Introductie tot de **TI-Innovator Hub**

Didier, Frank en Franklin

Inhoudsopgave

1 Inleiding **1**

2 Licht: discolicht **2**

3 Geluid: random muziek **4**

4 Licht en geluid: Disco en random muziek **6**

5 Toepassing: ééndimensionale muzikale random walk **6**

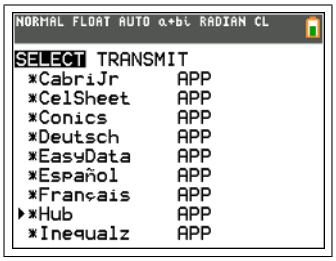
6 Toepassing: driedimensionale random walk **7**

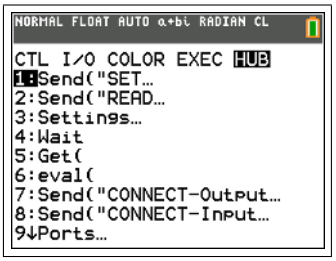
7 Toepassing: Simon says **8**

8 De lichtsensor: zelf random muziek maken **10**

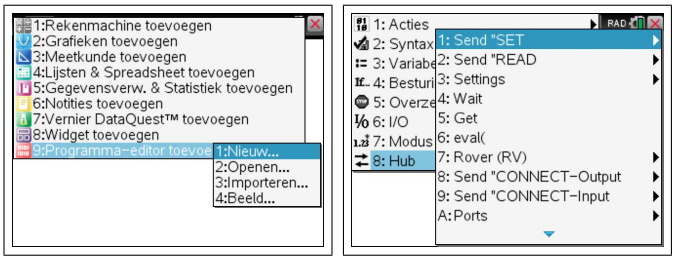
1 Inleiding

Voor je aan de slag kan met de **TI-Innovator Hub** moet je eerst de meegeleverde `app Hub` op je **TI-84 Plus Color** zetten. Dit kan vanaf een computer of via een andere **TI-84 Plus Color** waarop het reeds geïnstalleerd is. Eenmaal de `app` ook op jouw **TI-84 Plus Color** staat, zal ze het menu `[prgm]` in de programmeereditor uitbreiden met een nieuw submenu `[HUB]` waarin de commando's staan om met de **TI-Innovator Hub** te communiceren.





Voor de **TI-Nspire** moet je zorgen dat je besturingssysteem up-to-date is. Onder de programma-editor kan je een nieuw programma aanmaken. Onder de knop menu kan je dan een submenu [8:Hub] terugvinden.



Nu kunnen we, zowel voor **TI-84 Plus Color** als **TI-Nspire** , in een programma instructies doorsturen naar de **TI-Innovator Hub** . Dit gebeurt door berichten en commando's te zenden via `Send`. Wij zullen de volgende berichten gebruiken.

- `SET COLOR` r g b: De kleur LED wordt aangezet in een bepaalde kleur.
- `SET SOUND` f `TIME` t: Produceert een toon met frequentie f gedurende t seconden.

Je vindt deze berichten (of delen ervan) onder prgm [HUB] [Send("SET…] en prgm [HUB] [Settings…]. Andere combinaties van deze berichten zijn ook mogelijk maar we zullen ze hier niet gebruiken.

Naast de berichten naar de **TI-Innovator Hub** zijn er ook enkele commando's die gebruikt zullen worden. Deze vind je onder prgm [HUB].

- `Send(`*bericht*`)`: Verstuurt een bericht naar de **TI-Innovator Hub** .
- `Wait` t: Pauzeert de uitvoer van het programma gedurende t seconden.
- `eval(`*uitdrukking*`)`: Dit commando wordt binnen een bericht gebruikt om een uitdrukking te evalueren en om te zetten naar een getal. Dan pas wordt het bericht verstuurd naar de **TI-Innovator Hub** .

2 Licht: discolicht

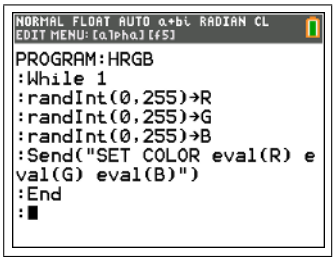
Uit de optica weet men dat elke kleur die ons oog kan waarnemen, gevormd kan worden door een mengsel van rood, groen en blauw licht. Daarom bestaat in de informatica een scherm uit pixels, elke pixel is opgemaakt uit drie LEDs: een rode, een groene en een blauwe. Op de **TI-Innovator Hub** is zo één

opstelling van drie LEDs beschikbaar. Op deze wijze kunnen we één pixel nabootsen.

De intensiteit van elke afzonderlijke LED wordt gegeven door een getal tussen 0 en 255, dit komt intern overeen met één byte (=8 bits). Elke kleur komt dus overeen met een RGB-code, dit zijn drie waarden (R, G, B), telkens tussen 0 en 255, die aangeven hoeveel van elke basiskleur aanwezig is. We kunnen dus meer dan 16 miljoen kleuren maken (reken dit uit).

Hoe doe je dit op de TI-84 Plus Color ?

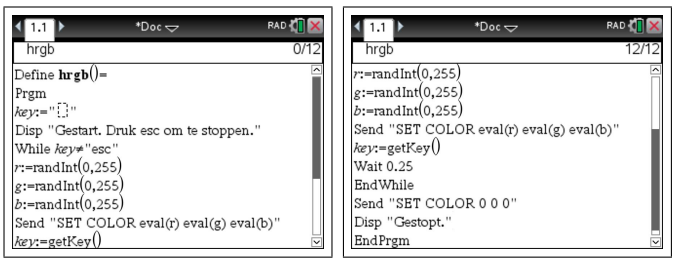
Met onderstaand programma wordt telkens een willekeurige waarde voor R, G en B genomen. We bekomen een disco flikkerlichtje.



- Met `While 1` wordt een oneindige lus gestart.
- Daarna worden met [math] `[prb] [randInt]` voor R, G en B willekeurige waarden tussen 0 en 255 gekozen.
- Tenslotte worden met `SET COLOR eval(R) eval(G) eval(B)` de kleur LEDs aangezet in de juiste kleurintensiteiten en wordt de lus gesloten.

Hoe doe je dit op de TI-Nspire ?

Met onderstaand programma wordt ook telkens een willekeurige waarde voor R, G en B genomen. We bekomen een disco flikkerlichtje. We gaan net nog een stapje verder dan bij het programma uit de **TI-84 Plus Color** en laten het programma eindigen als we op [esc] drukken.



- Met `While key` $\neq$ `"esc"` wordt een lus gestart die blijft lopen zolang we niet op esc drukken. Daarom dat we de variabele `key` eerder in het programma ook al leegmaakten met `key:=" "`.
- Daarna worden met `[randInt]` voor r, g en b willekeurige waarden tussen 0 en 255 gekozen.
- Vervolgens worden met `SET COLOR eval(r) eval(g) eval(b)` de kleur LEDs aangezet in de juiste kleurintensiteiten. We gebruiken `eval()` om de waarde uit de betreffende variabele te halen voor het commando `SET COLOR`.
- Daarna zullen we met `key:=getKey()` kijken of er op een knop is gedrukt. Denk er aan dat de voorwaarde om onze `While`-lus te sluiten hier aan gekoppeld is.
- Omdat we niet willen dat ons discolicht te snel verandert, laten we het programma 0,25 seconden wachten.
- Als we op esc drukken, wordt de lus doorbroken.
- Ten slotte gaan we met `SET COLOR 0 0 0` de LEDs weer uitzetten en displayen we dat het programma is gestopt.

3 Geluid: random muziek

De **TI-Innovator Hub** staat toe om een bepaalde toon op een zekere frequentie af te spelen.

Reeds sinds Pythagoras heeft men getracht muziek in een wiskundig formalisme te beschrijven. Pythagoras merkte op dat een snaar aanslaan zorgde voor een welbepaalde toon. Wanneer de snaar wordt gehalveerd krijgt men een consonante toon, juist één octaaf hoger. Pythagoras ging verder de snaar opdelen volgens welbepaalde breuken en bekwam aldus de *Pythagoraanse stemming*. In deze stemming kent men de gewoonlijke toonladder die 17 eeuwen lang werd gebruikt. Pas aan het begin van de middeleeuwse polyfonie merkte men dat de verhoudingen niet voor een regelmatige verdeling van de toonhoogten zorgden.

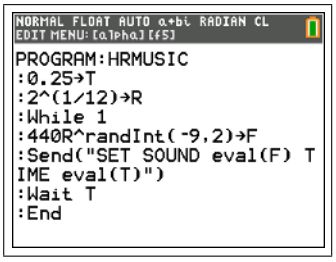
In de moderne muziektheorie wordt de *gelijkzwevende stemming* gebruikt. Hierbij merkt men op dat bij verhoging van één octaaf de frequentie verdubbelt. Gezien er twaalf halve tonen zijn in één octaaf worden de frequenties per halve toon telkens vermenigvuldigd met $\sqrt[12]{2}$. We bekomen een meetkundige rij:

$$f_{n+1} = \sqrt[12]{2} f_n$$

Het was ondermeer de bruggeling Simon Stevin die aan het einde van de 16de eeuw voor deze stemming pleitte. We zullen deze stemming ook gebruiken om met de **TI-Innovator Hub** muziek te spelen. Als standaard vertrekt men vaak van de la met frequentie 440Hz.

1	2	3	4	5	6	7	8	9	10	11	12
do	#do	re	#re	mi	fa	#fa	sol	#sol	la	#la	si
C	#C	D	#D	E	F	#F	G	#G	A	#A	B
$2^{-\frac{9}{12}}$	$2^{-\frac{8}{12}}$	$2^{-\frac{7}{12}}$	$2^{-\frac{6}{12}}$	$2^{-\frac{5}{12}}$	$2^{-\frac{4}{12}}$	$2^{-\frac{3}{12}}$	$2^{-\frac{2}{12}}$	$2^{-\frac{1}{12}}$	1	$2^{\frac{1}{12}}$	$2^{\frac{2}{12}}$

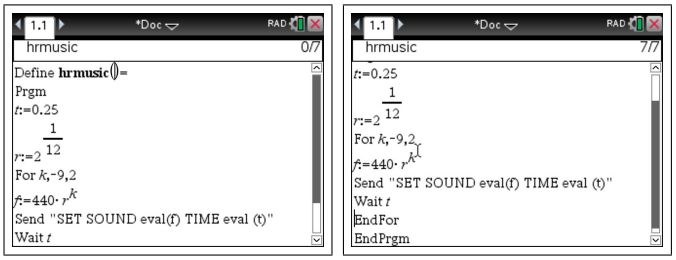
Hoe doe je dit op de TI-84 Plus Color ?



- Eerst wordt het tempo vastgelegd, elke noot zal $T = 0,25s$ duren. Het getal $\sqrt[12]{2}$ wordt eenmalig uitgerekend.
- Met `While 1` wordt een oneindige lus gestart.
- Er wordt een willekeurig getal gekozen tussen -9 en 2, waarmee de frequentie van de noot wordt berekend.
- De noot wordt gedurende een tijd T afgespeeld, het programma wacht voralleer verder te gaan met de lus.

Hoe doe je dit op de TI-Nspire ?

In het voorbeeld voor de **TI-Nspire** werken we niet met een oneindige `while` maar met een `for` om alle 12 tonen van de toonladder af te gaan.



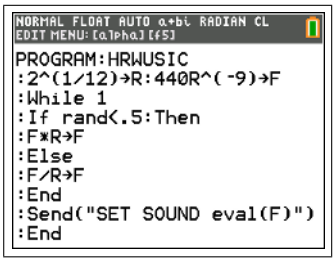
4 Licht en geluid: Disco en random muziek

Combineer nu zelf de twee vorige programma's.

5 Toepassing: ééndimensionale muzikale random walk

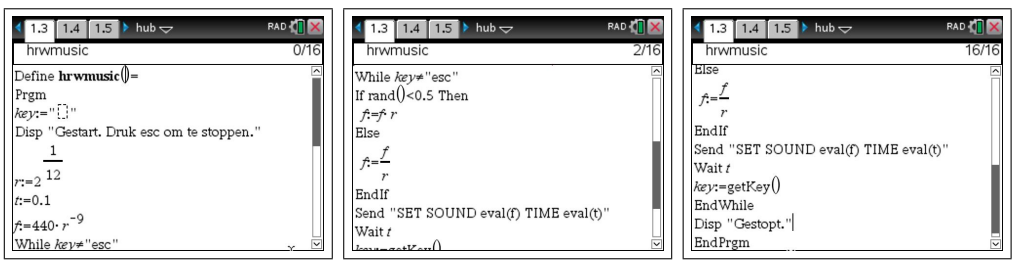
We zullen nu opieuw random muziek maken, maar deze keer vertrekken we van een do. Bij elke volgende noot wordt willekeurig een halve noot hoger of lager afgespeeld.

Hoe doe je dit op de TI-84 Plus Color ?



- Eerst wordt het getal $\sqrt[12]{2}$ eenmalig uitgerekend en de basisfrequentie van do wordt uitgerekend.
- Met `While 1` wordt een oneindige lus gestart.
- Met `rand` wordt er willekeurig een getal tussen 0 en 1 gekozen, indien dit getal kleiner is dan $0,5$ wordt de frequentie verhoogd met een factor $\sqrt[12]{2}$, anders wordt ze verlaagd met eenzelfde factor.
- Nadien wordt de noot gespeeld en wordt de lus gesloten.

Hoe doe je dit op de TI-Nspire ?

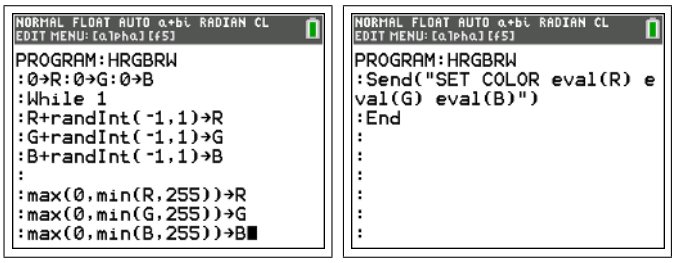


- Eerst wordt het getal $\sqrt[12]{2}$ eenmalig uitgerekend en de basisfrequentie van do wordt uitgerekend.
- We werken opnieuw met een `While` die wordt doorbroken door esc.
- Met `rand()` wordt er willekeurig een getal tussen 0 en 1 gekozen, indien dit getal kleiner is dan $0,5$ wordt de frequentie verhoogd met een factor $\sqrt[12]{2}$, anders wordt ze verlaagd met eenzelfde factor.
- Nadien wordt de noot gespeeld en wordt de lus gesloten.

6 Toepassing: driedimensionale random walk

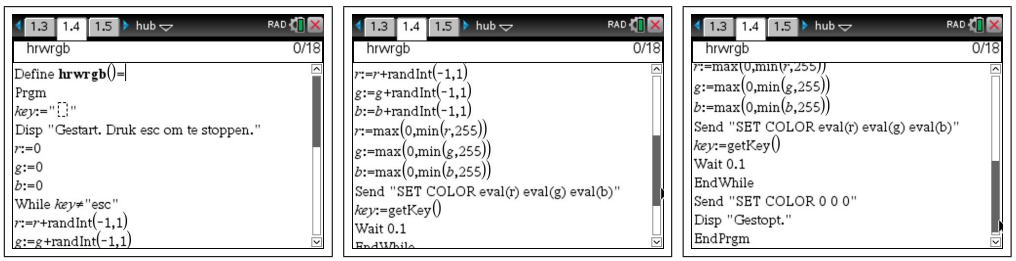
Deze keer gaan we een random walk programeren in de RGB-kleurenruimte. Een driedimensionale random walk dus. Omdat van richting veranderen in de ruimte ingewikkelder is dan in het vlak zullen we hierbij een andere aanpak hebben. Bij elke RGB component tellen we willekeurig $-1, 0$ of 1 op. Dit geeft een combinatie van drie ééndimensionale random walks in de ruimte.

Hoe doe je dit op de TI-84 Plus Color ?



- Eerst wordt de startkleur op zwart gezet, daarna begint een oneindige lus.
- Bij elke RGB component tellen we willekeurig $-1, 0$ of 1 op.
- Door $\max\{0, \min\{x, 255\}\}$ te gebruiken zorgen we ervoor dat x nooit kleiner kan zijn dan 0 of groter dan 255.
- Tenslotte worden de kleurleds aangezet en wordt de lus gesloten.

Hoe doe je dit op de TI-Nspire ?



- Eerst wordt de startkleur op zwart gezet, daarna begint de `While`-lus.
- Bij elke RGB component tellen we willekeurig $-1, 0$ of 1 op.
- Door $\max\{0, \min\{x, 255\}\}$ te gebruiken zorgen we ervoor dat x nooit kleiner kan zijn dan 0 of groter dan 255.
- Tenslotte worden de kleurleds aangezet en wordt de lus gesloten.
- Ook deze lus kunnen we weer stoppen met esc.

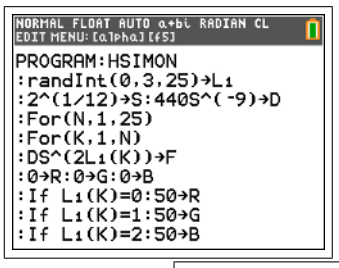
Opmerking

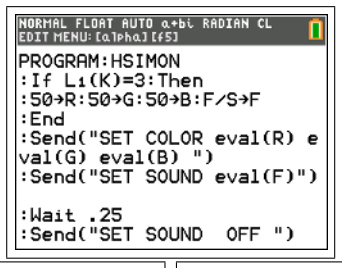
Uit de wiskunde is geweten dat een random walk in de ruimte niet per se terugkomt in haar startpunt. Omdat hier de waarden echter begrensd zijn door 0 en 255, zal men na lang wachten wel weer een moment vinden waar de leds terug uit zijn.

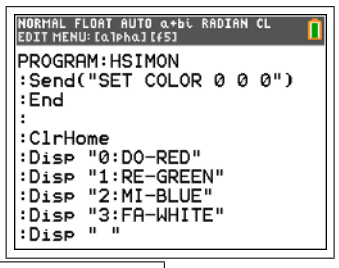
7 Toepassing: Simon says

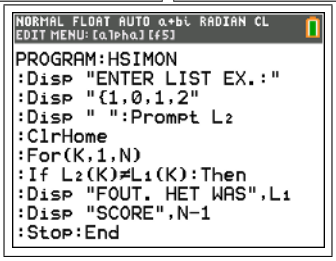
In het spelletje *Simon says* wordt een steeds langer wordende code gegeven die de speler telkens in correcte volgorde moet herhalen.

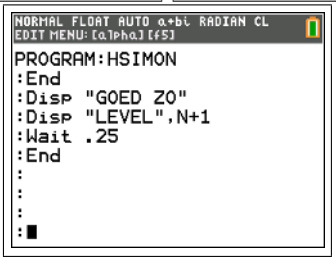
Hoe doe je dit op de TI-84 Plus Color ?





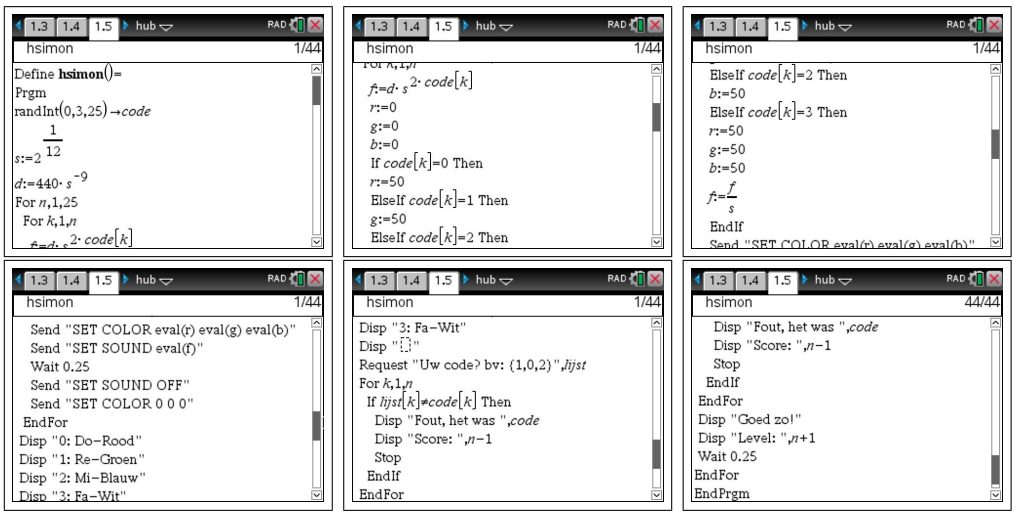






- Eerst wordt een de lijst L_1 gemaakt met een codesequentie van lengte 25. $\sqrt[12]{2}$ wordt eenmalig uitgerekend en de frequentie van do wordt bepaald.
- Met een `for`-lus over N wordt de getoonde sequentie steeds langer.
- De volgende `for`-lus zorgt voor het effectief tonen van de eerste N elementen van de code. Afhangende van de waarde $0, 1, 2, 3$ wordt de frequentie F van do, re, mi, fa gebruikt. De kleurcode R, G, B wordt op nul gezet en met drie verkorte `if`-structuren worden in de eerste drie gevalen de kleurcode op rood, groen, blauw gezet. In het laatste geval wordt de kleurcode op wit gezet en wordt de frequentie aangepast (anders zou je een halve toon boven fa zitten).
- Na de `if`-structuren wordt de kleurled aangezet en wordt de noot afgespeeld. Het programma wacht een kwartseconde en zet dan geluid en licht weer af. Daarna wordt dit herhaald voor de eerste N elementen van de code.
- Nadat de sequentie werd getoond, wordt het scerm leeggemaakt en komt een menu tevoorschijn met uitleg voor de gebruiker.
- Eenmaal de speler zijn antwoord heeft gegeven in L_2, gaat een `for`-lus na of er geen fout in zit. Indien het antwoord mis is, wordt de juiste sequentie getoond en stopt het programma.
- Indien er geen fouten gevonden werden, wordt de volgende level aangekondigd en start het spel opnieuw dankzij de `for`-lus over N.

Hoe doe je dit op de TI-Nspire ?

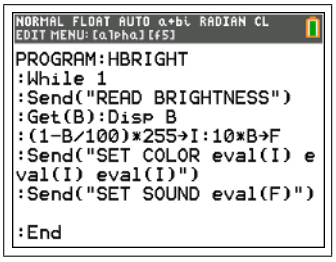


- De opbouw van dit programma is nagenoeg identiek aan de opbouw van het voorbeeld bij de **TI-84 Plus Color** .
- De **TI-Nspire** kent operator `ClrHome` niet dus het scherm zal steeds meer gevuld worden. Je kan het rekenblad wel handmatig leegmaken met menu `[1:Acties][5:Geschiedenis wissen]`.
- Hier maken we geen gebruik van de typische lijsten zoals bij de **TI-84 Plus Color** . Een variabele (voor ons zijn dat `code` en `lijst`) bij de **TI-Nspire** kan rechtsreeks een hele lijst aan getallen opslaan, in de vorm $\{x_1, x_2, x_3, \ldots, x_n\}$. Wij moeten onze poging dus ook in die vorm invoeren. Het n-de element van deze lijst vinden we dan door te werken met vierkante haakjes. Bijvoorbeeld: als in variabele `A` een lijst van 5 getallen zou zitten, nl. $\{2, 0, 5, 7, 3\}$, dan zou `A[4]` gelijk zijn aan 7.

8 De lichtsensor: zelf random muziek maken

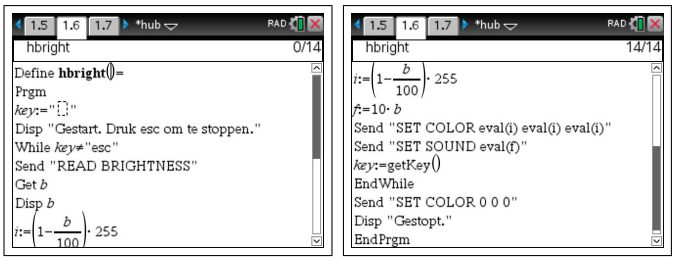
Op de **TI-Innovator Hub** zit ook een lichtsensor. Met het bericht `READ BRIGHTNESS` en het commando `Get` kan de lichtintensiteit gemeten worden. Dit geeft een waarde tussen 0 en 100. In volgend programma gebruiken we dit om de intensiteit van de kleur LEDs te regelen als het donker wordt en bovendien ook een toon van het omgevingslicht te laten afhangen.

Hoe doe je dit op de TI-84 Plus Color ?



- Een oneindige lus wordt gestart.
- De lichtintensiteit wordt gelezen, opgeslagen in B en afgedrukt.
- Met de formules wordt het licht I feller als donkerder het is, de toon F wordt dan lager.
- Tenslotte worden de kleur LEDs aangeschakeld en wordt de toon gespeeld.

Hoe doe je dit op de TI-Nspire ?



- De opbouw van dit programma is nagenoeg identiek aan de opbouw van het voorbeeld bij de **TI-84 Plus Color** .
- Ook deze lus kunnen we weer stoppen met esc.