De while-lus

Omschrijving

Het while-statement wordt ook een conditie gestuurde loop genoemd.

Bij deze vorm van loops wordt het uitvoeringsblok uitgevoerd zolang aan een bepaalde conditie wordt voldaan. Het aantal herhalingen moet vooraf niet gekend zijn, doch is afhankelijk van de voorwaarde.

In pseudocode wordt dit als volgt geschreven:

ZOLANG (Voorwaarde == true)

    SEQUENTIE S;

In Java:

```
while (expressie)
{
  statement1;
  statement2;
  …
}
```

Bij het uitvoeren van een while-statement zal op de eerste plaats de expressie (de conditie) geëvalueerd worden. Indien de expressie waar is, worden de statements uitgevoerd. Hierna wordt de expressie opnieuw geëvalueerd. Deze lus eindigt wanneer de expressie bij het evalueren onwaar is.
Een while-statement wordt dus nul of meer keren uitgevoerd (dit is ook zo bij een for-lus).

Voorbeeld:

```
public class TestWhile {
	public static void main(String[] args) {
		//for-lus ter vergelijking
		for(int i=1; i<=5; i++){
			System.out.println("de waarde van teller is "+ i);
		}

		System.out.println(" ");
		//de while lus
		int j=1;
		while (j<=5){
			System.out.println("de waarde van teller is "+ j);
			j++;
		}
	}
}
```

Output:

```
de waarde van teller is 1
de waarde van teller is 2
de waarde van teller is 3
de waarde van teller is 4
de waarde van teller is 5

de waarde van teller is 1
de waarde van teller is 2
de waarde van teller is 3
de waarde van teller is 4
de waarde van teller is 5
```

Een while-lus gebruikt dezelfde elementen van een for-lus **op een andere plaats**:
De teller wordt buiten de lus geïnitialiseerd.
Bij het binnengaan van de lus wordt de testexpressie geëvalueerd. Indien deze de waarde 'true' oplevert, worden de statements binnen de lus uitgevoerd. Wanneer deze de waarde 'false' oplevert, wordt de lus beëindigd.
De verhoogexpressie staat bij de while-lus in de body van de lus. Let op: dit is steeds de laatste expressie!

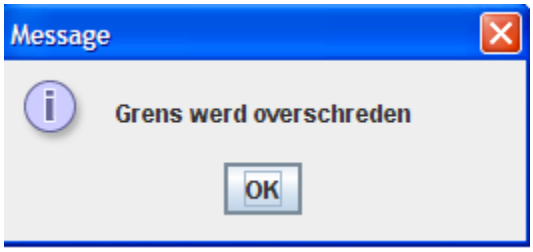
Oefening

Maak een project While.

Pas het voorbeeld aan zodat de getallen in omgekeerde volgorde worden afgedrukt.

Output met een dialoogvenster

Net zoals input met een dialoogvenster kan gebeuren, kan eveneens de output via een dialoogvenster in een JOptionPane weergegeven worden. In het programma wordt dan een apart venster geopend met daarin een mededeling aan de gebruiker.



In zijn meest eenvoudige vorm:

```
JOptionPane.showMessageDialog(null,message);
```

Voorbeeld: `JOptionPane.showMessageDialog(null,"Grens werd overschreden");`

`showMessageDialog()` is een methode met 2 argumenten:

- de vaste waarde null
- de boodschap die in het venster verschijnt

Oefening

Lees een aantal getallen in. De invoer wordt afgesloten door de gebruiker met 0.
0 moet niet verwerkt worden maar heeft de functie van een sentinel. Dit betekent dat dit geen echte waarde is, maar een waarde die aangeeft dat de invoer ten einde is.
Tel al deze ingevoerde getallen op.

Lus opstellen die voldoet aan de regels van gestructureerd programmeren

Een programma kan werken maar structureel gezien niet echt goed zijn.

1. Aantal herhalingen is vast

   Waneer je bijvoorbeeld weet hoeveel getallen moeten ingelezen worden, gebruik je bij voorkeur een for-lus. Belangrijk is dan dat de eerste instructie in de lus een lees-instructie is.

   Voorbeeld:

```
int getal;
int som = 0;
for (int i = 1; i <= 5; i++) {
   getal = Integer.parseInt(JOptionPane.showInputDialog("Geef getal"));
   som = som + getal;
}
System.out.println("som is " + som);
```

For-lus
{
LEES waarde
Reeks instructies
}

2. Aantal herhalingen is variabel

   Wanneer je bijvoorbeeld niet weet hoeveel getallen moeten ingelezen worden, omdat de input zal stoppen op basis van een ingevoerde waarde door de gebruiker, gebruik je bij voorkeur een while-lus. Belangrijk is dan dat de eerste lees-instructie vóór de lus gebeurt, en een nieuwe lees-instructie als laatste in de lus gebeurt.

   Voorbeeld:

```
int getal;
int som = 0;
getal = Integer.parseInt(JOptionPane.showInputDialog("Geef getal"));
while (getal != 0) {
   som = som + getal;
   getal = Integer.parseInt(JOptionPane.showInputDialog("Geef getal"));
 }
System.out.println("som is " + som);
JOptionPane.showMessageDialog(null,"som is "+ som);
```

```
LEES waarde
While-lus
    {
            Reeks instructies
             LEES waarde
    }
```

Oefeningen

Deze oefeningen lossen we op met een while lus.

Oefening 1

Lees een aantal gehele getallen in en druk het product hiervan af. 0 mag je niet ingeven. Als de gebruiker een 0 ingeeft, dan geef je een boodschap dat het getal verschillend van 0 moet zijn.
Geef STOP in wanneer je wil stoppen. Om twee strings met elkaar te vergelijken, gebruik je de functie equals.

Maak gebruik van onderstaande vertrekcode. Om lussen te oefenen mag je alles in de main-functie zetten.

```
import javax.swing.JOptionPane;
public class Oefening1 {

    public static void main(String[] args) {
        String input=JOptionPane.showInputDialog("Geef een getal in");
        while (!input.equals("STOP"))
        {
            int getal = Integer.parseInt(input);
            input=JOptionPane.showInputDialog("Geef een getal in");
        }
        JOptionPane.showMessageDialog(null,"output");
    }

}
```

Oefening 2

Lees een aantal getallen in en bepaal het grootste getal.
Let op, er kunnen ook negatieve getallen opgegeven worden.
Wanneer je STOP ingeeft, dan stopt de lus.

Oefening 3

Vraag een begin en een eindwaarde op. De eindwaarde moet minstens 50 hoger zijn dan de beginwaarde. Blijf dit net zo lang vragen tot de juiste eindwaarde wordt ingegeven. Hier zal je een lus voor moeten schrijven.
Schrijf vervolgens een tweede lus die alle veelvouden tussen begin- en eindwaarde afdrukt.

50 101

50*1
50*2

5 50
5*1
5*2
5*3
5*4

Oefening 4

Lees een getal in. Bepaal of dit getal een priemgetal is.
Je kan hiervoor een while-lus gebruiken die n deelt door alle getallen tussen 2 en n-1 en test of de deling een geheel getal oplevert – gebruik hiervoor de modulo-operator %.

Oefening 5 => vanaf nu kies je zelf of je while of for gebruikt.
In principe kan je elke lus met for schrijven.

Lees 2 gehele getallen in en bepaal de grootste gemene deler. De GGD is het grootste getal waar de gegeven getallen door gedeeld kunnen worden.
Voorbeeld:

```
de ggd van 8 en 12 is 4
```

het Algoritme van Euclides.
Werkwijze:
Aanname : getal1 > getal2
Deel eerste getal door het tweede. Werk verder met de rest en het kleinste getal zolang rest > 0.
Vb: 12 en 8 hebben als ggd 4.
12 en 8: 12/8 -> rest = 4, kleinste = 8
8 en 4: 8/4 -> rest = 0, kleinste = 4
4 en 0: einde -> ggd = 4

Oefening 6
//tip delen door 10 en werken met rest.

Tel de cijfers van een getal op.
Voor getal 4578 is dat 4+5+7+8 = 24

Oefening 7

Lees drie getallen in.
eerste getal = het spaargeld dat je op een rekening zet. Voorbeeld = 1000 euro
tweede getal = de verkregen rentevoet. Voorbeeld = 0.01
derde getal = wanneer je dit doel bereikt hebt, wil je het geld afhalen.
Genereer onderstaand overzicht en toon in een boodschap hoeveel jaren je moet sparen.
Gebruik onderstaande startcode.

```
jaar 1 : 1000.0
jaar 2 : 1010.0
jaar 3 : 1020.1
jaar 4 : 1030.301
jaar 5 : 1040.60401
jaar 6 : 1051.0100501
jaar 7 : 1061.5201506009998
jaar 8 : 1072.1353521070098
jaar 9 : 1082.85670562808
jaar 10 : 1093.6852726843606
jaar 11 : 1104.6221254112043
Voor een bedrag van 1100.0 moet je je spaargeld 11 jaren laten staan
```

```
public class Oefening6 {

    public static void main(String[] args) {
        // TODO code application logic here
        String spaarbedragString
                = JOptionPane.showInputDialog("Hoeveel wil je sparen?");
        String rentevoetString
                = JOptionPane.showInputDialog("Welke rentevoet heb je verkregen?");
        String doelString
                = JOptionPane.showInputDialog("Hoeveel moet je spaargeld opbrengen?");
        double spaarbedrag = Double.parseDouble(spaarbedragString);
        double rentevoet = Double.parseDouble(rentevoetString);
        double doel = Double.parseDouble(doelString);
}
```

Geneste lussen

Je kan for-lussen en while-lussen nesten. Een combinatie van een for-lus en een while-lus kan ook.

Geneste for-lus

Een for-lus kan binnen een andere for-lus genest worden.

Voorbeeld

```
public class TestForLussenGenest {

	public static void main(String[] args) {
		int kolom, rij;
		for (rij = 1; rij <= 2; rij++) {
			for (kolom = 1; kolom <= 3; kolom++) {
				System.out.print("*");
			}
			System.out.println();
		}
	}
}
```

Noteer welke waarden de variabele rij en kolom achtereenvolgens krijgen, en noteer de output van het programma in het kader ernaast.

rij	kolom

De binnenste lus gaat 3 sterretjes naast elkaar afdrukken (`System.out.print("*");` -> geen `println`)

De buitenste lus zorgt ervoor dat de binnenste lus 2 maal wordt uitgevoerd, telkens op een nieuwe lijn, dus 2 lijnen onder elkaar.

Oefeningen

Vanaf nu kies je zelf of je met een for of een while lus werkt.

1. Zorg voor volgende output

```
1 1 1
2 2 2
3 3 3
4 4 4
5 5 5
```

2. Zorg voor volgende output

```
1	2	3
4	5	6
7	8	9
10	11	12
13	14	15
```

Tip:

```
System.out.print("\t"+…..) ;
```

"\t": voegt tab toe

3. Zorg voor volgende output

```
1	2	3	4	5	6	7	8	9	10
2	4	6	8	10	12	14	16	18	20
3	6	9	12	15	18	21	24	27	30
4	8	12	16	20	24	28	32	36	40
5	10	15	20	25	30	35	40	45	50
6	12	18	24	30	36	42	48	54	60
7	14	21	28	35	42	49	56	63	70
8	16	24	32	40	48	56	64	72	80
9	18	27	36	45	54	63	72	81	90
10	20	30	40	50	60	70	80	90	100
```

4. Zorg voor volgende output

```
*
**
***
****
*****
******
*******
********
*********
**********
```

5. Bereken het gemiddelde van een aantal komma-getallen.
   Van welk aantal het gemiddelde berekend moet worden, wordt eerst ingelezen, en hierna de getallen zelf.
   De gebruiker moet telkens opnieuw een aantal kunnen ingeven met een reeks getallen, waarna het gemiddelde wordt berekend.
   Het programma eindigt wanneer 0 of een negatief getal als aantal wordt opgegeven.
   Maak gebruik van geneste lussen. Kies zelf dewelke je gebruikt.

Concreet.
-Je vraagt een aantal op. Voorbeeld : 3
-Vervolgens vraag je 3 getallen op waarvan het gemiddelde wordt berekend.
-Je vraagt opnieuw een aantal op. Voorbeeld : 10
-Vervolgens vraag je 10 getallen op waarvan het gemiddelde wordt berekend.
-Je vraagt opnieuw een aantal op. Voorbeeld : 0
Dit betekent : het programma stopt.

6. Bereken de faculteit van een getal. De gebruiker kan telkens weer een nieuw getal opgeven. Het programma eindigt wanneer de gebruiker een negatief getal of 0 ingeeft.
   De faculteit van een getal is het getal vermenigvuldigd met alle getallen kleiner dan zichzelf, uitgezonderd 0.
   De faculteit van 4, nl 4! = 4*3*2*1 = 24

Extra oefeningen

Oefening 1

Bereken de mediaan. Gebruik een 1-dim array.

Voorbeelden:
1, 2, 3, 4, 5, 6, 7, 8, 9
Mediaan is 5, het middelste getal. Links ervan zitten 4 getallen en rechts ervan zitten 4 getallen.
1, 2, 3, 4, 5, 6, 7, 8, 9, 10
De middelste getallen zijn 5 en 6. Daar tussen moet de Mediaan zitten. Je kunt dan doen: 5+6= 11 /2 is 5.5. Dat is de Mediaan

Oefening 2

Bereken de standaarddeviatie. Gebruik een 1-dim array.

Voorbeeld : gegeven de getallenreeks 2, 4, 5, 5, 6, 7, 9, 10

1. Bereken het gemiddelde Xgem:
Xgem = (2+4+5+5+6+7+9+10) / 8 = 6

2. Bereken van elk getal (xi) de deviatie di:
di = xi - Xgem, dus alle deviaties zijn:
(2-6),(4-6),(5-6),(5-6),(6-6),(7-6),(9-6),(10-6)
=> alle deviaties zijn dus: -4, -2, -1, -1, 0, 1, 3, 4

3. Neem nu van alle deviaties het kwadraat, deze zijn achtereenvolgens:
16, 4, 1, 1, 0, 1, 9, 16

4. Bereken nu het gemiddelde van al deze kwadraten:
=> (16+4+1+1+0+1+9+16) / 8 = 6

5. Neem nu de wortel van dit gemiddelde
σ = √ (∑(d2 / n)) = √6

Oefening 3

Maak volgende ruit.
Vraag met JOptionPane hoeveel sterretjes de ruit moet bevatten op haar breedste punt.

```
   *
  ***
 *****
*******
 *****
  ***
   *
```

Oefening 4

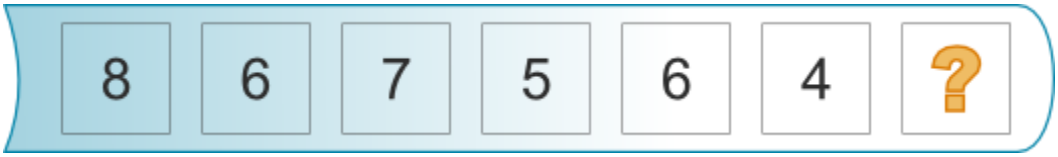
Vraag aan de gebruiker uit hoeveel rijen je kerstboom moet bestaan (zonder de stam)?

De stam neemt qua breedte 1 derde in van de onderkant van de boom
De stam is even hoog als 1 vierde van de kruin.
Maak een programma dat zich automatisch aanpast aan de vraag van de gerbuiker.

```
                    *
                   ***
                  *****
                 *******
                *********
               ***********
              *************
             ***************
            *****************
           *******************
                  ******
                  ******
```

Oefening 5

Terug in het CLB, bij de intelligentietestjes.
Schrijf een lus die de volgende reeks genereert.
Vraag aan de gebruiker met JOptionPane 2 zaken op.
-begin-getal
-uit hoeveel getallen moet de reeks bestaan?



Oefening 6

Definieer een array van 10 elementen.
Verschuif de inhoud van de array 2 posities naar rechts.

Voorbeeld.
De inhoud is {1, 2, 3, 4, 5, 6, 7, 8, 9, 10}
Na verschuiving
{3, 4, 5, 6, 7, 8, 9, 10, 1, 2}

Oefening 7

Uitbreiding vorige oefening.
Vraag aan de gebruiker hoeveel posities hij wil opschuiven.
Stel dat de array bestaat uit 10 posities, dan moet de vraagstelling als volgt gebeuren:
Hoeveel posities wil je opschuiven? Geef een getal van 1 tot 9.
Blijf aan de gebruiker een getal vragen totdat de gebruiker een getal tussen 1 en 9 opgeeft.