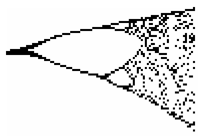
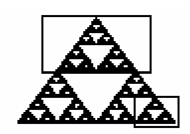
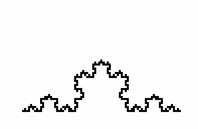
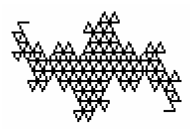
Cahiers T³ Europe
Vlaanderen nr. 17

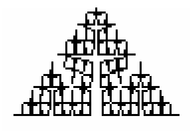
Fractalen met de TI-84+

Onderzoekscompetenties wiskunde/wetenschappen voor de 3de graad ASO

Didier Deses



KATHOLIEKE UNIVERSITEIT LEUVEN

Onderzoekscompetenties: Fractalen met de **TI-84+**

Dr Didier Deses[1]

[1]Leerkracht wiskunde K. A. Koekelberg, medewerker aan het departement wiskunde van de VUB

Voorwoord

Dit cahier is bedoeld als leidraad om onderzoekscompetenties in te richten voor polen wetenschappen en/of wiskunde. De bedoeling van de onderzoekscompetenties is dat een leerling zelf een deel wiskunde ontdekt. Omdat het ondenkbaar is dat een leerling ASO tweehonderd jaar wiskunde eventjes zelf uitwerkt, werd deze leidraad gemaakt. Het centrale onderwerp, *fractalen*, biedt ruim de mogelijkheid om verschillende takken van de wiskunde aan te halen (meetkunde, dimensietheorie, stochastiek, chaostheorie, algebra, ...) en haar vele toepassingen (economie, biologie, informatica, ...). Er werd bij het maken van deze tekst gestreefd naar een grote interdisciplinariteit. Leerlingen zullen in aanraking komen met economie, biologie en zelfs latijn! Zonder de moeilijkheidsgraad te laten stijgen zullen leerlingen op bepaalde domeinen relatief ver geraken, zo wordt bijvoorbeeld ook een eigen programmeertaal ontworpen.

Het praktische uitgangspunt van dit cahier is dat er zowel opzoekingswerk als eigen onderzoek van de leerlingen wordt verwacht. Er wordt geen zware wiskunde opgelegd. Het is aan de leerkracht om de verwachte diepgang en moeilijkheidsgraad te bepalen. Sommige onderwerpen doorkruisen de leerstof, bijvoorbeeld de driehoek van Pascal, complexe getallen, ... Ook hier zal de leerkracht eventueel rekening mee moeten houden. Er wordt aanbevolen om ofwel klassikaal het volledige cahier te behandelen, ofwel in groepjes te werken. Het eerste hoofdstuk en de eerste bijlage zijn dan wel nodig voor iedereen, maar daarna kan elk groepje een hoofdstuk zelfstandig afwerken. Evaluatie kan gebeuren door het cahier af te geven, door een (documentatie)mapje aan te maken en/of door een presentatie.

De inspiratie voor de leerstof en programma's voor deze tekst is ontstaan uit de eigen ervaring van de auteur. Hij is in zijn eigen schooltijd (eind '80, begin '90) in aanraking gekomen met het onderwerp fractalen. De programma's dateren van die periode, toen nog in BASIC op een Amstrad-computer met 64kb geheugen! Bij het maken van dit cahier is het opgevallen hoezeer de problemen van toen (zwart-wit scherm, beperkte rekencapaciteit) zich herhalen. De **TI-84+** is immers ongeveer even krachtig als de computers uit die periode! Tenslotte is ook gebleken hoezeer onderzoekscompetenties belangrijk zijn. Dit onderwerp is immers voor de auteur de motivatie geweest om verder wiskunde te studeren en onderzoek binnen de wiskunde te doen.

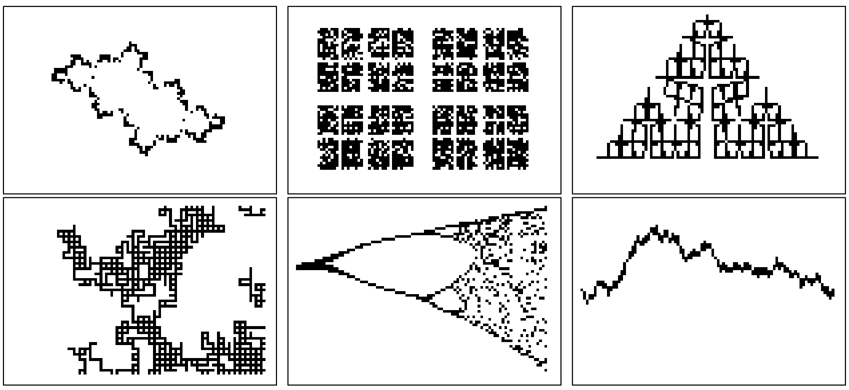
Inhoudsopgave

Hoofdstuk 1

Fractalen

In de wiskunde komen er enorm veel objecten voor, denk maar aan getallen, breuken, functies, grafieken, regelmatige veelhoeken, ... Meestal bestudeert men deze objecten zodat men hun goede eigenschappen kan gebruiken. Zo zal men naar continue functies kijken, die liefst ook afleidbaar zijn. Maar soms duiken er rariteiten op.

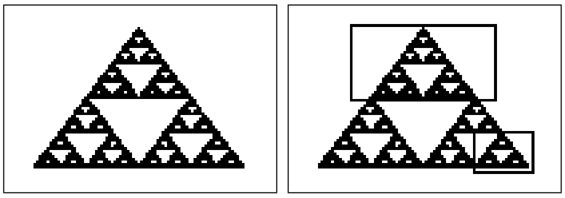
Fractalen zijn zulke curiositeiten. Het zijn meetkundige objecten, die een zeer grillige vorm hebben en waarin regelmaat soms zeer ver te zoeken is.



Fractalen zouden waarschijnlijk in het rariteitenkabinet van de wiskunde gebleven zijn, moesten ze niet zo vaak terug te vinden zijn in de natuur. Zo zijn bloemkolen, de kustlijn van een continent en de evolutie van de populatie van een bepaalde diersoort voorbeelden van fractalen. Een echt sluitende, wiskundige definitie geven van een fractaal is moeilijk. Wij zullen hier het begrip fractaal introduceren aan de hand van de typische kenmerken van deze objecten.

1.1 Zelfgelijkvormigheid

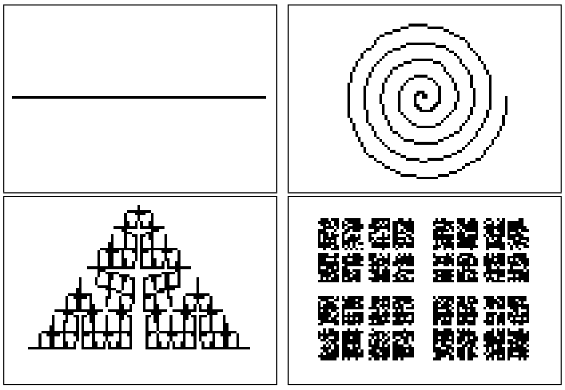
Het eerste en meest opvallende kenmerk van een fractaal is **zelfgelijkvormigheid**. Dit wil zeggen dat men in de fractaal delen terugvindt die een kopie zijn van de gehele fractaal zelf. Een van de beste voorbeelden om dit te tonen is de *zeef van Sierpinski*, genoemd naar de Poolse wiskundige Waclaw Sierpinski (1882-1969). Vanzelfsprekend is de werkelijke fractaal oneindig gedetailleerd!



In de twee kaders op de tweede tekening zie je telkens weer de linkse figuur verschijnen. Omdat de fractaal oneindig gedetailleerd is, kan je in de kleinere kopies binnen de kaders opnieuw dezelfde figuur herkennen, en zo tot in het oneindige door.

Opdracht 1. Zoek zelf op het internet naar enkele andere formuleringen van het begrip zelfgelijkvormigheid.

Opdracht 2. Zijn volgende figuren (deels) zelfgelijkvormig? Duid de gelijkvormige gebieden aan.



Opdracht 3. Zoek een aantal beelden van fractalen waar je de zelfgelijkvormigheid duidelijk ziet.

In zeer vele fractalen springt deze eigenschap van zelfgelijkvormigheid in het oog. Dit is niet alleen waar voor wiskundige fractalen, maar ook voor fractalen die natuurlijk voorkomen. Ongetwijfeld is het meest bekende voorbeeld een bloemkool. De tweede foto hieronder is eigenlijk een uitvergroting van de eerste.

Opdracht 4. Zoek een aantal beelden van zelfgelijkvormigheid die je in de natuur terugvindt.

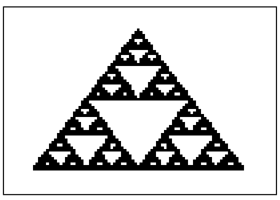
1.2 Fractale dimensie

Een andere gangbare definitie van een fractaal wordt gegeven aan de hand van de dimensie. Je weet waarschijnlijk reeds dat de dimensie van een rechte 1 is, die van een vlak 2 en van de ruimte 3. In de fysica wordt tijd ook als een dimensie gezien en wij leven dus in een 4-dimensionale wereld. Dit dimensiebegrip komt overeen met het minimaal aantal assen dat nodig is om een punt d.m.v. zijn coördinaten te bepalen ((x, y) in het vlak, (x, y, z) in de ruimte). Deze dimensie is altijd een natuurlijk getal.

In de wiskunde bestaan er echter nog vele andere dimensiebegrippen. Eén ervan is de **Hausdorffdimensie**, genoemd naar de Duitse wiskundige Felix Hausdorff (1868-1942). Deze dimensie is voor algemene verzamelingen te moeilijk om op het SO niveau te definiëren, maar voor sommige zelfgelijkvormige verzamelingen kan dit wel. We hebben gezien dat een fractaal kan bestaan uit n verschillende niet-overlappende kopies van zichzelf die telkens verkleind zijn met eenzelfde factor h (< 1). De dimensie is dan gegeven door volgende formule.

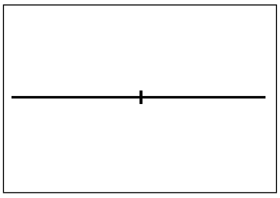
$$d_H = \frac{\ln n}{\ln \frac{1}{h}}$$

Als we het voorbeeld van de zeef van Sierpinski bekijken bestaat deze uit $n = 3$ kopies, die verkleind zijn met een factor $h = \frac{1}{2}$.



De dimensie is dus $d_H = \frac{\ln 3}{\ln 2} \approx 1.5849$. Deze dimensie hoeft dus geen natuurlijk getal meer te zijn. Een fractaal is dus een verzameling waarvan de Hausdorffdimensie geen natuurlijk getal is.

Merk op dat, in het geval van een lijnstuk, we de normale dimensie terugvinden. Een lijnstuk is immers opgebouwd uit twee lijnstukjes, die half zo groot zijn.



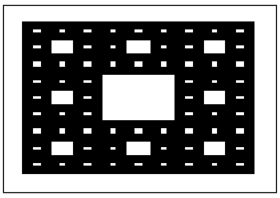
De dimensie is dus $d_H = \frac{\ln 2}{\ln 2} = 1$.

Opdracht 5. Toon aan dat de Hausdorffdimensie van een rechthoek 2 is en die van een balk 3.

Opdracht 6. Toon aan dat de Hausdorffdimensie van een gelijkzijdige driehoek 2 is.

Je kan nu besluiten dat de zeef van Sierpinski qua dimensie ligt tussen een lijnstuk en driehoek.

Opdracht 7. Bepaal de dimensie van het *Sierpinskitapijt*.

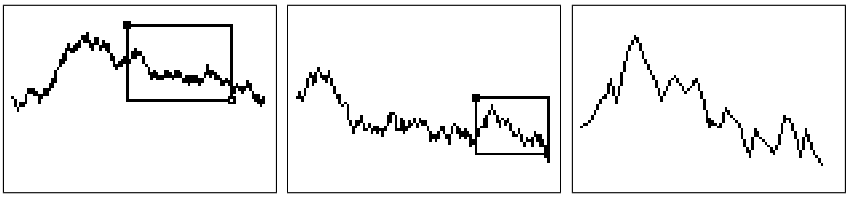


Opdracht 8. Het Sierpinskitapijt en de zeef van Sierpinski hebben beide ook ruimtelijke equivalenten: *de Mengerspons*, genoemd naar Karl Menger (1902-1985), *de Sierpinski tetraëder en de Sierpinski piramide*. Zoek beelden ervan en bepaal de dimensie.

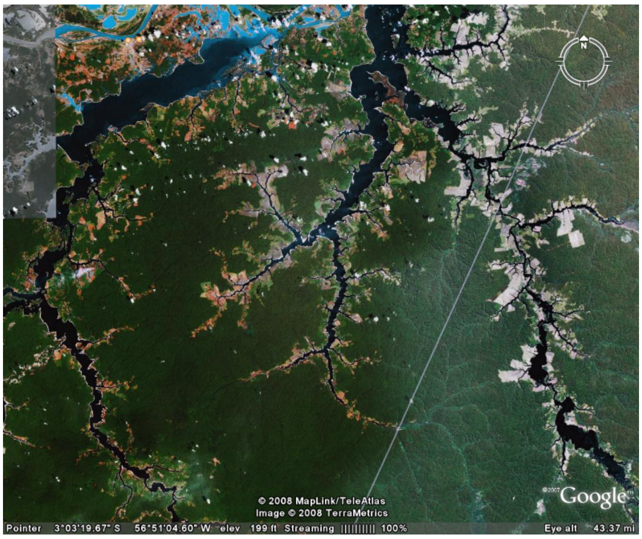
Opdracht 9. In de natuur komen fractalen ook veel voor. Een varenblad is een duidelijk voorbeeld. Ga op zoek naar een varenplant en bepaal de fractale dimensie van een aantal van haar bladeren. Wat is je conclusie?

1.3 Stochastische fractalen

Naast gelijkvormigheid en dimensie is er nog een derde factor die sommige fractalen karakteriseert: **toeval**. Wanneer je bijvoorbeeld beurskoersen bekijkt gedurende een jaar bekom je grafieken die zeer sterk op fractalen lijken. Er is wel geen exacte zelfgelijkvormigheid, maar op elke schaal is de koers even onvoorspelbaar. Zulke fractalen noemt men **stochastische fractalen**.



Stochastische fractalen komen veelvuldig voor in de natuur. Enkele voorbeelden zijn bliksems, kustlijnen of rivieren zoals hier in het Amazonegebied.



Opdracht 10. Zoek met Google Earth of Google Maps nog enkele voorbeelden van fractalen op onze planeet. Zoom telkens in en maak een fotoreportage.

Je hebt nu een goed idee van wat fractalen zijn, welke soorten er bestaan en waar ze kunnen voorkomen. In de volgende hoofdstukken zul je leren hoe je zelf fractalen kan maken. Volgende thema's worden behandeld:

- De driehoek van Pascal en de zeef van Sierpinski: getallen en de Hausdorffdimensie

- De dronkenman en de chaos: stochastische fractalen
- De schildpad en de draak: zelf een programmeertaal schrijven en de Hausdorffdimensie
- Julia Fractalen: de mooiste en beroemdste fractalen.

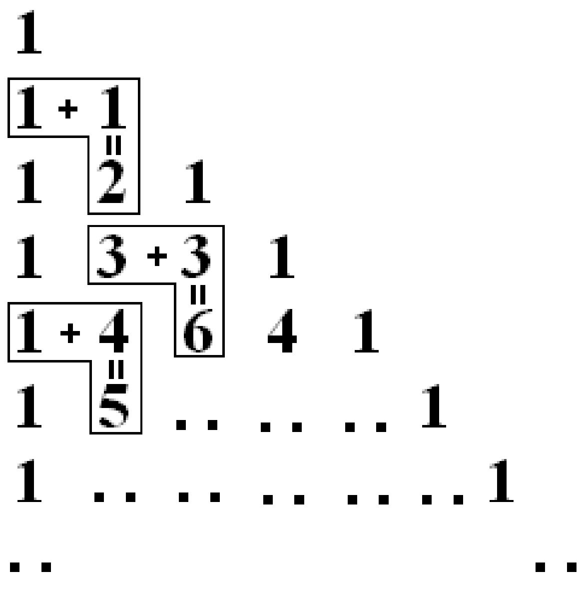
In elk van deze hoofdstukken zullen we op de TI-84+ enkele programma's maken om fractalen te berekenen. Hiervoor is een beetje programmeerkunst nodig maar dit is echt niet veel. Als je er nog niets van weet, is het nu tijd om de eerste bijlage eens te lezen!

Hoofdstuk 2

De driehoek van Pascal en de zeef van Sierpinski

2.1 De driehoek van Pascal

De *driehoek van Pascal* is een getallendriehoek, een schema van getallen in driehoekige vorm. Je maakt de driehoek als volgt. Een getal in de driehoek is de som van het getal erboven en het getal linksboven. Het begin en einde van elke rij is 1 en elke rij bevat één getal meer dan de vorige.



Opdracht 11. Neem een blad en maak daarop een raster van ruitjes van 1

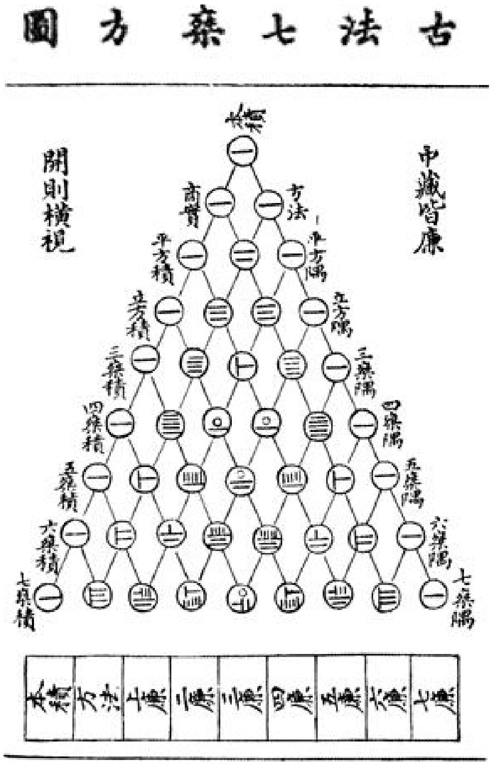
cm op 1 cm. Maak nu hierop de eerste 17 lijnen van de driehoek van Pascal. Maak er een drietal kopies van.

Opdracht 12. Nummer de rijen van je driehoek 0,1,2, ... en doe hetzelfde met de kolommen. Vergelijk nu het getal op de $n$de rij en de $k$de kolom met de combinatie $C_n^k = \binom{n}{k} = \frac{n!}{(n-k)!k!}$ uit de combinatoriek. Wat merk je?

Opdracht 13. In de driehoek van Pascal is een getal de som van de twee getallen erboven. Schrijf dit uit d.m.v. de combinaties uit de vorige opdracht. Bewijs nu de formule die je hebt gevonden.

Opdracht 14. Werk volgende veeltermen uit en schrijf de coëfficiënten die voorkomen op.
$(x+y)^0 = 1$
$(x+y)^1 = x+y$
$(x+y)^2 = x^2+2xy+y^2$
$(x+y)^3 =$
$(x+y)^4 =$
$(x+y)^5 =$
Wat merk je? Waarvoor kun je de driehoek van Pascal dus ook gebruiken? Deze stelling noemt men het *binomium van Newton.* Zoek ze op en formuleer ze correct. Zoek ook een bewijs.

Opdracht 15. De driehoek van Pascal was reeds gekend, lang voordat Blaise Pascal (1623-1662) hem gebruikte in de combinatoriek. Hieronder staat een blad uit een chinees handboek wiskunde uit de 11de eeuw. Herken je de driehoek van Pascal? Hoe noteerden de Chinezen toen hun cijfers?



2.2 De link met de zeef van Sierpinski

Het is duidelijk dat de driehoek van Pascal nuttig is in de algebra of in de combinatoriek, maar hoe komen we nu tot fractalen? We zullen hiervoor een andere manier van tellen invoeren. We definiëren een nieuwe optelling binnen de verzameling $\{0, 1\}$ als volgt:

$$0 + 0 = 0$$

$$0 + 1 = 1$$

$$1 + 0 = 1$$

$$1 + 1 = 0$$

Opdracht 16. Bewijs dat $(\{0, 1\}, +)$ een commutatieve groep is.

Opdracht 17. Maak opnieuw een blad met een raster van 1 cm op 1 cm. Vul nu de driehoek van Pascal in maar nu gerekend met de nieuwe optelling. Kleur vervolgens de 1'n donker.

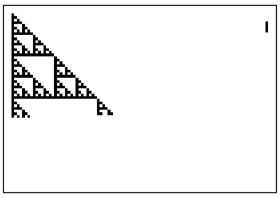
Opdracht 18. Neem één van je driehoeken en geef elk hokje waarin een oneven getal staat een donkere kleur, de even getallen laat je dus ongekleurd. Welke figuur zie je ontstaan?

Opdracht 19. Neem opnieuw één van je driehoeken en laat deze keer enkel de drievouden ongekleurd. Wat krijg je nu?

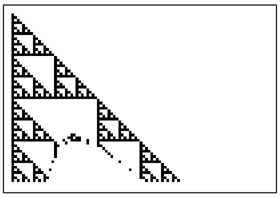
We gaan nu de **TI-84+** programmeren zodat we op de rekenmachine de bovenstaande opdrachten automatisch kan laten uitvoeren. We beginnen met de even getallen blanco te laten. Het programma ziet er dan als volgt uit:

```
PROGRAM:PASFRAC
:ClrDraw
:For(N,0,62)
:For(K,0,N)
:If fPart((N nCr
 K)/2)≠0:Pxl-On(
N,K)
:End:End
```

Eerst maken we het grafisch scherm leeg door $\boxed{\texttt{2nd}}$`[draw]` `[clrdraw]`. Daarna beginnen we twee for-lussen: één voor elke rij uit de driehoek en één voor elk elementje uit die rij. We berekenen dan het getal dat op de plaats (n, k) hoort in de driehoek door $\boxed{\texttt{math}}$`[prb]` `[nCr]` en delen dit door 2. Met $\boxed{\texttt{math}}$`[num]` `[fpart]` en $\boxed{\texttt{2nd}}$`[test]` `[≠]` gaan we na of het gedeelte na de komma nul is of niet, m.a.w. of het getal even is of niet. Wanneer het oneven is tekenen we het punt dat overeenstemt met (n, k) d.m.v. $\boxed{\texttt{draw}}$`[pxl-on]` (absolute schermcoördinaten, de werkelijke resolutie is 95 x 62, maar bij het maken van grafieken wordt de resolutie 94 x 62, één kolom pixels wordt gebruikt om te tonen dat de **TI-84+** bezig is). Het resultaat is de bekende zeef van Sierpinski.



Opdracht 20. Merk op dat als je voorbij de 42ste rij komt, er dan punten bijkomen die er in werkelijkheid niet zijn. Hoe zou dit komen? (Hint: reken maar eens $\binom{42}{26}$ uit)



Je hebt waarschijnlijk reeds gemerkt dat het programma relatief traag is. We kunnen het sneller laten gaan door gebruik van volgende eigenschap.

Opdracht 21. Bewijs dat $\binom{n}{k} = \binom{n}{n-k}$.

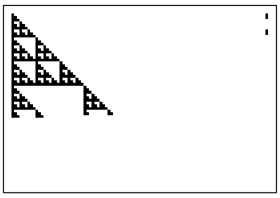
Opdracht 22. Leg uit waarom met volgende veranderingen het programma tweemaal sneller is.

```
PROGRAM:PASFRAC
:█lrDraw
:
:For(N,0,62)
:For(K,0,N/2)
:If fPart((N nCr
 K)/2)≠0
:Then
```

```
PROGRAM:PASFRAC
:Pxl-On(N,K)
:Pxl-On(N,N-K)
:End
:End:End
:
:
:█
```

We hebben reeds getoond dat de zeef van Sierpinski uit 3 kopies van zichzelf bestaat, die verkleind zijn met een factor $h = \frac{1}{2}$. De Hausdorffdimensie is dus $d_H = \frac{\ln(3)}{\ln(2)} \approx 1.5846$.

Wanneer we enkel de drievouden weghalen uit de driehoek van Pascal bekomen we een andere fractaal.



Opdracht 23. Toon aan dat voor dit voorbeeld $d_H \approx 1.631$.

Opdracht 24. Als je de vijfvouden weglaat krijg je $d_H = \frac{\ln(15)}{\ln(5)} \approx 1.683$. Hoe ziet de fractaal eruit?

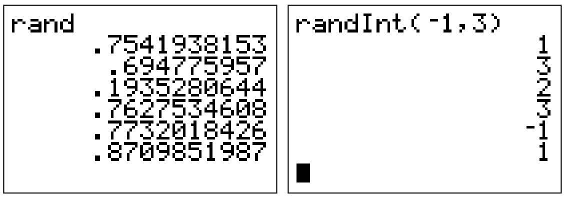
Opdracht 25. Merk op dat voor een n-voud, waar n niet priem is, de dimensie veel moeilijker te bepalen is. Kijk maar naar het geval voor viervouden. Wat is hierbij het probleem?

Hoofdstuk 3

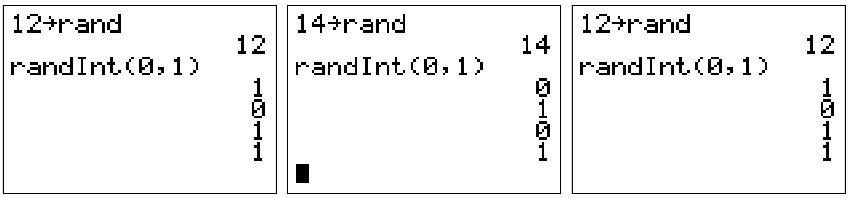
De dronkenman en de chaos

3.1 Toevallige getallen op de beurs

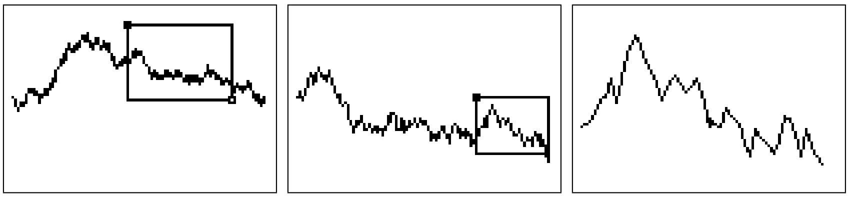
Stochastische fractalen ontstaan door toeval. Heel veel processen rondom ons hangen af van toeval. Om zulke fractalen te maken is het nodig dat we een reeks willekeurige getallen kunnen genereren op de **TI-84+** . Dit gebeurt als volgt. Het commando `math` `[prb]` `[rand]` geeft telkens een willekeurig getal in het interval $[0, 1[$. Met `math` `[prb]` `[randint]` kan je willekeurige gehele getallen aanmaken tussen twee grenzen.



Soms zal het nodig zijn om telkens weer dezelfde reeks toevalsgetallen te kunnen bestuderen. Dit kan door een startwaarde, de *seed*, te geven aan `rand`.



Waar stochastische fractalen een grote rol spelen is de beurs. Het aandeel van een bedrijf is op een bepaalde dag 100 euro waard. Veronderstel dat, onder normale omstandigheden, de aandeelprijs dagelijks met maximaal met 5 euro kan stijgen of dalen. Wanneer er (te) grote fluctuaties optreden wordt de beurs immers stilgelegd. De grafiek van de bekomen koers gedurende een jaar is een stochastische fractaal.



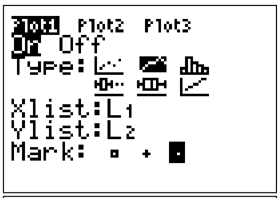
Het programma dat zulke grafieken maakt ziet er als volgt uit.

```
PROGRAM:BEURS
:ClrList L1,L2
:seq(K,K,1,365)→
L1:100→L2(1)
:For(K,2,365)
:L2(K-1)+randInt
(⁻5,5)→L2(K)
:End
```

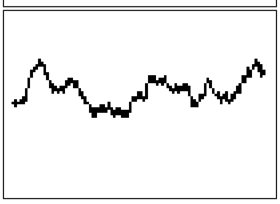
Eerst worden de lijsten L_1 en L_2 leeggemaakt. In L_1 komen nu de dagen, genummerd van 1 t.e.m. 365, dit wordt gedaan d.m.v. `2nd` `[list]` `[op]` `[seq]`. De eerste dag is de prijs van het aandeel 100 euro, dit wordt in L_2 onthouden. Daarna begint een lus, waarin de prijs van het aandeel wordt herberekend door bij de prijs van de vorige dag een willekeurig getal tussen -5 en 5 op te tellen, dit gebeurt d.m.v. `math` `[prb]` `[randint]`. Het resultaat van het programma zijn twee lijsten; L_1 bevat de dagen, L_2 de koers van het aandeel.

Geef nu een bepaalde seed in en start het programma. De berekening duurt een tijdje. Om dan de grafiek te maken gebruik je `2nd` `[statplot]` en `window` met de volgende instellingen. Daarna druk je op `graph`. Met `zoom` `[zoombox]` kun je inzoomen.

```
31→rand
                31
prgmBEURS
              Done
```




```
WINDOW
 Xmin=0
 Xmax=365
 Xscl=30
 Ymin=0
 Ymax=200
 Yscl=10
 Xres=1
```



Opdracht 26. Laat het programma een aantal keer draaien. Je krijgt telkens een andere koers te zien. Vind er eentje die stijgt, eentje die daalt en eentje waar de waarde van het aandeel na een jaar niet veel veranderd is. Geef telkens de bijhorende seed. Wat komt het meeste voor? Kan je dit verklaren?

Opdracht 27. Zoek een aantal grafieken van beurskoersen. Hebben deze de kenmerken van een fractaal? Soms worden ze radicaal beïnvloed door uitzonderlijke omstandigheden (oorlogen, schandalen, enz. ...). Kan je de invloed aflezen op de grafiek?

3.2 De Brownse beweging

De eerste die het effect van toeval zag en bestudeerde was de Schotse botanist Robert Brown (1773-1858). Hij bestudeerde kleine stuifmeelkorreltjes onder een microscoop. De korreltjes zaten in water en hij merkte dat ze willekeurige bewegingen maakten. Later heeft men hiervoor volgende verklaring gevonden. Het zijn de watermoleculen die onder de invloed van warmte rollen en botsen. Hoewel een stuifmeelkorrel veel groter is ondergaat ze toch ook de invloed van de botsing van de watermoleculen. Hierdoor beweegt de korrel op een onvoorspelbare manier.

Opdracht 28. (Enkel voor Latinisten.) Brown was zeker niet de eerste wetenschapper die opmerkte dat toevallige bewegingen in de natuur voorkomen. *Lucretius* (98VC-55VC) beschrijft een gelijkaardige beweging van stofdeeltjes in het zonlicht om de atoomtheorie van *Democritus* (460VC-375VC) te staven.

Contemplator fundunt radii per opaca domorum:
multa minuta modis multis per inana uidebis
corpora misceri radiorum lumine in ipso,
...
quod tales turbae motus quoque materiai
significant clandestinos caecosque subesse.
Multa uidebis enim plagis ibi percita caecis
commutare uiam retroque repulsa reuerti
...
Scilicet hic a principiis est omnibus error.

De Rerum Natura,

Lucretius,
boek II, verzen 114-117,127-130, 132.

Vertaal bovenstaande tekst, vraag eventueel hulp aan je leerkracht Latijn. Heb je dit fenomeen zelf al eens waargenomen? Kun je het nabootsen?

Spijtig genoeg wordt dit verschijnsel veroorzaakt door veranderingen van temperatuur, hetgeen de lucht in beweging zet. Deze thermische beweging is echter niet chaotisch van aard, ze volgt welbepaalde wetten. Het is dus geen zuivere Brownse beweging.

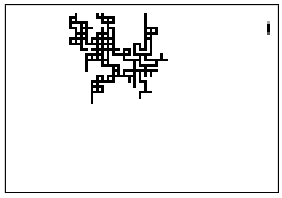
Een variant van de Brownse beweging is de het verhaal van de dronkenman. Die is volledig bezopen en staat in het midden van een plein. Hij wandelt verder en telkens zet hij een stap, ofwel naar voor of achter, ofwel naar links of rechts. De wandeling die hij aflegt is van dezelfde aard als de beweging van de stuifmeelkorrel. In het engels spreekt men van een *random walk*. Met volgend programma simuleren we deze wandeling van een dronkaard op de **TI-84+** .

```
PROGRAM:RWALK
:ClrDraw
:AxesOff
:.2→H
:0→X:0→Y
:
:Lbl 1
:X→A:Y→B
```

```
PROGRAM:RWALK
:rand→T
:If T<.5:X+H*ran
dInt(⁻1,1)→X
:If T≥.5:Y+H*ran
dInt(⁻1,1)→Y
:Line(A,B,X,Y)
:Goto 1
```

Eerst maken we het grafisch scherm leeg en halen de assen weg, dan wordt de stapgrootte op $H = 0.2$ gezet en de beginpositie op $(0, 0)$. Vervolgens zal een lus gestart worden d.m.v. een `label` (`prgm` `[ctl]` `[lbl]`). De huidige positie (X, Y) wordt opgeslagen als (A, B) en we beginnen met een stap te zetten. We kiezen met `math` `[prb]` `[rand]` een willekeurig getal T tussen 0 en 1. Is $T < 0.5$ dan zetten we een stap naar links of rechts, anders (als $T \geq 0.5$) zetten we een stap naar voor of achter. De stap wordt gezet door een willekeurig geheel getal te kiezen (`math` `[prb]` `[randint]`) tussen -1 en 1, dwz een getal uit $\{-1, 0, 1\}$ (het kan dus zijn dat de dronkaard eventjes blijft staan als 0 gekozen wordt). Tenslotte wordt een lijntje getekend vanuit de vroegere positie naar de nieuwe en wordt de lus gesloten d.m.v. `prgm` `[ctl]` `[goto]`.

Om het resultaat te zien gebruik je eerst `zoom` `[zdecimal]`. Daarna start je het programma. Het zal oneindig lang blijven draaien, je moet het zelf stopzetten door `on` te drukken en `[quit]` te kiezen.



Opdracht 29. Je kan het programma vereenvoudigen en versnellen door volgende veranderingen.

```
PROGRAM:RWALK
:ClrDraw
:AxesOff
:.2→H
:0→X:0→Y
:
:Lbl 1
:X→A:Y→B
```

```
PROGRAM:RWALK
:X+H*randInt(⁻1,
1)→X
:Y+H*randInt(⁻1,
1)→Y
:Line(A,B,X,Y)
:Goto 1
:█
```

Hiermee bekom je nog altijd een Brownse beweging. Wat mag de dronkenman nu ook nog doen? Waarom is dit programma sneller?

3.3 Het model van Verhulst

Een ander voorbeeld van een stochastische fractaal is het *model van Verhulst.* De Belgische wiskundige Pierre Verhulst (1804-1849) is onder invloed van Adolphe Quetelet (1796-1874) aan de studie begonnen over de groei van populaties. Hij ontdekte dat de aangroei van een populatie evenredig is met de grootte van de populatie, maar ook met overblijvende grondstoffen.

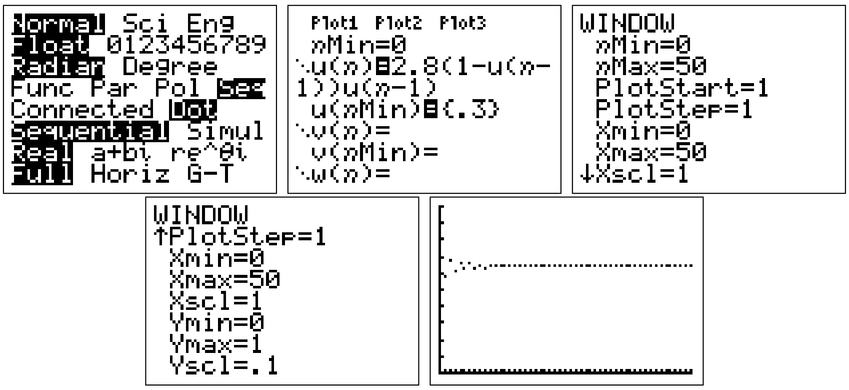
Opdracht 30. Zoek verdere informatie op over Verhulst en Quetelet.

We beschouwen de populatie van een diersoort op een beschikbaar grondgebied. De maximale populatie die op het gebied kan leven stellen we gelijk aan 100%. Op een bepaald tijdstip is de aanwezige populatie gegeven door x_n (in percent). Voor een bepaalde generatie is de populatie evenredig met de grootte van de populatie van de vorige generatie, dwz $x_n = cx_{n-1}$. We bekomen dus een meetkundige rij. Verhulst introduceerde een tweede factor. De populatie is immers ook evenredig met de overblijvende grondstoffen:

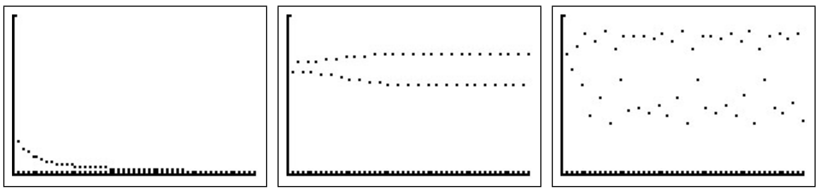
$$x_n = a(1 - x_{n-1})x_{n-1}$$

Het gedrag van deze rij is uiterst verbazend! Voor verschillende waarden van a verandert het (convergentie-)gedrag volledig.

We kunnen dit met de **TI-84+** gemakkelijk laten zien. Om een rij grafisch voor te stellen doe het volgende. Via `mode` selecteer je `[seq]` en `[dot]`. Als je nu op `y=` drukt kun je rijen invoeren. De `[nmin]` optie laat je toe om rijen te indexeren vanaf 0 of een andere positieve waarde. Je kan rijen geven door de algemene term of door recursie. Merk op dat n verkregen wordt door `X,T,θ,n` en u, v, w door bijvoorbeeld `2nd` `[u]` (`2nd` `7`). Voor je de grafiek maakt doe je er goed aan van met `window` de gewenste grenzen correct in te stellen.



We hebben in het voorbeeld gekozen voor een startwaarde van $x_0 = 0.3$, m.a.w. de beginpopulatie is 30% van de maximale populatie die het gebied kan dragen. Voor parameter a (die de voortplantingsfactor voorstelt) werd gekozen voor een waarde van 2.8. Op de grafiek zie je duidelijk dat de populatie eerst oscilleert, maar na een **stabilisatiefase** wordt ze stabiel op ongeveer 64%. We spreken hier van **convergentie** naar de waarde van 64%. Maar voor andere waarden van a kan je een heel ander gedrag krijgen.

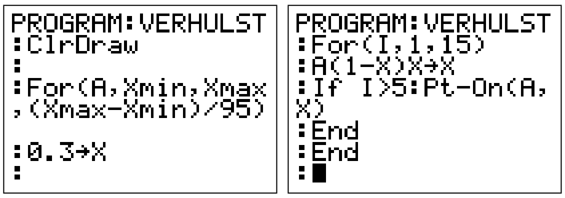


In het middelste voorbeeld is het alsof de rij naar twee verschillende waarden convergeert, in dit geval spreekt men van **adherentie**.

Opdracht 31. Als je weet dat de parameter a binnen $]0, 4[$ ligt, welke verschillende mogelijkheden kan je dan tegenkomen? Gebruik de startwaarde $x_0 = 0.3$.

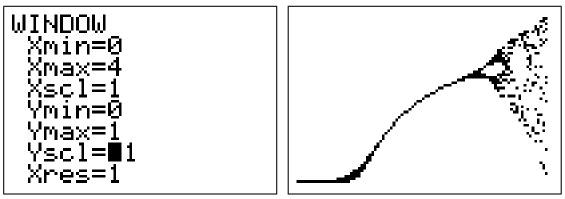
Opdracht 32. In welke mate kan de startwaarde voor verandering zorgen? Werk enkele voorbeelden uit.

Blijkbaar is het zo dat, afhangend van de waarde van a, de rij convergeert, adhereert aan meerdere waarden, of zelfs volledig chaotisch verloopt. Om beter te kunnen zien wat er gebeurt kan men een **bifurcatiediagram** maken. Op de x-as zetten we de parameter a en op de y as tekenen we de waarden die bereikt worden door de rij, na de stabilisatiefase. We bekijken het volgende programma.



In dit programma gaan we het grafisch scherm leegmaken, daarna beginnen we een lus voor A die langsheen de x-as zal variëren. We nemen de beginwaarde 0.3 en rekenen dan de eerste 15 termen van de rij uit. Vanaf de 5de term, tekenen we ook het overeenstemmende punt.

Om het programma te gebruiken stel je eerst via `window` het venster in, daarna roep je het programma aan. De volledige grafiek maken duurt wel een tijdje, de **TI-84+** moet immers telkens weer een aantal termen van de rij berekenen. Het resultaat is het volgende.

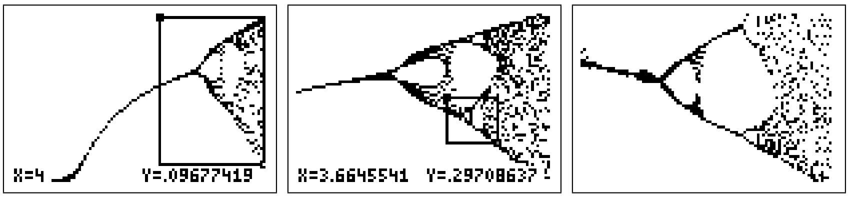


We zien dat voor kleine waarden van a de populatie uitsterft. Vanaf een kritieke waarde zal er convergentie zijn naar een stabiele populatie. Na opnieuw een kritieke waarde bekom je adherentie aan twee waarden. Daarna zal de grafiek zich steeds fijner opsplitsen, om tenslotte chaotisch te worden. Binnenin het chaotisch gebied zijn er toch weer gestructureerde delen.

Opdracht 33. Bepaal de kritieke waarden van a en duid de structuur binnen het chaotisch gedeelte aan.

De chaos die je hier terugvindt wordt binnen de wiskunde **deterministische chaos** genoemd. Het gaat hier om chaos die ontstaat uit een duidelijk berekenbaar deterministisch systeem, niet als resultaat van een stochastisch proces.

Je kan met `zoom` `[zoombox]` in de figuur inzoomen. Het venster wordt dan wel aangepast, maar je moet zelf wel opnieuw het programma starten. Let wel hoe meer detail je wenst, hoe meer termen in de rij je gaat moeten uitrekenen (zeker in het chaotisch gedeelte) en hoe langer de stabilisatiefase. Je zal dus geduld moeten hebben en de grenzen in het programma moeten aanpassen. Neem bijvoorbeeld `for(I,1,30)` en `if I>15:pt-on(A,X)`. Het programma zal wel veel meer tijd nodig hebben, maar het bifurcatiediagram zal veel nauwkeuriger worden.



Opdracht 34. Zoom in op verschillende delen. Illustreer het fractaal karakter, laat het chaotische deel en haar structuur zien, noteer telkens de waarden van a en geef een beschrijving.

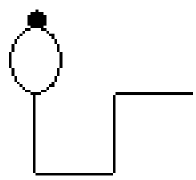
Hoofdstuk 4

De schildpad en de draak

4.1 Turtle graphics: een eigen programmeertaal maken!

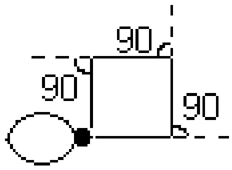
Vooraleer we zelf fractalen gaan maken zullen we eerst zelf een kleine programmeertaal ontwikkelen om grafieken mee te maken.

We beschouwen een schildpad in een veld. Het diertje verplaatst zich vooruit over een welbepaalde lengte. Tussen twee verplaatsingen kan ze van richting veranderen, ze kan zich draaien over een bepaalde hoek. Op die manier laat ze een spoor achter.



Je kan de schildpad nu “programmeren” om een bepaalde tekening te maken door te zeggen wanneer ze vooruit moet en wanneer ze moet draaien en over welke hoek. We spreken af dat 0 wil zeggen dat ze 1 m vooruit moet gaan en dat 90 wil zeggen dat ze over 90 graden moet draaien. We kunnen nu een vierkant tekenen door de schildpad volgende opdracht te geven.

$$0, 90, 0, 90, 0, 90, 0$$



De afspraken die we hebben gemaakt leggen een eenvoudige programmeertaal vast, het programma om een vierkant te tekenen bestaat uit de bovenstaande opdrachten.

Opdracht 35. Wat is het programma om een gelijkzijdige driehoek te maken? En een rechthoek met lengte 3 m en breedte 2 m?

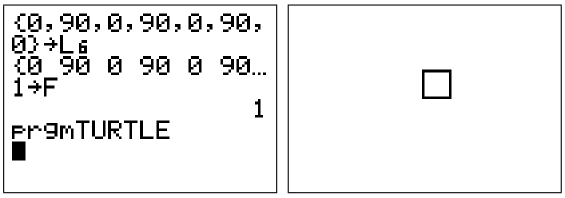
We zullen deze programmeertaal implementeren op de **TI-84+** . We gaan ervan uit dat de schildpad altijd in $(0, 0)$ start, met haar neus in de positieve x-richting. De stap die we telkens zetten zal in de veranderlijke F staan. Het programma is een lijst getallen, we zullen hiervoor de lijst L_6 gebruiken, zodat de andere lijsten vrij blijven. De richting waar de schildpad naartoe gaat wordt bijgehouden in de variabele T (de hoek gemeten vanaf de positieve x-as). De positie van de schildpad wordt bijgehouden in (A, B). Het programma ziet er als volgt uit.

```
PROGRAM:TURTLE
:ClrDraw:AxesOff
:Degree
:0→A:0→B:0→T
:For(I,1,dim(L6)
)
:If L6(I)=0
:Then
```

```
PROGRAM:TURTLE
:A→C:B→D
:A+Fcos(T)→A
:B+Fsin(T)→B
:Line(C,D,A,B)
:Else
:T+L6(I)→T
:End:End
```

Nadat we het scherm hebben leeggemaakt, schakelen we de **TI-84+** om naar graden. De startpositie is de oorsprong en de schildpad is naar de positieve x-as gericht. Nu wordt het schildpad-programma uit lijst L_6 gelezen. Als in L_6 een nul staat wordt de nieuwe positie berekend d.m.v. de stap F en de hoek T en wordt een lijnstukje getekend. Als L_6 een hoek bevat wordt de nieuwe richting in T opgeslagen.

Als we bovenstaand schildpad-programma willen uitvoeren om een vierkant te tekenen, steken we het in L_6 en leggen de afstand F vast. Daarna roepen we onze programmeertaal op. Je moet natuurlijk wel rekening houden met de grenzen van het grafisch venster.



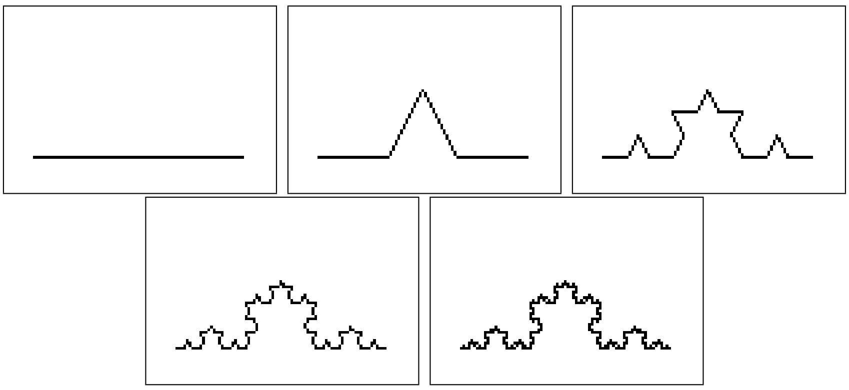
Opdracht 36. Gebruik je schildpad-programma om een gelijkzijdige driehoek en een rechthoek te tekenen.

Opdracht 37. Met welk schildpad-programma kan je een rechthoekige driehoek tekenen? Welke stelling(en) heb je hiervoor nodig?

Opdracht 38. Schrijf een schildpad-programma om een mannetje te tekenen.

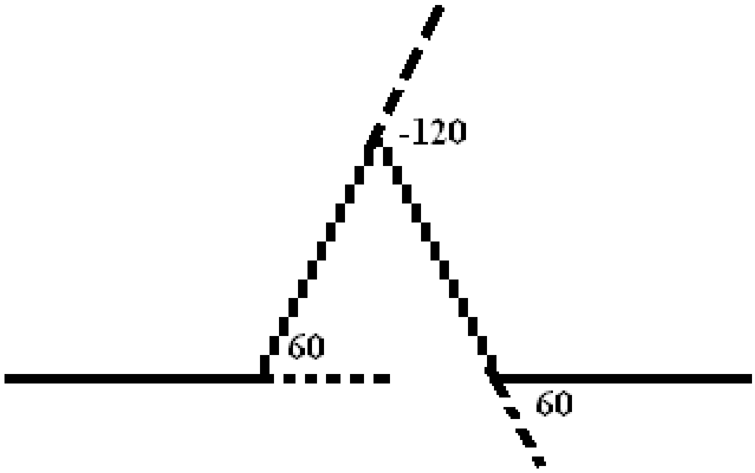
4.2 De Koch kromme

We zullen nu onze schildpad-programmeertaal gebruiken om fractalen te maken. Dit gaat als volgt. We vertrekken van een lijnstukje. Dit wordt vervangen door een bepaald patroon. Elk lijnstukje uit dit patroon wordt opnieuw vervangen door een verkleinde versie van dit patroon. Dit procédé blijft men herhalen. De uiteindelijke figuur is een fractaal.



De bovenstaande fractaal is de *Koch kromme*, genoemd naar de Zweedse wiskundige Niels Helge von Koch (1870-1924). De fractaal bestaat uit vier kopies van zichzelf, telkens verkleind. Het herhaalde patroon is gegeven door

$$0, 60, 0, -120, 0, 60, 0$$



Het programma om zulke fractalen te maken ziet er als volgt uit.

```
PROGRAM:TURTLEFR
:{0,60,0,-120,0,
60,0}→L1
:{0}→L2
:1/3→H
:1→F:L2→L6
:
:Lbl 1
```

```
PROGRAM:TURTLEFR
:prgmTURTLE
:Input
:1→K
:For(I,1,dim(L2)
)
:If L2(I)=0
:Then
```

```
PROGRAM:TURTLEFR
:For(J,1,dim(L1)
)
:L1(J)→L6(K)
:K+1→K
:End
:
:Else
```

```
PROGRAM:TURTLEFR
:L2(I)→L6(K)
:K+1→K
:End
:End
:
:L6→L2:F*H→F
:Goto 1
```

Eerst wordt het patroon opgeslagen in L_1, het startlijnstuk komt in L_2. Let erop dat de schildpad na het patroon te hebben doorlopen opnieuw met de neus in de positieve x-richting wijst! De schaalfactor H is in dit geval $\frac{1}{3}$. De staplengte F is 1, het beginprogramma voor de schildpad is het beginlijnstukje en komt in L_6. Daarna wordt een lus gestart. `prgmTURTLE` wordt aangeroepen om de eerste tekening te maken, met `prgm` `[io]` `[input]` wordt gewacht tot de gebruiker op `enter` drukt voor de berekening van de volgende stap. Hierna volgt een `for`-lus, voor elk element uit L_2 wordt nagegaan of dit een lijnstukje is (0) of een hoek. Als het een lijnstukje is, wordt d.m.v. een andere `for`-lus het patroon uit L_1 naar het nieuwe programma

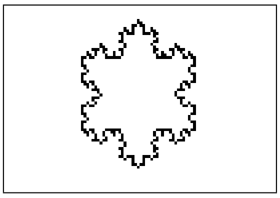
L_6 gekopiëerd. Als het een hoek is wordt die gewoon overgenomen. Met K hou je de lengte van het nieuwe programma bij. Uiteindelijk staat in L_6 het programma voor de volgende stap. Omdat dit het begin is van een volgende herhaling maken we een kopie ervan in L_2, daarna wordt de nieuwe staplengte gegeven en wordt de lus gesloten. De nieuwe tekening zal gemaakt worden en de volgende zal worden berekend. Het programma zal doorlopen tot je het met `on` onderbreekt of tot het geheugen vol is (lijsten kunnen maar 999 elementen bevatten). Let wel dat je vooraf een goed venster moet instellen met `window`, volgende grenzen zijn altijd wel een goed begin.

```
WINDOW
 Xmin=-.1
 Xmax=1.1
 Xscl=1
 Ymin=-.1
 Ymax=1.1
 Yscl=1
 Xres=1
```

Opdracht 39. Bepaal de dimensie van de Koch kromme.

Opdracht 40. Als je niet begint met een lijnstuk maar met volgende figuur krijg je een Koch sneeuwvlok of Koch eiland.

$$120, 0, -120, 0, -120, 0$$



Toon aan (d.m.v. driehoeken en meetkundige rijen) dat de omtrek van deze figuur oneindig is, maar dat de oppervlakte eindig is.

Opdracht 41. Welke fractaal bekom je als je vertrekt van een lijnstuk en volgend patroon gebruikt met een schaalfactor $\frac{1}{2}$?

$$0, 120, 0, -120, 0, -120, 0, 120, 0$$

Als je naar dit patroon kijkt wordt deze vervangen door vijf kopies van zichzelf. Wat is dan de dimensie? Wat kun je dus besluiten voor fractalen die na een aantal stappen zichzelf gaan overlappen?

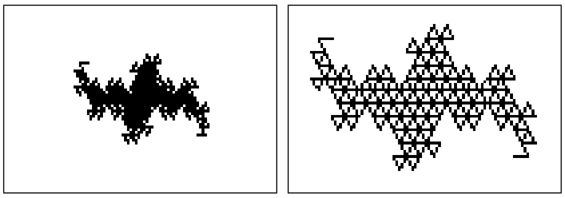
Opdracht 42. Maak zelf een aantal mooie fractalen, en geef de verschillende parameters.

4.3 Drakenkrommen

We beschouwen nu opnieuw een andere fractaal. Deze begint met een lijnstukje en het patroon is gegeven als volgt.

$$0, 120, 0, -120, 0$$

De schaalfactor is $\frac{1}{\sqrt{3}}$. De fractaal ziet er als volgt uit. (Laat je programma draaien, eenmaal je genoeg iteraties (=herhalingen) hebt stop je en gebruik je `zoom` `[zoombox]` om te vergroten. Laat nu opnieuw het programma draaien.)

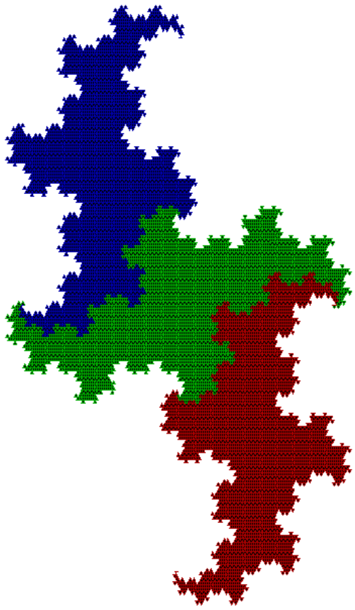


Dit is de *terdragon* fractaal. Als je tijdens het tekenen goed volgt zie dat hij niet overlapt, hij raakt sommige punten wel meerdere keren aan, maar doorloopt nooit een volledig lijnstuk.

Opdracht 43. Wat is de dimensie van de terdragon fractaal? Wat betekent dit? Met welke definitie van fractaal strookt dit niet?

Opdracht 44. De terdragon fractaal is een lijn die zo kronkelig is dat ze een gehele oppervlakte kan vullen. Zoek nog enkele van deze *vlakvullende krommen*.

De terdragon fractaal heeft nog een andere eigenschap. Kijk naar volgende afbeelding.



Je ziet dat drie terdragons samen opnieuw een terdragon vormen (vandaar de naam *ter*dragon). Je kan op deze manier het gehele vlak betegelen met terdragon-vormige tegels.

Opdracht 45. In de wiskunde bestaan er vele betegelingen van het vlak. Zoek er eens een paar op.

Opdracht 46. Je kan het vlak betegelen met regelmatige veelhoeken die telkens in een hoekpunt samenkomen, maar dit kan enkel met gelijkzijdige driehoeken, vierkanten en zeshoeken. Leg uit waarom. Zoek of maak een foto van zulke regelmatige betegelingen. Misschien kun je ze eens gaan bekijken in de metro-stations Delta en Yzer in Brussel.

Opdracht 47. De terdragon geeft een fractale betegeling die beschreven kan worden met ons programma. Maar er bestaan nog vele andere drakenfractalen, zoals bijvoorbeeld de *twindragon*. Ga na wat het verschil en wat de gelijkenis is tussen deze twee drakenfractalen.

Hoofdstuk 5

Julia fractalen

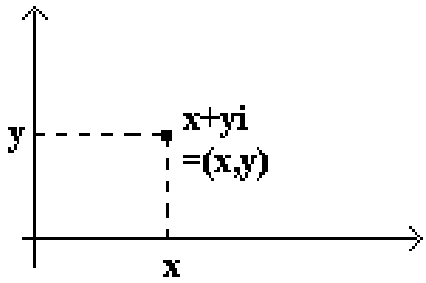
5.1 Iteratieve methode

We zullen nu de bekendste en mooiste fractalen gaan onderzoeken: de *Julia fractalen*. Deze werden beschreven door de Franse wiskundige Gaston Julia (1893-1978). Maar het is pas met het opkomen van de computers en het werk van Benoit Mandelbrot (1924-nu) dat men de schoonheid van fractalen heeft kunnen zien. Sindsdien duiken ze regelmatig op als versiering voor boeken, affiches, enz. ...

Deze fractalen steunen op berekeningen met complexe getallen. Herinner je dat de vergelijking $x^2 = -1$ geen oplossingen heeft in $\mathbb{R}$ omdat $\sqrt{-1}$ niet bestaat. Wiskundigen hebben dus een nieuw symbool i (de imaginaire eenheid) ingevoerd waarvoor geldt dat $i^2 = -1$. Dit geeft een uitbreiding van het begrip getal. Een **complex getal** z is een getal van de vorm $x + yi$ en bezit een reëel deel (x) en een imaginair deel (yi). Met deze getallen kan je gewoon rekenen, maar wel met de regel dat $i^2 = -1$. We geven een voorbeeldje.

$$(x+yi)^2 = x^2+2xyi+(yi)^2 = x^2+2xyi+y^2i^2 = x^2+2xyi-y^2 = (x^2-y^2)+(2xy)i$$

Bovendien heeft een complex getal $x+yi$ ook een meetkundige interpretatie, het komt namelijk overeen met het punt (x, y) van het vlak.

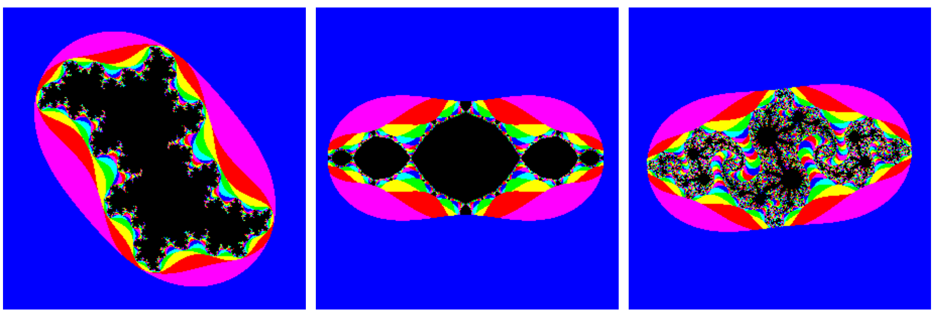


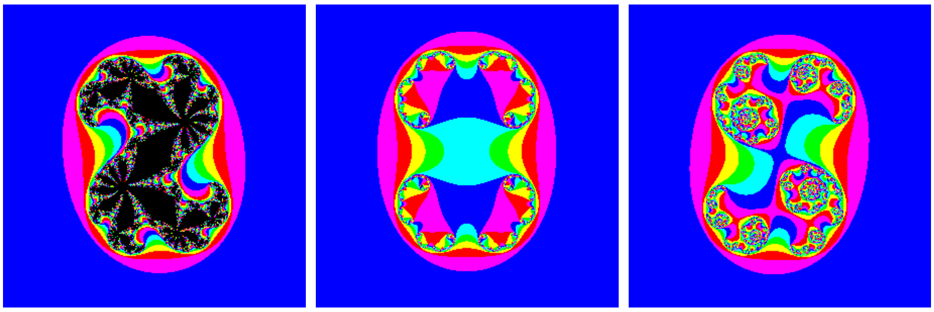
Hoe maak je nu fractalen? Kies een willekeurig complex getal z_0 als startwaarde en een vast complex getal c. Maak nu een rij complexe getallen als volgt.

$$z_n = z_{n-1}^2 + c$$

Je bekomt aldus een rij complexe getallen oftewel een rij punten in het vlak. Voor deze rij zijn er nu twee mogelijkheden. Ofwel zal na een aantal termen de rij zich verwijderen van de oorsprong, ofwel zal de rij voor eeuwig dicht bij de oorsprong blijven.

Julia fractalen komen als volgt tot stand. Neem een vast complex getal c. Alle punten van het vlak worden achtereenvolgens als startwaarde z_0 genomen voor de hierboven beschreven rij. Naargelang de snelheid waarmee de rij zich verwijdert van de oorsprong krijgt het startpunt z_0 een kleur, indien de rij zich niet verwijdert kleuren we z_0 zwart. Op deze manier krijgen alle punten van het vlak een bepaalde kleur en ontstaan er prachtige grafieken. Afhankelijk van de waarde van c krijg je zeer uiteenlopende vormen. Men moet hier opmerken dat de eigenlijke fractaal het zwarte gedeelte is, maar dat de vorm wordt vastgelegd door de randpunten van de Julia fractaal.





Merk op dat om deze grafieken te maken een computer nodig is, je moet immers voor elk punt (en voor een klein beeld van 100 pixels op 100 zijn dat er 10000!) telkens een honderdtal termen uit de rij berekenen. Dit is teveel voor een zakrekenmachine. De beeldjes hierboven kan je zelf maken met het programma op volgende website:

`http://www.kakoekelberg.be/wiskak/OCfrac/OCfrac.html`

Opdracht 48. Maak zelf een werkje over de theorie van complexe getallen. Wat houdt die in, wat zijn de belangrijkste stellingen en eigenschappen?

5.2 Backtracking methode

We hebben gezien dat de hierboven beschreven methode veel rekenwerk is, teveel voor de **TI-84+** . Gelukkig bestaat er een andere methode: *de backtracking methode*. We beschouwen opnieuw de rij

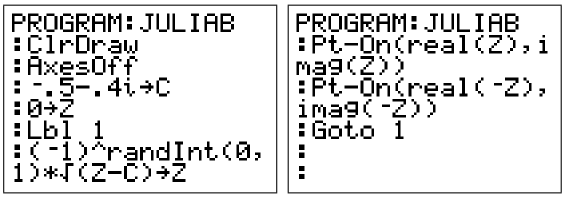
$$z_n = z_{n-1}^2 + c$$

Tot nu toe hebben we telkens we een term kenden, de volgende term uitgerekend met deze formule. Maar stel dat we een punt uit de rij kennen, dan kunnen we de vorige berekenen (dit heet *backtracking*).

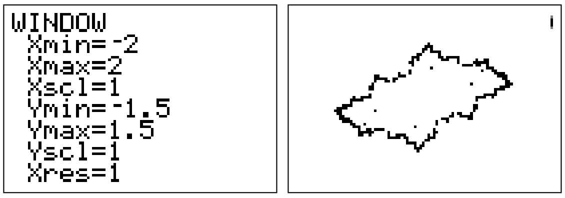
$$z_{n-1} = \pm\sqrt{z_n - c}$$

Er zijn hierbij enkele opmerkingen. Eerst moeten we de vierkantswortel trekken uit een complex getal en dit is niet zo eenvoudig, maar de **TI-84+** biedt hier de nodige hulp. Het tweede probleem is dat je niet één wortel krijgt maar twee. We zullen dus moeten kiezen welke van de twee we zullen gebruiken. Ten derde hebben we een startwaarde nodig die in de rij zit. Tenslotte hebben we dan als resultaat slechts één van de vele rijen.

Al deze problemen worden opgelost door het volgende. Men kan bewijzen dat je voor een Julia fractaal mag beginnen met een willekeurige startwaarde. Als je telkens willekeurig kiest tussen beide wortels uit de backtracking formule, dan zal de rij die je bekomt een rij punten zijn uit de rand van de fractaal. Als je dit lang genoeg doet bekom je een volledig beeld van de rand van de fractaal. Op de **TI-84+** ziet het programma er als volgt uit.



Nadat het scherm leeggemaakt is, wordt de parameter C ingesteld en wordt het startpunt $z_0 = 0$ genomen. Daarna begint de lus. Met $(-1)^{\texttt{randint}(0,1)}$ wordt telkens één van de twee voorgangers met de backtracking formule bepaald. Hierna worden toch beide wortels getekend, dit geeft immers tweemaal zoveel punten. De lus wordt dan gesloten en men gaat verder met de gekozen wortel. Het resultaat is als volgt:



Opdracht 49. Zoek op hoe je de twee vierkantswortels van een complex getal kan vinden.

Opdracht 50. Zoek zelf een aantal waarden van c waarvoor je een mooie Julia fractaal krijgt.

Opdracht 51. De Julia fractalen zijn punt-symmetrisch t.o.v. de oorsprong. Naargelang de aard van de parameter c kan je echter meer symmetrie hebben. Onderzoek dit.

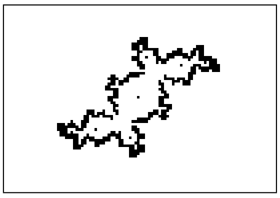
Opdracht 52. Wat is de invloed van de parameter b op de Julia fractaal met $c = a + bi$, voor een vaste waarde van a?

Opdracht 53. Wat is de invloed van de parameter a op de Julia fractaal met $c = a + bi$, voor een vaste waarde van b?

Opdracht 54. Maak de Julia fractalen voor volgende waarden van c. Verklaar hun naam.

- $c = i$ (dendriet)
- $c = -\frac{3}{4}$ (San Marco fractaal)
- $c = 1$ (stofwolk)

Opdracht 55. Volgende Julia fractaal is gekend onder de naam *konijn van Douady*, genoemd naar de Franse wiskundige Adrien Douady (1935-2006). Zoek op voor welke waarde van c men deze fractaal bekomt.



Bijlage A

Programmeren op de TI-84+

De **TI-84+** is een grafisch rekenmachine dat uitgerust is met een scherm met een grafische resolutie van 94 op 62 pixels. Bovendien bezit de **TI-84+** een eenvoudige programmeertaal, TIBasic. Dit is een dialect van BASIC, een programmeertaal waarmee Bill Gates (Microsoft) zijn faam heeft verworven. BASIC (Beginners All-purpose Symbolic Instruction Code) werd ontworpen om ook de leek in staat te stellen om kleine programma's te schrijven. Het TIBasic-dialect is trouw aan deze filosofie: er is geen enkele programmeer-ervaring nodig om programma's te schrijven voor de **TI-84+** . Enkel een beetje doorzettingsvermogen en zelfvertrouwen is nodig.

Om een programma te schrijven ga je naar `prgm`. Je kunt hier kiezen om een programma uit te voeren (`[exec]`), te veranderen (`[edit]`) of om een nieuw programma te schrijven (`[new]`). Indien je de laatste keuze maakt wordt er naar een naam gevraagd. Nadien kom je op de editor uit, waar je je programma kan invoeren.

Een programma bevat een opeenvolging van commando's die, wanneer we het programma laten draaien door de **TI-84+** , na elkaar zullen worden uitgevoerd. Elke programmeertaal bevat volgende drie concepten.

1. Variabelen. Op de **TI-84+** zijn dit `A,B,C, ...` en je kan deze een waarde toekennen d.m.v. `sto`.

   getal → *var*

2. Een als/dan-structuur:

   `if` *voorwaarde* : *commando*

Als aan de *voorwaarde* voldaan is dan zal het *commando* worden uitgevoerd, anders niet. Er bestaat ook een uitgebreidere versie waarin men een keuze maakt tussen twee groepen commando's (zie verder).

3. Een lus-structuur:

   `for(`*var, beginwaarde, eindwaarde*`)`
   commando
   `end`

   Hierin doorloopt de variabele *var* alle waarden tussen *beginwaarde* en *eindwaarde* en telkens wordt *commando* uitgevoerd. Andere varianten van lussen bestaan ook, bijvoorbeeld oneindige lussen (zie verder).

De commando's die met het programmeren te maken hebben zitten nu onder `prgm`. We geven hier een kort overzicht van de nuttigste programmeerfuncties.

prgm [ctl]	
[if]	Eerste vorm:
	`:if` *voorwaarde* : *commando*
[then]	Tweede vorm: uitgebreide `if` structuur:
[else]	`:if` *voorwaarde*
	`:then`
	: *commando's*
	`:else`
	: *commando's*
	`:end`
	de *voorwaarde* geef je in d.m.v. 2nd [test]
[for]	om lussen te maken
	`:for(` *var, beginwaarde, eindwaarde [, stapgrootte]* `)`
	: *commando's*
	`:end`
[end]	om bovenstaande blokken te eindigen
[Lbl]	om een punt in het programma te markeren (Label)
[Goto]	om te gaan naar een bepaald Label
	We gebruiken dit om en oneindige lus te maken:
	`:Lbl 1`
	: *commando's*
	`:Goto 1`
	De oneindige lus moet je zelf stopzetten door on te drukken.

prgm [i/o]	
[disp]	om een waarde/string op het scherm te printen
[prompt]	om de waarde van een variabele te vragen aan de gebruiker
[input]	om een text op het scherm te tonen en een waarde te vragen
	aan de gebruiker, deze kan ook een functie zijn.
	`:input` *"text", variabele* (bijv. $"Functie :", y_1$)
	ander gebruik (zonder argumenten):
	`:input`
	de gebruiker krijgt een cursor op het grafisch scherm
	met de pijltjestoetsen kan je een punt kiezen
	met enter worden de coördinaten opgeslagen in X en Y
	daarna gaat het programma verder.

Naast deze twee menu's uit prgm kunnen alle commando's uit andere menu's gebruikt worden. Hier volgen degene die we het meest zullen gebruiken.

`2nd` [draw] [draw]	
[clrdraw]	maakt het grafisch scherm leeg
[line]	tekent een lijnstuk tussen twee punten (A, B) en (C, D)
	:line(A,B,C,D)

`2nd` [draw] [points]	
[pt-on]	tekent een punt
[pxl-on]	tekent een pixel in absolute coördinaten.

Soms is het handiger om geen assen te hebben op het grafisch scherm, je kan deze uitzetten d.m.v. `2nd` [format] [axesoff].

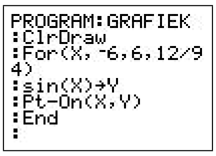
Nu we de nodige commando's kennen kunnen we een zeer eenvoudig voorbeeld behandelen, dat inzicht geeft over hoe de **TI-84+** grafieken maakt.

Gebruik een `for`-lus om een programma te schrijven dat de grafiek van een functie (bijvoorbeeld de sinusfunctie) maakt d.m.v. het volgend algoritme.

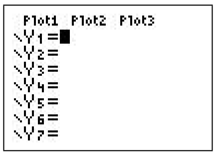
```
voor x gaande van -6 tot 6 met een stap van 1 pixel
bereken y=sin(x)
teken het punt (x,y)
sluit de lus
```

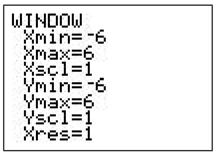
Gebruik de grafische commando's `2nd` [draw] [clrdraw] (om een leeg scherm te krijgen) en `2nd` [draw] [point] [pt-on] om een punt te tekenen. Vergeet niet van via `y=` alle functies weg te halen en om met `window` de grenzen aan te passen!

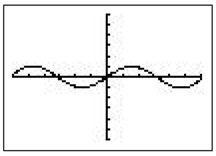
Het programma telt exact vijf regels:



We tekenen hier de functie op het interval $[-6, 6]$. Dit interval heeft lengte 12, het grafisch beeldscherm telt 94 pixels, de stapgrootte tussen twee pixels is dus 12/94. We hadden ook gewoon een voldoende kleine stap kunnen kiezen (bijvoorbeeld 0.1). Het resultaat van dit programma is hetzelfde als hetgeen we zouden verkrijgen door de ingebouwde functies te gebruiken.







Deze enkele commando's en menu's vormen de basis van hetgeen je kan gebruiken om op de **TI-84+** te programmeren, er is natuurlijk veel meer mogelijk. Een aantal programma's in deze tekst zijn wat langer, ze worden d.m.v. verschillende schermen getoond, deze moeten gelezen worden als een stripverhaal: van links naar rechts en van boven naar beneden. Je kan nu de programma's invoeren en naar eigen goeddunken veranderen.

Bijlage B

Referenties

Tenslotte vermelden we nog dat er op het internet tal van sites en programma's omtrent fractalen te vinden zijn. Volgende sites zijn ideaal als vertrekpunt:

`http://en.wikipedia.org/wiki/Fractal`
`http://en.wikipedia.org/wiki/List_of_fractals_by_Hausdorff_dimension`
`http://mathworld.wolfram.com/Fractal.html`

Een programma om de kleurrijke Julia-fractalen in het laatste hoofdstuk te maken, werd gemakshalve op een afzonderlijke website geplaatst:

`http://www.kakoekelberg.be/wiskak/OCfrac/OCfrac.html`

Verdere aanbevolen lectuur zijn de boeken over dit onderwerp van Hans Lauwerier en Robert L. Devaney:

Een wereld van fractals, Hans Lauwerier, Aramith, ISBN: 90-6834-076-X
Oneindigheid : een onbereikbaar ideaal, Hans Lauwerier, Aramith, ISBN: 90-6834-055-7
Chaos, fractals en dynamica : computer-experimenten in de wiskunde, Robert L. Devaney, Addison Wesley, ISBN: 90-6789-334-X

Bijlage C

Oplossingen van de opdrachten

Er worden hier beknopte oplossingen van een aantal van de opgaven uit deze bundel gegeven, dit slechts ter informatie voor de leerkracht. Er wordt aangeraden om de opdrachten zelf op te lossen. Alle informatie die hiervoor nodig is werd gehaald uit de geciteerde bronnen.

2. Alle figuren vertonen een zekere zelfgelijkvormigheid.
4. Bomen, varens, sommige rotsen, wolken, enz. ...
5. De middenloodlijnen op de zijden van een rechthoek verdelen die rechthoek in 4 kopies van zichzelf, verkleind met een factor $\frac{1}{2}$. De dimensie is dus $\frac{\ln 4}{\ln 2} = 2$. Een analoge verdeling van de balk in acht geeft dimensie $\frac{\ln 8}{\ln 2} = 3$.
6. Door de middens van de zijden te verbinden bekomen we een verdeling van een gelijkzijdige driehoek, die bestaat uit vier kopies, verkleind met een factor $\frac{1}{2}$. De dimensie is dus $\frac{\ln 4}{\ln 2} = 2$.
7. $\frac{\ln 8}{\ln 3} = 1.8927\ldots$
8. Mengerspons: $\frac{\ln 20}{\ln 3} = 2.7268\ldots$
Sierpinski-tetraëder: $\frac{\ln 4}{\ln 3} = 2$
Sierpinsky-piramide: $\frac{\ln 5}{\ln 2} = 1.4649\ldots$
12. Het getal op de $n$de rij, $k$de kolom is C_n^k.
13. $C_n^k = C_{n-1}^{k-1} + C_{n-1}^k$.
14. $(x+y)^n = \sum_{k=0}^{n} C_n^k x^{n-k} y^k, n \in \mathbb{N}$

17. & 18. De zeef van Sierpinski komt tevoorschijn.

19. Je bekomt een andere figuur, maar het is ook een fractaal.

20. De **TI-84+** kan zulke combinaties niet meer exact uitrekenen, de getallen worden te groot en er treden dus afrondingsfouten op.

22. Er wordt hier gebruik gemaakt van $C_n^k = C_n^{n-k}$ om slechts de helft van de berekeningen te maken.

23. $\frac{\ln 6}{\ln 3} = 1.6309\ldots$

25. Er treedt overlapping op tussen de verschillende driehoekjes, de figuur valt dus niet meer op eenvoudige wijze op te splitsen in niet overlappende delen.

26. Het stijgen of dalen van de koers kan gezien worden als een binomiaal proces X (met gelijke kansen op stijgen en dalen, n keer herhalen), de verwachtingswaarde ($E[X] = .5n$) geeft aan dat je gemiddeld evenveel zal stijgen als dalen. De situatie die men dus het meest zal waarnemen is een status quo.

29. In dit geval kan de dronkenman ook een stap diagonaal zetten, dit maakt het programma sneller omdat er minder `if`-structuren moeten nagegaan worden.

31. Convergentie, adherentie aan 2,3,4,... waarden en totale chaos.

32. Na een stabilisatie-fase kom je meestal op hetzelfde gedrag uit, onafhankelijk van de beginwaarden. Dit gebeurt bijna altijd maar er zijn uitzonderingen, bijvoorbeeld de startwaarden 0 of 1.

36. driehoek: $0, 120, 0, 120, 0$
    rechthoek: $0, 0, 0, 90, 0, 0, 90, 0, 0, 0, 90, 0, 0$

37. Enkel rechthoekige driehoeken waarbij de lengte van de zijden Pythagorische tripels zijn zullen getekend kunnen worden. Bovendien zal men minstens een van de scherpe hoeken moeten berekenen.

39. $\frac{\ln 4}{\ln 3} = 1.2619\ldots$

40. De fractaal zal de zeef van Sierpinsky zijn, maar omdat de delen elkaar overlappen kan je de eenvoudige formule voor de dimensie niet gebruiken.

43. De dimensie $\frac{\ln 3}{\ln \sqrt{3}} = 2$ is die van het vlak, men bekomt geen kommagetal en dus geen echte fractale dimensie.

44. O.a. de Peanokromme.

49. Men kan algemeen $n$de wortels trekken uit een complex getal door over te gaan op de goniometrische vorm. Voor vierkantswortels bestaan er ook eenvoudigere methoden. Als $(x + iy)^2 = a + bi$ dan worden de wortels van $a + bi$ gegeven door de oplossing van het stelsel

$$\begin{cases} x^2 - y^2 = a \\ 2xy = b \end{cases}$$

Dit stelsel oplossen geeft de formule:

$$\pm\frac{\sqrt{2}}{2}\left(\sqrt{\sqrt{a^2+b^2}+a} + i\mathrm{sign}(b)\sqrt{\sqrt{a^2+b^2}-a}\right)$$

51. Als de parameter c zuiver reeël is, dan is de fractaal ook symmetrisch ten opzichte van de x-as en de y-as.

52. De fractaal zal afhangend van b een "draaiing" ondergaan.

53. De fractaal zal afhangend van b een grotere of kleinere oppervlakte hebben. Ook de samenhang zal veranderen.

54. Dendriet (van het Griekse dendros: boom) betekent vertakking of uitloper. Deze fractaal ontstaat door verschillende vertakkingen. Wanneer een vloeistof doorheen een gesteente sijpelt bekomt men zulke fractalen.
De San Marco fractaal dankt zijn naam aan zijn vorm, die gelijkenis vertoont met het beeld van het San Marco plein in Venetië wanneer de gebouwen weerspiegeld worden in het water.
De fractaal is een totaal onsamenhangende verzameling punten, vergelijkbaar met een stofwolk.

55. $c = -0.123 + 0.745i$

Notities

………

………

………

………

………

………

………

………

………

………

………

………

………

………

………

………

………

………

………

………

………

………

………

………

………

………

………

………

Dit cahier gaat over het verwezenlijken van de onderzoekscompetenties wiskunde/wetenschappen met behulp van de TI-84+.

Als leidraad werd hiervoor het onderwerp fractalen gekozen. De tekst is opgedeeld in een aantal hoofdstukken waarvan het eerste hoofdstuk de nodige theorie en begrippen bevat. Daarna kan, naargelang de klassituatie, gekozen worden uit de volgende hoofdstukken waarin telkens een ander aspect van fractalen wordt uitgewerkt. Dit laat toe om zowel groepswerkjes te organiseren als klassikaal de thema's te behandelen.

DIDIER DESES is leerkracht wiskunde aan het Koninklijk Atheneum Koekelberg en geeft les aan de Wetenschappelijke (5u wisk/week) en de Latijnse richtingen (3u wisk/week). Hij is tevens wetenschappelijk medewerker aan de Vrije Universiteit Brussel.

Juni 2008