Henk Pfaltzgraff

Programmeren met de TI-83 Plus

Wiskunde met een extra dimensie

CTL I/O EXEC
1:If
2:Then
3:Else
4:For(
5:While
6:Repeat
7↓End

CTL I/O EXEC
1:Input
2:Prompt
3:Disp
4:DispGraph
5:DispTable
6:Output(
7↓getKey

Texas Instruments

Programmeren met de TI-83 Plus

Henk Pfaltzgraff

Voorwoord

Dit boekje wil op een vlotte manier het programmeren introduceren, meer bepaald het programmeren op een TI-83 Plus. Deze rekenmachine wordt heel wat gebruikt in de dagelijkse klassituatie.

Het programmeren en het gebruik van programma's geeft aan de TI-83 Plus een extra dimensie. Naast het gewone werk kun je met een programma het slaafse rekenwerk automatiseren waardoor er meer tijd vrijkomt voor het echte werk. En bovendien is het uitvoeren van een controle naar de juistheid van resultaten met een programma een fluitje van een cent.

Het programmeren heeft als voordeel dat het het probleemoplossend en algoritmiserend denkvermogen stimuleert. Om op een efficiënte manier een probleem te vertalen naar een programmeertaal is meer dan ooit voldoende inzicht in de materie nodig.

Koen Stulens
Texas Instruments

Inhoud

1 Inleiding

In de jaren '70 verscheen de eerste programmeerbare zakrekenmachine - kostprijs 1000 gulden. Dat kun je vergelijken met ongeveer 1200 euro nu. Zij was twee keer zo dik als de huidige programmeerbare rekenmachines en had slechts 20 geheugenplaatsen. Zij had geen alpha-gedeelte (voor de letters) en geen grafische mogelijkheden. Je kon haar programmeren met een beperkt aantal instructies. Het programma werd opgeslagen op een magnetisch kaartje dat, als een soort pinpasje, in en uit de rekenmachine geschoven kon worden.

Het heeft 25 jaar geduurd voor een betaalbare en hanteerbare programmeerbare rekenmachine op de Europese markt verscheen, nl. de grafische rekenmachine. Het grafische aspect valt direct op. Maar plaatjes (zoals de wiskundige ze smalend noemt) zijn geen wezenlijk element van de wiskunde. Nee, de grootste kracht van deze nieuwe rekenmachines, als de TI-83 Plus, is niet direct zichtbaar.

Druk op PRGM <NEW> ENTER A ENTER. Je hebt nu een nieuw programma met de naam "A" geopend. Druk weer op PRGM en er verschijnen onder CTL (*Control*) en onder I/O (*Input/Output*) een groot aantal opdrachten (*statements*) die je kunt gebruiken om een programma te bouwen. Daar gaat dit boekje over. Ga eerst met 2nd[QUIT] uit het lege programma, dat we nog even verwijderen met: 2nd[MEM] 2:Mem Mgmt/Del 7:Prgm. Zet de cursor voor de programmanaam A, druk DEL en 2:yes. Verlaat het MEMORY-menu weer met 2nd[QUIT].

De programmeertaal die Texas Instruments gebruikt lijkt sterk op de hogere en gebruiksvriendelijke programmeertaal BASIC en heet daarom TI-Basic. Laten we een paar TI-Basic-opdrachten eens nader bekijken.

- De *toekenningsopdracht*, die aangegeven wordt met een pijl →, krijg je met de toets [STO▸] . De waarde van wat links van de pijl staat wordt toegekend aan de letter die rechts staat. Zo betekent 3→X dat X de waarde 3 krijgt. En A+3→A wil zeggen dat de oude waarde van A plus 3 toegekend wordt aan A als nieuwe waarde. Korter gezegd: A wordt 3 groter.
- De *voorwaardelijke opdracht* If Then wordt alleen uitgevoerd als de voorwaarde achter If juist is.
- De *lus* For(*variabele, beginwaarde, eindwaarde, stapgrootte*) geeft je de kans om een aantal opdrachten te herhalen.

  For(A,1,7,2): *opdrachten* : End

  betekent dat *opdrachten* telkens uitgevoerd worden terwijl A de waarden 1, 3, 5 en 7 aanneemt.
- Input B of Prompt B betekent dat het programma wacht op de invoer van een waarde die aan B toegekend zal worden. Disp B toont de waarde van B op het scherm. Disp "tekst" schrijft "tekst" op het scherm. Tekst moet tussen aanhalingstekens staan.

Met dit gereedschap kun je het apparaat naar je hand zetten om problemen voor je op te lossen. De volgende twee voorbeelden hoef je niet uit te voeren, ze zijn bedoeld om je een beetje vertrouwd te maken met de klank van de taal. Leg je TI-83 Plus maar even opzij.

Karl Gauss (van de "Gauss kromme" – 1777-1855) zag, als zevenjarige al, dat de som: 1 + 2 + 3 + … + 99 + 100 geschreven kan worden als:

(1+100)+(2+99)+…+(50+51) = 101+101+…+101 = 50×101 = 50×100+50×1=5050.

Zo geniaal hoeven wij niet te zijn. Met een programma kan het ook!

```
Ø→S
For(X,1,1ØØ)
S+X→S
End
Disp S
```

Eerst krijgt S de waarde nul. (S van *SOM*). Voor X=1 wordt S+X de nieuwe S, dus S=Ø+1=1. Dan wordt X=2 en de nieuwe S wordt weer S+X, dus 1+2=3. Voor X=3 wordt de nieuwe S weer S+X m.a.w. 3+3=6, voor X=4 wordt de nieuwe S opnieuw S+X nl. 6+4=1Ø, ... Zodra X op 101 komt, verlaten we de For-lus en na End wordt S afgedrukt. Dat het ook kan met `sum(seq(X,X,1,1ØØ))` doet even niet ter zake.

Weet je wat een priemgetal is? Elk (positief) geheel getal heeft minstens twee delers, namelijk 1 en het getal zelf. Een priemgetal heeft alleen deze twee delers. Een priemgetal wordt ook wel eens een beetje slordig "ondeelbaar" genoemd.

Hieronder staat een programma dat vanaf een gegeven begingetal, N, de zes volgende priemgetallen berekent. Bekijk het even, zonder het in te toetsen. De prompt (of input) op regel 1 wacht op de invoer van het startgetal dat we N noemen. We nemen aan dat je voor N een geheel getal groter dan 2 invoert. Als N even is (deelbaar door 2) en dus geen priemgetal, wordt direct doorgeschakeld naar N+1.

```
Prompt N
Ø→I
If fPart(N/2)=Ø:N+1→N
Lbl A
If I=6:Stop
For(D,3,√(N),2)
If fPart(N/D)=Ø:Goto B
End
Disp N:I+1→I
Lbl B
N+2→N
Goto A
```

I telt het aantal priemgetallen.

Label A start met na te gaan hoeveel priemgetallen al gevonden zijn. De For-lus onderzoekt of er echte delers, D, zijn - beginnend met 3,dan 5, 7, 9, ... tot aan √(N).

Zodra een deler gevonden is (`fractionPart`=0 betekent dat de deling "opgaat") springen we naar label B om vervolgens N met 2 te verhogen. Als er geen deler gevonden is (na End van de For-lus) wordt N (als priemgetal) op het scherm gezet, N eveneens met 2 verhoogd en de teller I met 1. Terug naar label A en het spel begint overnieuw.

Programmeren is intellectuele creativiteit van een hoog niveau. Vindingrijkheid en systematiek spelen er een grote rol bij. De leerling kan zijn programma's gebruiken als probleemoplosser en automaat bij het huiswerk, de praktische opdracht, als verdieping van de wiskunde of als hobby. Gewoon, voor zijn of haar plezier.

2 Zelf een programma schrijven

Dit boekje gaat over in TI-Basic geschreven programmaatjes. We bekijken twee soorten:

(1) programma's om iets voor je uit te rekenen of op te lossen
(2) programma's om een experiment na te bootsen (simulaties), meestal een kansexperiment.

Als je je TI-83 Plus zelf programmeert, kun je maatwerk leveren. In de handleiding van de TI-83 Plus, hoofdstuk 16 (education.ti.com/guides), staat een volledig overzicht van alle instructies (statements) die je daarbij kunt gebruiken. Een stuk makkelijker maar lang niet zo leuk is het om gebruik te maken van kant en klare (vóórgeprogrammeerde) programma's, die je bijvoorbeeld van internet kunt halen (downloaden). Daarvoor moet je een verbinding maken tussen je rekenmachine en een computer, met een TI-GRAPH LINK™-kabel. Zie bv. www.henkshoekje.com voor kant en klare programma's.

Maar nu eerst een paar zelf te schrijven programma's om kennis te maken. Gebruik ◄ in combinatie met DEL om typfouten te wissen en voeg tekens en regels in met 2nd[INS]. Kijk telkens goed wat er op het scherm staat. Met ENTER voer je een programmaregel of een waarde in en met CLEAR veeg je een programmaregel uit.
Met 2nd[ALPHA] schakel je over op de (groen aangegeven) hoofdletters.

Doe PRGM<NEW> 1:Create New en tik dan bijvoorbeeld PRONAAM als naam voor het programma, gevolgd door ENTER. Gebruik maximaal 8 letters voor programmanamen.

Speciale programmeer instructies zoals If, Then, End, Input, Disp, Pause en Output vind je tijdens het programmeren onder de knop PRGM.

De symbolen =, ≠, ≤ en ≥ staan onder 2nd[TEST] en de knop VARS wordt gebruikt bij het instellen van het scherm, voor het invoeren van functies (in Y=) en van strings (tekst).

De instructies "onder" de knoppen en de daarbij horende menu's zijn alfabetisch gerangschikt terug te vinden achter in de handleiding van je TI-83 Plus, in de *tabel van functies en instructies*. Voorin en achterin dit boekje vind je een overzicht van de belangrijkste functies. Met 2nd[QUIT] ga je uit het programma terug naar het rekenscherm en met 2nd[MEM] 2:Mem Mgmt/Del 7:Prgm verwijder je PRONAAM.

2.1 De stelling van Pythagoras

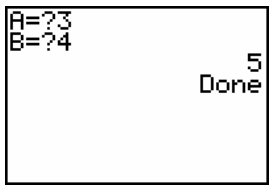
Ons eerste programma is van type (1). Je stopt er twee zijden van een rechthoekige driehoek in en het programma berekent met de stelling van Pythagoras de lengte van de derde zijde. Je kent hem nog wel: kwadrateer de rechthoekszijden, tel op en trek de wortel eruit en je krijgt de langste zijde (de *hypotenusa*). In formule: $C = \sqrt{(A^2 + B^2)}$.

Daar gaan we. Doe PRGM<NEW> 1: Create New en tik HYPO + ENTER.

De cursor knippert voor de eerste programmaregel. We beginnen met het schoonvegen van het scherm: PRGM<I/O> 8:ClrHome. ENTER plaatst het commando ClrHome op de programmaregel (de afkorting <I/O> is van Input/Output).

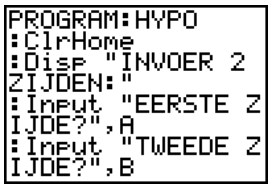
We vragen de zijden van de driehoek op met PRGM <I/O> 2:Prompt A,B en bekijken het eerste resultaat (druk na ENTER 2nd[QUIT] om het programma te verlaten).

Als het goed is, ben je nu terug in het gewone scherm. Met PRGM en het pijltje ▼ ga je omlaag tot aan de gewenste programmanaam HYPO en druk je twee keer ENTER. Achter A? typ je bijvoorbeeld 3 ENTER en achter B? 4 ENTER. Het woord Done geeft aan dat je programma gedaan is.

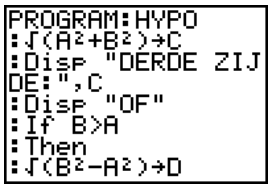


We gaan terug naar ons programmaatje met PRGM <EDIT> HYPO, zetten de cursor na twee keer ▼ op de onderste regel en voeren nu de stelling van Pythagoras in met 2nd[√] voor de wortel en [x²] voor een kwadraat. Met √(A² - B²) STO▶ C slaan we het resultaat op onder de letter C. Er is nu nog één instructie nodig voor de uitvoer van de waarde van C. Dat gaat met PRGM<I/O> 3:Disp C. We proberen het programma even uit. Eerst 2nd[QUIT] doen en daarna twee willekeurige getallen invoeren voor de rechthoeks-zijden A en B en het antwoord verschijnt.

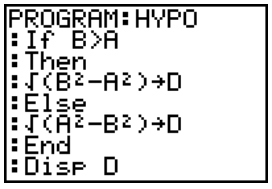
Dit was onze eerste, schuchtere, poging tot programmeren. Het programma rekende de langste zijde uit. Maar er is nog een mogelijkheid. Het zou kunnen dat niet C, maar A of B de langste zijde is van de drie. Als A bijvoorbeeld de hypotenusa is, wordt de derde zijde berekend via C=√(A² - B²).



Ga weer terug naar het programma HYPO en breng de veranderingen aan die hiernaast staan. Met INS kun je tekst invoegen, met DEL kun je een enkele letters uitvegen en met CLEAR veeg je een hele regel uit. Pas op met CLEAR! Een achteloze vinger op ▼ geeft verdriet.



Tussen aanhalingstekens is tekst aangebracht. De laatste regels bevatten de If-Then-Else-End-instructie, die waarschijnlijk geen verdere uitleg behoeft. Je kunt deze vinden onder PRGM<CTL>. We vermelden niet langer de ENTERs.



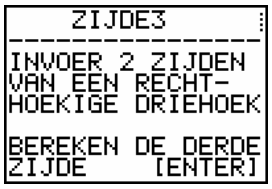
Het logische symbool > voor "groter dan" kun je, zoals eerder vermeld werd, vinden onder de knop 2nd[TEST]. En STO▶ D geeft de uitkomst van de wortel de naam D.

Hieronder nog even de *broncode* van ons programma.


```
ClrHome
Disp "INVOER 2 ZIJDEN:"
Input "EERSTE ZIJDE?",A
Input "TWEEDE ZIJDE?",B
√(A²+B²)→C
Disp "DERDE ZIJDE:",C
Disp "OF"
If B>A
Then
√(B²-A²)→D
Else
√(A²-B²)→D
End
Disp D
```

Ons programma heeft nog een zwak punt. Iemand die het onder ogen krijgt, kan namelijk niet zien waar het over gaat. Het programma heeft een *intro* nodig. Deze inleiding zetten we voorlopig even in een ander programma en plakken daar HYPO aan vast. Ons definitieve programma gaat ZIJDE3 heten. Druk PRGM<NEW> ZIJDE3 en voer de volgende programmaregels in:

```
ClrHome
Output(1,5,"ZIJDE3")
Output(2,1,"----------------")
Output(3,1,"INVOER 2 ZIJDEN")
Output(4,1,"VAN EEN RECHT-")
Output(5,1,"HOEKIGE DRIEHOEK")
Output(7,1,"BEREKEN DE DERDE")
Output(8,1,"ZIJDE    [ENTER]")
Pause
```

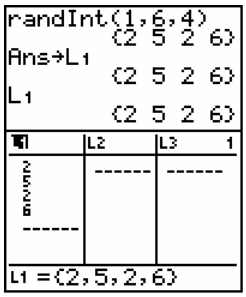


De opdracht Output(*regelnr, positienr, tekst*) is te vinden onder PRGM<I/O> 6:OutPut(. Op ons scherm is plaats voor 8 regels en 16 posities (tekens) per regel. De tekst op regel 2 bestaat uit 16 min-tekens, dat wordt straks een horizontale streep. Pause (te vinden onder PRGM<CTL> 8:Pause) wacht op ENTER.

Achter deze intro moet het programma HYPO geplakt worden. Daarvoor is de knop 2nd[RCL] (*recall* betekent oproepen) beschikbaar. Zet de cursor onder Pause en doe 2nd[RCL] PRGM<EXEC> HYPO. Tweemaal ENTER breidt het laatste programma uit met het vorige. Het programma HYPO kan verdwijnen: 2nd[MEM] 2:Mem Mgmt/Del 7:Prgm.

2.2 Muntje

We simuleren het werpen van een munstuk door gebruik te maken van een *randomgenerator*. Onder MATH<PRB> (denk aan *probability*, waarschijnlijkheid) staan enkele random-instructies. Voor ons is alleen 5:randInt van belang - syntax: randInt(*laagste,hoogste,aantal*). Random betekent willekeurig (lukraak) en Int staat staat voor integer of geheel getal.



randInt(1,6,4) produceert 4 gehele randomgetallen tussen 1 en 6. Vier worpen met een dobbelsteen dus. Probeer maar eens. Tussen accolades staat de rij uitkomsten. Deze kun je bijvoorbeeld in een lijst opslaan met STO▶. Na STO▶ 2nd[L1] (Ans→L1) wordt de rij opgeslagen in L1.

Kijk bijvoorbeeld via STAT 1:Edit... onder L1.

De lijst kun je weer uitvegen met STAT 4:ClrList 2nd[L1]. Wie maar één keer wil werpen, doet gewoon randInt(1,6).

Het werpen van een geldstuk ("kop"of "munt") kun je simuleren met randInt(Ø,1), waarbij de uitkomst Ø bijvoorbeeld staat voor "munt" en 1 voor "kop". Wie genoeg geduld heeft om een groot aantal keren met een muntje te gooien, maakt kennis met de *wet van de grote getallen*: op den duur gaat de relatieve frequentie (het aantal kop of munt gedeeld door het totale aantal worpen) steeds meer op 0,5 lijken.

We zullen een programma schrijven dat 100 keer met een munt werpt en de relatieve frequentie in een grafiek weergeeft. Eerst geven we de complete lijst met instructies, de programmalijst of broncode.

Open een nieuw programma met de naam MUNTJE via PRGM<NEW>. Geef op de eerste regel opdracht het scherm schoon te vegen met ClrHome en eventuele tekeningen te verwijderen met 2nd[DRAW] 1:ClrDraw. Je kunt beide opdrachten in één regel zetten door ze te scheiden met een dubbele punt: ClrHome:ClrDraw.

Om error-meldingen te voorkomen is het nuttig, indien er grafische output is, alle grafieken (2nd[CATALOG] FnOff) en statistische plots (2nd[STAT PLOT] 4:PlotsOff) uit te zetten

Ook moet het grafische scherm (WINDOW) ingesteld worden. Op de X-as komt straks het aantal worpen te staan (maximaal 100) en op de Y-as de relatieve frequentie (tussen 0 en 1). Ga naar VARS 1:Window… voor de schermvariabelen. De schaalvariabelen Xscl en Yscl geven de afstand tussen de streepjes (de schaalverdeling) op de assen.

MUNTJE

```
ClrHome:ClrDraw
FnOff:PlotsOff
Ø→Xmin:Ø→Ymin
1ØØ→Xmax:1→Ymax
Ø→Xscl:Ø.1→Yscl
"Ø.5"→Y₁
Text(Ø,3,"WET VAN DE GROTE AANTALLEN:")
Ø→M
For(X,1,1ØØ)
M+randInt(Ø,1)→M
Pt-On(X,M/X)
End
Text(55,68,"[ENTER]")
Text(7,82,M/1ØØ)
Pause: ClrHome
```

In de functievariabele Y₁ komt 0.5 te staan, zodat er in de grafiek een horizontale lijn op halve hoogte verschijnt. Dat doe je met "Ø.5"→Y₁.

In plaats van een intro zetten we dit keer een koptekst op het scherm. Dat wordt de kreet: "WET VAN DE GROTE AANTALLEN:". Zulke begeleidende tekst wordt kleiner afgedrukt in het tekenscherm als je in het menu 2nd[DRAW] kiest voor optie Ø:Text(, met de betekenis: Text(*regel,positie,tekst*). Het tekenscherm bevat 62 (verticaal) bij 94 (horizontaal) beeldpunten (pixels). Hierover later meer. Neem voor het moment maar even aan dat de koptekst netjes bovenaan verschijnt op de nulde regel, derde positie. Text(Ø,3,"WET VAN DE GROTE AANTALLEN").

Voor we aan het echte programma beginnen (de simulatie van 100 worpen) moet eerst nog even de teller M (die het aantal keren "munt" telt) op nul gezet worden: Ø→M. We gebruiken de teller X voor het aantal worpen.

De instructie For(*variabele,eerste,laatste*) wordt gebruikt voor het herhaaldelijk doorlopen van een lus. Zo'n For-lus moet afgesloten worden met End. For en End staan onder PRGM<CTL> 4:For(en 7:End. Onze For-lus laat X van 1 tot en met 100 lopen: For(X,1,1ØØ).

Onderwijl wordt telkens M (het aantal keer "munt") met randInt(Ø,1) verhoogd, met 0 of 1 dus. Hierbij staat de 1 voor "munt". Schrik niet van de volgende, voor programmeurs overbekende uitdrukking: M+randInt(Ø,1)→M.

Links van het pijltje staat de oude waarde van M (beginnend met M=0). Daar wordt iedere keer 0 of 1 bij opgeteld. Het resultaat M+randInt wordt opgeslagen (geteld) in de nieuwe waarde van M (rechts van het pijltje).

De tussenstand voor de relatieve frequentie M/X wordt als een punt op de grafiek getekend met Pt-On(X,M/X), 2nd[DRAW]<POINTS> 1:Pt-On(. Vóór de komma staat de x-coördinaat en achter de komma de y-coördinaat van het te tekenen punt.

De X wordt met 1 opgehoogd, tot en met X=1ØØ, waarna het programma achter End z'n weg vervolgt. Na de For-lus komt nog wat tekst op het scherm.

Op regel 55 (bijna onderaan) willen we de aanwijzing ENTER zetten en op regel 7 komt de eindfrequentie X/M. Als je het programma steeds weer wil gebruiken, met telkens iets andere resultaten, moet je aan het eind een Pause-opdracht geven samen met ClrHome. Dan kun je met twee keer ENTER iedere keer overnieuw beginnen met 100 nieuwe worpen van het muntje.

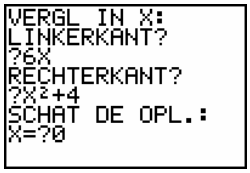
2.3 De vergelijkingoplosser

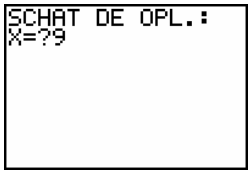
Typ het volgende programma ALOPLOS in. Rechts staat welke knoppen je moet gebruiken. Na de programmalijst volgt een toelichting.

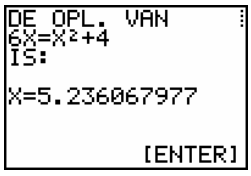
```
ClrHome                    PRGM<I/O> 8:ClrHome
Disp "VERGL IN X:
Disp "LINKERKANT?
Input Str1                 VARS 7:String 1:Str1
Str1→Y₁                    VARS<Y-VARS> 1:Y₁
Disp "RECHTERKANT?
Input Str2
Str2→Y₂

Lbl Ø                      PRGM<CTL> 9:Lbl
Disp "SCHAT DE OPL.:
Input "X=?",A              2nd[TEST] 1:=
ClrHome
Disp "DE OPL. VAN
Disp Str1+"="+Str2
Disp "IS:
Output(5,1,"X=")           PRGM<I/O> 6:Ouput
solve(Y₁-Y₂,X,A)→X         MATH Ø:solve( en
Output(5,3,X)              VARS<Y-VARS>
Output(8,1Ø,"[ENTER]
Pause                      PRGM<CTL> 8:Pause
ClrHome
Menu("----------           PRGM<CTL> C:Menu(
------","SCHAT OP          (16 streepjes)
NIEUW",Ø,"STOPPEN",1)

Lbl 1                      PRGM<CTL> 9:Lbl
ClrHome
""→Y₁:""→Y₂                VARS<Y-VARS>
```







Toelichting

Stel dat we de vergelijking 3X+1=7 willen oplossen.

`Input Str1` (regel 4) wacht op de invoer van een tekenreeks (string) die `Str1` genoemd gaat worden. Hier wordt dat de linkerkant van je vergelijking, 3X+1, wat in Y_1 wordt opgeslagen. De rechterkant van de vergelijking, `Str2`, komt in Y_2 (regel 7 en 8).

Label Ø markeert een plek in het programma. Een label is er om naar verwezen te worden, soms met een `Goto`-instructie maar in dit programma vanuit een menu. De naam van een label mag uit maximaal 2 karakters bestaan.

In regel 11 wordt een (grove) schatting van de oplossing gevraagd. Nul is meestal goed genoeg. En in regel 14 worden de twee strings verbonden door een gelijkheidsteken, met de bedoeling straks *linkerkant* = *rechterkant* oftewel $Y_1=Y_2$, de complete vergelijking, op het scherm te krijgen.

Regel 17, `solve(`Y_1`-`Y_2`,X,A)→X`, is het fundament van dit programma. Achter `solve` staat eerst de op nul herleide vergelijking, daarachter de variabele, X, en tenslotte de schatting, A. De oplossing wordt opgeslagen in X.

`Pause` wacht op `ENTER` en vier regels voor het einde haalt de `MENU`-instructie je uit het programma. De omschrijving van de `MENU`-instructie is:

`MENU("`*kopje*`","`*eerste keuze*`",`*eerste label*`,"`*tweede keuze*`",`*tweede label*`, …)`.

De laatste regel maakt Y_1 en Y_2 schoon, om daar later geen last van te hebben.

Opmerking

`2nd[MEM] 2:Mem Mgmt/Del` geeft je de mogelijkheid om (ruimtevretende) onderdelen, zoals plaatjes (`Pics`) en lijsten (`Lists`), te wissen. Wist je trouwens, dat je alle standaardinstellingen (*Defaults*) van de TI-83 Plus kunt herstellen met `2nd[MEM] 7:Reset 2:Defaults 2:Reset ?`

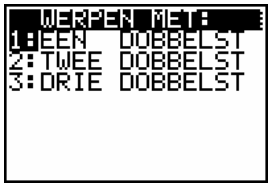
2.4 Dobbelstenen

Schrik niet van het typwerk voor het volgende programma, `DOB123`. Het valt erg mee dankzij het gebruik van de plak-knop `2nd[RCL]`.

We beginnen met wat vensterinstellingen, om dat straks niet drie keer te hoeven doen, en een menu. Maak eerst een programmaatje `MENU` als volgt.

`MENU`

```
⁻Ø.5→Xmin:Ø→Ymin:1→Xscl:Ø→Yscl
Lbl Ø
Menu("  WERPEN MET:   ",
"EEN  DOBBELST",1,
"TWEE DOBBELST",2,
"DRIE DOBBELST",3)
```

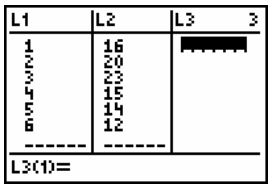


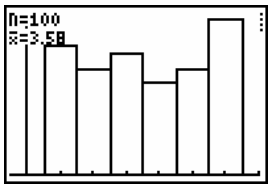
Probeer het even uit (eerst `2nd[QUIT]`). Er staan wat extra spaties in de menutekst, want het oog wil ook wat.

Nu creëren we het programma `DOB1`, dat het werpen met één dobbelsteen nabootst. De waarnemingsgetallen - 1, 2, 3, 4, 5 of 6 - komen in L_1, de frequenties in L_2. Een histogram wordt getekend en het gemiddelde $\bar{x}$ uitgerekend.

DOB1

```
ClrHome
Disp "WERPEN MET EEN"
Disp "DOBBELSTEEN:"
Disp "LEES AF IN L1 L2"
Disp "VIA STAT EDIT..."
Disp ""
Disp "HOEVEEL WORPEN:"
Input N
ClrHome
ClrAllLists                    2nd[MEM] 4:
seq(X,X,1,6)→L1                2nd[LIST]<OPS> 5:
6→dim(L2)                      2nd[LIST]<OPS> 3:
For(X,1,N)
randInt(1,6)→S
1+L2(S)→L2(S)
End
ClrHome
7→Xmax:1+max(L2)→Ymax          MATH<NUM> 7:
Plot1(Histogram,L1,L2)         2nd[STATPLOT]
DispGraph                      PRGM<I/O> 4:
Text(0,0,"N=",N)               2nd[DRAW] 0:
1-Var Stats L1,L2              STAT<CALC> 1:
Text(7,0,"x̄=",round(x̄,2))      MATH<NUM> 2:
Pause:PlotsOff                 2nd[STATPLOT] 4:
ClrHome:Stop                   PRGM<CTL> F:
```





We wissen de inhoud van alle lijsten met de opdracht `ClrAllLists` in regel 10. Vervolgens zetten we de rij 1 tot en met 6 in L_1 met het `seq`-commando uit het menu 2nd[LIST]<OPS> - `seq(`*uitdrukking, variabele, eerste, laatste*`)` . Het woord *sequence* betekent *rij*. De dimensie van de frequentierij, L_2, wordt zes. Het dim-commando vind je ook in 2nd[LIST]<OPS>.

In de dertiende regel begint een `For`-lus. `X` is het worpnummer en `N` het totaal aantal worpen. `X` loopt van 1 tot en met `N`. De aantal ogen van de worp noemen we `S` en $L_2(S)$ wordt met één verhoogd. Was je worp bijvoorbeeld 5, dan wordt $L_2(5)$ met 1 opgehoogd via $1+L_2(5)\rightarrow L_2(5)$.

Enkele regels lager hebben we de maximumwaarde, `max(`L_2`)`, van de frequentielijst nodig om het Y-bereik, `Ymax`, van het scherm te weten. De functie `max` haal je uit het `MATH<NUM>`-menu.

Het definiëren van het histogram doe je als volgt: 2nd[STATPLOT]<PLOTS> `1:Plot1(` gevolgd door 2nd[STATPLOT]<TYPE> `3:Histogram` , L_1 , L_2). Het commando `DispGraph` geeft de opdracht het gedefinieerde histogram te plotten.

Drie regels lager bereken je het gemiddelde $\bar{x}$ via STAT<CALC> `1:1-Var Stats`. Het gemiddelde $\bar{x}$ vind je uit VARS `5:Statistics 2:`$\bar{x}$.

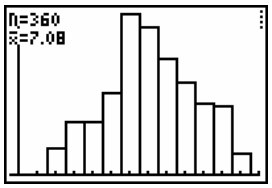
Even ademhalen, het ergste is achter de rug.

De compositie van DOB2, het werpen met twee dobbelstenen is meer een arrangement dan een compositie. Open eerst PRGM<NEW> DOB2 en roep DOB1 op met 2nd[RCL] PRGM<EXEC> DOB1.

Nu staat er een kopie van DOB1 onder de naam DOB2 en daar gaan we een paar dingen in veranderen. We gebruiken hiervoor de cursorpijltjes, [DEL], [CLEAR] en 2nd[INS].

We wijzigen de volgende regels:

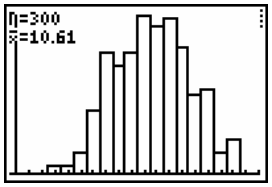
regel 2: *TWEE* i.p.v. EEN
regel 3: *STENEN* i.p.v. STEEN
regel 11: seq(X,X,1,*12*)
regel 12: *12*→dim(L2)



Regel 14 moet worden: randInt(1,6)*+randInt(1,6)*→S voor twee dobbelstenen en regel 18 wordt *13*→Xmax.

Verder niets meer veranderen en probeer of DOB2 werkt.

Voor het werpen met drie dobbelstenen gaan we DOB3 maken. Begin weer met het plakken van DOB2 via RCL. Kijk nu naar het cursief gedrukte hierboven en verander achtereenvolgens de regels 2, 11, 12, 14 en 18 in:




```
DRIE
seq(X,X,1,18)
18→dim(L2)
randInt(1,6)+randInt(1,6)+randInt(1,6)
19→Xmax
```

Nu komt het *grote plakken*! Creëer DOB123, het verzamelprogramma. Eerst plakken we (met RCL) MENU. Daarna komt een regel Lbl 1 met daarachter geplakt DOB1. Doe hetzelfde voor Lbl 2 en DOB2 en Lbl 3 en DOB3. En klaar is Kees!

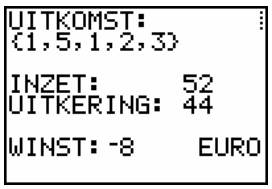
2.5 Een kansspelletje

We proberen zoveel mogelijk zessen te gooien met vijf dobbelstenen (zie onderstaande figuur van de intro). Het werpen met de 5 dobbelstenen wordt gesimuleerd in de eerste regel na Lbl H. randInt(1,6,5) komt in lijst L_1. In de daarop volgende instructie, L_1=6→L_2, wordt een rijtje nullen en enen geproduceerd en in L_2 opgeslagen: een 1 als er een zes stond en een Ø als er geen zes stond in L_1. De som S van L_2 levert het aantal zessen.

Je kunt uitrekenen dat de verwachte uitkering ongeveer 1,093 euro per spelletje is. Dat is niet zo moeilijk als je 2nd[DISTR] Ø:binompdf(5,1/6,X) gebruikt. Maar je zult merken dat je in het algemeen wel erg lang op de grote klapper van €1000 moet wachten die al je verliezen goed maakt.

```
ClrHome
Disp "5 DOBBELSTENEN"
Disp "INZET:1 EURO"
Disp "UITKERING:"
Disp "66...  2 EURO"
Disp "666..  1Ø EURO"
Disp "6666.  1ØØ EURO"
Disp "66666  1ØØØ EURO"
Output(8,14,"==>"
Pause:ClrHome
Ø→I:Ø→U
Output(1,1,"UITKOMST:")
Output(4,1,"INZET:")
Output(5,1,"UITKERING:")
Output(7,1,"WINST:      EURO")
```

```
Lbl H
randInt(1,6,5)→L1:L1=6→L2
I+1→I
Output(2,1,L1)
sum(L2)→S
If S=2:U+2→U
If S=3:U+1Ø→U
If S=4:U+1ØØ→U
If S=5:U+1ØØØ→U
Output(4,12,I)
Output(5,12,"     ")
Output(7,7,"      ")
Output(5,12,U)
Output(7,7,U-I)
Pause:Goto H
```



3 Het scherm van de TI-83 Plus

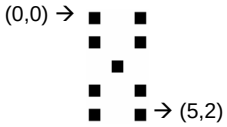
Soms is het scherm (de display) een beetje te klein. Anders gezegd, er passen wat weinig letters en cijfers (karakters) op.

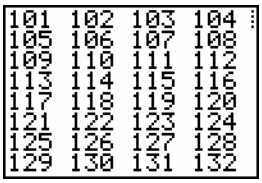
Met de `Output`-instructie, `Output`(*rij*,*kolom*,"*tekst*" of *waarde*), kun je maximaal 8 rijen met elk 16 karakters karakters op het scherm krijgen - dus 8 × 16 = 128.

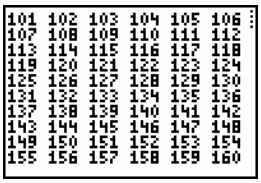
Via de `Text`-instructie, `Text`(*rij*,*kolom*,"*tekst*") die in het DRAW<DRAW>-menu te vinden is, is het mogelijk om het aantal rijen tot 10 en het aantal karakters per regel tot 23 op te voeren, bijna het dubbele aantal.

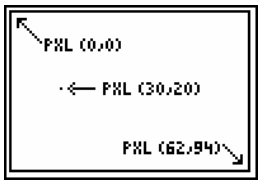
Het scherm is opgebouwd uit 62 × 94 pixels (beeldschermpunten - eigenlijk zijn het vierkantjes). Hieronder is het scherm gevuld met getallen van drie cijfers opgebouwd met beide instructies.

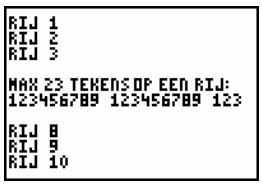
De letter X, met de Text-instructie, is opgebouwd uit 9 pixels.





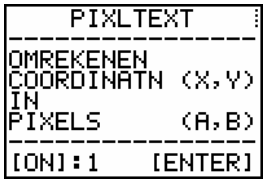


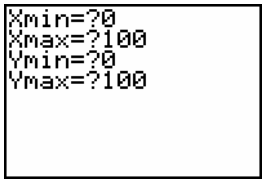


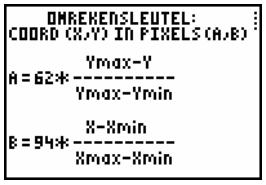


De positie van een pixel wordt met coördinaten (a,b) weergegeven, met als oorsprong (0,0) het punt linksboven. Het punt op het scherm rechtsonder krijgt de coördinaten (62,94). De eerste coördinaat a geeft het aantal pixels van bovenaf en de tweede coördinaat b hoeveel pixels je naar rechts bent gegaan.

Het programma `PIXLTEXT.83P` op www.henkshoekje.com geeft de omrekenings-formules:







4 Games

Naast het programmeerwerk als hulpmiddel voor rekenwerk, kun je je TI-83 Plus ook gebruiken voor wat meer ludiek programmeerwerk zoals bijvoorbeeld games. Het programmeren van zo'n game is vanzelfsprekend niet enkel ludiek maar hiermee ontwikkel je bovendien je probleemopslossend denken. Om een game te vertalen naar de TI-83 Plus zal je heel wat moeten algoritmiseren en structureren. Bovendien speelt de grafische creativiteit hierin ook een belangrijke rol.

Om je een idee te geven van wat allemaal mogelijk is, kun je misschien eens een bezoek brengen aan één van de volgende sites:

Texas Instruments:	education.ti.com/us/student/games/games.html
	ftp.ti.com/pub/graph-ti/calc-apps/83plus/
Detached Solutions:	www.detachedsolutions.com/puzzpack/
TiCT Headquarters:	tict.ticalc.org
TI-83 Games Basic:	bulbanos.digitalrice.com/ti83.html

De auteur is niet verantwoordelijk voor de inhoud van de hierboven vermelde sites.

We bekijken even het spelletje *Even versus Oneven*. Twee spelers (A en B) kiezen een getal uit {1, 2, 3, 4, 5}. Als de som even is, scoort A een punt en anders B. Dit is een strategisch interessant spelletje! De kansen op even en oneven zijn namelijk niet gelijk. Er is dus een strategie te bedenken die op de lange duur winst oplevert.

Als extra aspect voegen we de optie om te spelen tegen de TI-83 Plus toe.

We splitsen ons programmeerwerk op in vier delen: de intro, input, de verwerking en de output.

INTRO

We starten met een elementair openingsscherm gevolgd door een keuze om tegen een reële tegenstander te spelen of tegen de TI-83 Plus.

```
ClrHome
Disp "---------------"
Disp "EVEN <-> ONEVEN"
Disp "---------------"
Output(8,1Ø,"[ENTER]"):Pause:ClrHome
Disp "SPELERS A EN B":Disp "KIEZEN EEN GETAL"
Disp "UIT {1,2,3,4,5}":Disp ""
Disp "A SCOORT ALS SOM"
Disp "EVEN IS, B ALS"
Disp "SOM ONEVEN IS..."
Output(8,1Ø,"[ENTER]"):Pause
Menu("TEGENSTANDER:","SPELER",Ø,"TI-83 PLUS",1)
Lbl Ø
Ø→Z:Goto 2
Lbl 1
1→Z
```

INPUT

Vervolgens zetten we de tellers E en O voor respectievelijk het aantal keren even en oneven op Ø.

```
Lbl 2
ClrHome:Ø→E:Ø→O
```

```
seq(X,X,1,5)→L1
```

We vervolgen met de input van de spelers. Voor Z=Ø spelen twee spelers tegen elkaar en in het geval Z=1 wordt er tegen de TI-83 Plus gespeeld. Bovendien wordt er getest of een speler effectief een waarde tussen 1 en 5 ingeeft.

```
Lbl 3
Input "GETAL A? ",A:ClrHome
If sum(L1=A)=Ø
Then
Disp "NIET VALSSPELEN":Goto 3
End
If Z=1
Then
randInt(1,5)→B
Else
Lbl 4
Input "GETAL B? ",B:ClrHome
If sum(L1=B)=Ø
Then
Disp "NIET VALSSPELEN":Goto 4
End
End
```

VERWERKING

De verwerking is het eenvoudigste deel van het programma. De input van de spelers, A+B, wordt opgeteld en toegekend aan C. Met `C/2 - int(C/2)` wordt getest of de som even of oneven is.

```
A+B→C:C/2-int(C/2)→D
If D=Ø
Then
E+1→E
Else
O+1→O
End
```

OUTPUT

Als output tonen we het aantal keer even en oneven samen met het laatste resultaat. We sluiten af met een keuzemenu met als keuzes stoppen of opnieuw spelen.

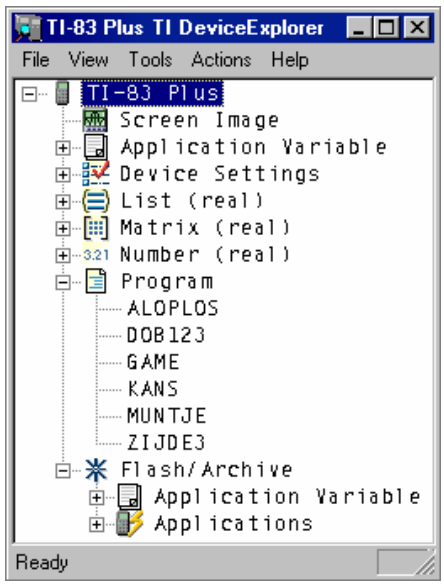
```
ClrHome
Output(1,1,"TOTAALSCORE")
Output(3,1,A): Output(3,2,"+"):Output(3,3,B)
Output(3,4,"="):Output(3,5,C)
Output(5,1,"EVEN:"):Output(5,9,E)
Output(6,1,"ONEVEN:"):Output(6,9,O)
Output(8,10,"[ENTER]"):Pause
Menu("EVEN <-> ONEVEN","OPNIEUW",5,"STOP",6)
Lbl 5:ClrHome:Goto 3
Lbl 6:ClrHome
```

5 Internet

Hieronder vind je enkele adressen van *sites* met heel wat TI-83 Plus programma's.

Texas Instruments:	ftp.ti.com/pub/graph-ti/calc-apps/83plus
Dick Klingens:	www.pandd.demon.nl
Scholierensite:	www.scholieren.com
Mathematics & Statistics:	www.hsu.edu/faculty/lloydm/ti/prgmtabl.html
TICALC.org:	www.ticalc.org

Om programma's van deze site op je TI-83 Plus te installeren, download je ze eerst op je computer. Installeer dan de software TI Connect™ die je gratis kan downloaden via education.ti.com (Products → computer software).



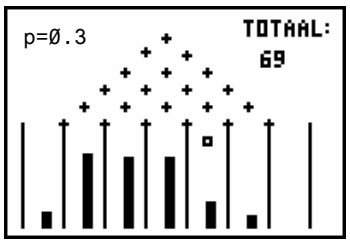
Het enige wat je dan nog moet doen is je TI-83 Plus met je computer verbinden met een TI-GRAPH LINK™-kabel. Het installeren van de programma's is nu niet anders meer dan het verslepen van het programma van de map op je computer naar het DeviceExplorer-venster van TI Connect of naar de TI Connect-icoon op het bureaublad.

Voor meer informatie kan je terecht op de site van Texas Instruments – education.ti.com.

Een site met meer dan 160 aan wiskunde verwante TI-83 Plus-programma's, speciaal bedoeld voor scholieren en studenten, is te vinden op www.henkshoekje.com.

Een voorbeeld hieruit:

GALTON.83P bootst het spijkerbord van Galton na. Een groot aantal knikkertjes wordt boven ingebracht, via een soort trechter. De knikkertjes botsen tegen de spijkers en komen in een van de onderste zeven bakjes terecht. Dit programma kan gebruikt worden voor de simulatie van een binomiaal kansexperiment. De kans p op "rechtsaf" kan gevarieerd worden, zodat een scheve verdeling kan ontstaan.



Henk Pfaltzgraff

Programmeren met de TI-83 Plus

Wiskunde met een extra dimensie

CTL I/O EXEC
1:If
2:Then
3:Else
4:For(
5:While
6:Repeat
7↓End

CTL I/O EXEC
1:Input
2:Prompt
3:Disp
4:DispGraph
5:DispTable
6:Output(
7↓getKey

Texas Instruments