'n=periode\n'
form.mainloop()
```



158. Annuïteitstabel + grafiek annuiteitstabelTK

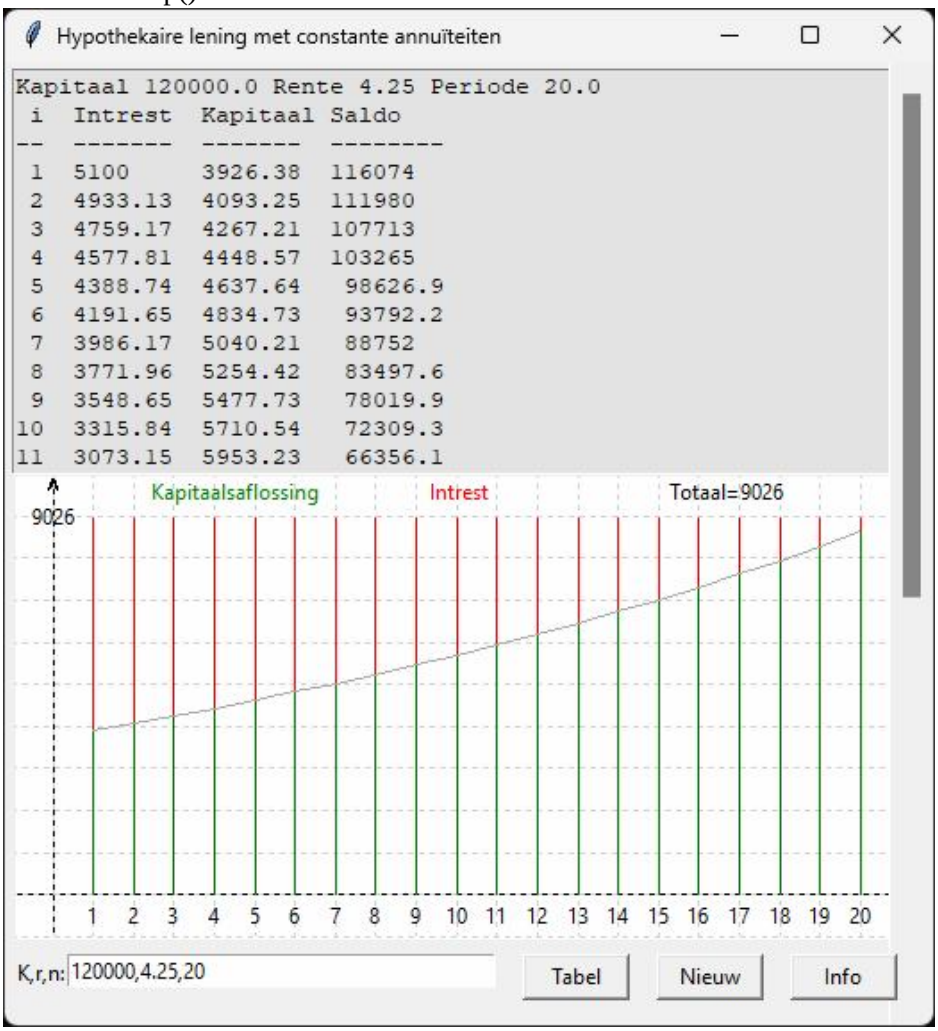
Zie voor de formules naar een eerder programma over annuïteiten. Merk hier vooral bij hogere rentevoeten op dat in het begin de kapitaalsaflossing klein en de intrest groot is.

Programma

```
# annuiteiten
from tkinter import *;from tabulate import tabulate;from tkinter import messagebox
def mult(s):
    l=s.split(','); co=[]
    for i in l:co.append(float(i))
    return co
def transx(x):return (x-xmi)*breedteC/(xma-xmi)
def transy(y):return hoogteC-hoogteC*(y-ymi)/(yma-ymi)
def ad(lin):tex.insert(INSERT,lin+'')
def ap(li,x,y):X=transx(x);Y=transy(y);li.append(X);li.append(Y)
```

```
def lijn(j):
        li=[]
        li.append(str(j+1))
        li.append(format(intrest[j],'.2f'))
        li.append(format(kapafl[j],'.2f'))
        li.append(format(saldo[j],'.2f'))
        return li
def bereken():
        global intrest,kapafl,saldo,xmi,ymi,xma,yma
        C.delete(ALL);tex.delete('1.0',END)
        k,r,n=mult(E1.get())
        ad('Kapitaal '+str(k)+' Rente '+str(r)+' Periode '+str(n)+'\n')
        intrest=[];kapafl=[];saldo=[]
        i=r/100;u=1+i
        kapafl.append(i*k/(u**n-1))
        intrest.append(i*k)
        saldo.append(k-kapafl[0])
        ad(" i  Intrest  Kapitaal Saldo\n")
        tabel=[];tabel.append(lijn(0));n= int(n)
        for j in range(1,n):
                intrest.append(i*saldo[j-1])
                kapafl.append(u*kapafl[j-1])
                saldo.append(saldo[j-1]-kapafl[j])
                tabel.append(lijn(j))
        ad(tabulate(tabel))
        pa=intrest[0]+kapafl[0]
        ad("\nPeriodieke afbetaling="+format(pa,'.2f'))
        xmi=-1;xma=n+2;ymi=-1;yma=10
        schaal=(yma-1)/pa
        kl='black'
        # assen
        X=transx(0);C.create_line(X,0,X,hoogteC,fill=kl,arrow='first')
        Y=transy(0);C.create_line(0,Y,breedteC,Y,fill=kl,arrow='last')
        # rooster
        for x in range(int(xmi),int(xma+1)):
             X=transx(x);C.create_line(X,0,X,hoogteC,fill='lightgrey',dash=[1,2])
        for y in range(int(ymi),int(yma+1)):
                Y=transy(y);C.create_line(0,Y,breedteC,Y,fill='lightgrey',dash=[1,2] );lisgr=[]
        for i in range(0,n):
                k=kapafl[i]*schaal;j=i+1
                lisk=[];lisp=[];lisco=[];ap(lisk,j,0);ap(lisk,j,k)
                ap(lisp,j,k);ap(lisp,j,yma-1);ap(lisco,j,-0.5)
                ap(lisgr,j,k)
                C.create_line(lisk,fill='green')
                C.create_line(lisp,fill='red')
                C.create_text(lisco,fill='black',text=str(j))
        lipa=[];ap(lipa,0,yma-1);C.create_text(lipa,text=format(pa,'.0f'))
        C.create_line(lisgr,fill='darkgrey')
        C.create_text(125,10,fill='green',text='Kapitaalsaflossing')
        C.create_text(250,10,fill='red',text='Intrest')
        C.create_text(400,10,text='Totaal='+format(pa,'.0f'))
```

```
def info():messagebox.showinfo(tit+': info',infstr);return
def nieuw():
        C.delete(ALL);tex.delete('1.0',END)
        E1.delete(0,len(E1.get()))
# hoofdprogramma
breedteC=520;hoogte=540;hoogteC=260;hoInv1=hoogte-40
form=Tk();form.geometry('520x540')
tit='Hypothekaire lening met constante annuïteiten'
form.title(tit)
C = Canvas(form, bg="white", height=hoogteC, width=breedteC)
C.place(x=4,y=230)
tex=Text(form,bg='grey89',height=14,width=61,wrap=WORD);tex.place(x=4,y=3)
vsb=Scrollbar(form,orient=VERTICAL,command=tex.yview,width=25)
tex['yscrollcommand']=vsb.set;vsb.pack(side=RIGHT,fill='y')
Label(form,text='K,r,n:').place(x=4,y=hoInv1)
E1=Entry(form);E1.place(x=35,y=hoInv1,width=240);E1.insert(0,'120000,4.25,20')
Button(form,text='Tabel',command=bereken).place(x=290,y=hoInv1,width=60)
Button(form,text='Nieuw',command=nieuw).place(x=365,y=hoInv1,width=60)
Button(form,text='Info',command=info).place(x=440,y=hoInv1,width=60)
infstr='K=geleend kapitaal\n'
infstr=infstr+'r=rentevoet\n'
infstr=infstr+'n=periode\n'
form.mainloop()
```

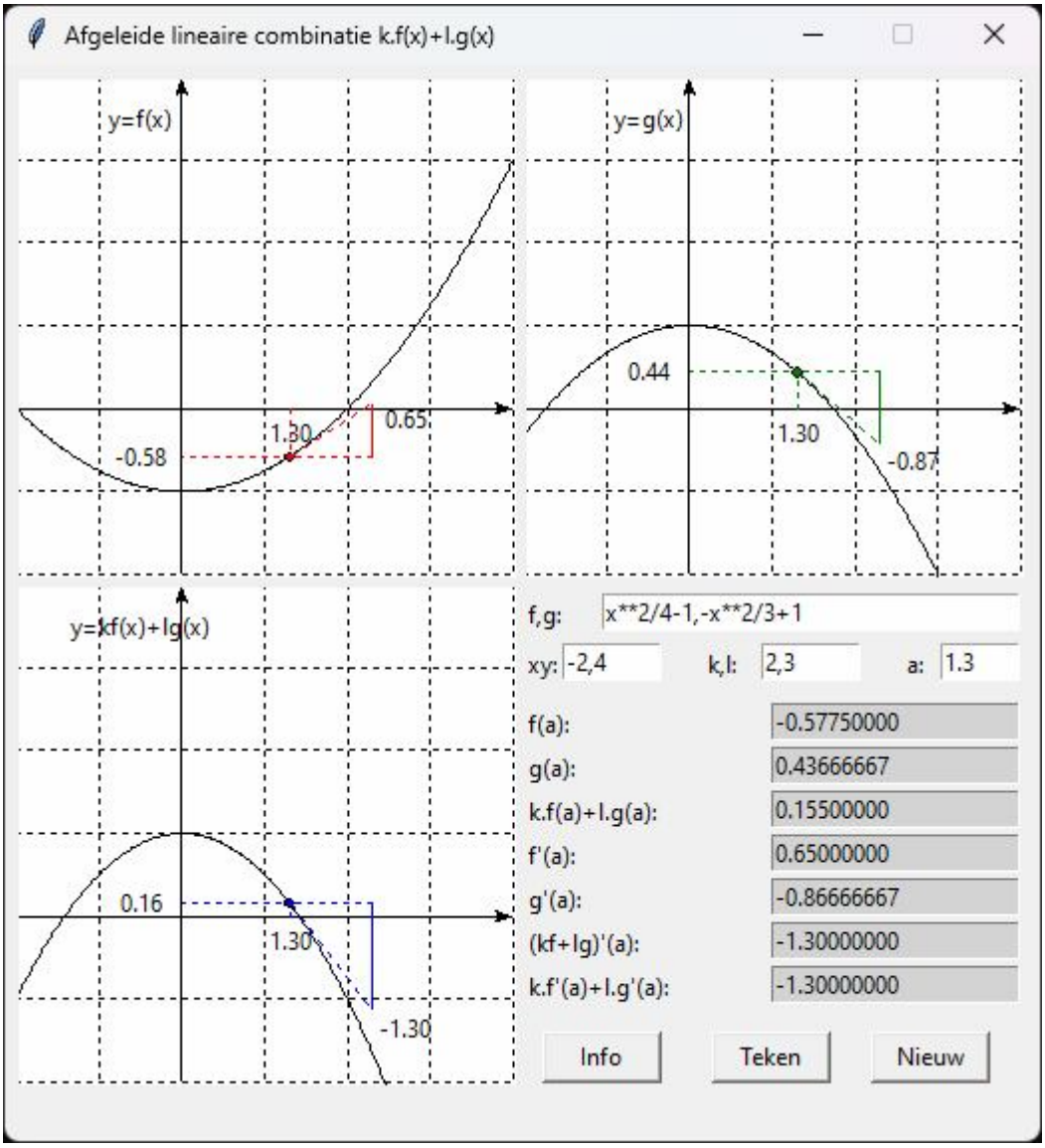


De volgende 4 programma's illustreren numeriek de formules voor de afgeleiden van een lineaire combinatie van 2 functies, het product van 2 functies, het quotiënt van 2 functies en de samengestelde van 2 functies. De listings zijn bijna hetzelfde.

159. Afgeleide van k.f(x)+l.g(x) afgeleide lineair combi

```
# afgeleide k.f(x)+l.g(x) (met tkinter)
from math import *;from tkinter import *;from tkinter import messagebox
# meervoudige invoer
def mult(s):
    l=s.split(','); co=[]
    for i in l:co.append(float(i))
    return co
# omzetten wiskundige coördinaten x,y,z => schermcoördinaten X,Y
def transx(x):return (x-xmi)*breedteC/(xma-xmi)
def transy(y):return hoogteC-hoogteC*(y-ymi)/(yma-ymi)
# berekenen functie en afgeleide functie
def f(x):return eval(fs)
def g(x):return eval(gs)
def kf_lg(x):return eval('k*('+fs+')+l*('+gs+')')
def df(x):return (f(x+dx)-f(x-dx))/dx/2
def dg(x):return (g(x+dx)-g(x-dx))/dx/2
def dkf_lg(x):return(kf_lg(x+dx)-kf_lg(x-dx))/dx/2
def prin(E,x):E.insert(0,format(x,'.8f'))
def init(Cv,te):
    global kl;Cv.delete(ALL);kl='black'
# assen
    X=transx(0);Cv.create_line(X,0,X,hoogteC,fill=kl,arrow='first')
    Y=transy(0);Cv.create_line(0,Y,breedteC,Y,fill=kl,arrow='last')
# rooster
    for x in range(int(xmi),int(xma+1)):
        X=transx(x);Cv.create_line(X,0,X,hoogteC,fill=kl,dash=[1,2])
    for y in range(int(ymi),int(yma+1)):
        Y=transy(y);Cv.create_line(0,Y,breedteC,Y,fill=kl,dash=[1,2] )
# tekst canvas
    lis=[];ap(lis,xmi+1.5,yma-0.5);Cv.create_text(lis,text=te)
# toevoegen punt aan lijst [x1,y1,x2,y2,...] van een kromme (of rechte)
def ap(li,x,y):X=transx(x);Y=transy(y);li.append(X);li.append(Y)
def teken(C,lis,tek,col):C.create_line(lis,fill=col);C.create_text(35,10,text=tek,fill=col)
def bereken():
#krommen
    global xmi,xma,ymi,yma,fs,gs,x,y,k,l
    xmi,xma=mult(E1.get());ymi,yma=xmi,xma
    for E in (E3,E4,E5,E6,E7,E8,E9):wis(E)
    fs,gs=E0.get().split(',');k,l=mult(E1b.get())
    init(C1,'y=f(x)');   init(C2,'y=g(x)');   init(C3,'y=kf(x)+lg(x)');
    lis1,lis2,lis3=[],[],[];stap=0.02;x=xmi;y=ymi
    while x<xma:x=x+stap;ap(lis1,x,f(x));ap(lis2,x,g(x));ap(lis3,x,kf_lg(x))
    C1.create_line(lis1,fill=kl);C2.create_line(lis2,fill=kl);C3.create_line(lis3,fill=kl)
    a=float(E2.get());prin(E3,f(a));prin(E4,g(a));prin(E5,kf_lg(a))
    prin(E6,df(a));prin(E7,dg(a));prin(E8,dkf_lg(a));prin(E9,(k*df(a)+l*dg(a)))
```

```
    rkl(C1,a,f(a),df(a),'red')
    rkl(C2,a,g(a),dg(a),'green')
    rkl(C3,a,kf_lg(a),dkf_lg(a),'blue')
    return
def prinX(C,a):lis=[];ap(lis,a,-0.3);C.create_text(lis,text=format(a,'.2f'))
def prinY(C,a):lis=[];ap(lis,-0.5,a);C.create_text(lis,text=format(a,'.2f'))
def rkl(C,a,b,ri,col):
        #punt
        lis=[];r=0.05;ap(lis,a-r,b-r);ap(lis,a+r,b+r);C.create_oval(lis,fill=col)
        #driehoek
        yf=b+ri
        lis=[];ap(lis,a+1,b);ap(lis,a,b);ap(lis,a+1,yf);C.create_line(lis,fill=col,dash=[1,2])
        lis=[];ap(lis,a+1,b);ap(lis,a+1,yf);C.create_line(lis,fill=col)
        lis=[];ap(lis,a,0);ap(lis,a,b);ap(lis,0,b);C.create_line(lis,fill=col,dash=[1,2])
        #tekst in canvas
        prinX(C,a);prinY(C,b)
        lis=[];ap(lis,a+1.4,b+ri-0.2);C.create_text(lis,text=format(ri,'.2f'))
def wis(E):E.delete(0,len(E.get()))
def nieuw():
    for E in (E0,E1,E2,E3,E4,E5,E6,E7,E8,E9):wis(E)
    for C in (C1,C2,C3):C.delete(ALL)
def info():messagebox.showinfo(tit+': info',inf)
def canv(xp,yp):C=Canvas(form, bg="white", height=hoogteC, width=breedteC);C.place(x=xp,y=yp);return C
def ent(te,i):
        Label(form,text=te+':').place(x=260,y=320+i*22)
        E=Entry(form,bg=lg);E.place(x=385,y=320+i*22,width=125)
        return E
# hoofdprogramma
tit="Afgeleide lineaire combinatie k.f(x)+l.g(x)"
dx,dy=1E-6,1E-6;breedte,hoogte=520,540;breedteC,hoogteC=250,250;lg='lightgrey';nwk='.4f'
form=Tk();form.geometry(str(breedte)+'x'+str(hoogte));form.title(tit);form.resizable(False,False)
C1=canv(5,5);C2=canv(260,5);C3=canv(5,260)
Label(form,text='f,g:').place(x=260,y=265)
E0=Entry(form);E0.place(x=300,y=265,width=210);E0.insert(0,'x**2/4-1,-x**2/3+1')
Label(form,text='xy:').place(x=260,y=290)
E1=Entry(form);E1.place(x=280,y=290,width=50);E1.insert(0,'-2,4')
Label(form,text='k,l:').place(x=350,y=290)
E1b=Entry(form);E1b.place(x=380,y=290,width=50);E1b.insert(0,'2,3')
Label(form,text='a:').place(x=450,y=290)
E2=Entry(form);E2.place(x=470,y=290,width=40);E2.insert(0,'1.3')
E3=ent('f(a)',0);E4=ent('g(a)',1)
E5=ent("k.f(a)+l.g(a)",2);E6=ent("f'(a)",3)
E7=ent("g'(a)",4);E8=ent("(kf+lg)'(a)",5)
E9=ent("k.f'(a)+l.g'(a)",6)
inf="Invullen   f(x),g(x)\nxy min,max      a    k,l\n\n"
inf=inf+"Steeds blijkt:\n(kf+lg)'(a)= k.f'(a)+l.g'(a)"
Button(form,text="Info",command=info).place(x=270,y=485,width=60)
Button(form,text="Teken",command=bereken).place(x=355,y=485,width=60)
Button(form,text="Nieuw",command=nieuw).place(x=435,y=485,width=60)
form.mainloop()
```



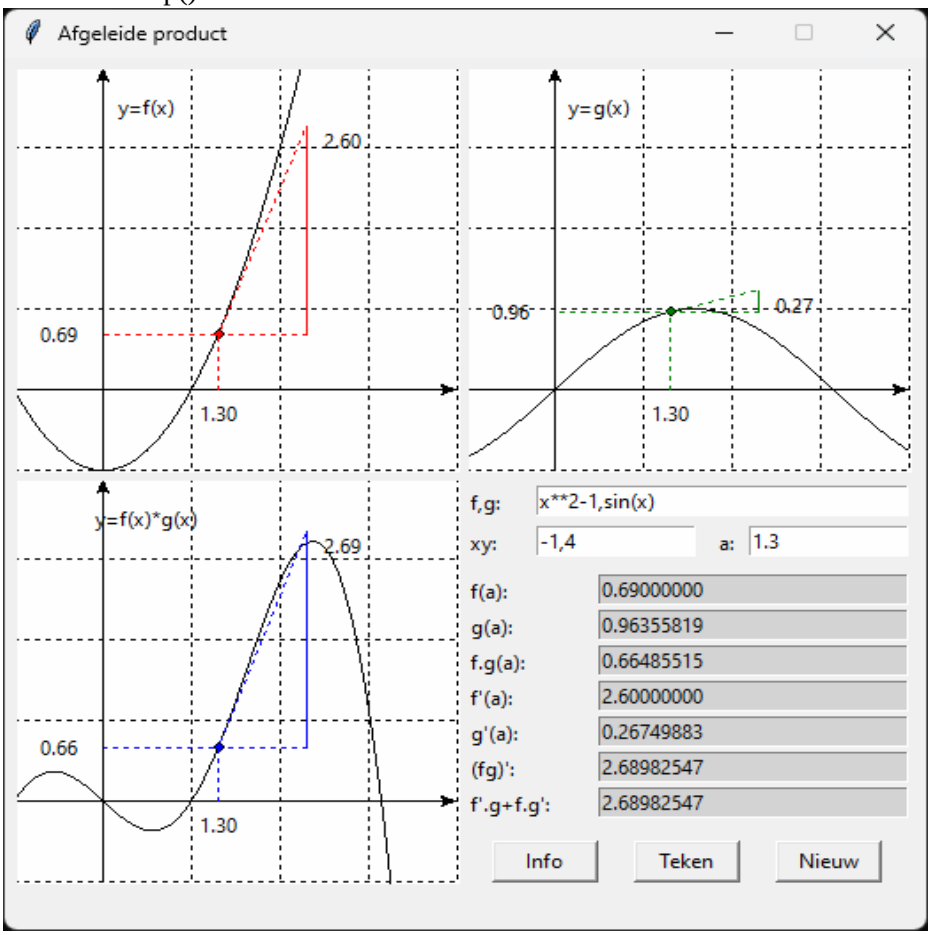
160. Afgeleide van een product afgeleide product

Programma

```
# afgeleide product (met tkinter)
from math import *;from tkinter import *;from tkinter import messagebox
# meervoudige invoer
def mult(s):
    l=s.split(','); co=[]
    for i in l:co.append(float(i))
    return co
# omzetten wiskundige coördinaten x,y,z => schermcoördinaten X,Y
def transx(x):return (x-xmi)*breedteC/(xma-xmi)
def transy(y):return hoogteC-hoogteC*(y-ymi)/(yma-ymi)
# berekenen functie en afgeleide functie
def f(x):return eval(fs)
def g(x):return eval(gs)
def fg(x):return eval('('+fs+')*('+gs+')')
def df(x):return (f(x+dx)-f(x-dx))/dx/2
```

```
def dg(x):return (g(x+dx)-g(x-dx))/dx/2
def dfg(x):return(fg(x+dx)-fg(x-dx))/dx/2
def prin(E,x):E.insert(0,format(x,'.8f'))
def init(Cv,te):
   global kl;Cv.delete(ALL);kl='black'
# assen
   X=transx(0);Cv.create_line(X,0,X,hoogteC,fill=kl,arrow='first')
   Y=transy(0);Cv.create_line(0,Y,breedteC,Y,fill=kl,arrow='last')
# rooster
   for x in range(int(xmi),int(xma+1)):
      X=transx(x);Cv.create_line(X,0,X,hoogteC,fill=kl,dash=[1,2])
   for y in range(int(ymi),int(yma+1)):
      Y=transy(y);Cv.create_line(0,Y,breedteC,Y,fill=kl,dash=[1,2] )
# tekst canvas
   lis=[];ap(lis,xmi+1.5,yma-0.5);Cv.create_text(lis,text=te)
# toevoegen punt aan lijst [x1,y1,x2,y2,...] van een kromme (of rechte)
def ap(li,x,y):X=transx(x);Y=transy(y);li.append(X);li.append(Y)
def teken(C,lis,tek,col):C.create_line(lis,fill=col);C.create_text(35,10,text=tek,fill=col)
def bereken():
#krommen
   global xmi,xma,ymi,yma,fs,gs,x,y,z;xmi,xma=mult(E1.get());ymi,yma=xmi,xma
   for E in (E3,E4,E5,E6,E7,E8,E9):wis(E)
   fs,gs=E0.get().split(',')
   init(C1,'y=f(x)');   init(C2,'y=g(x)');   init(C3,'y=f(x)*g(x)');
   lis1,lis2,lis3=[],[],[];stap=0.02;x=xmi;y=ymi
   while x<xma:x=x+stap;ap(lis1,x,f(x));ap(lis2,x,g(x));ap(lis3,x,fg(x))
   C1.create_line(lis1,fill=kl);C2.create_line(lis2,fill=kl);C3.create_line(lis3,fill=kl)
   a=float(E2.get());prin(E3,f(a));prin(E4,g(a));prin(E5,fg(a))
   prin(E6,df(a));prin(E7,dg(a));prin(E8,dfg(a));prin(E9,(df(a)*g(a)+f(a)*dg(a)))
   rkl(C1,a,f(a),df(a),'red')
   rkl(C2,a,g(a),dg(a),'green')
   rkl(C3,a,fg(a),dfg(a),'blue')
   return
def prinX(C,a):lis=[];ap(lis,a,-0.3);C.create_text(lis,text=format(a,'.2f'))
def prinY(C,a):lis=[];ap(lis,-0.5,a);C.create_text(lis,text=format(a,'.2f'))
def rkl(C,a,b,ri,col):
        #punt
        lis=[];r=0.05;ap(lis,a-r,b-r);ap(lis,a+r,b+r);C.create_oval(lis,fill=col)
        #driehoek
        yf=b+ri
        lis=[];ap(lis,a+1,b);ap(lis,a,b);ap(lis,a+1,yf);C.create_line(lis,fill=col,dash=[1,2])
        lis=[];ap(lis,a+1,b);ap(lis,a+1,yf);C.create_line(lis,fill=col)
        lis=[];ap(lis,a,0);ap(lis,a,b);ap(lis,0,b);C.create_line(lis,fill=col,dash=[1,2])
        #tekst in canvas
        prinX(C,a);prinY(C,b)
        lis=[];ap(lis,a+1.4,b+ri-0.2);C.create_text(lis,text=format(ri,'.2f'))
def wis(E):E.delete(0,len(E.get()))
def nieuw():
   for E in (E0,E1,E2,E3,E4,E5,E6,E7,E8,E9):wis(E)
   for C in (C1,C2,C3):C.delete(ALL)
def info():messagebox.showinfo(tit+': info',inf)
```

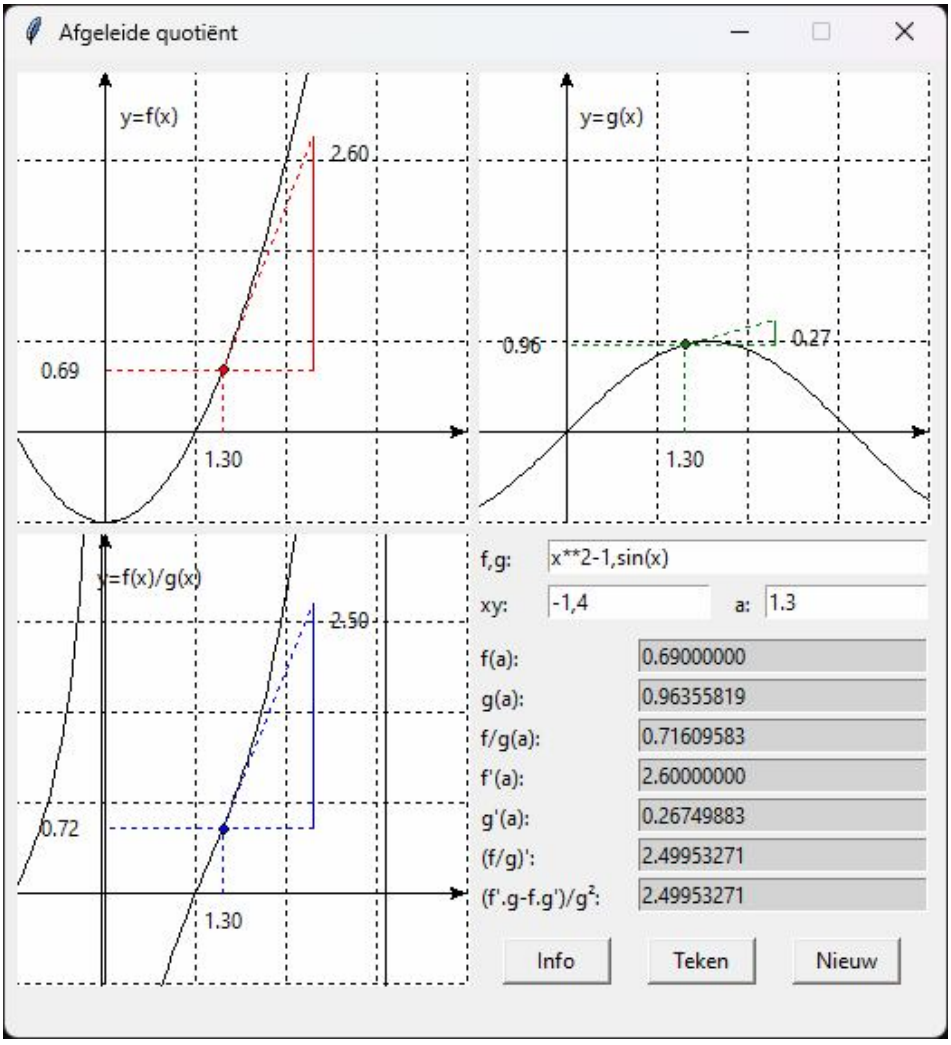
```
def canv(xp,yp):C=Canvas(form, bg="white", height=hoogteC, width=breedteC);C.place(x=xp,y=yp);return C
def ent(te,i):
        Label(form,text=te+':').place(x=260,y=320+i*22)
        E=Entry(form,bg=lg);E.place(x=335,y=320+i*22,width=175)
        return E
# hoofdprogramma
tit="Afgeleide product"
dx,dy=1E-6,1E-6;breedte,hoogte=520,540;breedteC,hoogteC=250,250;lg='lightgrey';nwk='.4f'
form=Tk();form.geometry(str(breedte)+'x'+str(hoogte));form.title(tit);form.resizable(False,False)
C1=canv(5,5);C2=canv(260,5);C3=canv(5,260)
Label(form,text='f,g:').place(x=260,y=265)
E0=Entry(form);E0.place(x=300,y=265,width=210);E0.insert(0,'x**2-1,sin(x)')
Label(form,text='xy:').place(x=260,y=290)
E1=Entry(form);E1.place(x=300,y=290,width=90);E1.insert(0,'-1,4')
Label(form,text='a:').place(x=400,y=290)
E2=Entry(form);E2.place(x=420,y=290,width=90);E2.insert(0,'1.3')
E3=ent('f(a)',0);E4=ent('g(a)',1)
E5=ent("f.g(a)",2);E6=ent("f'(a)",3)
E7=ent("g'(a)",4);E8=ent("(fg)'",5)
E9=ent("f'.g+f.g'",6)
inf="Invullen   f(x),g(x)\nxy min,max          a\n\n"
inf=inf+"Steeds blijkt:\n(f*g)'(a)= f'(a)*g(a)+f(a)*g'(a)"
Button(form,text="Info",command=info).place(x=275,y=485,width=60)
Button(form,text="Teken",command=bereken).place(x=355,y=485,width=60)
Button(form,text="Nieuw",command=nieuw).place(x=435,y=485,width=60)
form.mainloop()
```



161. Afgeleide van een quotiënt afgeleide quotient

```
# afgeleide quotiënt (met tkinter)
from math import *;from tkinter import *;from tkinter import messagebox
# meervoudige invoer
def mult(s):
    l=s.split(','); co=[]
    for i in l:co.append(float(i))
    return co
# omzetten wiskundige coördinaten x,y,z => schermcoördinaten X,Y
def transx(x):return (x-xmi)*breedteC/(xma-xmi)
def transy(y):return hoogteC-hoogteC*(y-ymi)/(yma-ymi)
# berekenen functie en afgeleide functie
def f(x):return eval(fs)
def g(x):return eval(gs)
def f_g(x):return eval('('+fs+')/('+gs+')')
def df(x):return (f(x+dx)-f(x-dx))/dx/2
def dg(x):return (g(x+dx)-g(x-dx))/dx/2
def df_g(x):return(f_g(x+dx)-f_g(x-dx))/dx/2
def prin(E,x):E.insert(0,format(x,'.8f'))
def init(Cv,te):
    global kl;Cv.delete(ALL);kl='black'
# assen
    X=transx(0);Cv.create_line(X,0,X,hoogteC,fill=kl,arrow='first')
    Y=transy(0);Cv.create_line(0,Y,breedteC,Y,fill=kl,arrow='last')
# rooster
    for x in range(int(xmi),int(xma+1)):
        X=transx(x);Cv.create_line(X,0,X,hoogteC,fill=kl,dash=[1,2])
    for y in range(int(ymi),int(yma+1)):
        Y=transy(y);Cv.create_line(0,Y,breedteC,Y,fill=kl,dash=[1,2] )
# tekst canvas
    lis=[];ap(lis,xmi+1.5,yma-0.5);Cv.create_text(lis,text=te)
# toevoegen punt aan lijst [x1,y1,x2,y2,...] van een kromme (of rechte)
def ap(li,x,y):X=transx(x);Y=transy(y);li.append(X);li.append(Y)
def teken(C,lis,tek,col):C.create_line(lis,fill=col);C.create_text(35,10,text=tek,fill=col)
def bereken():
#krommen
    global xmi,xma,ymi,yma,fs,gs,x,y,z;xmi,xma=mult(E1.get());ymi,yma=xmi,xma
    for E in (E3,E4,E5,E6,E7,E8,E9):wis(E)
    fs,gs=E0.get().split(',')
    init(C1,'y=f(x)');   init(C2,'y=g(x)');   init(C3,'y=f(x)/g(x)');
    lis1,lis2,lis3=[],[],[];stap=0.049;x=xmi-0.07;y=ymi
    while x<xma:x=x+stap;ap(lis1,x,f(x));ap(lis2,x,g(x));ap(lis3,x,f_g(x))
    C1.create_line(lis1,fill=kl);C2.create_line(lis2,fill=kl);C3.create_line(lis3,fill=kl)
    a=float(E2.get());prin(E3,f(a));prin(E4,g(a));prin(E5,f_g(a))
    prin(E6,df(a));prin(E7,dg(a));prin(E8,df_g(a));prin(E9,(df(a)*g(a)-f(a)*dg(a))/g(a)**2)
    rkl(C1,a,f(a),df(a),'red')
    rkl(C2,a,g(a),dg(a),'green')
    rkl(C3,a,f_g(a),df_g(a),'blue')
    return
def prinX(C,a):lis=[];ap(lis,a,-0.3);C.create_text(lis,text=format(a,'.2f'))
```

```
def prinY(C,a):lis=[];ap(lis,-0.5,a);C.create_text(lis,text=format(a,'.2f'))
def rkl(C,a,b,ri,col):
        #punt
        lis=[];r=0.05;ap(lis,a-r,b-r);ap(lis,a+r,b+r);C.create_oval(lis,fill=col)
        #driehoek
        yf=b+ri
        lis=[];ap(lis,a+1,b);ap(lis,a,b);ap(lis,a+1,yf);C.create_line(lis,fill=col,dash=[1,2])
        lis=[];ap(lis,a+1,b);ap(lis,a+1,yf);C.create_line(lis,fill=col)
        lis=[];ap(lis,a,0);ap(lis,a,b);ap(lis,0,b);C.create_line(lis,fill=col,dash=[1,2])
        #tekst in canvas
        prinX(C,a);prinY(C,b)
        lis=[];ap(lis,a+1.4,b+ri-0.2);C.create_text(lis,text=format(ri,'.2f'))
def wis(E):E.delete(0,len(E.get()))
def nieuw():
   for E in (E0,E1,E2,E3,E4,E5,E6,E7,E8,E9):wis(E)
   for C in (C1,C2,C3):C.delete(ALL)
def info():messagebox.showinfo(tit+': info',inf)
def canv(xp,yp):C=Canvas(form, bg="white", height=hoogteC, width=breedteC);C.place(x=xp,y=yp);return C
def ent(te,i):
        Label(form,text=te+':').place(x=260,y=320+i*22)
        E=Entry(form,bg=lg);E.place(x=350,y=320+i*22,width=160)
        return E
# hoofdprogramma
tit="Afgeleide quotiënt"
dx,dy=1E-6,1E-6;breedte,hoogte=520,540;breedteC,hoogteC=250,250;lg='lightgrey';nwk='.4f'
form=Tk();form.geometry(str(breedte)+'x'+str(hoogte));form.title(tit);form.resizable(False,False)
C1=canv(5,5);C2=canv(260,5);C3=canv(5,260)
Label(form,text='f,g:').place(x=260,y=265)
E0=Entry(form);E0.place(x=300,y=265,width=210);E0.insert(0,'x**2-1,sin(x)')
Label(form,text='xy:').place(x=260,y=290)
E1=Entry(form);E1.place(x=300,y=290,width=90);E1.insert(0,'-1,4')
Label(form,text='a:').place(x=400,y=290)
E2=Entry(form);E2.place(x=420,y=290,width=90);E2.insert(0,'1.3')
E3=ent('f(a)',0);E4=ent('g(a)',1)
E5=ent("f/g(a)",2);E6=ent("f'(a)",3)
E7=ent("g'(a)",4);E8=ent("(f/g)'",5)
E9=ent("(f'.g-f.g')/g²",6)
inf="Invullen   f(x),g(x)\nxy min,max          a\n\n"
inf=inf+"Steeds blijkt:\n(f/g)'(a)= (f'(a)*g(a)-f(a)*g'(a))/g(a)²"
Button(form,text="Info",command=info).place(x=275,y=485,width=60)
Button(form,text="Teken",command=bereken).place(x=355,y=485,width=60)
Button(form,text="Nieuw",command=nieuw).place(x=435,y=485,width=60)
form.mainloop()
```



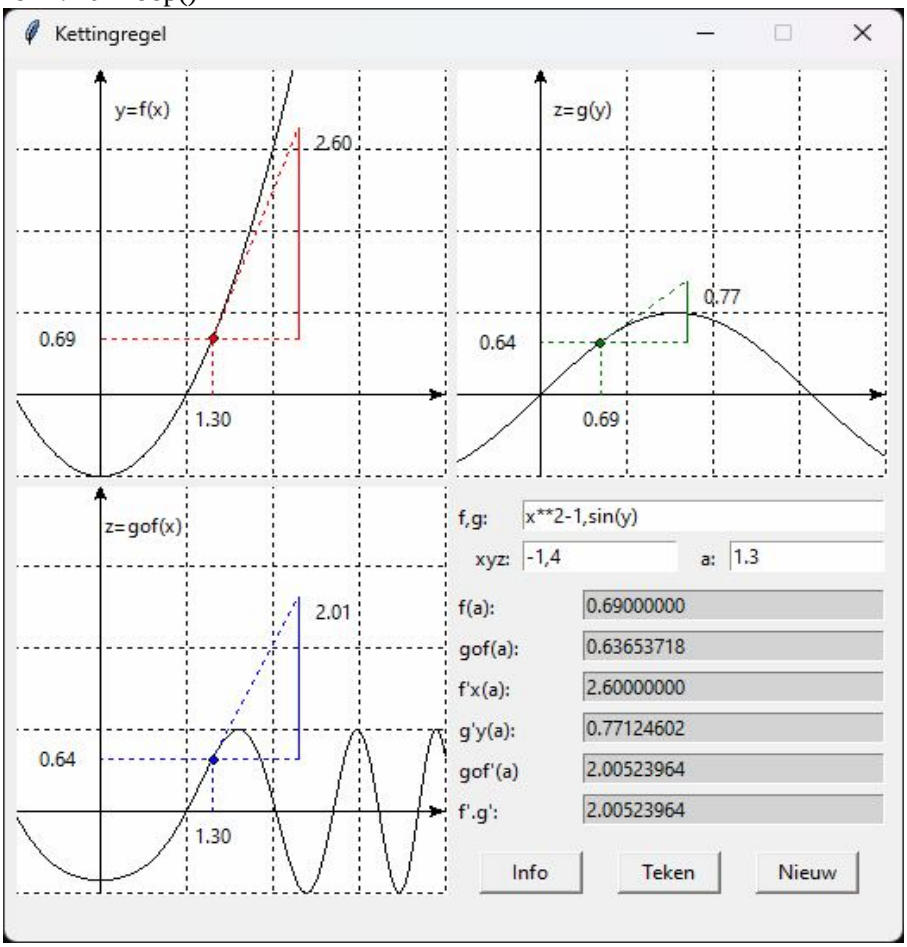
162. Kettingregel kettingregel_afgeleiden

Programma

```
# kettingregel (met tkinter)
from math import *;from tkinter import *;from tkinter import messagebox
# meervoudige invoer
def mult(s):
    l=s.split(','); co=[]
    for i in l:co.append(float(i))
    return co
# omzetten wiskundige coördinaten x,y,z => schermcoördinaten X,Y
def transx(x):return (x-xmi)*breedteC/(xma-xmi)
def transy(y):return hoogteC-hoogteC*(y-ymi)/(yma-ymi)
# berekenen functie en afgeleide functie
def f(x):return eval(fs)
def g(y):return eval(gs)
def df(x):return (f(x+dx)-f(x-dx))/dx/2
def dg(y):return (g(y+dy)-g(y-dy))/dy/2
def dgof(x):return(g(f(x+dx))-g(f(x-dx)))/dx/2
def prin(E,x):E.insert(0,format(x,'.8f'))
def init(Cv,te):
```

```
    global kl;Cv.delete(ALL);kl='black'
# assen
    X=transx(0);Cv.create_line(X,0,X,hoogteC,fill=kl,arrow='first')
    Y=transy(0);Cv.create_line(0,Y,breedteC,Y,fill=kl,arrow='last')
# rooster
    for x in range(int(xmi),int(xma+1)):
        X=transx(x);Cv.create_line(X,0,X,hoogteC,fill=kl,dash=[1,2])
    for y in range(int(ymi),int(yma+1)):
        Y=transy(y);Cv.create_line(0,Y,breedteC,Y,fill=kl,dash=[1,2] )
# tekst canvas
    lis=[];ap(lis,xmi+1.5,yma-0.5);Cv.create_text(lis,text=te)
# toevoegen punt aan lijst [x1,y1,x2,y2,...] van een kromme (of rechte)
def ap(li,x,y):X=transx(x);Y=transy(y);li.append(X);li.append(Y)
def teken(C,lis,tek,col):C.create_line(lis,fill=col);C.create_text(35,10,text=tek,fill=col)
def bereken():
    #krommen
    global xmi,xma,ymi,yma,fs,gs,x,y,z;xmi,xma=mult(E1.get());ymi,yma=xmi,xma
    for E in (E3,E3b,E4,E5,E6,E7):wis(E)
    fs,gs=E0.get().split(',')
    init(C1,'y=f(x)');   init(C2,'z=g(y)');   init(C3,'z=gof(x)');
    lis1,lis2,lis3=[],[],[];stap=0.02;x=xmi;y=ymi
    while x<xma:x=x+stap;ap(lis1,x,f(x));ap(lis3,x,g(f(x)))
    while y<yma:y=y+stap;ap(lis2,y,g(y))
    C1.create_line(lis1,fill=kl);C2.create_line(lis2,fill=kl);C3.create_line(lis3,fill=kl)
    a=float(E2.get());prin(E3,f(a));prin(E3b,g(f(a)));prin(E4,df(a))
    prin(E5,dg(f(a)));prin(E6,dgof(a));prin(E7,dg(f(a))*df(a))
    rkl(C1,a,f(a),df(a),'red')
    rkl(C2,f(a),g(f(a)),dg(f(a)),'green')
    rkl(C3,a,g(f(a)),dgof(a),'blue')
    return
def prinX(C,a):lis=[];ap(lis,a,-0.3);C.create_text(lis,text=format(a,'.2f'))
def prinY(C,a):lis=[];ap(lis,-0.5,a);C.create_text(lis,text=format(a,'.2f'))
def rkl(C,a,b,ri,col):
        lis=[];r=0.05;ap(lis,a-r,b-r);ap(lis,a+r,b+r);C.create_oval(lis,fill=col)
        yf=b+ri
        lis=[];ap(lis,a+1,b);ap(lis,a,b);ap(lis,a+1,yf);C.create_line(lis,fill=col,dash=[1,2])
        lis=[];ap(lis,a+1,b);ap(lis,a+1,yf);C.create_line(lis,fill=col)
        lis=[];ap(lis,a,0);ap(lis,a,b);ap(lis,0,b);C.create_line(lis,fill=col,dash=[1,2])
        prinX(C,a);prinY(C,b)
        lis=[];ap(lis,a+1.4,b+ri-0.2);C.create_text(lis,text=format(ri,'.2f'))
def wis(E):E.delete(0,len(E.get()))
def nieuw():
    for E in (E0,E1,E2,E3,E3b,E4,E5,E6,E7):wis(E)
    for C in (C1,C2,C3):C.delete(ALL)
def info():messagebox.showinfo(tit+': info',inf)
# hoofdprogramma
tit="Kettingregel"
dx,dy=1E-6,1E-6;breedte,hoogte=520,540;breedteC,hoogteC=250,250;lg='lightgrey';nwk='.4f'
form=Tk();form.geometry(str(breedte)+'x'+str(hoogte));form.title(tit);form.resizable(False,False)
C1=Canvas(form, bg="white", height=hoogteC, width=breedteC);C1.place(x=5,y=5)
C2=Canvas(form, bg="white", height=hoogteC, width=breedteC);C2.place(x=260,y=5)
```

```
C3=Canvas(form, bg="white", height=hoogteC, width=breedteC);C3.place(x=5,y=260)
Label(form,text='f,g:').place(x=260,y=270)
E0=Entry(form);E0.place(x=300,y=270,width=210);E0.insert(0,'x**2-1,sin(y)')
Label(form,text='xyz:').place(x=270,y=295)
E1=Entry(form);E1.place(x=300,y=295,width=90);E1.insert(0,'-1,4')
Label(form,text='a:').place(x=400,y=295)
E2=Entry(form);E2.place(x=420,y=295,width=90);E2.insert(0,'1.3')
Label(form,text='f(a):').place(x=260,y=325)
E3=Entry(form,bg=lg);E3.place(x=335,y=325,width=175)
Label(form,text='gof(a):').place(x=260,y=350)
E3b=Entry(form,bg=lg);E3b.place(x=335,y=350,width=175)
Label(form,text="f'x(a):").place(x=260,y=375)
E4=Entry(form,bg=lg);E4.place(x=335,y=375,width=175)
Label(form,text="g'y(a):").place(x=260,y=400)
E5=Entry(form,bg=lg,);E5.place(x=335,y=400,width=175)
Label(form,text="gof'(a)").place(x=260,y=425)
E6=Entry(form,bg=lg);E6.place(x=335,y=425,width=175)
Label(form,text="f'.g':").place(x=260,y=450)
E7=Entry(form,bg=lg);E7.place(x=335,y=450,width=175)
inf="f: f(x) , g: f(y)   \nxyz: min,max xyz\nf'x:naar x    g'y:naar y\n"
inf=inf+"gof: g[f(x)] gof':naar x\nf'.g': f' naar x, g' naar y"
Button(form,text="Info",command=info).place(x=275,y=485,width=60)
Button(form,text="Teken",command=bereken).place(x=355,y=485,width=60)
Button(form,text="Nieuw",command=nieuw).place(x=435,y=485,width=60)
form.mainloop()
```



163. Middelwaardestelling van Cauchy cauchy

Hebben we 2 functies f en g die continu zijn op een interval [a,b] en is $\Delta f = f(b)-f(a)$ en $\Delta g = g(b)-g(a)$, dan

$$\exists c \in]a,b[: \frac{f(b)-f(a)}{g(b)-g(a)} = \frac{f'(c)}{g'(c)}$$

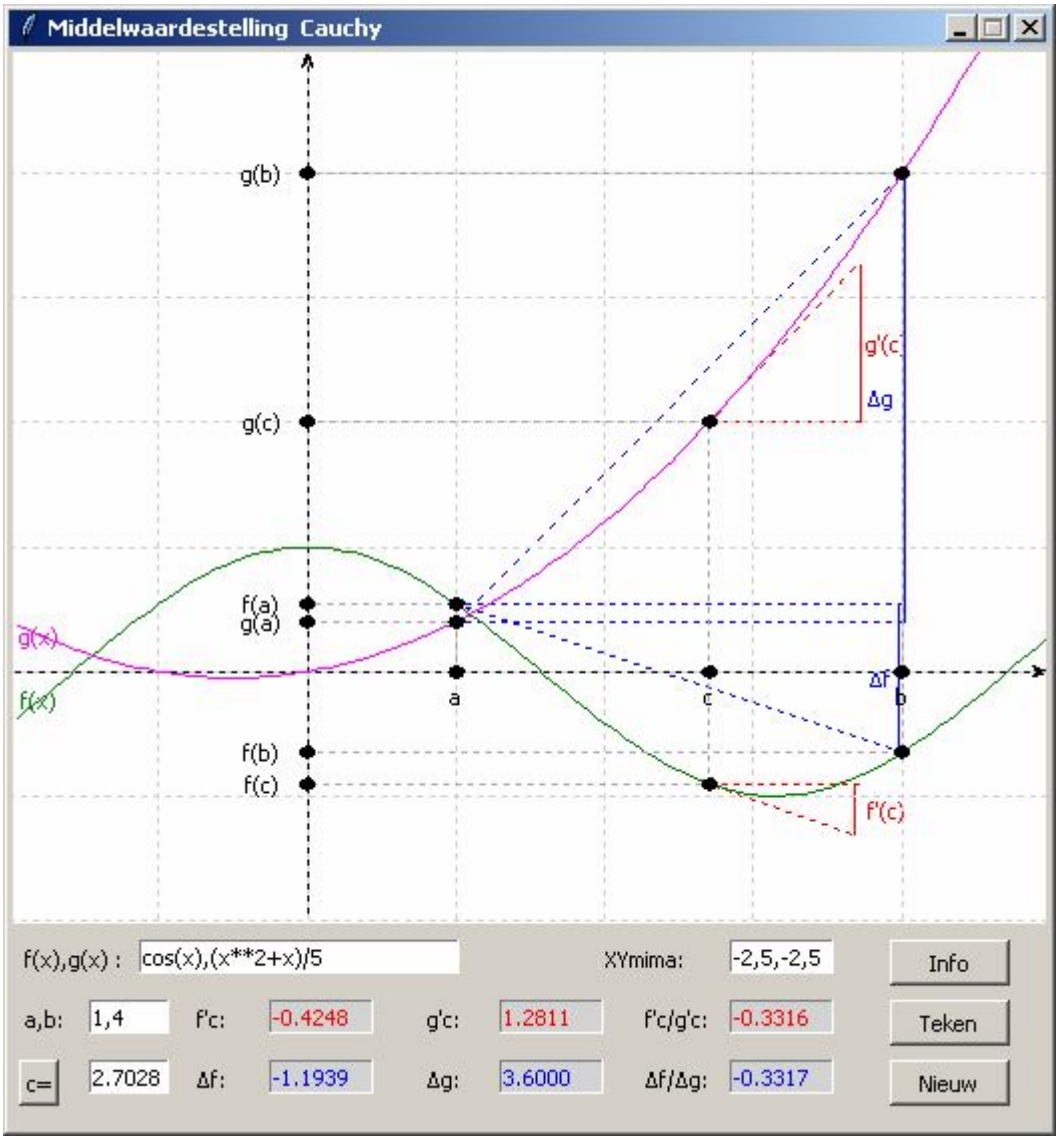
In het volgende programma kan je een c in]a,b[zoeken, die aan deze voorwaarde voldoet. Zoals in de andere programma's over middelwaardestellingen is ook hier een knop voorzien c= die de exacte waarde van c berekent. Deze stelling kan bvb. gebruikt worden bij het bewijs van de regel van de l'Hopital.

Programma:

```
# middelwaardestelling Cauchy
from math import *;from tkinter import *;from tkinter import messagebox
def mult(s):
    l=s.split(',');co=[]
    for i in l:co.append(float(i))
    return co
def transx(x):return (x-xmi)*breedte/(xma-xmi)
def transy(y):return hoogteC-hoogteC*(y-ymi)/(yma-ymi)
def f(x):return eval(fs)
def g(x):return eval(gs)
def df(x):return (f(x+dx)-f(x-dx))/dx/2
def d2f(x):return(f(x+dx)+f(x-dx)-2*f(x))/dx**2
def dg(x):return (g(x+dx)-g(x-dx))/dx/2
def d2g(x):return(g(x+dx)+g(x-dx)-2*g(x))/dx**2
def bereken():
    global fs,gs,xmi,xma,ymi,yma,a,b,c;nwk='.4f'
    fs,gs=E1.get().split(',')
    xmi,xma,ymi,yma=mult(E2.get())
    a,b=mult(E3.get());c=float(E4.get())
    if a>=b:C.delete(ALL);messagebox.showinfo('Fout','Herstel: a<b');return
    C.delete(ALL);raakl();koord();kromme()
def raakl():
    ricoRf=df(c);wis(E5);E5.insert(0,format(ricoRf,'.4f'))
    ricoRg=dg(c);wis(E6);E6.insert(0,format(ricoRg,'.4f'))
    if ricoRg==0:C.delete(ALL);messagebox.showinfo('Fout',"g'(c)=0");return
    wis(E7);E7.insert(0,format(ricoRf/ricoRg,'.4f'))
    yf=f(c)+ricoRf;yg=g(c)+ricoRg
    lis=[];ap(lis,c+1,f(c));ap(lis,c,f(c));ap(lis,c+1,yf)
    C.create_line(lis,fill='red',dash=[1,2])
    lis=[];ap(lis,c+0.98,f(c));ap(lis,c+0.98,yf);C.create_line(lis,fill='red')
    lis=[];ap(lis,c,0);ap(lis,c,f(c));C.create_line(lis,dash=[1,2])
    lis=[];ap(lis,c+1.2,f(c)+ricoRf/2);C.create_text(lis,fill='red',text="f'(c)")
    lis=[];ap(lis,c+1,g(c));ap(lis,c,g(c));ap(lis,c+1,yg)
    C.create_line(lis,fill='red',dash=[1,2])
    lis=[];ap(lis,c+1.02,g(c));ap(lis,c+1.02,yg);C.create_line(lis,fill='red')
    lis=[];ap(lis,c,0);ap(lis,c,g(c));C.create_line(lis,dash=[1,2])
    lis=[];ap(lis,c+1.2,g(c)+ricoRg/2);C.create_text(lis,fill='red',text="g'(c)")
def koord():
```

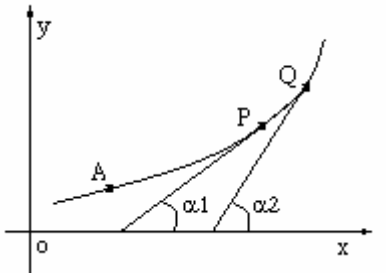
```
    delKf=f(b)-f(a);wis(E8);E8.insert(0,format(delKf,'.4f'))
    delKg=g(b)-g(a);wis(E9);E9.insert(0,format(delKg,'.4f'))
    if delKg==0:C.delete(ALL);messagebox.showinfo('Fout',"Δg=0");return
    wis(E10);E10.insert(0,format(delKf/delKg,'.4f'))
    lis=[];ap(lis,b,f(a));ap(lis,a,f(a));ap(lis,b,f(b));C.create_line(lis,fill='blue',dash=[1,2])
    lis=[];ap(lis,b-0.02,f(a));ap(lis,b-0.02,f(b));C.create_line(lis,fill='blue')
    lis=[];ap(lis,b-0.15,(f(a)+f(b))/2);C.create_text(lis,fill='blue',text="Δf")
    lis=[];ap(lis,b,g(a));ap(lis,a,g(a));ap(lis,b,g(b));C.create_line(lis,fill='blue',dash=[1,2])
    lis=[];ap(lis,b+0.02,g(a));ap(lis,b+0.02,g(b));C.create_line(lis,fill='blue')
    lis=[];ap(lis,b-0.15,(g(a)+g(b))/2);C.create_text(lis,fill='blue',text="Δg")
def newton():
    mi=1000;delx=(b-a)/100;x=a;delquot=(f(b)-f(a))/(g(b)-g(a))
    while x<b:
        x=x+delx;dif=abs(df(x)-dg(x)*delquot)
        if dif<mi:mi=dif;xmi=x
    x=xmi;xo=x+1
    while abs(x-xo)>eps:xo=x;x=x-(df(x)-dg(x)*delquot)/(d2f(x)-d2g(x)*delquot)
    wis(E4);E4.insert(0,format(x,'.4f'))
def kromme():
#assen
    X=transx(0);C.create_line(X,0,X,hoogteC,fill='black',arrow='first')
    Y=transy(0);C.create_line(0,Y,breedte,Y,fill='black',arrow='last')
# rooster
    for x in range(int(xmi),int(xma+1)):
        X=transx(x);C.create_line(X,0,X,hoogteC,fill='lightgray',dash=[1,2])
    for y in range(int(ymi),int(yma+1)):
        Y=transy(y);C.create_line(0,Y,breedte,Y,fill='lightgray',dash=[1,2] )
#kromme
    lis=[];stap=0.05;x=xmi
    while x<xma:
        x=x+stap;ap(lis,x,f(x))
    C.create_line(lis,fill='green')
    lis=[];x=xmi+0.2;ap(lis,x,f(x));C.create_text(lis,fill='green',text='f(x)')
    lis=[];stap=0.05;x=xmi
    while x<xma:
        x=x+stap;ap(lis,x,g(x))
    C.create_line(lis,fill='magenta')
    lis=[];x=xmi+0.2;ap(lis,x,g(x));C.create_text(lis,fill='magenta',text='g(x)')
#stippels
    for p in (a,b,c):
        lis=[];ap(lis,p,0);ap(lis,p,f(p));ap(lis,0,f(p));C.create_line(lis,dash=[1,2],fill='darkgrey')
        lis=[];ap(lis,p,0);ap(lis,p,g(p));ap(lis,0,g(p));C.create_line(lis,dash=[1,2],fill='darkgrey')
#punten
    for p in (a,b,c):rondje(p,f(p));rondje(p,0);rondje(0,f(p));rondje(p,g(p));rondje(0,g(p))
#tekst
    for ps
in('a','b','c'):naamX(eval(ps),ps);psyf='f('+ps+')';naamY(eval(psyf),psyf);psyg='g('+ps+')';naamY(eval(psyg),psyg)
def naamX(p,pn):lis=[];ap(lis,p,-0.2);C.create_text(lis,text=pn)
def naamY(p,pn):lis=[];ap(lis,-0.3,p);C.create_text(lis,text=pn)
def rondje(a,b):lis=[];r=0.05;ap(lis,a-r,b-r);ap(lis,a+r,b+r);C.create_oval(lis,fill='black')
def info():h=messagebox.showinfo(tit,infstr)
```

```
def ap(li,x,y):X=transx(x);Y=transy(y);li.append(X);li.append(Y)
def wis(E):E.delete(0,len(E.get()))
def ent(tek,kl,xp,yp):
   Label(form,text=tek).place(x=xp,y=yp)
   E=Entry(form,width=8,bg=gr,fg=kl);E.place(x=xp+45,y=yp)
   return E
def nieuw():
   for E in (E1,E2,E3,E4,E5,E6,E7,E8,E9,E10):wis(E)
   C.delete(ALL)
#Hoofdprogramma
tit="Middelwaardestelling  Cauchy";gr='lightgrey';eps=1E-7;dx=1E-6
breedte=520;hoogte=540;hoogteC=hoogte-105;hoInv1=hoogte-95;hoInv2=hoogte-65;hoInv3=hoogte-35
form=Tk();form.geometry(str(breedte)+'x'+str(hoogte));form.title(tit);form.resizable(False,False)
C = Canvas(form, bg="white", height=hoogteC, width=breedte);C.pack()
#invoer
Label(form,text='f(x),g(x) :').place(x=5,y=hoInv1)
E1=Entry(form,width=26);E1.place(x=65,y=hoInv1);E1.insert(0,'cos(x),(x**2+x)/5')
Label(form,text='XYmima:').place(x=295,y=hoInv1)
E2=Entry(form,width=8);E2.place(x=360,y=hoInv1);E2.insert(0,'-2,5,-2,5')
Label(form,text='a,b:').place(x=5,y=hoInv2)
E3=Entry(form,width=6);E3.place(x=40,y=hoInv2);E3.insert(0,'1,4')
Button(form,text='c=',command=newton,width=1).place(x=5,y=hoInv3,width=20)
E4=Entry(form,width=6);E4.place(x=40,y=hoInv3);E4.insert(0,'2.5')
#uitvoer
E5=ent("  f'c:",'red',85,hoInv2);E6=ent("  g'c:",'red',200,hoInv2); E7=ent("f'c/g'c:",'red',315,hoInv2)
E8=ent("  Δf:",'blue',85,hoInv3);E9=ent("  Δg:",'blue',200,hoInv3);E10=ent("Δf/Δg:",'blue',315,hoInv3)
Button(form,text='Teken',command=bereken).place(x=440,y=hoInv2,width=60)
Button(form,text='Info',command=info).place(x=440,y=hoInv1,width=60)
Button(form,text='Nieuw',command=nieuw).place(x=440,y=hoInv3,width=60)
infstr="Probeer een c ∈]a,b[ in te vullen zodat\nf'(c)/g'(c) = Δf/Δg=[f(b)-f(a)]/[g(b)-g(a)]\n"
infstr=infstr+"Hint: wijzig c een beetje\nAls f'(c)/g'(c) dichter bij Δf/Δg\n"
infstr=infstr+"komt zit je al in de goede richting.\nMet de knop 'c=' kan je uiteindelijk\n"
infstr=infstr+"de exacte waarde van c berekenen."
form.mainloop()
```



164. Kromming - kromtecirkel - evolute kromming

Op de kromme y=f(x) is de gemiddelde kromming tussen P en Q intuïtief $= \dfrac{\alpha_2 - \alpha_1}{s_{AQ} - s_{AP}}$



Hieruit volgt: de kromming van y=f(x) in $P(x_0,y_0)$ $= k = \lim_{h\to 0} \dfrac{\alpha(s+h) - \alpha(s)}{h} = \alpha'_s$

Bepaling van k in functie van y' en y'' :

$y'_x = \tan(\alpha)$, dus $\alpha = \operatorname{atan}(y'_x)$ zodat $\alpha'_x = \dfrac{y''_x}{1+y'^2_x}$. Ook geldt: $s'_x = \sqrt{1+y'^2_x}$

$$k = \alpha'_s = \alpha'_x \cdot x'_s = \frac{\alpha'_x}{s'_x} = \frac{y''_x}{\left(\sqrt{1+y'^2_x}\right)^3}$$

De kromtestraal definiëren we als het omgekeerde van de kromming: r=1/k
Om de kromtecirkel KC te bepalen, nemen we de normaal in P op y=f(x). Het middelpunt M van de kromtecirkel ligt op afstand r op deze normaal. De kromtecirkel heeft dus in P dezelfde raaklijn als y=f(x). De verzameling van alle middelpunten van kromtecirkels die we bekomen als we P de kromme y=f(x) laten doorlopen, noemen we de evolute E
We bepalen nu ook de coördinaten van $M(x_1,y_1)$:

$$r = \frac{1}{k} = \frac{\left(\sqrt{1+y'^2_x}\right)^3}{y''_x}$$

Als β de hoek is tussen normaal en positieve x-as, dan is tanβ= rico normaal $= \frac{-1}{y'_x}$

Voor M geldt: $x_1=x_0 + r\cdot\cos\beta$ en $y_1=y_0 + r\cdot\sin\beta$

met $\cos^2\beta = \frac{1}{1+\tan^2\beta} = \frac{y'^2_x}{1+y'^2_x}$.

Stijgt de kromme, dan is $y'_x>0$, maar $\beta > \pi/2$
Daalt de kromme, dan is $y'_x<0$, maar $\beta < \pi/2$

Dus geldt steeds: $\cos\beta = \frac{-y'_x}{\sqrt{1+y'^2_x}}$ en $\sin\beta = \cos\beta\cdot\tan\beta = \frac{-y'_x}{\sqrt{1+y'^2_x}}\cdot\frac{-1}{y'_x} = \frac{1}{\sqrt{1+y'^2_x}}$

Voor M geldt dan ook:

$$x_1 = x_0 + \frac{\left(\sqrt{1+y'^2_x}\right)^3}{y''_x}\cdot\frac{-y'_x}{\sqrt{1+y'^2_x}} = x_0 - \frac{\left(1+y'^2_x\right)\cdot y'_x}{y''_x}$$

$$y_1 = y_0 + \frac{\left(\sqrt{1+y'^2_x}\right)^3}{y''_x}\cdot\frac{1}{\sqrt{1+y'^2_x}} = y_0 + \frac{\left(1+y'^2_x\right)}{y''_x}$$

Dit geldt enkel als $y'' \neq 0$. Is y''=0, m.a.w. in een buigpunt wordt de straal van de kromtecirkel ∞ groot. M ligt dan ook op +∞ of -∞ van de normaal.
T.o.v. een punt P(x,y) van de kromme y=f(x) worden de coördinaten (x_1,y_1) van M berekend met:
if d2f(x)==0: return []
else:cof=(1+df(x)**2)/d2f(x);x1=x-cof*df(x);y1=y+cof;return [x1,y1]

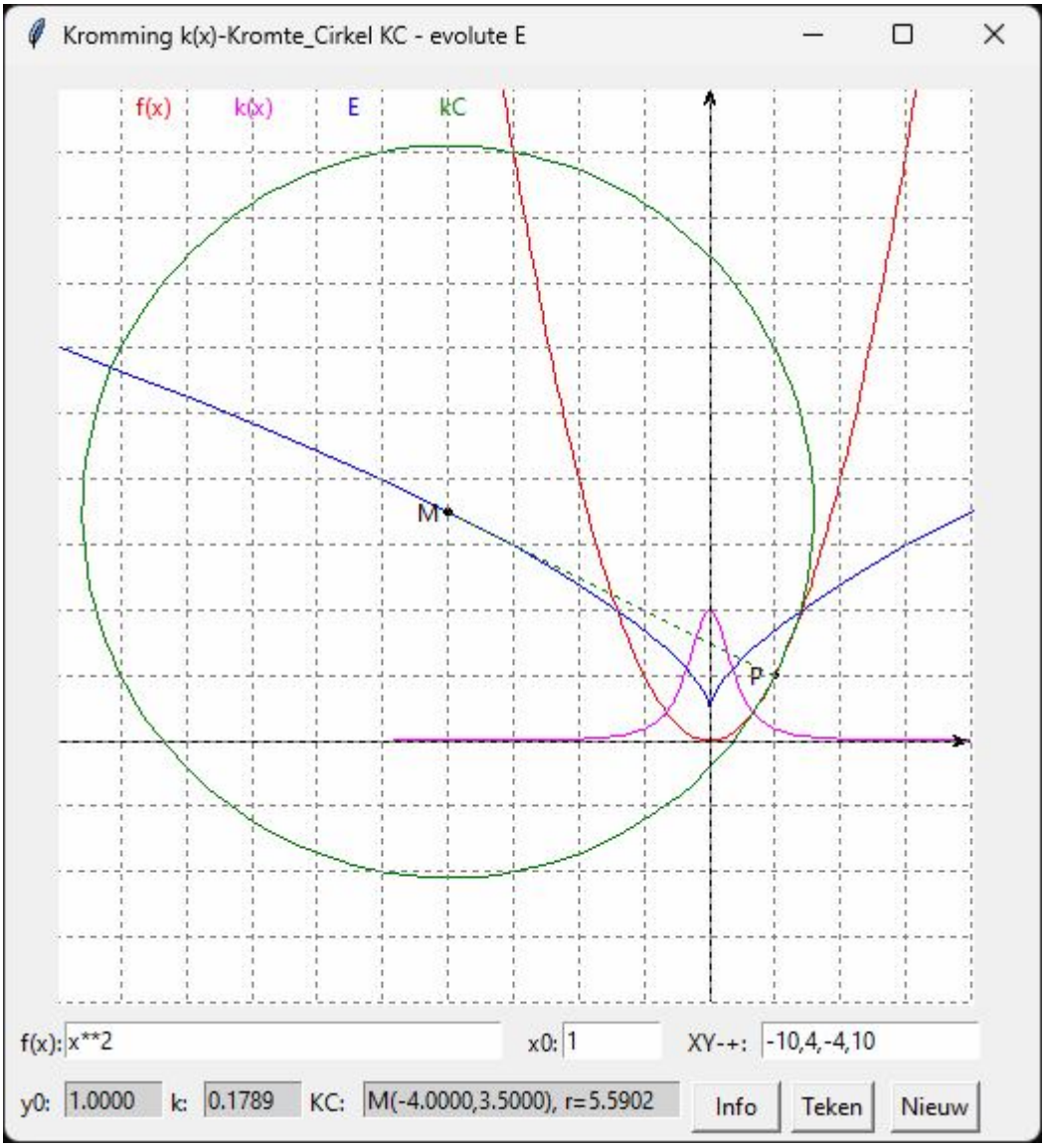
In het volgende programma wordt voor een functie y=f(x) behalve de kromme y=f(x) ook de kromming k(x), de evolute E en een kromtecirkel getekend.

Programma

```
# kromming, kromtecirkel, evolute met tkinter
from math import *;from tkinter import *;from tkinter import messagebox
def mult(s):
   l=s.split(',');co=[]
   for i in l:co.append(float(i))
   return co
def transx(x):return (x-xmi)*breedteC/(xma-xmi)
def transy(y):return hoogteC-hoogteC*(y-ymi)/(yma-ymi)
def f(x):return eval(E1.get())
```

```
def df(x):return (f(x+dx)-f(x-dx))/dx/2
def d2f(x):return (f(x+d2x)+f(x-d2x)-2*f(x))/d2x**2
def kr(x):return d2f(x)/sqrt(1+df(x)**2)**3
def mid_evol(x,y):
   if d2f(x)==0: return []
   else:cof=(1+df(x)**2)/d2f(x);x1=x-cof*df(x);y1=y+cof;return [x1,y1]
def teken():
   global xmi,xma,ymi,yma
   xmi,xma,ymi,yma=mult(E2.get());C.delete(ALL)
   for E in (E4,E5,E6):E.delete(0,len(E.get()))
#krommen
   lisF=[];lisK=[];lisE=[];stap=0.05;x=xmi
   while x<xma:
      x=x+stap;y=f(x);ap(lisF,x,y)
      ap(lisK,x,kr(x))
      compev=mid_evol(x,y)
      if compev!=[]:x1,y1=compev;ap(lisE,x1,y1)
   C.create_line(lisF,fill='red')       # functie
   C.create_line(lisK,fill='magenta')   # kromming
   C.create_line(lisE,fill='blue')      # evolute
#kromtecirkel
x0=float(E3.get());k=kr(x0);r=abs(1/k);y0=f(x0);E4.insert(0,format(y0,nwk));E5.insert(0,format(k,nwk))
   x1,y1=mid_evol(x0,y0)
   lisKC=[];stap=0.05;t=-stap
   while t<2*pi:
      t=t+stap;x2,y2=x1+r*cos(t),y1+r*sin(t)
      ap(lisKC,x2,y2)
   C.create_line(lisKC,fill='green')
   punt(x0,y0,'P');punt(x1,y1,'M')
   lis=[];ap(lis,x0,y0);ap(lis,x1,y1);C.create_line(lis,fill='green',dash=[1,2])
   E6.insert(0,'M('+format(x1,nwk)+','+format(y1,nwk)+'), r='+format(r,nwk))
   C.create_text(50,10,text='f(x)',fill='red')
   C.create_text(100,10,text='k(x)',fill='magenta')
   C.create_text(150,10,text='E',fill='blue')
   C.create_text(200,10,text='kC',fill='green')
#assen
   X=transx(0);line = C.create_line(X,0,X,hoogteC,fill='black',arrow='first')
   Y=transy(0);line = C.create_line(0,Y,breedteC,Y,fill='black',arrow='last')
#rooster
   for x in range(int(xmi),int(xma+1)):
      X=transx(x);line = C.create_line(X,0,X,hoogteC,fill='gray',dash=[1,2])
   for y in range(int(ymi),int(yma+1)):
      Y=transy(y);line = C.create_line(0,Y,breedteC,Y,fill='gray',dash=[1,2] )
def punt(a,b,pn):
   lis=[];ap(lis,a-0.3,b);C.create_text(lis,text=pn)
   lis=[];r=0.05;ap(lis,a-r,b-r);ap(lis,a+r,b+r);C.create_oval(lis,fill='black')
def ap(li,x,y):X=transx(x);Y=transy(y);li.append(X);li.append(Y)
def info():h=messagebox.showinfo(tit,infstr)
def nieuw():
   for E in (E1,E2,E3,E4,E5,E6):E.delete(0,len(E.get()))
   C.delete(ALL)
```

```
#hoofdprogramma
tit='Kromming k(x)-Kromte_Cirkel KC - evolute E'
lg='lightgrey';nwk='.4f';dx=1E-6;d2x=1E-4
breedte=520;hoogte=540;breedteC=460;hoogteC=460;hoInv1=hoogte-60;hoInv2=hoogte-30
form=Tk();form.geometry('520x540');form.title(tit)
C = Canvas(form, bg="white", height=hoogteC, width=breedteC);C.place(x=25,y=10)
Label(form,text='f(x):').place(x=5,y=hoInv1)
E1=Entry(form);E1.place(x=30,y=hoInv1,width=220);E1.insert(0,'x**2')
Label(form,text='XY-+:').place(x=340,y=hoInv1)
E2=Entry(form);E2.place(x=380,y=hoInv1,width=110);E2.insert(0,'-10,4,-4,10')
Label(form,text='x0:').place(x=260,y=hoInv1)
E3=Entry(form);E3.place(x=280,y=hoInv1,width=50);E3.insert(0,'1')
Label(form,text='y0:').place(x=5,y=hoInv2)
E4=Entry(form,bg=lg);E4.place(x=30,y=hoInv2,width=50)
Label(form,text='k:').place(x=80,y=hoInv2)
E5=Entry(form,bg=lg);E5.place(x=100,y=hoInv2,width=50)
Label(form,text='KC:').place(x=150,y=hoInv2)
E6=Entry(form,bg=lg);E6.place(x=180,y=hoInv2,width=160)
Button(form,text='Info',command=info).place(x=345,y=hoInv2,width=45)
Button(form,text='Teken',command=teken).place(x=395,y=hoInv2)
Button(form,text='Nieuw',command=nieuw).place(x=445,y=hoInv2)
infstr="k=kromming=α's , r=1/k= straal\n van de kromtecirkel KC aan de\n"
infstr=infstr+'kromme y=f(x) in het punt P(x0,y0)\n'
infstr=infstr+"E= evolute= verzameling\n middelpunten M(x1,y1) van KC's\n"
infstr=infstr+'M ligt op de normaal van y=f(x) in P\n'
infstr=infstr+'MP raakt aan de evolute E in M'
form.mainloop()
```



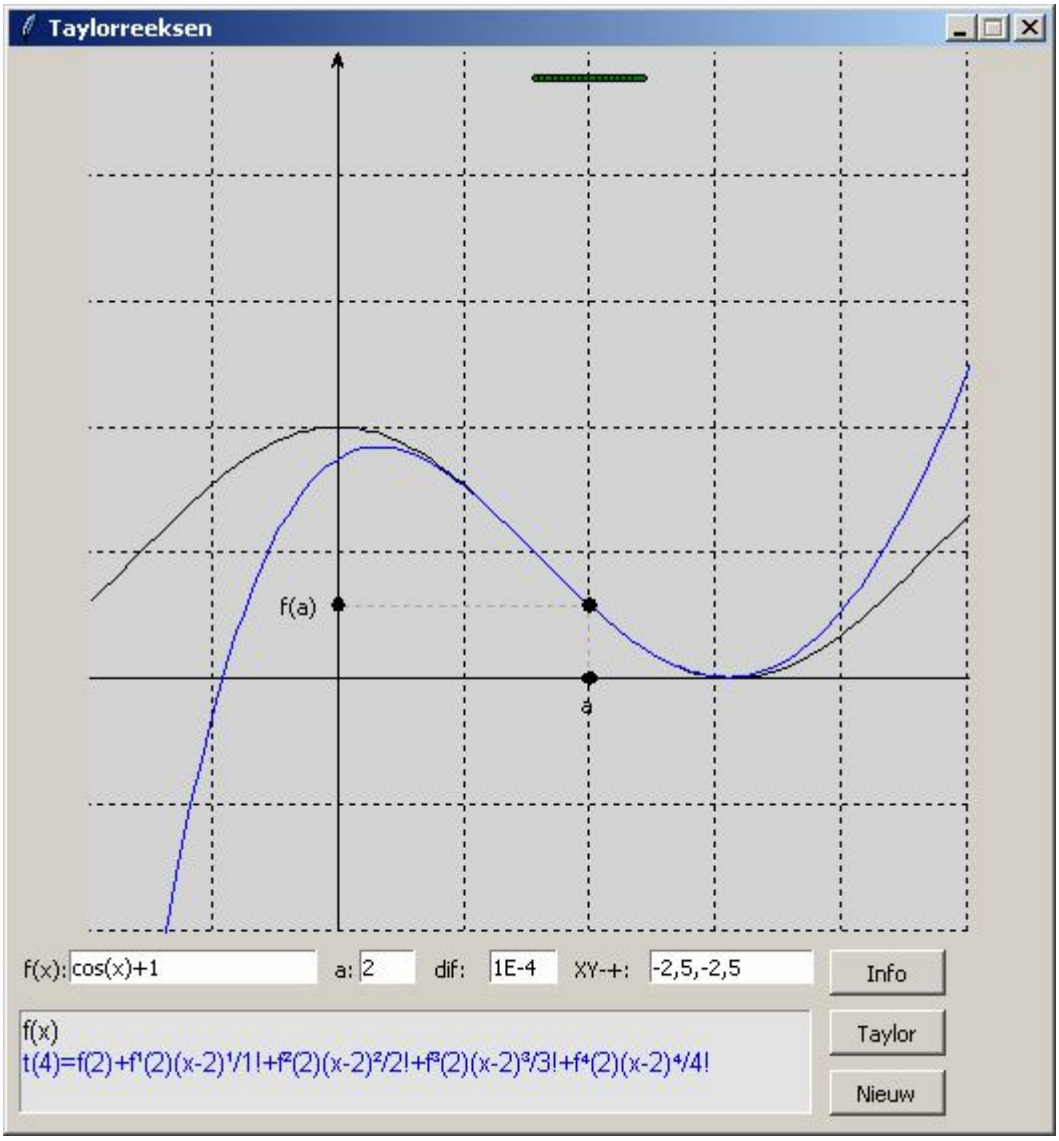
165. Taylorreeksen Taylor_grafiek

In hoeverre is een Taylorfunctie een benadering van f(x) in een omgeving van a ?
Behalve de grafiek van een functie f(x) wordt met dit programma ook de grafiek getekend van een stukje van de overeenstemmende Taylorfunctie in een gegeven punt a.
Je ziet dan dat, hoe meer termen de Taylorfunctie bevat, het interval (groen lijnstuk) waarin | f(x)-t(x,i)|<dif groter wordt

```
# test taylor
from math import *;from tkinter import *;from tkinter import messagebox
def mult(s):
    l=s.split(',');co=[]
    for i in l:co.append(float(i))
    return co
def transx(x):return (x-xmi)*breedteC/(xma-xmi)
def transy(y):return hoogteC-hoogteC*(y-ymi)/(yma-ymi)
def v(x):
    if x<=0:sgn='+'
```

```
    else:sgn='-'
    return sgn+str(abs(x))
def fac(j):
    if j==1:return 1
    else: return j*fac(j-1)
def f(x):return eval(E1.get())
def df(x,i):
    dx=10**(i-6)
    if i==0:return f(x)
    else: return(df(x+dx,i-1)-df(x-dx,i-1))/dx/2
def t(x,i):
    T=f(a)
    for j in range(1,i+1):
        T=T+df(a,j)*(x-a)**j/fac(j)
    return T
def bereken():
    global xmi,xma,ymi,yma,a,i,col;C.delete(ALL)
    xmi,xma,ymi,yma=mult(E4.get())
    a=float(E2.get());dif=float(E3.get())
    if a==int(a):a=int(a)
#assen
    X=transx(0);C.create_line(X,0,X,hoogteC,fill='black',arrow='first')
    Y=transy(0);C.create_line(0,Y,breedte,Y,fill='black',arrow='last')
# rooster
    for x in range(int(xmi),int(xma+1)):
        X=transx(x);C.create_line(X,0,X,hoogteC,fill='black',dash=[1,2])
    for y in range(int(ymi),int(yma+1)):
        Y=transy(y);C.create_line(0,Y,breedte,Y,fill='black',dash=[1,2] )
#kromme
    lis=[];stap=0.05;x=xmi
    while x<xma:
        x=x+stap;ap(lis,x,f(x))
    C.create_line(lis,fill='black')
#taylor
    col=['red','green','orange','blue','magenta','darkblue']
    lis=[];stap=0.05;x=xmi
    while x<xma:
        x=x+stap;ap(lis,x,t(x,i))
    C.create_line(lis,fill=col[i-1])
    taylverg='t('+str(i)+')=f('+str(a)+')'
    for j in range(1,i+1):
        taylverg=taylverg+'+f'+supdig[j]+'('+str(a)+')(x'+v(a)+')'+supdig[j]+'/'+str(j)+'!'
    ad('f(x)\n'+taylverg)
#stippels
    p=a;lis=[];ap(lis,p,0);ap(lis,p,f(p));ap(lis,0,f(p));C.create_line(lis,dash=[1,2],fill='darkgrey')
#punten
    p=a;rondje(p,f(p));rondje(p,0);rondje(0,f(p))
#tekst
    ps='a';naamX(eval(ps),ps);psy='f('+ps+')';naamY(eval(psy),psy)
#interval
    stap=0.05;x=xmi
```

```
    while x<xma:
        x=x+stap
        if abs(t(x,i)-f(x))<dif:lis=[];ap(lis,x-0.05,yma-0.25);ap(lis,x+0.05,yma-0.20);C.create_oval(lis,fill='green')
    i=i%6+1
def naamX(p,pn):lis=[];ap(lis,p,-0.2);C.create_text(lis,text=pn)
def naamY(p,pn):lis=[];ap(lis,-0.3,p);C.create_text(lis,text=pn)
def rondje(a,b):lis=[];r=0.05;ap(lis,a-r,b-r);ap(lis,a+r,b+r);C.create_oval(lis,fill='black')
def info():h=messagebox.showinfo(tit,infstr)
def ap(li,x,y):X=transx(x);Y=transy(y);li.append(X);li.append(Y)
def
ad(lin):tex.delete('1.0',END);tex.insert(INSERT,lin);tex.tag_add("start","2.0","3.0");tex.tag_config("start",foregro
und=col[i-1])
def wis(E):E.delete(0,len(E.get()))
def nieuw():
    for E in (E1,E2,E3,E4):wis(E)
    C.delete(ALL);tex.delete('1.0',END)
#Hoofdprogramma
supdig=['⁰','¹','²','³','⁴','⁵','⁶','⁷'];i=1
tit='Taylorreeksen';gr='lightgrey'
breedte=520;hoogte=540;hoogteC=hoogte-100;breedteC=hoogteC;hoInv1=hoogte-90;hoInv2=hoogte-
60;hoInv3=hoogte-30
form=Tk();form.geometry(str(breedte)+'x'+str(hoogte));form.title(tit);form.resizable(False,False)
C = Canvas(form, bg=gr, height=hoogteC, width=breedteC);C.pack()
Label(form,text='f(x):').place(x=5,y=hoInv1)
E1=Entry(form,width=20);E1.place(x=30,y=hoInv1);E1.insert(0,'cos(x)+1')
Label(form,text='a:').place(x=160,y=hoInv1)
E2=Entry(form,width=4);E2.place(x=175,y=hoInv1);E2.insert(0,'2')
Label(form,text='dif:').place(x=210,y=hoInv1)
E3=Entry(form,width=5);E3.place(x=240,y=hoInv1);E3.insert(0,'1E-4')
Label(form,text='XY-+:').place(x=280,y=hoInv1)
E4=Entry(form,width=13);E4.place(x=320,y=hoInv1);E4.insert(0,'-2,5,-2,5')
tex=Text(form,bg='grey89',height=3,width=56,wrap=WORD,font=('"TkDefaultFont"','10'));tex.place(x=5,y=hoIn
v2)
Button(form,text='Taylor',command=bereken,width=8).place(x=410,y=hoInv2)
Button(form,text='Info',command=info,width=8).place(x=410,y=hoInv1)
Button(form,text='Nieuw',command=nieuw,width=8).place(x=410,y=hoInv3)
infstr='"Telkens je op 'Taylor' tikt, krijg je\n"
infstr=infstr+'de volgende term van de taylorreeks\n'
infstr=infstr+'en de grafiek van f(x) en t(x) te zien\n'
infstr=infstr+'(tot en met de 7de term)\n'
infstr=infstr+'Bovenaan wordt het interval getoond\n'
infstr=infstr+'waarbinnen |f(x)-t(x)|<dif'
form.mainloop()
```



166. Onderlinge stand van 2 rechten in het vlak rechten

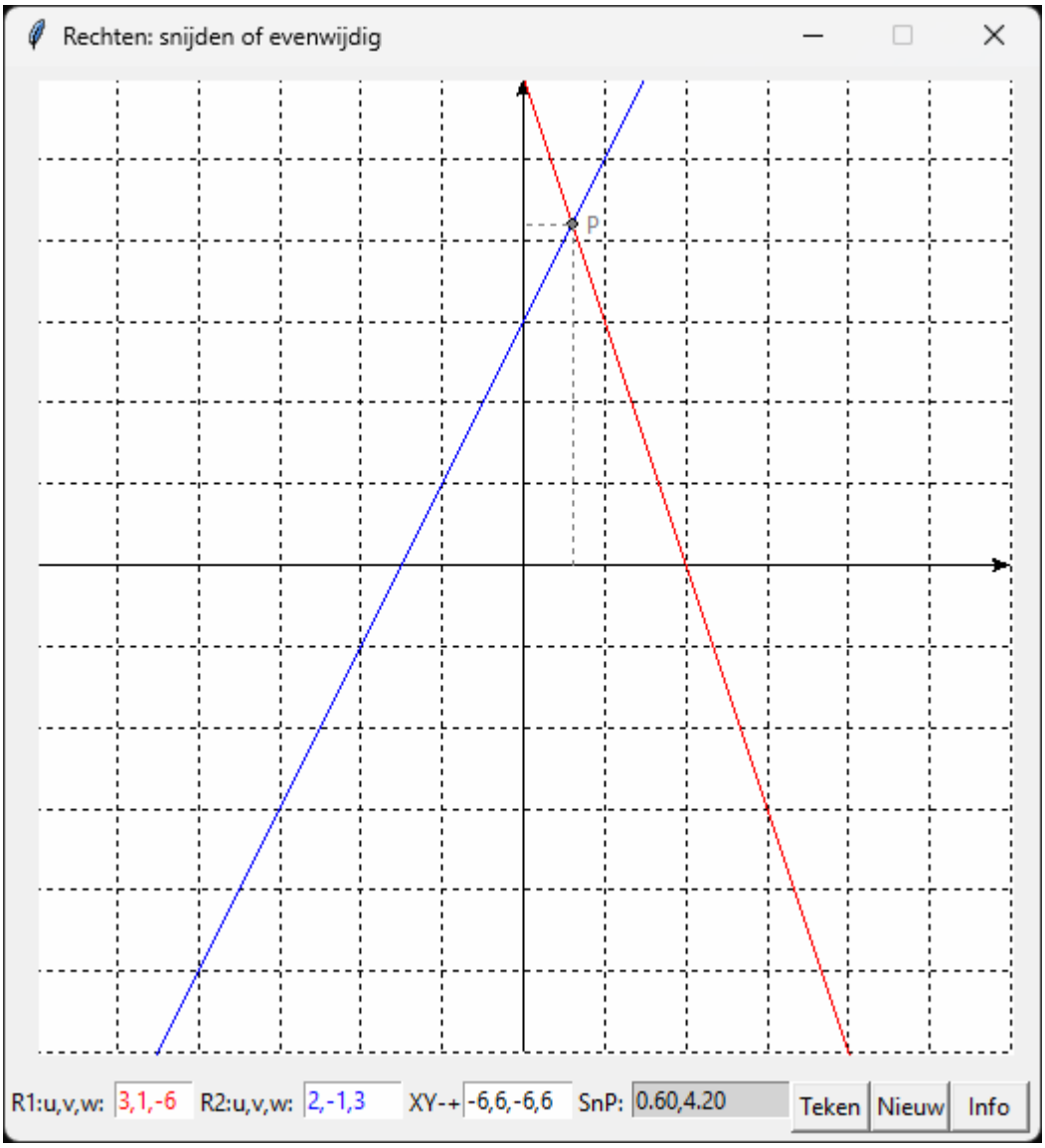
Een eenvoudig programma om de eventuele snijpunten te berekenen van 2 rechten:
er zijn 3 mogelijkheden: snijden (als het stelsel van de vergelijkingen 1 oplossing heeft)
evenwijdig (als het stelsel strijdig is), samenvallend (als het stelsel ∞ veel oplossingen heeft)
Afzonderlijk moeten we het geval behandelen als één rechte verticaal is.

Programma:

```
#snijden van rechten
from math import *;from tkinter import *;from tkinter import messagebox
def mult(s):
    l=s.split(',');co=[]
    for i in l:co.append(float(i))
    return co
def transx(x):return (x-xmi)*breedteC/(xma-xmi)
def transy(y):return hoogteC-hoogteC*(y-ymi)/(yma-ymi)
def teken():
    global xmi,xma,ymi,yma,r
```

```
    xmi,xma,ymi,yma=mult(E3.get());C.delete(ALL);bl='black'
#assen
    X=transx(0);C.create_line(X,0,X,hoogteC,fill=bl,arrow='first')
    Y=transy(0);C.create_line(0,Y,breedteC,Y,fill=bl,arrow='last')
#rooster
    for x in range(int(xmi),int(xma+1)):
        X=transx(x);C.create_line(X,0,X,hoogteC,fill=bl,dash=[1,2])
    for y in range(int(ymi),int(yma+1)):
        Y=transy(y);C.create_line(0,Y,breedteC,Y,fill=bl,dash=[1,2] )
#rechten tekenen
    u1,v1,w1=mult(E1.get());u2,v2,w2=mult(E2.get())
    if (u1==0 and v1==0) or (u2==0 and v2==0):
        messagebox.showinfo('Foutieve invoer','Minstens één van beide\nis geen rechte')
    if v1!=0:
        lis=[];ap(lis,xmi,(-u1*xmi-w1)/v1);ap(lis,xma,(-u1*xma-w1)/v1)
    else:
        lis=[];ap(lis,-w1/u1,ymi);ap(lis,-w1/u1,yma)
    C.create_line(lis,fil='red')
    if v2!=0:
        lis=[];ap(lis,xmi,(-u2*xmi-w2)/v2);ap(lis,xma,(-u2*xma-w2)/v2)
    else:
        lis=[];ap(lis,-w2/u2,ymi);ap(lis,-w2/u2,yma)
    C.create_line(lis,fill='blue')
#oplossen stelsel (snijpunt)
    d=u1*v2-u2*v1
    if d!=0:
        xd=-w1*v2+w2*v1;yd=-u1*w2+w1*u2
        x=xd/d;y=yd/d; opl=format(x,nwk)+','+format(y,nwk)
        rondje(x,y);lis=[];ap(lis,x,0);ap(lis,x,y);ap(lis,0,y);C.create_line(lis,fill='grey',dash=[1,2])
    elif v1*w2-w1*v2!=0 or u1*w2-w1*u2!=0:opl='Evenwijdig'
    else: opl='Samenvallend'
    wis(E4);E4.insert(0,opl)
def rondje(a,b):
    Rr=0.05;lis=[];ap(lis,a-Rr,b-Rr);ap(lis,a+Rr,b+Rr);C.create_oval(lis,fill='grey')
    lis=[];ap(lis,a+5*Rr,b);C.create_text(lis,fill='grey',text='P')
def ap(li,x,y):X=transx(x);Y=transy(y);li.append(X);li.append(Y)
def info():messagebox.showinfo(tit+': info',infstr)
def wis(E):E.delete(0,len(E.get()))
def nieuw():
    for E in (E1,E2,E3,E4):wis(E)
    C.delete(ALL)
# hoofdprogramma
tit='Rechten: snijden of evenwijdig';sc=1;nwk='.2f'
breedte=(520*sc);hoogte=(540*sc);hoogteC=490*sc;breedteC=hoogteC;hoInv=hoogte-30*sc
form=Tk();form.geometry(str(breedte)+'x'+str(hoogte))
form.title(tit);form.resizable(False,False)
C = Canvas(form, bg="white", height=hoogteC, width=breedteC)
C.place(x=15,y=5);Label(form,text='R1:u,v,w:',).place(x=0,y=hoInv)
E1=Entry(form,fg='red');E1.place(x=55*sc,y=hoInv,width=40*sc);E1.insert(0,'3,1,-6')
Label(form,text='R2:u,v,w:',).place(x=95*sc,y=hoInv)
E2=Entry(form,fg='blue');E2.place(x=150*sc,y=hoInv,width=50*sc);E2.insert(0,'2,-1,3')
```

```
Label(form,text='XY-+:').place(x=200*sc,y=hoInv)
E3=Entry(form);E3.place(x=230*sc,y=hoInv,width=60*sc);E3.insert(0,'-6,6,-6,6')
Label(form,text='SnP:').place(x=285*sc,y=hoInv)
E4=Entry(form,bg='lightgrey');E4.place(x=315*sc,y=hoInv,width=80*sc)
Button(form,text='Teken',command=teken).place(x=395*sc,y=hoInv,width=40)
Button(form,text='Nieuw',command=nieuw).place(x=435*sc,y=hoInv,width=40)
Button(form,text='Info',command=info).place(x=475*sc,y=hoInv,width=40)
infstr='Gegeven 2 rechten\nu1.x+v1.y+w1=0\nu2.x+v2.y+w2=0'
infstr=infstr+'\nDe oplossing van het stelsel\nvan deze 2 vergelijkingen geeft'
infstr=infstr+'\nde coördinaten van het snijpunt P\n'
infstr=infstr+'Heeft het stelsel geen oplossing,\ndan zijn de rechten // of samenvallend'
form.mainloop()
```



167. Regelmatige veelhoeken veelhoeken

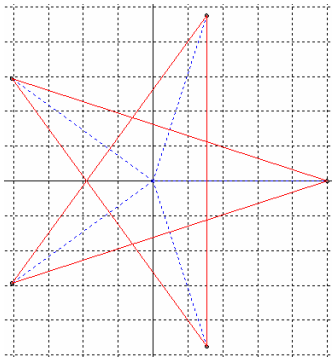
In dit programma wordt de grafiek van een regelmatige convexe of concave veelhoek getekend. Stel x_1,y_1=middelpunt regelmatige veelhoek, n het aantal hoekpunten, p het verschil van de indices van opeenvolgende hoekpunten , r de straal van de omgeschreven cirkel.

De 5 hoekpunten worden gegeven door x,y= x_1+r*cos(t),y_1+r*sin(t) waarbij $t = \frac{2 \cdot \pi \cdot p.i}{n}$

In de tekening is x_1,y_1=0,0 n=5 p=2 of 3 r=5

In het volgende programmaonderdeel wordt in de while lus de lijst liss1 telkens opgevuld met de coördinaten van middelpunt en 1 hoekpunt.
Deze liss1 wordt onmiddellijk (binnen de lus) getekend (blauwe stippellijn).
De lijst lis wordt binnen de lus achtereenvolgens opgevuld met de coördinaten van al de hoekpunten. Pas na de lus wordt de veelhoek getekend (rode volle lijn)

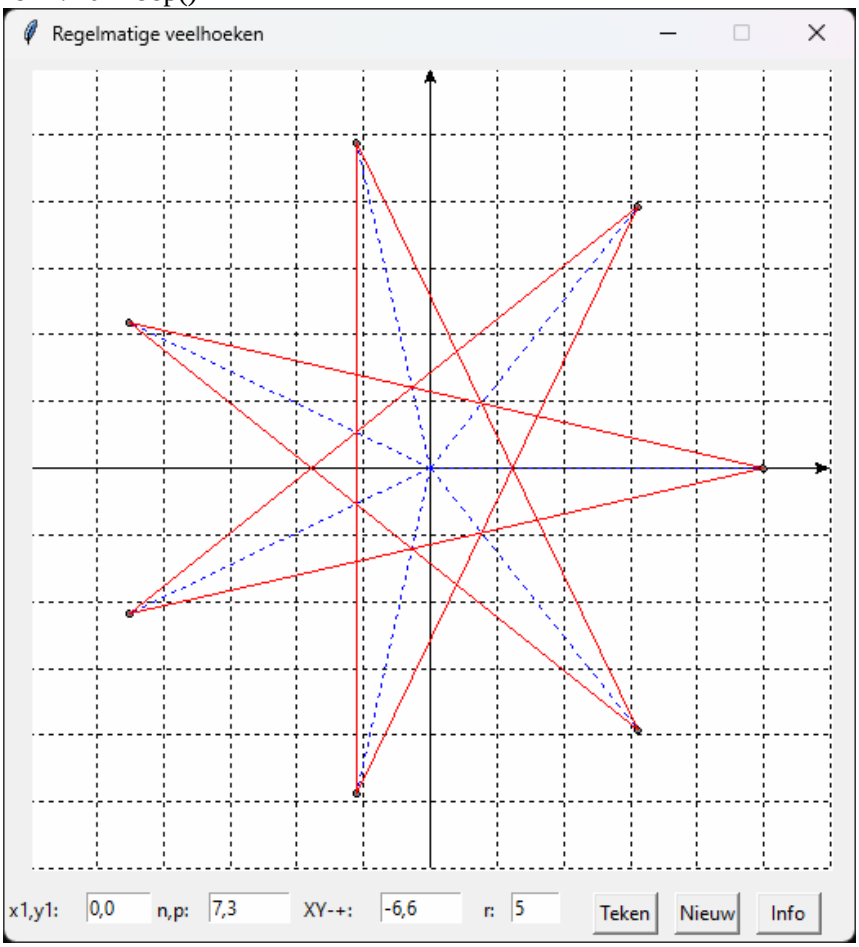
```
lis=[];stap=2*pi*p/n;t=-stap
if n<=p:messagebox.showinfo('Fout','Kies n>p');return
while t<2*pi*p:
    x2,y2=x1+r*cos(t),y1+r*sin(t);rondje(x2,y2)
    lissl=[];ap(lissl,x1,y1);ap(lissl,x2,y2)
    C.create_line(lissl,fill='blue',dash=[1,2])
    ap(lis,x2,y2);t=t+stap
C.create_line(lis,fill='red')
```



Programma:

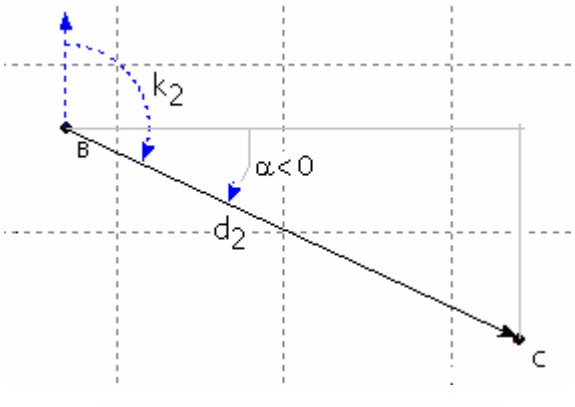
```
#regelmatige veelhoeken
from math import *;from tkinter import *;from tkinter import messagebox
def mult(s):
   l=s.split(',');co=[]
   for i in l:co.append(float(i))
   return co
def transx(x):return (x-xmi)*breedteC/(xma-xmi)
def transy(y):return hoogteC-hoogteC*(y-ymi)/(yma-ymi)
def teken():
   global xmi,xma,ymi,yma,r
   xmi,xma=mult(E3.get());ymi,yma=xmi,xma;C.delete(ALL);bl='black'
#assen
   X=transx(0);C.create_line(X,0,X,hoogteC,fill=bl,arrow='first')
   Y=transy(0);C.create_line(0,Y,breedteC,Y,fill=bl,arrow='last')
#rooster
   for x in range(int(xmi),int(xma+1)):
      X=transx(x);C.create_line(X,0,X,hoogteC,fill=bl,dash=[1,2])
   for y in range(int(ymi),int(yma+1)):
      Y=transy(y);C.create_line(0,Y,breedteC,Y,fill=bl,dash=[1,2] )
#veelhoeken
   x1,y1=mult(E1.get())
   n,p=mult(E2.get());r=float(E4.get())
   lis=[];stap=2*pi*p/n;t=-stap
   if n<=p:messagebox.showinfo('Fout','Kies n>p');return
   while t<2*pi*p:
      x2,y2=x1+r*cos(t),y1+r*sin(t);rondje(x2,y2)
      lissl=[];ap(lissl,x1,y1);ap(lissl,x2,y2)
      C.create_line(lissl,fill='blue',dash=[1,2])
      ap(lis,x2,y2);t=t+stap
   C.create_line(lis,fill='red')
def rondje(a,b):Rr=0.05;lis=[];ap(lis,a-Rr,b-Rr);ap(lis,a+Rr,b+Rr);C.create_oval(lis,fill='grey')
```

```
def ap(li,x,y):X=transx(x);Y=transy(y);li.append(X);li.append(Y)
def info():messagebox.showinfo(tit+': info',infstr)
def wis(E):E.delete(0,len(E.get()))
def nieuw():
    for E in (E1,E2,E3):wis(E)
    C.delete(ALL)
# hoofdprogramma
tit='Regelmatige veelhoeken';sc=1
breedte=(520*sc);hoogte=(540*sc);hoogteC=490*sc;breedteC=hoogteC;hoInv=hoogte-30*sc
form=Tk();form.geometry(str(breedte)+'x'+str(hoogte))
form.title(tit);form.resizable(False,False)
C = Canvas(form, bg="white", height=hoogteC, width=breedteC)
C.place(x=15,y=5);Label(form,text='x1,y1:',).place(x=0,y=hoInv)
E1=Entry(form);E1.place(x=50*sc,y=hoInv,width=40*sc);E1.insert(0,'0,0')
Label(form,text='n,p:',).place(x=90*sc,y=hoInv)
E2=Entry(form);E2.place(x=125*sc,y=hoInv,width=50*sc);E2.insert(0,'7,3')
Label(form,text='XY-+:').place(x=180*sc,y=hoInv)
E3=Entry(form);E3.place(x=230*sc,y=hoInv,width=50*sc);E3.insert(0,'-6,6')
Label(form,text='r:').place(x=290*sc,y=hoInv)
E4=Entry(form);E4.place(x=310*sc,y=hoInv,width=30*sc);E4.insert(0,'5')
Button(form,text='Teken',command=teken).place(x=360*sc,y=hoInv,width=40)
Button(form,text='Nieuw',command=nieuw).place(x=410*sc,y=hoInv,width=40)
Button(form,text='Info',command=info).place(x=460*sc,y=hoInv,width=40)
infstr='Wijzig x1,y1   n,p\n xmi,xma     r\n'
form.mainloop()
```



168. **Verkenner** verkenner

Vanuit een startpunt A(0,0) vertrekt een verkenner en legt een afstand d_1 af met kompaskoers k_1. Daarna een 2de afstand d_2 met kompaskoers k_2, enz....



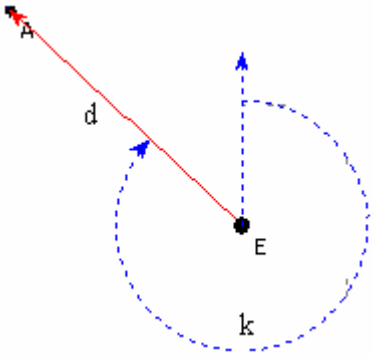
$$A(0,0) \xrightarrow{d_1,k_1} B(x_1,y_1) \xrightarrow{d_2,k_2} B(x_2,y_2).....$$

We zoeken eerst een formule om bvb. van B naar C te gaan.

Steeds geldt: $\alpha = \frac{\pi}{2} - k$

$x_2=x_1+d_2\cdot\cos\alpha = x_1+d_2\cdot\sin\alpha$; $y_2=y_1+d_2\cdot\sin\alpha = y_1+d_2\cdot\cos\alpha$

Maar hoe bepalen we de afstand d en kompashoek k om van een eindpunt, bvb. E(x,y) regelrecht terug te keren naar het startpunt A(0,0) ? : $d = \sqrt{x^2+y^2}$ en k =atan(x/y)



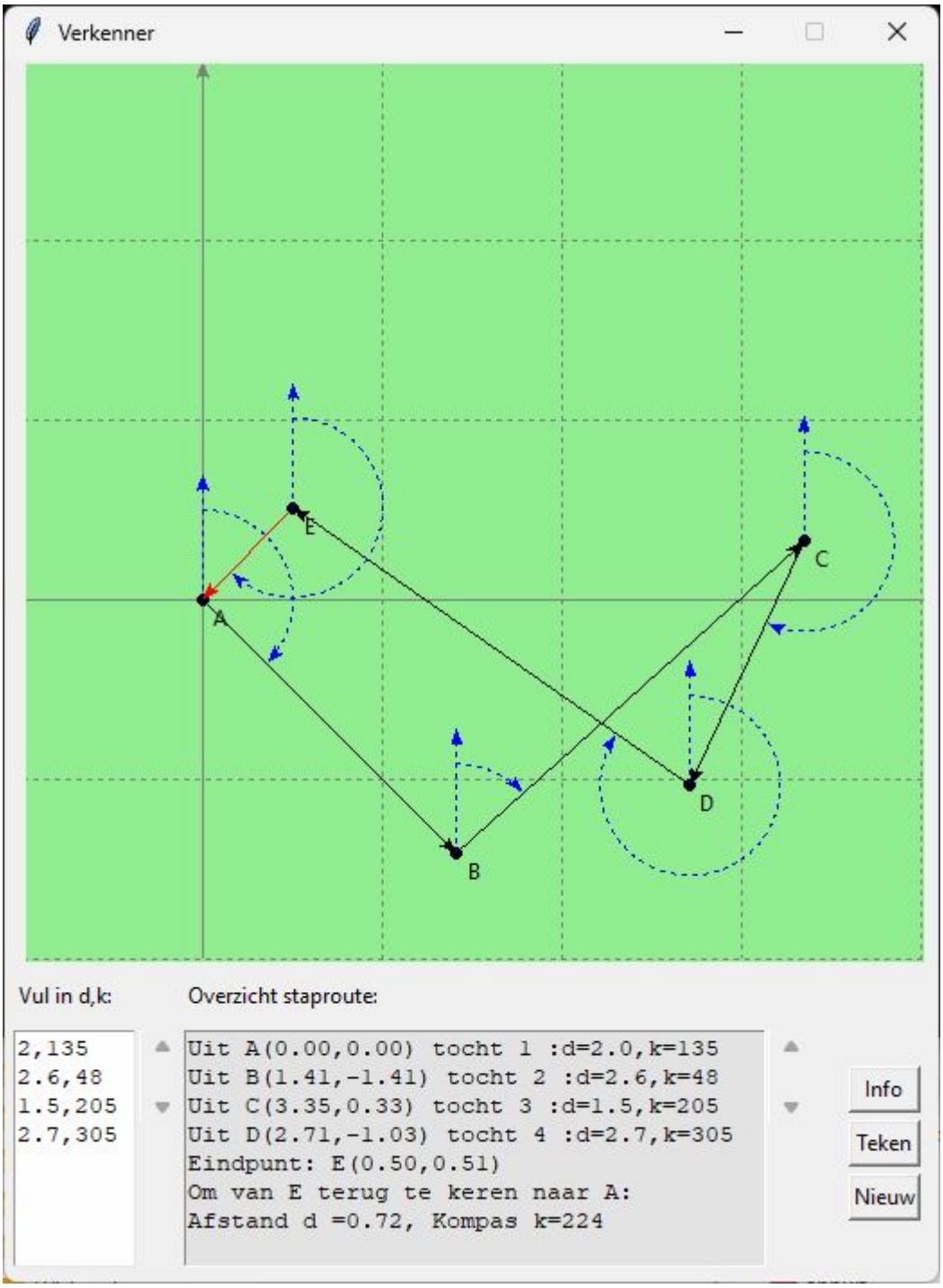
Als y>0 dan is k =k+180 , zoniet als x>0 dan is k=k+360

Programma

```
# verkenner
from math import *;from tkinter import *;from tkinter import messagebox
class Punt:
    def __init__(self,naam,xco,yco):self.naam=naam;self.x = xco;self.y = yco
class Rak:
    def __init__(self,naam,afstand,koers):self.naam=naam;self.d = afstand;self.k = koers
def mult(s):
    l=s.split(',');co=[]
    for i in l:co.append(float(i))
    return co
def transx(x):return (x-xmi)*breedteC/(xma-xmi)
def transy(y):return hoogteC-hoogteC*(y-ymi)/(yma-ymi)
def zoom(x,y):
    global xmi,ymi,xma,yma
    if x<xmi:xmi=int(x-1)
    if y<ymi:ymi=int(y-1)
    if x>xma:xma=int(x+1)
    if y>yma:yma=int(y+1)
def norm():
    global xmi,ymi,xma,yma
    if (yma-ymi)<(xma-xmi):yma=ymi+xma-xmi
    if (yma-ymi)>(xma-xmi):xma=xmi+yma-ymi
def teken():
    global xmi,xma,ymi,yma,k,d,dim
    xmi,xma,ymi,yma=-1,1,-1,1;C.delete(ALL);tex2.delete('1.0',END)
    lijst=tex1.get("1.0","end-1c");afst_koers=lijst.split('\n');lk=len(afst_koers)-1
    while afst_koers[lk]=="":afst_koers.pop();lk=lk-1
    x,y=0,0;start=Punt('A',0,0)
    inp='  ';tocht=[];p=[start]
```

```
    for i in range(0,len(afst_koers)):
        d,k=mult(afst_koers[i]);tocht.append(Rak(chr(97+i),d,k))
        x=x+d*sin(radians(k));y=y+d*cos(radians(k));zoom(x,y)
        p.append(Punt(chr(66+i),x,y))
        tex2.insert(INSERT,'Uit '+p[i].naam+'('+format(p[i].x,'.2f')+','+format(p[i].y,'.2f')+') tocht '+str(i+1)+'
:d='+str(tocht[i].d)+',k='+format(tocht[i].k,'.0f')+'\n')
    norm();dim=xma-xmi;p.append(Punt(chr(67+i),x,y))
    tex2.insert(INSERT,'Eindpunt: '+p[i+1].naam+'('+format(p[i+1].x,'.2f')+','+format(p[i+1].y,'.2f')+')\n')
#rooster
    for x in range(int(xmi),int(xma+1)):
        X=transx(x);C.create_line(X,0,X,hoogteC,fill='grey',dash=[1,2])
    for y in range(int(ymi),int(yma+1)):
        Y=transy(y);C.create_line(0,Y,breedte,Y,fill='grey',dash=[1,2] )
#assen
    X=transx(0);C.create_line(X,0,X,hoogteC,fill='grey',arrow='first')
    Y=transy(0);C.create_line(0,Y,breedte,Y,fill='grey',arrow='last')
    for i in range(0,len(afst_koers)):rondje(p[i].x,p[i].y,p[i].naam,tocht[i].k)
    for i in range(0,len(p)-2):lis=[];ap(lis,p[i].x,p[i].y);ap(lis,p[i+1].x,p[i+1].y);C.create_line(lis,arrow='last')
#hoe terug naar het startpunt?
    xE,yE=p[i+1].x,p[i+1].y
    lis=[];ap(lis,0,0);ap(lis,xE,yE);C.create_line(lis,arrow='first',fill='red')
    dE=sqrt(xE**2+yE**2);kE=int(degrees(atan(xE/yE)))
    if yE>0:kE=kE+180
    elif xE>0:kE=kE+360
    tocht.append(Rak(chr(98+i),dE,kE))
    rondje(p[i+1].x,p[i+1].y,p[i+1].naam,tocht[i+1].k)
    tex2.insert(INSERT,'Om van '+ p[i+1].naam+' terug te keren naar A:\nAfstand d ='+format(dE,'.2f')+', Kompas
k='+str(kE))
def rondje(a,b,te,koers):
    lis=[];ap(lis,a+0.02*dim,b-0.02*dim);C.create_text(lis,text=te)
    lis=[];r=0.03;ap(lis,a-r,b-r);ap(lis,a+r,b+r);C.create_oval(lis,fill='black')
    lis=[];ap(lis,a,b);ap(lis,a,b+0.7);C.create_line(lis,arrow='last',fill='blue',dash=[1,2])
    lis=[];hk=0
    while hk<koers:ap(lis,a+0.5*sin(radians(hk)),b+0.5*cos(radians(hk)));hk=hk+1
    C.create_line(lis,arrow='last',fill='blue',dash=[1,2])
def ap(li,x,y):X=transx(x);Y=transy(y);li.append(X);li.append(Y)
def info():messagebox.showinfo(tit+', info:',helstr)
def nieuw():C.delete(ALL);tex1.delete('1.0',END);tex2.delete('1.0',END)
#hoofdprogramma
breedte=520;hoogte=680;breedteC=500;hoogteC=500
form=Tk();form.geometry(str(breedte)+'x'+str(hoogte));form.resizable(False,False)
tit='Verkenner'
form.title(tit);hgT=hoogteC+40
C = Canvas(form, bg="lightgreen", height=hoogteC, width=breedteC);C.place(x=10,y=0)
Label(form,text='Vul in d,k:              Overzicht staproute:').place(x=5,y=hgT-30)
tex1=Text(form,bg='white',height=8,width=8,wrap=WORD);tex1.place(x=5,y=hgT)
tex1.insert(INSERT,'2,135\n2.6,48\n1.5,205\n2.7,305\n')
vsb1=Scrollbar(form,orient=VERTICAL,command=tex1.yview,width=25)
tex1['yscrollcommand']=vsb1.set;vsb1.place(x=75,y=hgT)
tex2=Text(form,bg='grey89',height=8,width=40,wrap=WORD);tex2.place(x=100,y=hgT)
vsb2=Scrollbar(form,orient=VERTICAL,command=tex2.yview,width=25)
```

```
tex2['yscrollcommand']=vsb2.set;vsb2.place(x=425,y=hgT)
B1=Button(form,text=' Info ',command=info);B1.place(x=470,y=hoogte-120,width=40)
B2=Button(form,text='Teken',command=teken);B2.place(x=470,y=hoogte-90,width=40)
B3=Button(form,text='Nieuw',command=nieuw);B3.place(x=470,y=hoogte-60,width=40)
helstr='Vanuit het startpunt A(0,0)\n'
helstr=helstr+'legt een verkenner na elkaar\nenkele rechtlijnige afstanden af:\n'
helstr=helstr+'in het linker venster vul je\nafstanden en kompaskoersen in,\n'
helstr=helstr+'gescheiden door een komma\nDe grafiek toont de afgelegde weg\n'
helstr=helstr+'en berekent afstand en kompas om\nterug te keren naar het startpunt'
form.mainloop()
```



169. Afgeknotte piramide afgeknotte piramide

We kiezen voor de eenvoud onder- en bovenvlak // met het XY-vlak. We maken een draadmodel, dwz.ribben en zijden van onder- en bovenvlak worden voorgesteld door stippellijnen.
Zijn x0,y0,z0,r0 en x1,y1,z1,r1 resp. de coördinaten en de straal van onder- en bovenvlak en n het aantal hoekpunten van onder-of bovenvlak, dan kunnen we (binnen de for-loop) de ribben tekenen met de lijsten lisrb.
Onder- en bovenvlak worden getekend met de lijsten lisgr en lisbo

```
def cl(li):C.create_line(li,fill='red',dash=[1,2])
.....
   n=int(E2.get())
   stap=2*pi/n;t=-0.1;lisgr=[];lisbo=[]
   for i in range(0,n+1):
         t=t+stap;lisrb=[]
         x=x0+r0*cos(t);y=y0+r0*sin(t);z=z0
         ap(lisgr,x,y,z);ap(lisrb,x,y,z)
         x=x1+r1*cos(t);y=y1+r1*sin(t);z=z1
         ap(lisbo,x,y,z);ap(lisrb,x,y,z)
         cl(lisrb)
   cl(lisgr);cl(lisbo)
```

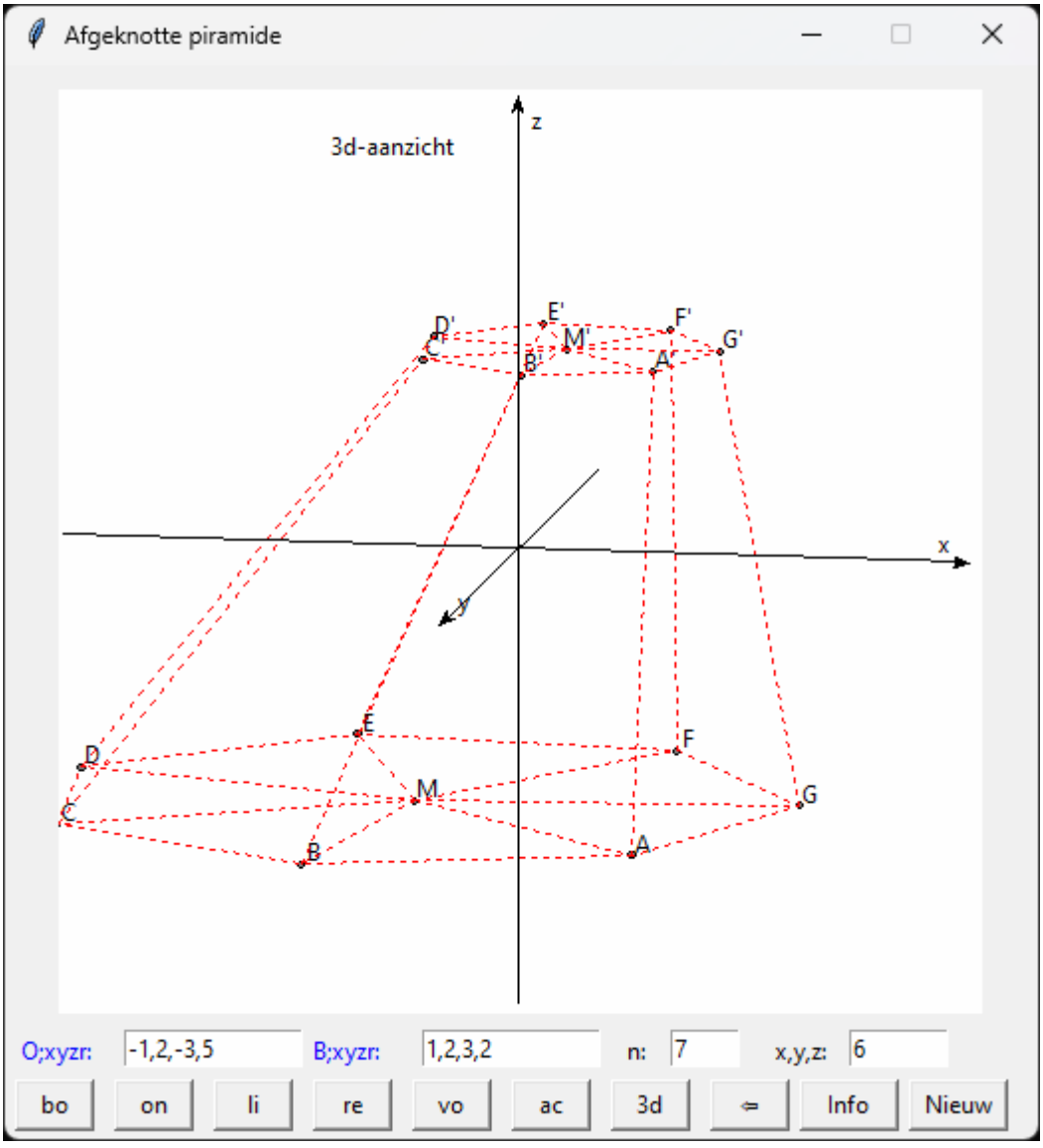
Programma:

```
# 3 dimensionale grafieken: afgeknotte piramide met grondvlak // XY
from math import *;from tkinter import *;from tkinter import messagebox
def mult(s):
   l=s.split(',');co=[]
   for i in l:co.append(float(i))
   return co
def transx(x):return (x-mi)*breedteC/(ma-mi)
def transy(y):return hoogteC-hoogteC*(y-mi)/(ma-mi)
def hoek(y,x):
   if y==0 and x>=0:c=0
   elif y==0 and x<0:c=pi
   elif x==0 and y>0:c=pi/2
   elif x==0 and y<0:c=3*pi/2
   else:
      c=atan(y/x)
      if x<0:c=c+pi
   return(c)
def omzet(x,y,z):
   c=hoek(y,x);r=sqrt(x*x+y*y);y1=sin(a+c)*r;d=hoek(y1,z)
   r1=sqrt(y1*y1+z*z);X=transx(cos(a+c)*r);Y=transy(cos(b+d)*r1)
   return [X,Y]
def asteken(x,y,z):
   lis=[];ap(lis,-x,-y,-z);ap(lis,x,y,z)
   C.create_line(lis,fill=bl,arrow='last')
def cl(li):C.create_line(li,fill='red',dash=[1,2])
def ct(lis,t):C.create_text(lis,text=t,fill=bl)
def puntP(x,y,z,i,j):
   lis=[];d=0.2;ap(lis,x+d,y+d,z+d);ct(lis,chr(65+i)+ac[j])
```

```
    lis=[];Rr=d/4;ap(lis,x-Rr,y-Rr,z-Rr);ap(lis,x+Rr,y+Rr,z+Rr);C.create_oval(lis,fill='grey')
def teken():
    global mi,ma,x0,y0,z0,a,b,x1,y1,z1,n,r0,r1,lis
    lis=[170,30];ct(lis,graftit)
    a=radians(draaixy);b=radians(draaiyz)
    x0,y0,z0,r0=mult(E0.get())
    x1,y1,z1,r1=mult(E1.get())
    n=int(E2.get());ma=int(E3.get());mi=-ma
    puntP(x1,y1,z1,12,1);puntP(x0,y0,z0,12,0)
    stap=2*pi/n;t=-0.1;lisgr=[];lisbo=[]
    for i in range(0,n+1):
        t=t+stap;lisrb=[];listrg=[];listrb=[]
        ap(listrg,x0,y0,z0);ap(listrb,x1,y1,z1)
        x=x0+r0*cos(t);y=y0+r0*sin(t);z=z0
        ap(lisgr,x,y,z);ap(lisrb,x,y,z)
        ap(listrg,x,y,z)
        if i<n:puntP(x,y,z,i,0)
        x=x1+r1*cos(t);y=y1+r1*sin(t);z=z1
        ap(lisbo,x,y,z);ap(lisrb,x,y,z)
        ap(listrb,x,y,z)
        if i<n:puntP(x,y,z,i,1)
        cl(lisrb);cl(listrg);cl(listrb)
    cl(lisgr);cl(lisbo)
    asteken(ma,0,0)
    asteken(0,ma,0);asteken(0,0,ma)
#xyz
    d=0.3;mad=ma-d
    lis=[];ap(lis,mad,d,d);ct(lis,'x')
    lis=[];ap(lis,d,mad,d);ct(lis,'y')
    lis=[];ap(lis,d,d,mad);ct(lis,'z')
def rich(grt,rig):
        global draaixy,draaiyz,graftit;graftit=grt+'aanzicht';C.delete(ALL);draaixy=rig[0];draaiyz=rig[1];teken()
def keuze(i):rich(naam[i],riget[i])
def ap(li,x,y,z):X,Y=omzet(x,y,z);li.append(X);li.append(Y)
def draai():global draaixy;draaixy=draaixy+7;C.delete(ALL);teken()
def info():messagebox.showinfo(tit+': info',infstr)
def nieuw():
    for E in (E0,E1,E2,E3):E.delete(0,len(E.get()))
    C.delete(ALL);global draaixy,draaiyz;draaixy=10;draaiyz=10
def but(i,nm,xp):te=nm[0:2];Button(form,text=te,command=lambda:
keuze(i)).place(x=5+50*i,y=hoInv2,width=40)
#hoofdprogramma
tit='Afgeknotte piramide';bl='black';gr='green';ac=['',"'"]
form=Tk();form.geometry('520x540');form.title(tit)
hoogte=540;breedte=520;hoogteC=465;breedteC=465;hoInv=hoogte-55;hoInv2=hoogte-30
form.title(tit);form.resizable(False,False)
C = Canvas(form,bg='white', height=hoogteC, width=breedteC)
C.place(x=25,y=10);draaixy=10;draaiyz=10
Label(form,text="O;xyzr:",fg='blue').place(x=5,y=hoInv)
E0=Entry(form);E0.place(x=60,y=hoInv,width=90);E0.insert(0,'-1,2,-3,5')
Label(form,text="B;xyzr:",fg='blue').place(x=152,y=hoInv)
```

```
E1=Entry(form);E1.place(x=210,y=hoInv,width=90);E1.insert(0,'1,2,3,2')
Label(form,text='n:',).place(x=310,y=hoInv)
E2=Entry(form);E2.place(x=335,y=hoInv,width=35);E2.insert(0,'7')
Label(form,text='x,y,z:').place(x=385,y=hoInv)
E3=Entry(form);E3.place(x=425,y=hoInv,width=50);E3.insert(0,'6')
naam=['boven','onder','linker','rechter','voor','achter','3d-']
riget=[[0,90],[0,-90],[-90,0],[90,0],[0,0],[180,0],[10,10]]
for i in range(0,len(naam)):but(i,naam[i],5+i*50)
```

```
Button(form,text='⇦',command=draai,width=1).place(x=355,y=hoInv2,width=40)
Button(form,text='Info',command=info).place(x=400,y=hoInv2,width=50)
Button(form,text='Nieuw',command=nieuw).place(x=455,y=hoInv2,width=50)
infstr='bo:bovenaanzicht\non:onderaanzicht\nli:linkeraanzicht\n'
infstr=infstr+'re:rechteraanzicht\nvo:vooraanzicht\nac:achteraanzicht\n'
infstr=infstr+'3d:3d-aanzicht\nO:x,y,z,r:co.midden\nen straal ondervlak\n'
infstr=infstr+'B:x,y,z,r:co.midden\nen straal bovenvlak\nn: # opstaande ribben\nxyz: maxim +/- xyz'
form.mainloop()
```



170. Doorsnede piramide met vlak. doorsnede_piramide_vlak

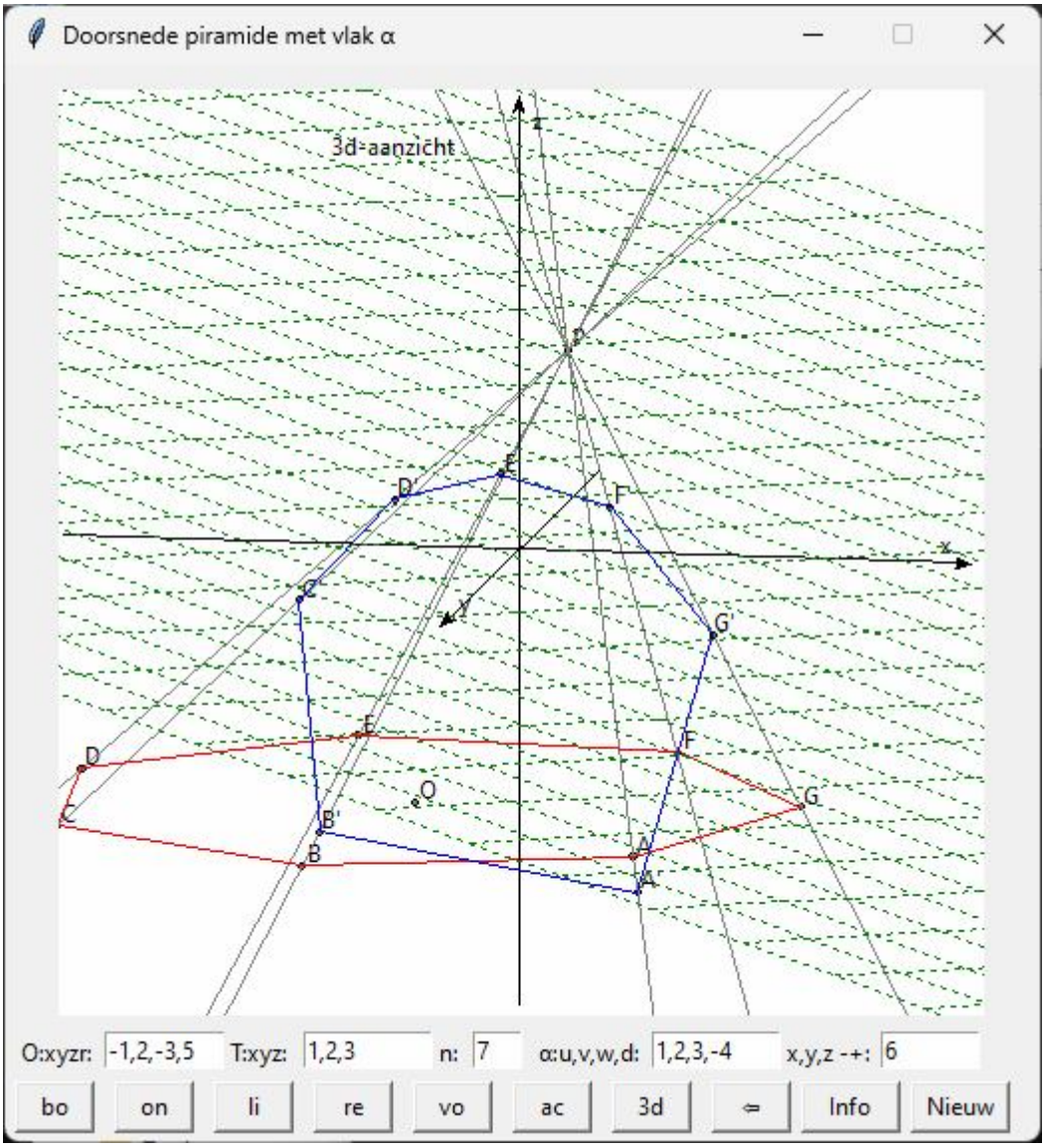
We nemen een bijzondere soort piramide: waarvan de top P wordt verbonden met de hoekpunten van een regelmatige veelhoek die // is met het xy-vlak. Van deze veelhoek moet het middelpunt x,y,z en de straal r ingevoerd worden. Deze veelhoek ABC.... wordt getekend in het rood, de opstaande ribben in het grijs. Van een vlak α: ux+vy+wz+d=0 moeten u,v,w,d gegeven worden. De doorsnede A'B'C'... van vlak en kegel wordt in het blauw getekend. Ook van deze grafiek kunnen alle aanzichten getoond worden.

Programma:

```
# 3 dimensionale grafieken: doorsnede piramide met vlak ux+vy+wz+d=0
from math import *;from tkinter import *;from tkinter import messagebox
def mult(s):
    l=s.split(',');co=[]
    for i in l:co.append(float(i))
    return co
def transx(x):return (x-mi)*breedteC/(ma-mi)
def transy(y):return hoogteC-hoogteC*(y-mi)/(ma-mi)
def hoek(y,x):
    if y==0 and x>=0:c=0
    elif y==0 and x<0:c=pi
    elif x==0 and y>0:c=pi/2
    elif x==0 and y<0:c=3*pi/2
    else:
        c=atan(y/x)
        if x<0:c=c+pi
    return(c)
def omzet(x,y,z):
    c=hoek(y,x);r=sqrt(x*x+y*y);y1=sin(a+c)*r;d=hoek(y1,z)
    r1=sqrt(y1*y1+z*z);X=transx(cos(a+c)*r);Y=transy(cos(b+d)*r1)
    return [X,Y]
def asteken(x,y,z):
    lis=[];ap(lis,-x,-y,-z);ap(lis,x,y,z)
    C.create_line(lis,fill=bl,arrow='last')
def cl(li,kl):C.create_line(li,fill=kl)
def ct(lis,t):C.create_text(lis,text=t,fill=bl)
def puntP(x,y,z,i,j):
    lis=[];d=0.2;ap(lis,x+d,y+d,z+d);ct(lis,chr(65+i)+ac[j])
    lis=[];Rr=d/4;lis=[];ap(lis,x-Rr,y-Rr,z-Rr);ap(lis,x+Rr,y+Rr,z+Rr);C.create_oval(lis,fill='grey')
def teken():
    global mi,ma,x0,y0,z0,a,b,x1,y1,z1,n,r0,r1,lis
    u,v,w,d=mult(E3.get())
    if u==0 and v==0 and w==0:messagebox.showinfo('Fout','u,v en w niet tegelijkertijd 0 stellen');return
    lis=[170,30];ct(lis,graftit)
    a=radians(draaixy);b=radians(draaiyz)
    x0,y0,z0,r0=mult(E0.get())
    x1,y1,z1=mult(E1.get())
    n=int(E2.get());ma=int(E4.get());mi=-ma
    puntP(x1,y1,z1,15,0);puntP(x0,y0,z0,14,0)
    stap=2*pi/n;t=-0.1;lisgr=[];lisds=[]
    for i in range(0,n+1):
```

```
            t=t+stap;lisrb=[]
            x=x0+r0*cos(t);y=y0+r0*sin(t);z=z0
            k=10;ap(lisrb,k*x+(1-k)*x1,k*y+(1-k)*y1,k*z+(1-k)*z1)
            k=-10;ap(lisrb,k*x+(1-k)*x1,k*y+(1-k)*y1,k*z+(1-k)*z1)
            ap(lisgr,x,y,z)
            if i<n:puntP(x,y,z,i,0)
            cl(lisrb,'grey') # tekenen ribben
            m0=u*x0+v*y0+w*z0+u*r0*cos(t)+v*r0*sin(t)
            m1=u*x1+v*y1+w*z1
            if m0!=m1:
                    k=(m0+d)/(m0-m1)
                    x=(x0+r0*cos(t))*(1-k)+x1*k
                    y=(y0+r0*sin(t))*(1-k)+y1*k
                    z=z0*(1-k)+z1*k
                    ap(lisds,x,y,z)
                    if i<n:puntP(x,y,z,i,1)
    cl(lisgr,'red');cl(lisds,'blue') # tekenen grondvlak,doorsnede met vlak
    asteken(ma,0,0);asteken(0,ma,0);asteken(0,0,ma)
#vlak ux+vy+wz+d=0 ( // z-as)
    stap=ma/15
    if u!=0:
       z=mi
       while z<ma:
          z=z+stap;lis=[];y=mi
          while y<ma:
             y=y+stap;x=(-d-v*y-w*z)/u;ap(lis,x,y,z)
          lijn(lis)
       y=mi
       while y<ma:
          y=y+stap;lis=[];z=mi
          while z<ma:
             z=z+stap;x=(-d-v*y-w*z)/u;ap(lis,x,y,z)
          lijn(lis)
    elif v!=0:
       z=mi
       while z<ma:
          z=z+stap;lis=[];x=mi
          while x<ma:x=x+stap;y=(-d-w*z)/v;ap(lis,x,y,z)
          lijn(lis)
       x=mi
       while x<ma:
          x=x+stap;lis=[];z=mi
          while z<ma:z=z+stap;y=(-d-w*z)/v;ap(lis,x,y,z)
          lijn(lis)
    elif w!=0:
       x=mi
       while x<ma:
          x=x+stap;lis=[];y=mi
          while y<ma:y=y+stap;z=-d/w;ap(lis,x,y,z)
          lijn(lis)
       y=mi
```

```
        while y<ma:
            y=y+stap;lis=[];x=mi
            while x<ma:x=x+stap;z=-d/w;ap(lis,x,y,z)
            lijn(lis)
#xyz
    d=0.3;mad=ma-d
    lis=[];ap(lis,mad,d,d);ct(lis,'x')
    lis=[];ap(lis,d,mad,d);ct(lis,'y')
    lis=[];ap(lis,d,d,mad);ct(lis,'z')
def lijn(lis):C.create_line(lis,fill=gr,dash=[1,2])
def rich(grt,rig):
            global draaixy,draaiyz,graftit;graftit=grt+'aanzicht';C.delete(ALL);draaixy=rig[0];draaiyz=rig[1];teken()
def keuze(i):rich(naam[i],riget[i])
def ap(li,x,y,z):X,Y=omzet(x,y,z);li.append(X);li.append(Y)
def draai():global draaixy;draaixy=draaixy+7;C.delete(ALL);teken()
def info():messagebox.showinfo(tit+': info',infstr)
def nieuw():
    for E in (E0,E1,E2,E3,E4):E.delete(0,len(E.get()))
    C.delete(ALL);global draaixy,draaiyz;draaixy=10;draaiyz=10
def but(i,nm,xp):te=nm[0:2];Button(form,text=te,command=lambda:
keuze(i)).place(x=5+50*i,y=hoInv2,width=40)
#hoofdprogramma
tit='Doorsnede piramide met vlak α';bl='black';gr='green';ac=['',"""]
form=Tk();form.geometry('520x540');form.title(tit)
hoogte=540;breedte=520;hoogteC=465;breedteC=465;hoInv=hoogte-55;hoInv2=hoogte-30
form.title(tit);form.resizable(False,False)
C = Canvas(form,bg='white', height=hoogteC, width=breedteC)
C.place(x=25,y=10);draaixy=10;draaiyz=10
Label(form,text="O:xyzr:").place(x=5,y=hoInv)
E0=Entry(form);E0.place(x=50,y=hoInv,width=65);E0.insert(0,'-1,2,-3,5')
Label(form,text="T:xyz:").place(x=110,y=hoInv)
E1=Entry(form);E1.place(x=150,y=hoInv,width=65);E1.insert(0,'1,2,3')
Label(form,text='n:').place(x=215,y=hoInv)
E2=Entry(form);E2.place(x=235,y=hoInv,width=25);E2.insert(0,'7')
Label(form,text='α:u,v,w,d:').place(x=265,y=hoInv)
E3=Entry(form);E3.place(x=325,y=hoInv,width=65);E3.insert(0,'1,2,3,-4')
Label(form,text='x,y,z -+:').place(x=390,y=hoInv)
E4=Entry(form);E4.place(x=440,y=hoInv,width=50);E4.insert(0,'6')
naam=['boven','onder','linker','rechter','voor','achter','3d-']
riget=[[0,90],[0,-90],[-90,0],[90,0],[0,0],[180,0],[10,10]]
for i in range(0,len(naam)):but(i,naam[i],5+i*50)
Button(form,text='⇦',command=draai,width=1).place(x=355,y=hoInv2,width=40)
Button(form,text='Info',command=info).place(x=400,y=hoInv2,width=50)
Button(form,text='Nieuw',command=nieuw).place(x=455,y=hoInv2,width=50)
infstr='bo:bovenaanzicht\non:onderaanzicht\nli:linkeraanzicht\n'
infstr=infstr+'re:rechteraanzicht\nvo:vooraanzicht\nac:achteraanzicht\n'
infstr=infstr+'3d:3d-aanzicht\nO:x,y,z,r:co.midden\nen straal ondervlak\n'
infstr=infstr+' α:ax+by+cz+d=0:\nvergelijking snijvlak\nn: # opstaande ribben\nxyz: maxim +/- xyz'
form.mainloop()
```



171. Doorsnede van vlakken in de ruimte doorsnede_vlakken

3 vlakken α,β en γ kunnen elkaar snijden in een punt P dat meteen ook het snijpunt is van de snijlijnen A,B en C van deze 3 vlakken. Maar 2 (of 3) vlakken kunnen ook evenwijdig zijn of samenvallen

```
# Vergelijkingen van rechte en vlak in de ruimte
from math import *;from tkinter import *;from tkinter import messagebox;from copy import deepcopy
def mult(s):
    l=s.split(',');co=[]
    for i in l:co.append(float(i))
    return co
def transx(x):return (x-xmi)*widC/(xma-xmi)
def transy(y):return heigC-heigC*(y-ymi)/(yma-ymi)
def det(mat):
    le=len(mat)
    if le==2:d=mat[0][0]*mat[1][1]-mat[0][1]*mat[1][0]
    else:
```

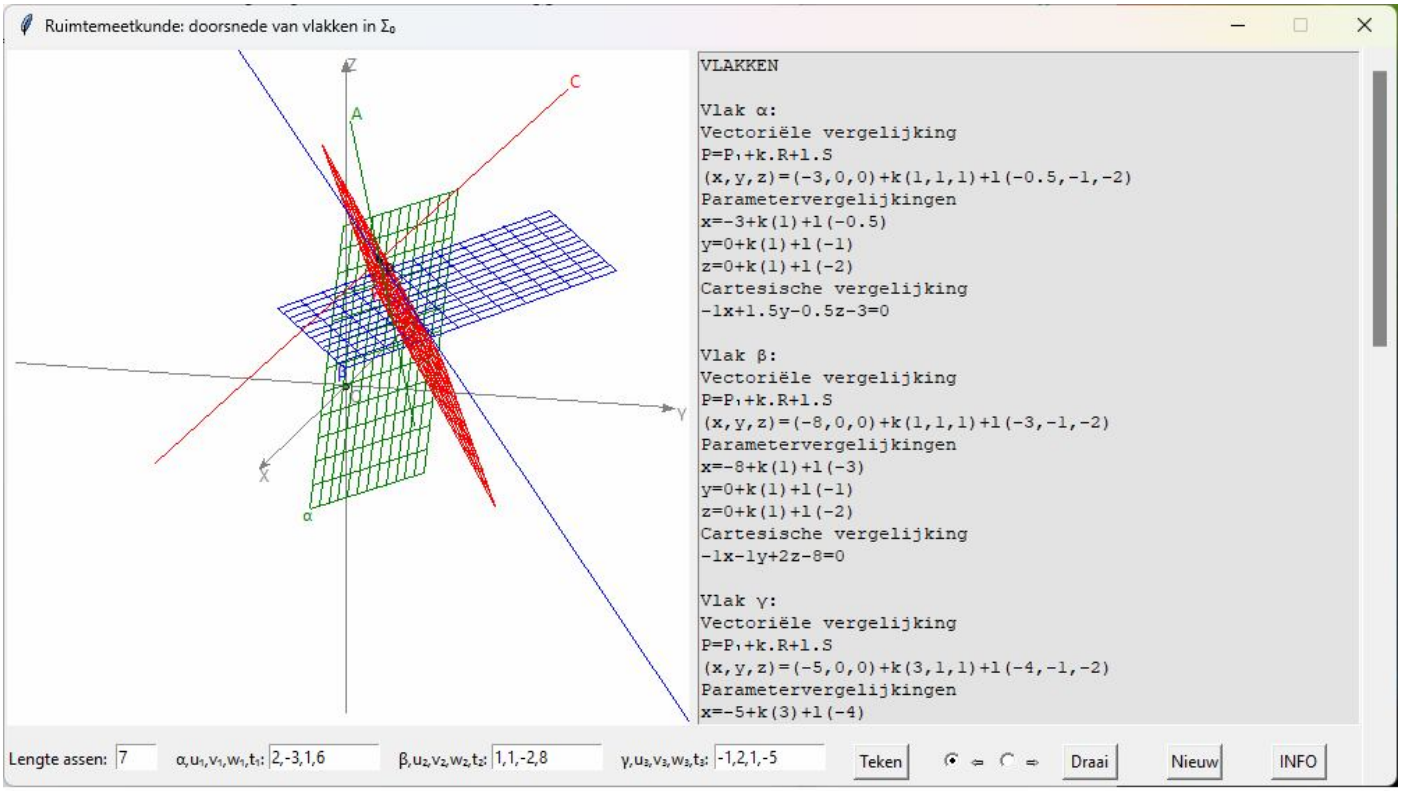
```
        d=0
        for i in range(0,le):
            s=[]
            for k in range(0,le):
                if i!=k:s.append(mat[k][1:])
            d=d+mat[i][0]*(-1)**i*det(s)
    return d
def ad(lin):tex.insert(INSERT,lin+'\n')
def vg(x):
    if x==int(x):x=int(x)
    return str(round(x,3))
def sg(a):
    if a==int(a):a=int(a)
    if a>0 or abs(a)<0.0001:t='+'
    else:t=''
    return t+str(round(a,3))
def recfun(x):return b*(x-x1)/a+y1
def appen(x,y,z):X,Y=omzet(x,y,z);lis.append(X);lis.append(Y)
def punt(x,y,z,nm):
    col='black';X,Y=omzet(x,y,z);C.create_oval(X-2,Y-2,X+2,Y+2,fill=col)
    C.create_text(X+7,Y+7,text=nm,fill=col)
def stelsel(coefvgl):
    global dst;acop=deepcopy(coefvgl);m=[];B=[];n=3
    for i in range(0,n):B.append(-coefvgl[i][n])
    for i in range(0,n):acop[i].pop()
    for k in range(0,n):m.append([])
    for k in range(0,n):
        m[k]=deepcopy(acop)
        for j in range(0,n):
            m[k][j].pop(k)
            m[k][j].insert(k,B[j])
    dst=det(acop);a=det(m[0]);b=det(m[1]);c=det(m[2])
    if dst==0:ad('Vlakken snijden niet in één punt');return(0,0,0)
    else: return(a/dst,b/dst,c/dst)
def stelsel2x2(u1,u2,u3,v1,v2,v3):
    d=u1*v2-u2*v1;da=u3*v2-u2*v3;db=u1*v3-u3*v1
    if d==0:return[0,0]
    return [da/d,db/d]
# definities i.v.m. GRAFIEKEN
def hoek(x,y):
    if x==0 and y>=0:c=0
    elif x==0 and y<0:c=pi
    elif y==0 and x>0:c=pi/2
    elif y==0 and x<0:c=3*pi/2
    else:
        c=atan(x/y)
        if y<0:c=c+pi
    return(c)
def omzet(x,y,z):
    c=hoek(x,y)
    r=sqrt(x*x+y*y);x1=sin(ah+c)*r
```

```
    d=hoek(x1,z)
    r1=sqrt(x1*x1+z*z)
    X=transx(cos(ah+c)*r);Y=transy(cos(bh+d)*r1)
    return [X,Y]
def asteken(nm,x,y,z):
    lis=[]
    X,Y=omzet(-x,-y,-z);lis.append(X);lis.append(Y)
    X,Y=omzet(x,y,z);lis.append(X);lis.append(Y)
    line = C.create_line(lis,fill='grey',arrow='last')
    C.create_text(X+4,Y+4,text=nm,fill='grey')  # x,y,z bij assen schrijven
def grafinit():
    global axlen,xmi,ymi,zmi,xma,yma,zma,ah,bh,u1,v1,w1,t1,u2,v2,w2,t2,u3,v3,w3,t3,lis

lis=[];axlen=float(E1.get());u1,v1,w1,t1=mult(E2.get());u2,v2,w2,t2=mult(E3.get());u3,v3,w3,t3=mult(E4.get())
    xma,yma,zma=axlen,axlen,axlen;xmi,ymi,zmi=-axlen,-axlen,-axlen
    ah=radians(drxy);bh=radians(dryz)
    C.delete(ALL);tex.delete('1.0',END)
#assen + oorsprong
    asteken('X',axlen,0,0);asteken('Y',0,axlen,0);asteken('Z',0,0,axlen)
    X,Y=omzet(0,0,0);
    C.create_oval(X-2,Y-2,X+2,Y+2,fill='grey') # punt tekenen
    C.create_text(X+7,Y+7,text='O',fill='grey') # naam bijzetten
def grafrechte(x1,y1,z1,a,b,c,col,te):
    global lis
# teken grafiek rechte
    lis=[];t=-axlen+1;appen(x1+t*a,y1+t*b,z1+t*c)
    t=axlen-1;appen(x1+t*a,y1+t*b,z1+t*c)
    tek(lis,col)
    lis=[];appen(x1+t*a+0.2,y1+t*b+0.2,z1+t*c+0.2);C.create_text(lis,text=te,fill=col)
# tekst rechte
    ad('Vectoriële vergelijking');ad('P=P\u2081+k.R');ad('(x,y,z)=('+vg(x1)+',
'+vg(y1)+','+vg(z1)+')+k('+vg(a)+','+vg(b)+','+vg(c)+')')
    ad('Parametervergelijkingen');ad('x='+vg(x1)+'+k('+vg(a)+')');
ad('y='+vg(y1)+'+k('+vg(b)+')');ad('z='+vg(z1)+'+k('+vg(c)+')');
    ad('Cartesische vergelijking')
    cv1='';cv2=''
    if a==0:cv2=cv2+' x='+vg(x1)
    else:cv1=cv1+'(x'+sg(-x1)+')/'+vg(a)+'='
    if b==0:cv2=cv2+' y='+vg(y1)
    else:cv1=cv1+'(y'+sg(-y1)+')/'+vg(b)+'='
    if c==0:cv2=cv2+' z='+vg(z1)
    else:cv1=cv1+'(z'+sg(-z1)+')/'+vg(c)
    ad(cv1+'...   '+cv2)
    return
def leng(u,v,w):return sqrt(u*u+v*v+w*w)
def vecricvec(u,v,w,t):
    if u!=0: x1,y1,z1,a,b,c,d,e,f=-t/u,0,0,(-v-w)/u,1,1,(v+2*w)/u,-1,-2
    elif v!=0: x1,y1,z1,a,b,c,d,e,f=0,-t/v,0,1,(-u-w)/v,1,-1,(u+2*w)/v,-2
    elif w!=0:x1,y1,z1,a,b,c,d,e,f=0,0,-t/w,1,1,(-u-v)/w,-1,-2,(u+2*v)/w
    return x1,y1,z1,a,b,c,d,e,f
def teken():
```

```
    global x1,y1,z1,a,b,c,d,e,f;grafinit()
    if u1==v1==w1==0 or  u2==v2==w2==0 or  u3==v3==w3==0: messagebox.showinfo('FOUT:','0,0,0,... geeft
geen vlak');return
    ad('VLAKKEN')
    x1,y1,z1,a,b,c,d,e,f=vecricvec(u1,v1,w1,t1)
    grafvlak(x1,y1,z1,a,b,c,d,e,f,'green','α')
    x1,y1,z1,a,b,c,d,e,f=vecricvec(u2,v2,w2,t2)
    grafvlak(x1,y1,z1,a,b,c,d,e,f,'blue','β')
    x1,y1,z1,a,b,c,d,e,f=vecricvec(u3,v3,w3,t3)
    grafvlak(x1,y1,z1,a,b,c,d,e,f,'red','γ')
    vgl=[[u1,v1,w1,t1],[u2,v2,w2,t2],[u3,v3,w3,t3]]
    x,y,z=stelsel(vgl)
    if dst!=0:punt(x,y,z,'P');ad('\nSnijpunt P: ('+vg(x)+','+vg(y)+','+vg(z)+')')
    ad('\nSNIJLIJNEN:')
    snijlijn(u1,v1,w1,t1,u2,v2,w2,t2,'red','C');snijlijn(u1,v1,w1,t1,u3,v3,w3,t3,'blue','B');snijlijn(u3,v3,w3,t3,
u2,v2,w2,t2,'green','A');
def snijlijn(u1,v1,w1,t1,u2,v2,w2,t2,col,te):
    if te=='A':tvl='β,γ'
    if te=='B':tvl='α,γ'
    if te=='C':tvl='α,β'
    ad('\nSnijlijn '+te+' van '+tvl+' :\n')
    if (u1*v2==v1*u2 and u1*w2==w1*u2) or ((u1==u2==v1==v2==0) or(u1==u2==w1==w2==0) or
(w1==w2==v1==v2==0)):
        if u1*t2==t1*u2 and v1*t2==v2*t1 and w1*t2==w2*t1 : ad(tvl+' vallen samen')
        else:ad(tvl+' zijn evenwijdig')
        return
# u1x+v1y+w1z+t1=0
# u2x+v2y+w2z+t2=0
# x,y,z=co steunvector , a,b,c=co richtingsvector
    elif u1==v1==0:
            z=-t1/w1;c=0
            if u2!=0:y=1;x=(-v2-t2-w2*z)/u2;b=1;a=-v2/u2
            else:x=1;y=(-u2-t2-w2*z)/v2;a=1;b=0
            print(x,y,z,a,b,c)
    elif u1==w1==0:
            y=-t1/v1;b=0
            if u2!=0:z=1;x=(-w1-t2-v2*y)/u2;c=1;a=-w2/u2
            else:x=1;z=(-u2-t2-v2*y)/w2;a=1;c=0
    elif v1==w1==0:
            x=-t1/u1;a=0
            if w2!=0:y=1;z=(-v2-t2-u2*x)/w2;b=1;c=-v2/w2
            else:z=1;y=(-w2-t2-u2*x)/v2;c=1;b=0
    elif u2==v2==0:
        z=-t2/w2;c=0
        if u1!=0:y=1;x=(-v1-t1-w1*z)/u1;b=1;a=-v1/u1
        else:x=1;y=(-u1-t1-w1*z)/v1;a=1;b=0
    elif u2==w2==0:
        y=-t2/v2;b=0
        if u1!=0:z=1;x=(-w1-t1-v1*y)/u1;c=1;a=-w1/u1
        else:x=1;z=(-u1-t1-v1*y)/w1;a=1;c=0
    elif v2==w2==0:
```

```
        x=-t2/u2;a=0
        if w1!=0:y=1;z=(-v1-t1-u1*x)/w1;b=1;c=-v1/w1
        else:z=1;y=(-w1-t1-u1*x)/v1;c=1;b=0
    else: # willekeurige stand vlakken
        z=1;x,y=stelsel2x2(u1,v1,-t1-w1,u2,v2,-t2-w2)
        c=1;a,b=stelsel2x2(u1,v1,-w1,u2,v2,-w2)
    grafrechte(x,y,z,a,b,c,col,te)
def grafvlak(x1,y1,z1,a,b,c,d,e,f,kl,te):
# grafiek van het vlak
# constructie van 2 richtingsvectoren met lengte 1 die loodrecht op elkaar staan
    global lis,vlak;vlak=2
    noem=((a+d)*d+(b+e)*e+(c+f)*f)
    if noem==0:
        d=2*d;e=2*e;f=2*f
        noem=((a+d)*d+(b+e)*e+(c+f)*f)
    n=((a+d)*a+(b+e)*b+(c+f)*c)/noem
    x2,y2,z2=a+d,b+e,c+f;x3,y3,z3=-a+n*d,-b+n*e,-c+n*f
    leng2=leng(x2,y2,z2);leng3=leng(x3,y3,z3)
    x2,y2,z2=x2/leng2,y2/leng2,z2/leng2
    x3,y3,z3=x3/leng3,y3/leng3,z3/leng3
    tma=int(axlen/2);stap=0.5;t=-tma-stap
#niveaulijnen x2,...
    while t<tma:
        lis=[];t=t+stap
        appen(x1-tma*x2+t*x3,y1-tma*y2+t*y3,z1-tma*z2+t*z3)
        appen(x1+tma*x2+t*x3,y1+tma*y2+t*y3,z1+tma*z2+t*z3)
        tek(lis,kl)
    t=-tma-stap
#niveaulijnen x3,...
    while t<tma:
      lis=[];t=t+stap
      appen(x1-tma*x3+t*x2,y1-tma*y3+t*y2,z1-tma*z3+t*z2)
      appen(x1+tma*x3+t*x2,y1+tma*y3+t*y2,z1+tma*z3+t*z2)
      tek(lis,kl);
    t=tma+.1;lis=[];appen(x1+t*(x2+x3),y1+t*(y2+y3),z1+t*(z2+z3));C.create_text(lis,text=te,fill=kl)
# tekst
    ad('\nVlak '+te+':\nVectoriële vergelijking');ad('P=P\u2081+k.R+l.S')
    ad('(x,y,z)=('+vg(x1)+','+vg(y1)+','+vg(z1)+')+k('+vg(a)+','+vg(b)+','+vg(c)+') +l('+vg(d)+','+vg(e)+','+vg(f)+')')
    ad('Parametervergelijkingen');ad('x='+vg(x1)+'+k('+vg(a)+')+l('+vg(d)+')')
    ad('y='+vg(y1)+'+k('+vg(b)+')+l('+vg(e)+')');ad('z='+vg(z1)+'+k('+vg(c)+')+l('+vg(f)+')');
    ad('Cartesische vergelijking');u=b*f-c*e;v=-a*f+c*d;w=a*e-b*d;t=-det([[x1,y1,z1],[a,b,c],[d,e,f]])
    ad(sg(u)+'x'+sg(v)+'y'+sg(w)+'z'+sg(t)+'=0')
    return
def draai():
    global drxy;draaihoek=10
    if rich==0:drxy=drxy+draaihoek
    if rich==1:drxy=drxy-draaihoek
    C.delete(ALL);teken()
    return
def appen(x,y,z):X,Y=omzet(x,y,z);lis.append(X);lis.append(Y)
def tek(li,kl):line = C.create_line(li,fill=kl)
```

```
def tekvec(li):line = C.create_line(li,arrow='last',fill='red')
def tekstip(li):line = C.create_line(li,dash=[1,2],fill='green')
def tekstipvec(li):line = C.create_line(li,dash=[1,2],fill='green',arrow='last')
def selec():global rich;rich=var.get()
def info():
   helstr='Doorsnede van vlakken\n'
   helstr=helstr+' Van 3 vlakken α β en γ\n'
   helstr=helstr+'wordt de vergelijking gegeven\n'
   helstr=helstr+'in de vorm: ux+vy+wz+t=0\n'
   helstr=helstr+"Met 'Teken' krijg je de snij-\n"
   helstr=helstr+'lijnen A,B en C en het snijpunt P\n'
   helstr=helstr+'Snijlijn blauw,rood = groen...'
   h=messagebox.showinfo('INFO:',helstr)
def nieuw():
   for en in [E1,E2,E3,E4]:en.delete(0,len(en.get()))
   C.delete(ALL);tex.delete('1.0',END);drxy=15
def but(xp,yp,t,co):Button(form,text=t,command=co).place(x=xp,y=yp,width=40)
#hoofdprogramma
wid=490;heig=530;heigC=heig-45;widC=wid;hoInv=heig-30;drxy=15;dryz=15
form=Tk();form.geometry(str(1000)+'x'+str(heig));form.resizable(False,False)
form.title('Ruimtemeetkunde: doorsnede van vlakken in Σ\u2080')
C = Canvas(form, bg="white", height=heigC, width=widC);C.place(x=0,y=0)
L1=Label(form,text='Lengte assen:',);L1.place(x=0,y=hoInv)
E1=Entry(form);E1.place(x=80,y=hoInv,width=30);E1.insert(0,'7')
L2=Label(form,text='α, u1,v1,w1,t1:');L2.place(x=110,y=hoInv)
E2=Entry(form);E2.place(x=180,y=hoInv,width=60);E2.insert(0,'2,-3,1,6')
L3=Label(form,text='β, u2,v2,w2,t2:',);L3.place(x=240,y=hoInv)
E3=Entry(form);E3.place(x=310,y=hoInv,width=60);E3.insert(0,'1,1,-2,8')
L4=Label(form,text='γ, u3,v3,w3,t3:',);L4.place(x=370,y=hoInv)
E4=Entry(form);E4.place(x=440,y=hoInv,width=60);E4.insert(0,'-1,2,1,-5')
tex=Text(form,bg='grey89',height=30,width=59,wrap=WORD);tex.place(x=wid+8,y=3)
vsb=Scrollbar(form,orient=VERTICAL,command=tex.yview,width=25)
tex['yscrollcommand']=vsb.set;vsb.pack(side=RIGHT,fill='y')
rich=0;var = IntVar();pijl=['\u21E6','\u21E8'] #unicode pijlen
for i in range(0,2):
   rb=Radiobutton(form,text=pijl[i],variable=var,value=i,command=selec)
   rb.place(x=650+i*40,y=hoInv)
but(590,hoInv,'Teken',teken)
but(730,hoInv,'Draai',draai);but(790,hoInv,'Nieuw',nieuw)
but(910,hoInv,'INFO',info)
form.mainloop()
```



172. Binomium Binomium1

In deze programma's wordt $(a\cdot x+b\cdot y)^n$ berekend voor $a,b\in Z$ en $n\in N_0$

Formule:

$$(a\cdot x+b\cdot y)^n = C_n^0\cdot a^n\cdot x^n\cdot b^0\cdot y^0 + C_n^1\cdot a^{n-1}\cdot x^{n-1}\cdot b^1\cdot y^1 + ... + C_n^i\cdot a^{n-i}\cdot x^{n-i}\cdot b^i\cdot y^i + ...$$

$$= \sum_{i=0}^{n} C_n^i\cdot a^{n-i}\cdot x^{n-i}\cdot b^i\cdot y^i$$

De coëfficiënt van de algemene term is $C_n^i\cdot a^{n-i}\cdot b^i$. Het tweede deel is $x^{n-i}\cdot y^i$

Een eerste programma werkt in de 'shell' :

Programma:

```
# Binomium
def u(i):return numli[i]
def mult(s):
    l=s.split(',');co=[]
    for i in l:co.append(int(i))
    return co
def C(n,p):
    c=1
    for i in range(0,p):c=c*(n-i)//(i+1)
    return c
def vc(a):
    if a<0:return(str(a))
    else:return('+'+str(a))
def ex(g):
    if g==0:return u(0)
```

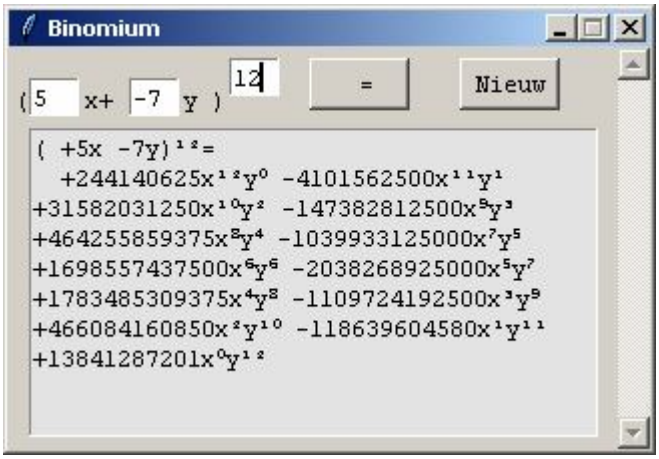
```
    else:
        exst='';d=g
        while d>0:
            g=d;d=g//10;q=g%10;exst=u(q)+exst
        return exst
numli=['\u2070','\u00b9','\u00b2','\u00b3','\u2074','\u2075','\u2076','\u2077','\u2078','\u2079']; ns='\u207f'
a,b,n=mult(input('Uitwerken (ax+by)'+ns+' : geef a,b,n:'))
antw='('+vc(a)+'x'+vc(b)+'y)'+ex(n)+'= '
for i in range(0,n+1):
    antw=antw+vc(C(n,i)*a**(n-i)*b**i)+'x'+ex(n-i)+'y'+ex(i)
print(antw)

>>>
Uitwerken (ax+by)ⁿ : geef a,b,n:3,-4,5
(+3x-4y)⁵= +243x⁵y ⁰-1620x⁴y¹+4320x³y²-5760x²y³+3840x¹y⁴-1024x⁰y⁵
>>>
numli is een lijst van de superscripts van 0 tot 9
De functie ex(g) zet een willekeurig geheel getal om naar zijn 'superscript' -equivalent
De functie vc(a) maakt een stringversie van a maar als a>0, met toevoeging van '+'
```

173. Tkinterversie binomium Binomium2

Programma

```
# Binomium
from math import sqrt;from tkinter import *
def u(i):return numli[i]
def mult(s):
    l=s.split(',');co=[]
    for i in l:co.append(int(i))
    return co
def ad(lin):tex.insert(INSERT,lin+' ')
def C(n,p):
    c=1
    for i in range(0,p):c=c*(n-i)//(i+1)
    return c
def vc(a):
    if a<0:return(' '+str(a))
    else:return(' +'+str(a))
def ex(g):
    if g==0:return u(0)
    else:
        exst='';d=g
        while d>0:
            g=d;d=g//10;q=g%10;exst=u(q)+exst
    return exst
def bereken():
    tex.delete('1.0',END)
    a=int(E1.get());b=int(E2.get());n=int(E3.get())
    antw='('+vc(a)+'x'+vc(b)+'y)'+ex(n)+'=\n '
    for i in range(0,n+1):
```

```
            antw=antw+vc(C(n,i)*a**(n-i)*b**i)+'x'+ex(n-i)+'y'+ex(i)
    ad(antw)
def nieuw():
    for E in (E1,E2,E3):E.delete(0,len(E.get()))
    tex.delete('1.0',END)
#hoofdprogramma
numli=['\u2070','\u00b9','\u00b2','\u00b3','\u2074','\u2075','\u2076','\u2077','\u2078','\u2079'];ns='\u207f'
tit='Binomium';wid=320;heig=200
form=Tk();form.geometry(str(wid)+'x'+str(heig));form.title(tit)
form.resizable(False,False);form.option_add('*Font',('Courier','9'))
L1=Label(form,text='(   .x+    .y )  ',);L1.place(x=0,y=15)
E1=Entry(form,width=3);E1.place(x=10,y=15);E1.insert(0,'2')
E2=Entry(form,width=3);E2.place(x=60,y=15);E2.insert(0,'3')
E3=Entry(form,width=3);E3.place(x=110,y=5);E3.insert(0,'4')
tex=Text(form,bg='grey89',height=10,width=40,wrap=WORD);tex.place(x=10,y=40)
vsb=Scrollbar(form,orient=VERTICAL,command=tex.yview,width=20)
tex['yscrollcommand']=vsb.set;vsb.pack(side=RIGHT,fill='y')
B1=Button(form,text=' =',command=bereken);B1.place(x=150,y=5,width=50)
B2=Button(form,text=' Nieuw ',command=nieuw);B2.place(x=225,y=5,width=50)
form.mainloop()
```



174. Kansbomen kansboom

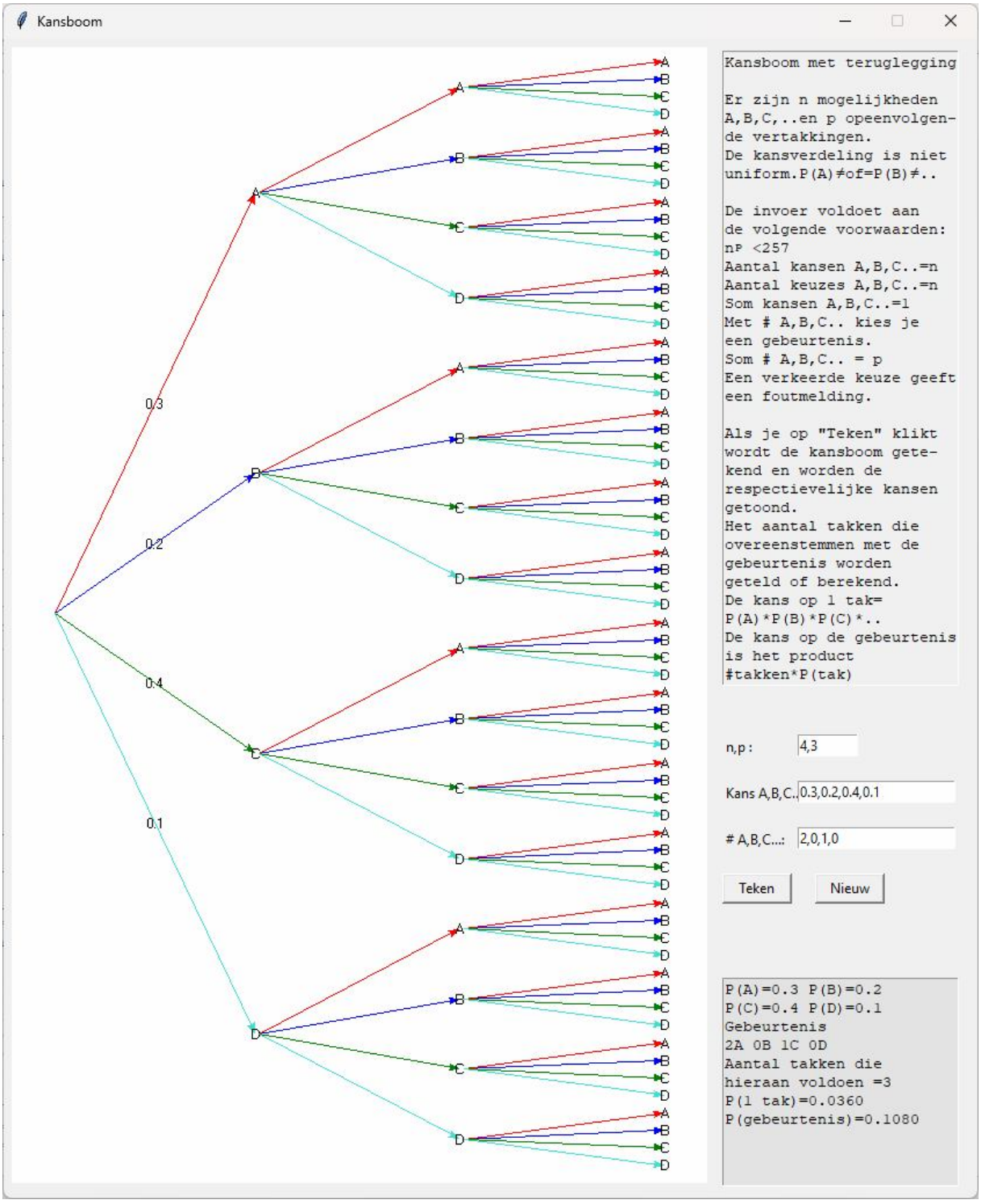
We hebben de keuze tussen n ≠ mogelijkheden (voorgesteld door kleuren of letters A,B C,..) , waarvan de kansen niet uniform verdeeld zijn. In ieder geval is P(A)+P(B)+P(C)+..=1
We moeten p keer kiezen (met teruglegging zodat de kansen dezelfde blijven)
Wat is dan de keuze op een bepaalde keuze: bvb. als n=4 en p=3, wat is de kans om 2 x A en 1 x C te kiezen. Om deze kans te berekenen, moeten we een kansboom tekenen, waarvan het aantal eindtakken
= n·n.. = n^p = 4^3 = 64. Het programma tekent deze boom en berekent de kans P(2 x A, 1 x C)
Het programma kan de kans berekenen voor elke waarde van n en p. Maar omwille van een duidelijke tekening, beperken we n en p zodat n^p<257 en n<10. Maar dit kan je zonder moeite veranderen.

Programma

```
# kansboom
from math import *;from tkinter import *;from tkinter import messagebox
class Pijl:
    def __init__(self,naam,kleur,kans):self.naam=naam;self.kleur = kleur;self.kans = kans
```

```
def mult(s):
    l=s.split(',');co=[]
    for i in l:co.append(float(i))
    return co
def Comb(n,p):
    c=1
    for i in range(0,p):c=c*(n-i)//(i+1)
    return c
def transx(x):return (x-xmi)*breedteC/(xma-xmi)
def transy(y):return hoogteC-hoogteC*(y-ymi)/(yma-ymi)
def ap(li,x,y):X=transx(x);Y=transy(y);li.append(X);li.append(Y)
def ad(lin):tex.insert(INSERT,lin+'')
def tak(lis,py):
    C.create_line(lis,fill=py.kleur,arrow='last' )
    if p==pma:C.create_text(lis[2:4],text=py.naam,font=('Small Fonts','8'))
def boom(x,y,p,n):
    br=n**p;d=0.02
    for i in range(0,n):
        yn=y+(n-1)*br/2/n-i*br/n
        if p==pma:lis=[];ap(lis,x+0.5,yn/2);C.create_text(lis,text=str(kans[i]),font=('Small Fonts','8'))
        lis=[];ap(lis,x+d,y);ap(lis,x+1,yn);tak(lis,pyl[i])
        if p>1:boom(x+1,yn,p-1,n)
def wis(E):E.delete(0,len(E.get()))
def nieuw():
    for E in (E1,E2,E3):wis(E)
    C.delete(ALL);tex.delete('1.0',END)
def teken():
    global n,p,pma,ka,xmi,xma,ymi,yma,pyl,br,kans
    C.delete(ALL);tex.delete('1.0',END)
    n,p=mult(E1.get());kans=mult(E2.get());aant=mult(E3.get());fo='Foutieve invoer'
    pyl=[];n,p=int(n),int(p);pma=p
    if n**p>256:messagebox.showinfo(fo,'n^p='+str(n**p)+'>256');return
    if len(kans)!=n:messagebox.showinfo(fo,'Aantal kansen='+str(len(kans))+'≠n');return
    if len(aant)!=n:messagebox.showinfo(fo,'Aantal keuzes='+str(len(aant))+'≠n');return
    somkans=0;somaant=0
    for i in range(0,n):somkans=somkans+kans[i]
    if abs(somkans-1)>1E-6:messagebox.showinfo(fo,'Som kansen='+format(somkans,'.4f')+'≠1');return
    for i in range(0,n):somaant=somaant+aant[i]
    if somaant!=p:messagebox.showinfo(fo,'Som aantal='+str(int(somaant))+'≠p');return
    #definiëren pijlen
    for i in range(0,n):
        pyl.append(Pijl(chr(i+65),kl[i],kans[i]))
    xmi=-0.2;xma=p+0.2;br=n**p/2;yma=br+0.4;ymi=-yma
    boom(0,0,p,n) #tekenen kansboom
    #berekenen kans
    for i in range (0,n):ad('P('+chr(65+i)+')='+str(kans[i])+' ')
    ad('\nGebeurtenis \n')
    aantalGeb=1;rest=p;kansGebeurt=1
    for i in range (0,n):at=int(aant[i]);aantalGeb=aantalGeb*Comb(rest,at);rest=rest-at;ad(str(at)+chr(65+i)+' ')
    ad('\nAantal takken die\nhieraan voldoen ='+str(aantalGeb))
    for i in range(0,n):kansGebeurt=kansGebeurt*kans[i]**aant[i]
```

```
    ad('\nP(1 tak)='+format(kansGebeurt,'.4f'))
    ad('\nP(gebeurtenis)='+format(aantalGeb*kansGebeurt,'.4f'))
    return
# hoofdprogramma
tit="Kansboom";gr='lightgrey';eps=1E-7;dx=1E-6
breedte=840;hoogte=1000;hoogteC=980;breedteC=600
brInv=620;hoInv0=10;hoInv1=600;hoInv2=640;hoInv3=680;hoInv4=720;hoInv5=810
form=Tk();form.geometry(str(breedte)+'x'+str(hoogte));form.title(tit);form.resizable(False,False)
C = Canvas(form, bg="white", height=hoogteC, width=breedteC);C.place(x=5,y=5)
#invoer
Label(form,text='n,p :').place(x=brInv,y=hoInv1)
E1=Entry(form,width=8);E1.place(x=brInv+65,y=hoInv1);E1.insert(0,'4,3')
Label(form,text='Kans A,B,C...:').place(x=brInv,y=hoInv2)
E2=Entry(form,width=22);E2.place(x=brInv+65,y=hoInv2);E2.insert(0,'0.3,0.2,0.4,0.1')
Label(form,text='# A,B,C...:').place(x=brInv,y=hoInv3)
E3=Entry(form,width=22);E3.place(x=brInv+65,y=hoInv3);E3.insert(0,'2,0,1,0')
tex=Text(form,bg='grey89',height=11,width=25,wrap=WORD);tex.place(x=brInv,y=hoInv5)
texinfo=Text(form,bg='grey94',height=34,width=25,wrap=WORD);texinfo.place(x=brInv,y=hoInv0)
infstr='Kansboom met teruglegging\n\nEr zijn n mogelijkheden\nA,B,C,..en p opeenvolgen- de vertakkingen.\n'
infstr=infstr+'De kansverdeling is niet\nuniform.P(A)≠of=P(B)≠..\n\n'
infstr=infstr+'De invoer voldoet aan\nde volgende voorwaarden:\nn^p <257\nAantal kansen A,B,C..=n\n'
infstr=infstr+'Aantal keuzes A,B,C..=n\nSom kansen A,B,C..=1\nMet # A,B,C.. kies je\neen gebeurtenis.\n'
infstr=infstr+'Som # A,B,C.. = p\nEen verkeerde keuze geeft\neen foutmelding.\n\nAls je op "Teken" klikt\n'
infstr=infstr+'wordt de kansboom gete-\nkend en worden de respectievelijke kansen\n'
infstr=infstr+'getoond.\nHet aantal takken die\novereenstemmen met de gebeurtenis worden\ngeteld of
berekend.\n'
infstr=infstr+'De kans op 1 tak= P(A)*P(B)*P(C)*..\nDe kans op de gebeurtenis\nis het product #takken*P(tak)'
texinfo.insert(INSERT,infstr)
Button(form,text='Teken',command=teken,).place(x=brInv,y=hoInv4,width=60)
Button(form,text='Nieuw',command=nieuw,).place(x=brInv+80,y=hoInv4,width=60)
kl=['red','blue','green','turquoise','violet','coral','orange','pink','yellow']
form.mainloop()
```



175. Meetkundige voorstelling van bewerkingen met complexe getallen
complexe getallen_meetkunde

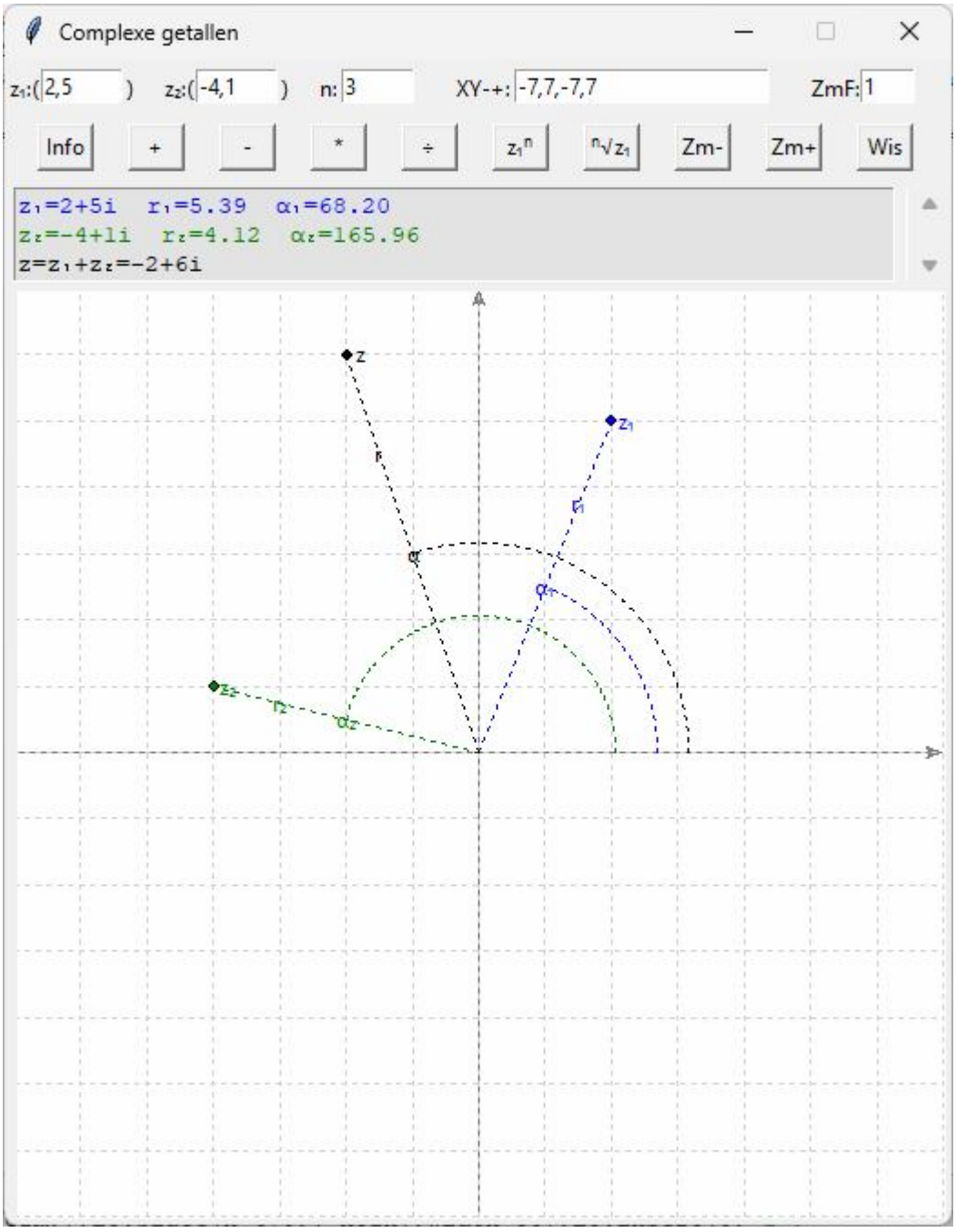
In dit programma moet je 2 complexe getallen $z_1(a,b)$ en $z_2(c,d)$ invoeren
Je kan ze met elkaar optellen, aftrekken, vermenigvuldigen en delen.
Van z_1 kan je ook de n-de macht en de n-de machtswortels berekenen.
In de grafiek zie je de meetkundige voorstelling van de gekozen bewerking.

Programma

```
# Complexe getallen, meetkundige voorstelling met tkinter
from math import *;from tkinter import *;from tkinter import messagebox
def multF(s):
    l=s.split(',');co=[]
    for i in l:co.append(float(i))
    return co
def multI(s):
    l=s.split(',');co=[]
    for i in l:co.append(int(i))
    return co
def transx(x):return (x-xmi)*breedteC/(xma-xmi)
def transy(y):return hoogteC-hoogteC*(y-ymi)/(yma-ymi)
def ad(lin):
    tex.insert(INSERT,lin)
    for j in range(0,3):
        tex.tag_add(str(j),str(j+1)+".0",str(j+1)+'.50');tex.tag_config(str(j),foreground=col[j])
def sg(x):
    if x==int(x):x=int(x);sx=str(x)
    else:sx=format(x,'.2f')
    if x>=0:return '+'+sx
    else: return sx
def sg2(x):
    if x==int(x):x=int(x);sx=str(x)
    else:sx=format(x,'.2f')
    return sx
def ap(li,x,y):X=transx(x);Y=transy(y);li.append(X);li.append(Y)
def punt(x,y,te,kl):
    diamx=(xma-xmi)/200;diamy=(yma-ymi)/200;d=3*diamx;r=sqrt(x*x+y*y);h=arg(x,y)
    diam=0.1;lis=[];ap(lis,x-diamx,y-diamy);ap(lis,x+diamx,y+diamy);C.create_oval(lis,fill=kl)
    lis=[];ap(lis,x+d,y);C.create_text(lis,text='z'+te,fill=kl)
    lis=[];ap(lis,0,0);ap(lis,x,y);C.create_line(lis,fill=kl,dash=[1,2])
    lis=[];
    if h<0:stap=-0.01
    else:stap=0.01
    t=-stap
    while abs(t-h)>abs(stap):t=t+stap;ap(lis,r*cos(t)/2,r*sin(t)/2)
    C.create_line(lis,fill=kl,dash=[1,2])
    lis=[];ap(lis,r*cos(h)/2,r*sin(h)/2);C.create_text(lis,text='α'+te,fill=kl)
    lis=[];ap(lis,.75*r*cos(h),.75*r*sin(h));C.create_text(lis,text='r'+te,fill=kl)
def arg(a,b):
    if a==0 and b>0:h=pi/2
    elif a==0 and b<0:h=3*pi/2
    else:
        h=atan(b/a)
        if a<0:h=h+pi
    return h
def bereken(i):
    global xmi,ymi,xma,yma,a,b,c,d,zf
    C.delete(ALL);tex.delete('1.0',END)
```

```
    a,b=multF(E1.get());c,d=multF(E2.get());n=int(E3.get())
    xmi,xma,ymi,yma=multI(E4.get());zf=int(E5.get())
    if (xmi>=0 or xma<=0 or ymi>=0 or yma<=0):messagebox.showinfo('Toon het assenstelsel','Verander XY-
+');return
    if (a==0 and b==0) or (c==0 and d==0):messagebox.showinfo('Ongeldige invoer','z1 en/of z2 ≠ 0');return
    #assen
    li=[];ap(li,xmi,0);ap(li,xma,0);C.create_line(li,fill='grey',arrow='last')
    li=[];ap(li,0,ymi);ap(li,0,yma);C.create_line(li,fill='grey',arrow='last')
    for x in range(int(xmi),int(xma+1)):
        li=[];ap(li,x,ymi);ap(li,x,yma);C.create_line(li,fill='lightgrey',dash=[1,2])
    for y in range(int(ymi),int(yma+1)):
        li=[];ap(li,xmi,y);ap(li,xma,y);C.create_line(li,fill='lightgrey',dash=[1,2])
    r1=sqrt(a*a+b*b);r2=sqrt(c*c+d*d)
    h1=arg(a,b);h2=arg(c,d)
    if i==0:x=a+c;y=b+d
    if i==1:x=a-c;y=b-d
    if i==2:r=r1*r2;h=h1+h2
    if i==3:r=r1/r2;h=h1-h2
    if i==4:r=r1**n;h=h1*n
    if i==5:r=r1**(1/n);h=h1/n
    if i>1:x=r*cos(h);y=r*sin(h)
    h1g=180*h1/pi;h2g=180*h2/pi
    if i>1:hg=180*h/pi
    punt(a,b,'1',col[0]);ad('z1='+sg2(a)+sg(b)+'i  r1='+sg2(r1)+'  α1='+sg2(h1g))
    if i<4:punt(c,d,'2',col[1]);ad('\nz2='+sg2(c)+sg(d)+'i  r2='+sg2(r2)+'  α2='+sg2(h2g))
    else:ad('\n')
    ad('\nz='+bewerktek[i]+'='+sg2(x)+sg(y)+'i')
    if i<5:punt(x,y,'',col[2])
    if i>1:ad('  r='+sg2(r)+'  α='+sg2(hg))
    if i==5:
        ad(sg(360/n)+'*k ( k=0..'+sg2(n-1)+')\n')
        for j in range(1,n):ad('z='+sg2(r*cos(h+2*j*pi/n))+sg(r*sin(h+2*j*pi/n))+'i ')
        lis=[];stap=0.01;t=-stap
        while t<2*pi:t=t+stap;ap(lis,r*cos(t),r*sin(t))
        C.create_line(lis,fill=col[2],dash=[1,2])
        stap=2*pi/n;t=h-stap
        while t<2*pi+h:t=t+stap;punt(r*cos(t),r*sin(t),'',col[2])
def but(i):Button(form,text=bew[i],width=3,command=lambda:bereken(i)).place(x=68+50*i,y=35)
def nieuwXY():
    if xmi<0 and xma>0 and ymi<0 and yma>0:wis(E4);E4.insert(0,str(xmi)+','+str(xma)+','+str(ymi)+','+str(yma))
def XYmin():global xmi,xma,ymi,yma;zf=int(E5.get());xma-=zf;yma-=zf;xmi+=zf;ymi+=zf;nieuwXY()
def XYplus():global xmi,xma,ymi,yma;zf=int(E5.get());xma+=zf;yma+=zf;xmi-=zf;ymi-=zf;nieuwXY()
def wis(E):E.delete(0,len(E.get()))
def nieuw():
     for E in (E1,E2,E3,E4,E5):wis(E)
     C.delete(ALL);tex.delete('1.0',END)
def info():h=messagebox.showinfo(tit,infstr)
#hoofdprogramma
bew=['+','-','*','÷','z1^n','^n√z1']
bewerktek=['z1+z2','z1-z2','z1*z2','z1÷z2','z1^n','^n√z1']
col=['blue','green','black']
```

```
xmi,xma,ymi,yma=-5,5,-5,5;zf=1
breedte=520;hoogte=640;tit='Complexe getallen'
hoogteC=510;breedteC=510;hoInv=5
form=Tk();form.geometry(str(breedte)+'x'+str(hoogte));form.title(tit)
form.resizable(False,False)
C = Canvas(form, bg="white", height=hoogteC, width=breedteC);C.place(x=5,y=125)
L1=Label(form,text='z1:(              )',);L1.place(x=0,y=hoInv)
E1=Entry(form);E1.place(x=20,y=hoInv,width=45);E1.insert(0,'2,5')
L2=Label(form,text='z2:(              )',);L2.place(x=85,y=hoInv)
E2=Entry(form);E2.place(x=105,y=hoInv,width=45);E2.insert(0,'-4,1')
L3=Label(form,text='n:',);L3.place(x=170,y=hoInv)
E3=Entry(form);E3.place(x=185,y=hoInv,width=40);E3.insert(0,'3')
L4=Label(form,text='XY-+:',);L4.place(x=245,y=hoInv)
E4=Entry(form,);E4.place(x=280,y=hoInv,width=140);E4.insert(0,'-7,7,-7,7')
L5=Label(form,text='ZmF:',);L5.place(x=440,y=hoInv)
E5=Entry(form,);E5.place(x=470,y=hoInv,width=30);E5.insert(0,'1')
tex=Text(form,bg='grey89',height=3,width=60,wrap=WORD);tex.place(x=5,y=70)
vsb=Scrollbar(form,orient=VERTICAL,command=tex.yview,width=25)
tex['yscrollcommand']=vsb.set;vsb.place(x=495,y=70)
for keu in range(0,6):but(keu)
Button(form,text='Info',width=3,command=info).place(x=18,y=35)
Button(form,text='Zm-',width=3,command=XYmin).place(x=368,y=35)
Button(form,text='Zm+',width=3,command=XYplus).place(x=418,y=35)
Button(form,text='Wis',width=3,command=nieuw).place(x=468,y=35)
infstr='Bepaal met +,-,*,÷ :\nz=z1+z2 , z=z1-z2\nz=z1*z2 , z=z1÷z2\n'
infstr=infstr+'Met z1^n : z=z1^n en\nmet ^n√z1 : z=^n√z1\n'
infstr=infstr+'Zm- of Zm+:\nxmi,xma,ymi,yma\n'
infstr=infstr+'ZmF kleiner of groter'
form.mainloop()
```


Complexe getallen
z₁:(2,5) z₂:(-4,1) n:3 XY-+:-7,7,-7,7 ZmF:1
Info + - * ÷ z₁ⁿ ⁿ√z₁ Zm- Zm+ Wis
z₁=2+5i r₁=5.39 α₁=68.20
z₂=-4+1i r₂=4.12 α₂=165.96
z=z₁+z₂=-2+6i
z
z₁
r
r₁
α
α₁
z₂
r₂
α₂

176. Programma's blz. - naam