Unit 4: Lussen | **Oefenblad 2: While**

In Deze les leer je over de flexibele van de lussen: **While**…

Doelen:

- Een eenvoudige **while**-lus schrijven
- De **while**-lus gebruiken om er zeker van te zijn dat er geldige gegevens worden ingevoerd

Het machtige While…

De lus **While…EndWhile** zal zich blijven herhalen zolang de <voorwaarde> ervan waar is. Dat ziet er zo uit:

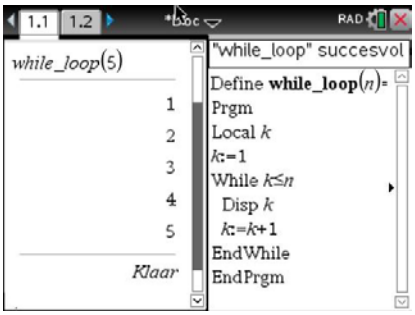
<initialiseer de voorwaarde>

While *<voorwaarde>*

<kern van de lus>

EndWhile

- *Initialiseren* verwijst naar het instellen van een of meer variabelen zodat de **While**-opdracht in staat is de voorwaarde de eerste keer correct uit te werken. Het initialiseren stelt een waarde van waar of onwaar vast voor de variabele. Als de beginvoorwaarde onwaar is dan wordt de lus volledig overgeslagen. Als de voorwaarde waar is dan wordt de kern van de lus verwerkt.
- De <voorwaarde> is een logische uitdrukking, bijvoorbeeld **X>0**.
- De <kern van de lus> is een willekeurige verzameling opdrachten, die andere lussen en **If** structuren kan bevatten. De <kern van de lus> wordt verwerkt telkens als de <voorwaarde> waar is.
- Het sleutelwoord **EndWhile** wordt gebruikt om het einde (de bodem) van de <kern van de lus> aan te duiden. Bij de opdracht **EndWhile** maakt het programma een 'lus' terug naar de opdracht **While** en test de <voorwaarde> opnieuw. Als de voorwaarde onwaar is wordt de lus overgeslagen. Als de voorwaarde waar is wordt de kern van de lus opnieuw verwerkt.



De **k:=1** aan het begin van het programma stelt de beginvoorwaarde in op een bekende waarde van onwaar. Zonder deze initialisatie, zou de variabele **k** een of andere opgeslagen waarde kunnen hebben en daarmee een onbekende waarde in het programma kunnen introduceren.

Ergens in de <kern van de lus> moet een opdracht voorkomen die effect heeft op de voorwaarde zodat de lus uiteindelijk zal stoppen en de opdrachten na de lus zullen worden verwerkt. Gewoonlijk staat deze opdracht dichtbij het eind (de bodem) van de <kern van de lus>. **k:=k+1** zorgt ervoor dat **k** zal toenemen en uiteindelijk groter wordt dan **n**.

Het programma demonstreert het **While** -equivalent van de **For**-lus:

```
For k,1, n
    Disp k
EndFor
```

Docenten Tip: VOORZICHTIG! Oneindige lussen zijn gevaarlijk! Als een programma vast zit in een oneindige lus, druk dan op de toets **ON/HOME** en houd deze ingedrukt tot het programma stopt (onderbroken wordt).).
Wees bijzonder voorzichtig met de computersoftware omdat de toets on/Home niet op dezelfde manier werkt als op een rekenmachine.
Docenten Tip: Het is belangrijk om te benadrukken dat de **While**-lus mogelijk helemaal niet wordt verwerkt. In de volgende les bespreken we de **Repeat**-lus die altijd minstens één keer wordt verwerkt. Dit is een subtiel maar belangrijk onderscheid.
Er zijn drie componenten nodig om een succesvolle **While**-lus te maken: ***Initialiseren, testen en veranderen***:

- ***Initialiseren*** van een variabele (beginwaarde instellen).
- ***Testen*** van een voorwaarde gebaseerd op die variabele.
- ***Veranderen*** van de variabele zodat uiteindelijke de voorwaarde onwaar wordt, zodat de lus zal stoppen.

Nagaan of Input geldig is met While…End

We zullen een gedeelte van een programma (code-segment) schrijven dat ervoor zorgt dat de gebruiker een positief getal invoert, haar vertelt wanneer de invoer ongeldig is en dan verzoekt om een andere waarde in de plaats ervan in te voeren.

De uitvoer van dit programmagedeelte staat hier rechts waarbij enkele niet-positieve getallen zijn ingevoerd om het effect te zien. Kijk eens of je dit programmagedeelte kunt schrijven zonder te spieken op de volgende pagina!

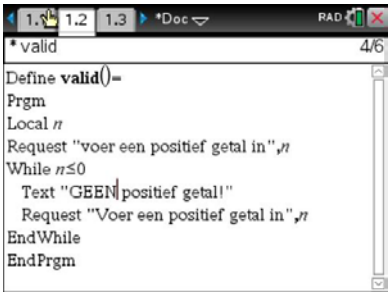
1. We beginnen met het maken van een nieuw programma. De naam die we hier hebben gebruikt is **valid.**
2. Creëer een lokale variabele **n** en gebruik de opdracht **Request** om een waarde te ontvangen van de gebruiker. Merk op dat er gevraagd wordt om een *positief getal*.

3. Voeg een **While**-structuur in vanuit het menu **Besturing**. Zowel het **While-** als het **EndWhile**-commando worden in het programma geplakt en de cursor verschijnt achter het woord **While**.

4. Typ de voorwaarde **n<=0**.
   - Krijg de <= bewerking door ctrl = te selecteren.

5. Voltooi tenslotte de *kern van de lus* door te zorgen voor een foutmelding met behulp van een **Text** –opdracht en door een extra **Request**-opdracht op te nemen om de gebruiker opnieuw een waarde voor *n* in te laten voeren.

Het programma **valid( )** dat je hier rechts ziet, leverde de interactie zoals beschreven op de vorige pagina.



Merk op dat er TWEE **Request**-opdrachten zijn.

- De eerste wordt gebruikt om de voorwaarde (n<=0) te initialiseren. Als er hier een positief getal is ingevoerd, dan zal de lus helemaal niet worden verwerkt..
- Maar als er 0 of een negatief getal is ingevoerd dan zal de kern van de lus de foutmelding weergeven en om een andere waarde vragen.

De lus zal herhaald worden zolang er een niet-positieve waarde wordt ingevoerd.

Docenten Tip: Het is geweldig om te kunnen kopiëren en plakken in de editor!
Je kunt '>=' typen op een computer en ctrl-B (Syntax controleren & opslaan) zet deze twee tekens om in één.