Programmeren op de Casio fx-9860G

Praktische opdracht
september 2007

Inleiding

Een programma is een reeks instructies die aangeven wat de computer, en in ons geval de grafische rekenmachine (GR), moet doen. Je kunt het vergelijken met een kookrecept, als je zonder nadenken stap voor stap de instructies volgt, dan zul je een lekkere maaltijd aan het eind krijgen.

Zo gaat dat met een programma ook. De GR die volgt domweg de instructies op en aan het eind krijg je exact datgene wat je wilde. Het enige waar je op hoeft te letten is dus, dat jij erg zorgvuldig die instructies formuleert.

Bij deze praktische opdracht zullen we zelf een programma bedenken en die in onze GR programmeren. Je kunt denken aan een programma die uitrekent, nadat jij de drie zijden van een driehoek hebt ingevoerd, of het een rechthoekige driehoek is of niet. Een programma die een reeks van ingevoerde getallen van klein naar groot sorteert. Een programma die uitrekent dat de 324^{ste} dag van 2007 een dinsdag is.

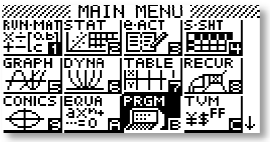
We beginnen met een gemeenschappelijk deel, waarin we de meest voorkomende commando's samen bestuderen. Dit gedeelte heb je nodig om op basis hiervan straks je eigen programma te kunnen maken. Wat je in dit gemeenschappelijk deel leert, is zeker niet alles wat je nodig hebt voor je eigen programma.
Bij een praktische opdracht wordt er van je verwacht dat je zelf onderzoek doet (internet, boeken, handleiding van Casio, vrienden/familie/buren, etc) naar wat de mogelijkheden zijn en dus zelf uitzoekt wat je nodig hebt om je programma werkende te krijgen. In de lessen kun je vragen stellen, maar de vragen zullen pas beantwoord worden, als duidelijk blijkt dat je eerst zelf gezocht hebt. Op een vraag die bijvoorbeeld onder de eerste hit in Google te vinden is, zal geen antwoord op gegeven worden.

Ook zal ik hier bij lange na niet alle mogelijkheden bespreken, kom je een knop tegen waarvan je niet weet wat hij doet, klik er gerust op en probeer op deze manier te ontdekken wat er gebeurt. Zoek anders in de handleiding van je GR of op internet, er zullen vele opties bestaan waarvan je het bestaan niet afweet en die erg handig kunnen worden voor je eigen programma.

DEEL 1

De eerste stapjes

Als je jouw GR aandoet en op het knopje [MENU] klikt, dan zie je het scherm hiernaast.



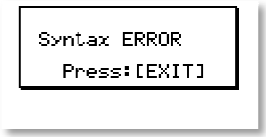
Als je vervolgens naar [PRGM] gaat en erop klikt dan krijg je als je nog geen programma's hebt het scherm hiernaast.

Mocht je al wel wat programma's hebben, dan zul je iets soortgelijks zien als hiernaast is afgebeeld. De getallen achter de programma's geven aan hoeveel bytes het programma groot is.

Als we nu op [NEW-F3] klikken, dan maken we een nieuw programma. Geef dit programma de naam die jij wilt (max. 8 tekens) en klik op [EXE]. Eventueel kun je door op het 'sleuteltje' [F5] een wachtwoord opgeven.

In dit scherm gaan we ons eerste programmaatje schrijven, het programma dat niets anders kan dan HELLO WORLD op het scherm tonen.

De dubbele aanhalingstekens (“), die vertelt de GR dat hij deze tekst op het scherm moet tonen.



Vergeet je deze aanhalingstekens (probeer het maar) dan krijg je een foutmelding.

Als je nu weer terug gaat naar het hoofdscherm dan kun je hierin het programmaatje verwijderen (DEL-[F4]) of een andere naam geven (REN-

[F6]-[F2]) (REN, van rename). Probeer zelf even uit wat deze toetsen doen.

Het eerste echte programmaatje

Hierboven hebben we kennisgemaakt met het invoerscherm en hoe je een programma aanmaakt. Nu zullen we ons bezig gaan houden met de instructies die we een programma kunnen geven.

De meeste programma's hebben een invoer, deze invoer die wordt verwerkt om vervolgens de gewenste uitvoer te geven.

Als voorbeeld gaan we een programma maken dat twee getallen met elkaar vermenigvuldigt (wat natuurlijk niet een erg nuttig programmaatje zal gaan worden, want dat kan ook in het RUN menu, maar om te leren programmeren is het zeker erg nuttig).

Open het programma-menu en maak een nieuw programma met de naam VERM.

De vraagteken (?) is een invoercommando voor je GR. De syntax[1]

?→A

vraagt om een invoer (die de gebruiker dient in te voeren) die hij dan toekent aan de variabele A. Je rekenmachine heeft heel veel variabelen die allemaal een waarde kunnen hebben (de rode letters op je GR). Als ik de letter A de waarde 2 en de letter B de waarde 3 toeken dan zal AxB als antwoord 6 geven.

[1] Voor een programmeertaal is de syntax de "taal" van het programmeren. Het bekende Syntax Error betekent dus dat de GR het niet begrepen heeft.

Combineren we dit met de aanhalingstekens die we hierboven geleerd hebben, dan kunnen we de gebruiker van ons programma vragen om iets in te voeren:

"Voer een getal in"?→A

De GR zet nu 'Voer een getal in' op het scherm en wacht totdat jij een getal ingetypt hebt. Nadat jij dit getal ingetypt hebt, kent hij de letter A (het geheugen A) deze ingevoerde waarde toe.

Het programma hiernaast vraagt om twee getallen en vermenigvuldigt deze twee met elkaar en geeft dat als uitvoer. Bekijk dit programma goed, voer 'm in je GR en wees er zeker van dat je 'm begrijpt.

OPDRACHT 1:
Schrijf een programma DELEN dat twee getallen op elkaar deelt.

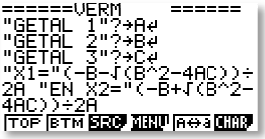
Als laatste van dit eerste deel gaan we proberen een programma te schrijven die de oplossingen van de ABC-formule voor ons uitrekent.

Zoals je weet kan de ABC-formule de snijpunten vinden van een tweedegraadsvergelijking met de x-as.
Voor de vergelijking $ax^2 + bx + c = 0$ zijn de snijpunten met de x-as

$$x_1 = \frac{-b - \sqrt{b^2 - 4ac}}{2a} \quad \wedge \quad x_2 = \frac{-b + \sqrt{b^2 - 4ac}}{2a}$$

OPDRACHT 2:
Schrijf een programma dat (alleen) de oplossing x_1 uitrekent met behulp van de ABC-formule. Maak nu dus gebruik van 3 keer het invoercommando (éénmaal voor a, b en c).

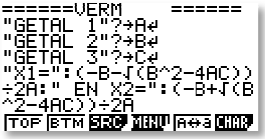
Als we beide oplossingen in het uitvoerscherm willen krijgen dan zou je verwachten dat je moeiteloos het volgende programma zou kunnen gebruiken.



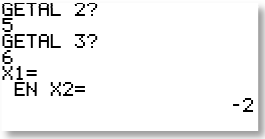
Het programma is wel bijna goed. Er zijn een aantal dingen waar je op moet letten. Je GR wil geen spaties in zijn programmeercode, natuurlijk wel tussen aanhalingstekens, want dat is gewoon letterlijke tekst die hij op het scherm laat zien. Maar bijvoorbeeld **A+ B,** met tussen de + en de B een spatie, accepteert hij niet. De spatie moet weg.

Ook wil je GR geen codes achter elkaar geplakt. Je moet bij elke code een nieuwe regel beginnen of het commando : (dubbele punt) gebruiken. Een nieuwe regel beginnen of een dubbele punt gebruiken, betekent voor je GR hetzelfde. Vaak is het overzichtelijker om vaak nieuwe regels te gebruiken.
Dus **"X1=" (–B–…** mag niet en moet **"X1=" : (–B–…** zijn, aangezien "X1=" één statement[2] is en **(–B–…** een andere.

Hiermee hebben we twee veel voorkomende kleine foutjes behandeld waar je heel goed op moet letten in het vervolg. Maar als ik deze foutjes er uithaal, dan werkt het programma nog steeds niet goed en geeft hij maar één oplossing.



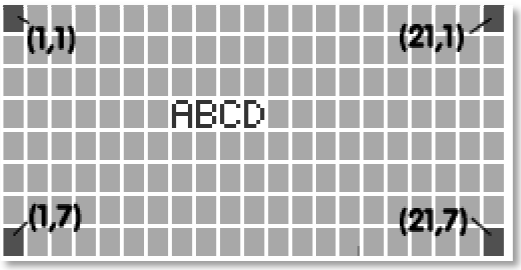
Waarom staat er achter **X1=** niets?
Dit is om de hele eenvoudige reden dat je GR over zichzelf heen schrijft. Eerst heeft hij de uitvoer van X1 uitgerekend (wat -3 is in dit voorbeeld) en vervolgens die van X2 die hij over die van X1 heen schrijft in zijn scherm.



[2] Uitvoerbare instructie.

Hiervoor bestaat het commando **Locate.** Locate zorgt ervoor dat je zelf kunt kiezen waar op het scherm je je uitvoert zet.

Het scherm is verdeeld in 21 bij 7 hokjes waar letters of cijfers in kunnen staan. Zie de figuur hiernaast. Verwar dit niet met de pixels van je GR, want dit zijn er 127 bij 63. De syntax voor locate is de volgende:




```
Locate 8,4,"ABCD"
```

De waarde die A heeft, wordt op plekje 8,4 gezet (dus 8 naar rechts, 4 omlaag).

Let op dat je niet gewoon Locate intoets met de 'rode letters', de rode letters zijn geheugenplekjes en als je Locate intypt, dan zal je GR dit niet herkennen als het commando Locate. Dit commando staat onder [SHIFT][VARS][F6][I/O-F4][Lcte-F1].

Als ik nu dus voor ons programma van de ABC formule dit commando gebruik, kan ik ervoor zorgen dat de antwoorden niet over elkaar heenkomen, want ik bepaal zelf waar ik ze neerzet. Dit kan ik doen door bijvoorbeeld:

```
Locate 20,6,(-B-√(B^2-4AC)/2A)
```

te gebruiken.

OPDRACHT 3:
Pas je ABC-formule programma dusdanig aan, dat hij nu wel goed werkt.

DEEL 2

Het echte werk

Nu we de basis van het programmeren onder de knie hebben, zullen we gaan kijken naar een paar commando's die ons het programmeerwerk mogelijk en makkelijker zullen maken.

Een heel belangrijke commando dat in bijna elk programma wel terugkomt is het zgn. programmeringscommando. Hier zijn een aantal typen van. In dit deel zullen we aan de hand van een aantal voorbeelden en opdrachten de werking van deze commando's leren.

- **If-Then**
  Dit is een commando die bestaat uit een conditie (de If) en een statement (de Then). Het statement wordt alleen maar uitgevoerd als de If-conditie waar is. If en Then kun je vinden onder [SHIFT][VARS][COM-F1].

  Voorbeeld:
  Als je wilt dat je GR twee getallen bij elkaar optelt en als de uitkomst groter is dan 100 hij dit aangeeft, dan kun je dit commando goed voor gebruiken.
  De conditie is dan dat het getal groter moet zijn dan honderd, het statement is dan dat je GR aangeeft dat dit getal groter is dan 100. In woorden: ALS A+B groter is dan 100, DAN geef het aan.

```
If A+B>100
Then "Het getal is groter dan 100"
```

OPDRACHT 4:
Verwerk dit voorbeeld in een werkend programma dat om invoer vraagt en dit dan verwerkt (zie opdracht 1).

Dit commando lijkt erg op die hierboven, met als enig verschil dat deze nu ook een Else statement heeft. Met andere woorden, als er aan de conditie voldaan wordt, dan gebeurt er iets, en als er niet aan de conditie voldaan wordt, dan gebeurt er iets anders.

Voorbeeld:
Als we doorgaan op het voorbeeld hierboven en we willen dat ons programma ook aangeeft dat het antwoord kleiner dan 100 is kunnen we het volgende voor gebruiken. In woorden: ALS A+B groter is dan 100, DAN geef dit aan en ANDERS dat.

```
If A+B>100
Then "Het getal is groter dan 100"
Else "Het getal is kleiner dan 100"
```

OPDRACHT 5:

a. Verwerk dit voorbeeld in een werkend programma dat om invoer vraagt en dit dan verwerkt.
b. Wat gebeurt er als de uitkomst exact 100 is? Hoe kunnen we deze fout oplossen? Verbeter je programma.

- **If-Then-(Else)-IfEnd**
  Als je de If-Then of de If-Then-Else afsluit met een IfEnd, dan zal het IfEnd statement altijd worden uitgevoerd na het Then of Else statement.
  Handig voor als je wilt dat iets zeker wordt uitgevoerd.
- **For-To-Next**
  Alles tussen het For statement en het Next statement wordt herhaalt, waarbij de waarde van een controle variabele telkens

met één wordt verhoogt. Je stelt een eindwaarde in, waarbij de uitvoering stopt.

Voorbeeld:
Stel je wilt uitrekenen hoeveel 1+2+3+4+… is. Deze straf kreeg de wiskunde Gauss van zijn leraar toen hij negen jaar was. Hij moest voor straf tot en met honderd gaan. Hij deed het toen op een geniale manier (die hier niet aan bod zal komen), maar wij kunnen er moeiteloos een programmaatje voor schrijven:

```
For 1→A To 100
A+B→B
Next
```

A is de controle-variabele die begint bij 1 (de 1→A geeft deze variabele de waarde 1 zoals je weet) en doorgaat tot 100. Elke keer als je bij Next aangekomen bent, begin je weer opnieuw waarbij A eentje hoger wordt. Als A 100 is geworden, stopt het programma.
We hebben een tweede variabele nodig die steeds het antwoord onthoudt wat er uit de vorige berekening kwam. Dat is precies wat jij zou doen als je 1+2+3+4+… uit je hoofd zou uitrekenen:

1+2=**3**
3+3=**6**
6+4=**10**
10+5=**15**
Enz.

B is nu steeds de uitkomst (zie dikgedrukt hierboven).

OPDRACHT 6:

a. Pas het programma hierboven aan zodat je 100+101+102+…+200 uitrekent.
b. Pas het programma aan zodat de gebruiker gevraagd wordt tot hoeveel hij wil optellen.

- **For-To-Step-Next**

  Dit commando doet exact hetzelfde als For-To-Next met als enig verschil dat je de stapgrootte in kunt stellen. For-To-Next laat de controlevariabele steeds met één toenemen, met Step kun je zelf bepalen met hoeveel je de controlevariabele wilt laten toenemen.

  Voorbeeld:

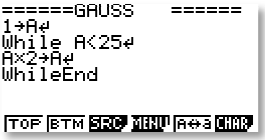
  Stel je wilt 0,1+0,2+…+0,9+1,0 bij elkaar optellen, dan kun je de code hiernaast gebruiken.

- **Do-LpWhile en While-WhileEnd**

  Deze commando's blijven herhalen totdat er niet meer aan de conditie voldaan wordt.

  Voorbeeld:

  Stel je wilt weten hoevaak je 1 moet verdubbelen (1,2,4,8,16,enz) om boven de 25 uit te komen, dan kun je heel goed Do-LpWhile of While-WhileEnd voor gebruiken:



  Bekijk goed de (geringe) verschillen van deze twee methodes.

Opdracht 7:
Waarom geeft dit programma 32 als uitkomst terwijl ik zeg dat hij zichzelf alleen maar mag herhalen als de waarde van A kleiner is dan 25?

Waar ik eigenlijk in geïnteresseerd ben, is hoe vaak ik mag verdubbelen en toch onder de 25 te blijven. Dit kan ik eenvoudig op deze manier tellen:

Wees er zeker van dat je dit begrijpt (waarom N-1 en niet N tonen?).

OPDRACHT 8:
Schrijf een programma dat de gebruiker vraagt om een invoer, wat als uitvoer heeft hoe vaak er verdubbeld is om daarboven te komen.

<u>Nog een paar handige commando's</u>
Als laatste nog een paar handige commando's die je misschien goed kunt gebruiken in je eigen programma.

- **Prog ([SHIFT][VARS][CTL-F2][PROG-F1])**
  Dit commando kun je gebruiken als je een ander programma wilt aanroepen in je programma.
- **⊿ ([SHIFT][VARS][F5])**
  Dit is een uitvoercommando en laat tussentijdse resultaten zien tijdens je programma. Kan soms erg handig zijn als je bijvoorbeeld een fout aan het opsporen bent. Het doet eigenlijk hetzelfde als [EXE] in het gewone RUN menu.

- **STOP ([SHIFT][VARS][CTL-F2][Stop-F4])**
  Dit commando stopt de uitvoer van een programma. Soms erg handig om in een loop te hebben, zodat de loop[3] stopt wanneer jij wilt.
- **Goto-Lbl ([SHIFT][VARS][JUMP-F3])**
  Met dit commando kun je zelf een loop specificeren.

[3] Een loop wordt in de informatica het repeteren van een aantal statements genoemd, je hebt dus een If-Then loop, een For-To-Next loop, enz.

DEEL 3

Je eigen programma

Als je de delen 1 en 2 goed begrepen hebt, ben je er nu aan toe om aan je eigen programma te beginnen.

Je moet iets bedenken wat je wilt programmeren, dit moet goedgekeurd worden door je docent voordat je begint. Je mag pas aan je programma beginnen als de acht opdrachten hierboven zijn afgetekend door je docent. Laat dus elke keer het (werkende) programma aan je docent zien.

De enige twee eisen aan je programma is, dat hij minimaal één programmeringscommando moet bevatten en minimaal 10 regels lang moet zijn.

Wat moet je inleveren?

Je maakt een kort verslagje waarbij je begint met een inleiding waarin je beschrijft wat het doel van je programma is en waarom je hiervoor gekozen hebt. Je levert een printje van je programma in[4], met begeleidende tekst waarbij je per regel uitlegt wat het programma doet.

De beoordeling

Je wordt beoordeeld op de volgende punten:

1. De afgetekende opdrachten (5 punten). Bij het aftekenen van de opdrachten kunnen er vragen gesteld worden die je juist moet beantwoorden.

[4] Met de kabels en het programma FA124 die je bij je GR gekregen hebt, kun je moeiteloos je GR aan je computer aansluiten en zo een screenshot maken van het scherm van je GR.

2. Je bedenkt iets wat je wilt programmeren, vertelt dit aan je docent en afhankelijk van de moeilijkheidsgraad van jouw programma kun je 30, 35 of 40 punten verdienen (35 is standaard). Een te makkelijk programma wordt niet goedgekeurd. Je programma en de uitleg ervan zullen je dus 30, 35 of 40 punten op kunnen leveren. Voor je inleiding/doel van het programma kun je 10 punten krijgen en voor het programma zelf met uitleg 25.
3. Dit brengt het totaal aantal te behalen punten op 40. Het cijfer wordt verkregen door jouw punten keer 0,25 te doen.
4. De mogelijkheid bestaat om bonuspunten te krijgen. Mocht je een heel goed, moeilijk, origineel, enz, programma gemaakt hebben, dan kun je je eindcijfer met 1,0 cijferpunten verhogen.
5. Bewaar je programma in je GR, want de docent wil het werkende programma ook even testen en niet alleen van papier.

BIJLAGE

Uitgewerkt voorbeeld

Naam

Kees Schoenmaker

Inleding

Vaak is het handig om variabelen te kunnen wisselen van waarde. Als ik bijvoorbeeld op zoek ben naar het verschil tussen twee waarden, dan wil ik graag de grootste min de kleinste doen. Mijn GR weet als ik getalA-getalB doe, niet welke de grootste en de kleinste is. Ook kan het handig zijn voor het programmeren van de stelling van Pythagoras. De schuine zijde is altijd de langste, dus het grootste getal. Een gedeelte en uitbreiding van dit programma zou daar ook heel geschikt voor kunnen zijn.

Doel van mijn programma

Het doel van mijn programma is dat hij twee waarden vraagt waarvan de eerste (getal A) altijd de grootste moet zijn. Als de gebruiker voor A een kleiner getal dan voor B invult, dan wisselt mijn programma de waarde van A en B om. Als het ingevoerde getal A als groter is dan B, dan hoeft mijn programma niets te doen en zegt hij dat A al groter is dan B. Mocht de gebruiker twee even grote getallen invoeren, dan geeft mijn GR dat ook aan.

De programmeercode

Regel 1: Er wordt gevraagd om getal A, de aanhalingstekens (“) die

zorgen ervoor dat de tekst letterlijk op het scherm afgedrukt wordt. De vraagteken (?) die zorgt ervoor dat het programma wacht op een invoer van de gebruiker en de A die zorgt ervoor dat deze invoer in de variabele A wordt opgeslagen.

Regel 2: Als regel 1.

Regel 3: Ik gebruik hier het programmeringscommando If-Then-Else, met als conditie A<B

Regel 4: Als A<B is, dan zet hij de waarde van A in een tijdelijke variabele C (door A C te gebruiken), dus A is nu leeg en kan variabele B overgezet in variabele A (door B A te gebruiken). Nu is variabele B leeg en kan dus variabele C (die de waarde die A eerst had heeft) overgezet worden in variabele B. De dubbele punt tussendoor, koppelt de verschillende statements aan elkaar. Ik zou hier ook een return (nieuwe regel) kunnen gebruiken, maar daar wordt mijn programma onoverzichtelijk van.

Regel 5 en 6: Vervolgens geeft het programma aan dat de inhoud van de variabelen zijn verwisseld (dat heeft hij namelijk in regel 4 gedaan).

Regel 7 en 8: Hier toont hij op het scherm de nieuwe verwisselde waarden van de variabelen.

Regel 9: De Else geeft hier aan wat hij anders moet doen als A niet kleiner is dan B. Ik gebruik een nieuwe If-Then om ook te testen of ze niet toevallig gelijk zijn. Want als A niet kleiner dan B is, dan kan hij groter zijn of even groot.

Regel 10 en 11: Als A gelijk is aan B, dan drukt hij dit af op het scherm.

Regel 12: De Else die kijkt naar als dat niet zo is (als A niet gelijk is aan B)

Regel 13, 14 en 15: Aangezien A dus niet gelijk is aan B en ook niet kleiner, dan moet A wel groter zijn dan B. Dus mijn programma hoeft helemaal niets meer te doen.