Hoofdstuk 5

Lussen

For-lus

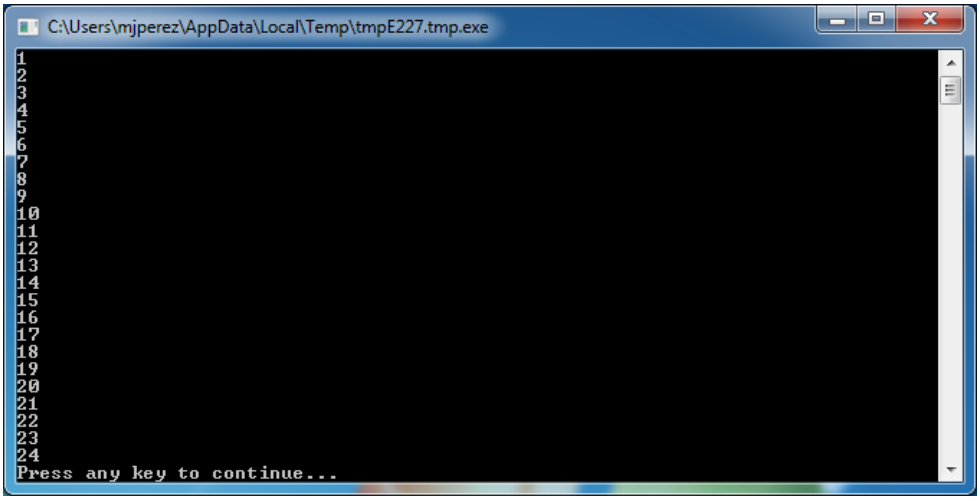
Laten we nog eens kijken naar het programma dat we in het vorige hoofdstuk hebben geschreven.

```
i = 1
start:
TextWindow.WriteLine(i)
i = i + 1
If (i < 25) Then
  Goto start
EndIf
```

Met dit programma worden de getallen van 1 tot 24 in volgorde naar het tekstvenster geschreven. Dit proces waarmee variabelen toenemen wordt veel gebruikt tijdens het programmeren en daarom vind je in programmeertalen normaal gesproken een gemakkelijkere manier om dit te doen. Het bovenstaande programma is gelijk aan het programma hieronder:

```
For i = 1 To 24
  TextWindow.WriteLine(i)
EndFor
```

En de uitvoer is:



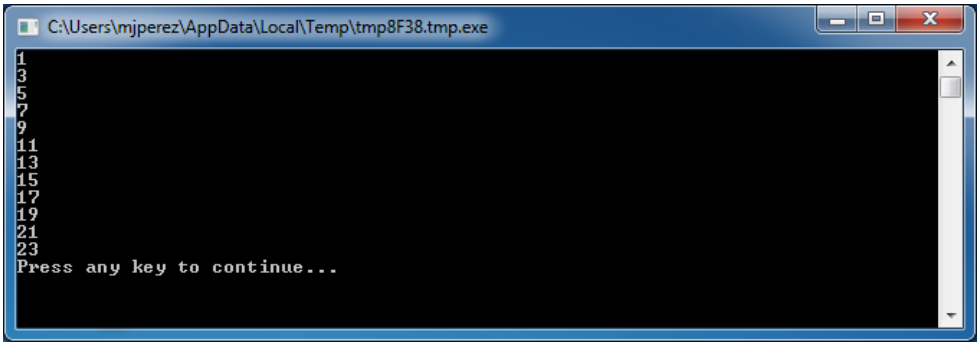
Afbeelding 19 – De For-lus gebruiken

Valt het je op dat we het achtregelige programma hebben gereduceerd tot een vierregelig programma en dat dit programma nog steeds precies hetzelfde doet als het achtregelige programma? Misschien kun je je nog herinneren dat we eerder hebben gezegd dat er vaak meerdere manieren zijn om hetzelfde te doen. Dit is een goed voorbeeld.

For.EndFor wordt, in programmeertermen, een *lus* genoemd. Hiermee kun je een variabele een begin- en eindwaarde geven en de computer deze variabele laten toenemen. Telkens wanneer de computer de variabele laat toenemen, worden de instructies tussen **For** en **EndFor** uitgevoerd.

Je kunt de lus ook gebruiken als je de variabele met 2 wilt laten toenemen in plaats van met 1, als je bijvoorbeeld alle oneven getallen tussen 1 en 24 naar het tekstvenster wilt schrijven.

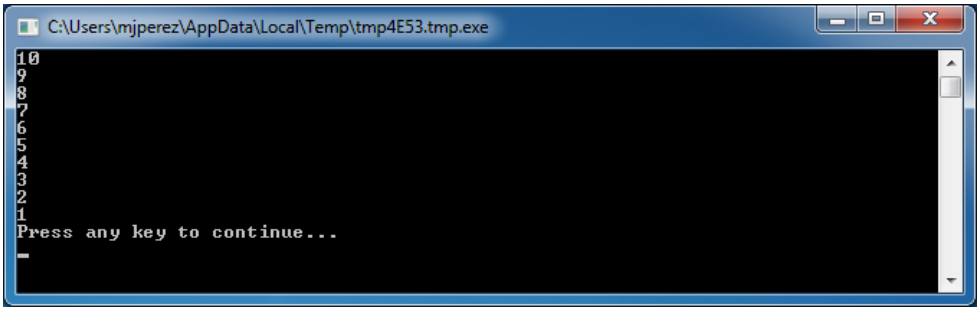
```
For i = 1 To 24 Step 2
  TextWindow.WriteLine(i)
EndFor
```



Afbeelding 20 – Alleen de oneven getallen

Met het gedeelte **Step 2** van de instructie **For** neemt de waarde van **i** met 2 toe in plaats van met de gebruikelijke waarde 1. Met **Step** kun je elke toename specificeren die je wilt. Je kunt ook een negatieve waarde voor de stap specificeren en hiermee de computer terug laten tellen, zoals in het voorbeeld hieronder:

```
For i = 10 To 1 Step -1
  TextWindow.WriteLine(i)
EndFor
```

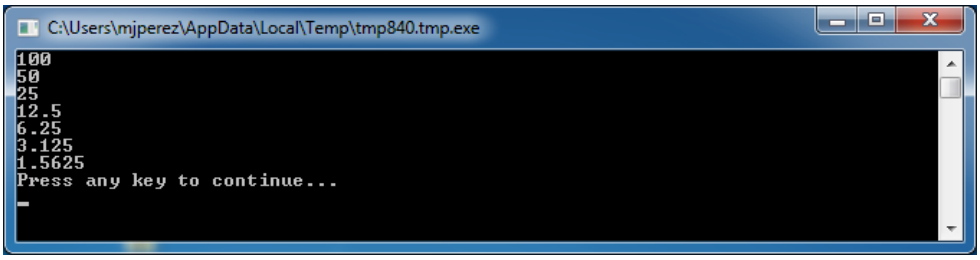


Afbeelding 21 – Terugtellen

While-lus

De While-lus is een lusmethode die handig is wanneer je het aantal lussen niet van tevoren weet. Daar waar de For-lus een van tevoren vastgesteld aantal keren wordt uitgevoerd, wordt de While-lus uitgevoerd totdat een gegeven voorwaarde waar is. In het voorbeeld hieronder halveren we het getal totdat het resultaat groter is dan 1.

```
getal = 100
While (getal > 1)
  TextWindow.WriteLine(getal)
  getal = getal / 2
EndWhile
```



Afbeelding 22 – Halverende lus

In het programma hierboven hebben we de waarde 100 aan *getal* toegewezen en voeren we de While-lus net zolang uit totdat getal groter is dan 1. We schrijven het getal binnen de lus naar het tekstvenster en delen het vervolgens door twee, waarmee we het halveren. En zoals verwacht, de uitvoer van het programma zijn getallen die toenemend na elkaar worden gehalveerd.

Het is moeilijk om dit programma te schrijven met een For-lus, omdat we niet weten hoe vaak de lus zal worden uitgevoerd. Met een While-lus kun je een voorwaarde gemakkelijk controleren en de lus voortzetten of beëindigen.

Het is ook interessant om te vermelden dat elke While-lus kan worden uitgepakt in een If..Then-instructie. Het programma hierboven kan bijvoorbeeld als volgt worden herschreven zonder het eindresultaat te beïnvloeden.

```
getal = 100
startLabel:
TextWindow.WriteLine(getal)
getal = getal / 2

If (getal > 1) Then
  Goto startLabel
EndIf
```

Elke While-lus wordt in feite intern herschreven naar instructies die If..Then samen met een of meer Goto-instructies gebruiken.