

C# Les 3

Arrays

Als je bijeenhorende waarden hebt kan je (in plaats van de waarden te gaan bijhouden in heel veel verschillende variabelen) een lijst van deze waarden in één variabele plaatsen. Een lijst van floats (= komma-getallen) maak je zo:

```
float[] prijzen = { 50, 20.5F, 7 };
```

Je kan de aparte elementen in de lijst opvragen met de vierkante haken (dat is de *array access operator*). Zoals in de meeste programmeertalen start het tellen van de elementen in een array vanaf nul.

```
float[] prijzen = { 50, 20.5F, 7 };
```

```
Console.WriteLine(prijzen[0]);  
Console.WriteLine(prijzen[1]);  
Console.WriteLine(prijzen[2]);
```

Veranderen van elementen in de lijst kan ook.

```
prijzen[0] = 70;
```

Anders dan in sommige andere programmeertalen heeft een lijst in C# een vaste lengte die je niet kan veranderen na het aanmaken van de array 😞. Een element in de lijst *toevoegen* kan dus niet met een array (maar wel met collections, hierover later meer).

Je kan wel de lengte van een array te weten komen door er `.Length` achter te plaatsen:

```
Console.WriteLine($"De lengte van de lijst is {prijzen.Length}");
```

for

Naast de while-loop heb je ook nog de for-loop. Deze loop lijkt in het begin misschien wat lastig om te schrijven maar eens je hem onder de knie hebt ga je hem vast vaak gebruiken! De loop is handig als je een teller-variabele nodig hebt binnen de loop en als je op voorhand weet hoeveel keer je wil herhalen. Deze is dus extra handig in combinatie met arrays.

Hij ziet er zo uit:

```
for(int i=0; i<10; i=i+1)
```

```
{  
    Console.WriteLine(i);  
}
```

Tussen de ronde haken van de for-loop zie je 3 delen (gescheiden door 2 punt-komma's).

Wat gebeurt er precies bij de for-loop?

1. Het eerste stukje maakt een variabele i:

```
int i=0
```

2. Daarna komt een test (zoals bij de while). Er wordt gekeken of deze uitspraak klopt of niet (true of false is). Als de test slaagt wordt de code tussen de accolades uitgevoerd, anders stopt de for-loop en gaat het programma verder onder de accolades.

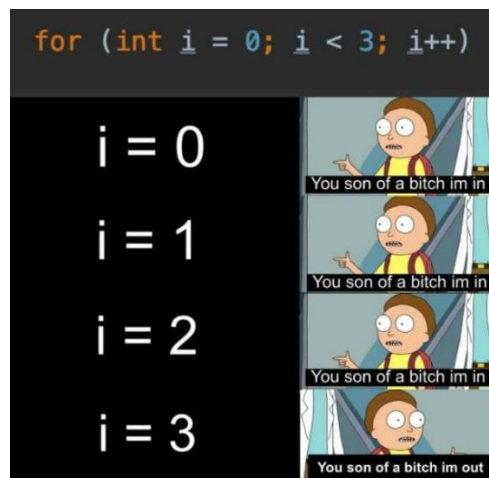
```
i<10
```

In ons voorbeeld is bij de eerste keer de variabele i net op nul gezet en nul is inderdaad kleiner dan 10 (dus de uitspraak is true). Dus de code die tussen de accolades staat wordt een eerste keer uitgevoerd.

3. Na elke uitvoer van de code tussen de accolades gaat het laatste stukje tussen de ronde haken worden uitgevoerd:

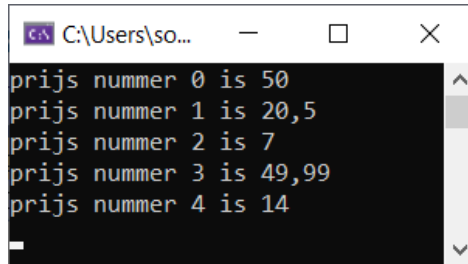
```
i=i+1
```

i wordt verhoogd met 1 (meestal wordt dit korter geschreven als i++). Het programma gaat nu terug naar punt 2 en herhaalt totdat de test niet meer klopt.



In combinatie met een lijst van waarden ziet de loop er zo uit:

```
float[] prijzen = { 50, 20.5F, 7, 49.99F, 14 };  
  
for (int i = 0; i < prijzen.Length; i++)  
{  
    Console.WriteLine($"prijs nummer {i} is {prijzen[i]}");  
}
```



```
C:\Users\so...  
prijs nummer 0 is 50  
prijs nummer 1 is 20,5  
prijs nummer 2 is 7  
prijs nummer 3 is 49,99  
prijs nummer 4 is 14
```

Oefening 1: prijzen filteren

Kan je door een if in de for-loop te plaatsen er voor zorgen dat enkel de prijzen lager dan 20 worden getoond?

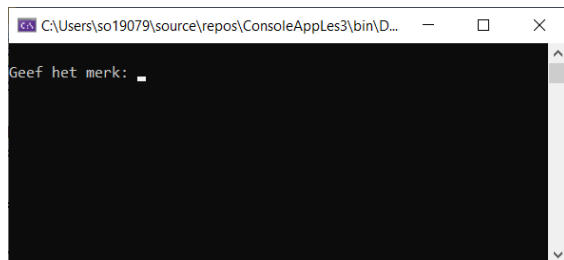
Oefening 2: CLASSIC AUTOSHOP - bijeenhorende lijsten filteren

Vertrek van onderstaande bijeenhorende lijsten. Alle eerste elementen horen bij elkaar (Volvo, model 66, bouwjaar 1975, twee-deurs en sedan), alle tweede elementen horen bij elkaar (Fiat model Panda van 1981, drie-deurs hatchback) enzovoort...

```
string[] merk = { "Volvo", "Fiat", "BMW", "Volvo", "Volvo", "BMW" };  
string[] model = { "66", "Panda", "732i", "265", "340", "525" };  
int[] bouwjaar = { 1975, 1981, 1979, 1980, 1982, 1974 };  
int[] aantalDeuren = { 2, 3, 3, 5, 5, 4 };  
string[] type = { "sedan", "hatchback", "sedan", "wagon", "hatchback", "sedan" };  
};
```

A. Schrijf een programma zodat je op merk kan filteren.

TIP: bouw je progamma stap voor stap op: probeer bijvoorbeeld eerst al de merken te tonen en bouw dan verder.

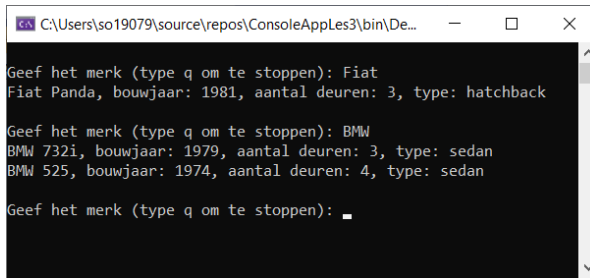


```
C:\Users\so19079\source\repos\ConsoleAppLes3\bin\D...  
Geef het merk: _
```



```
C:\Users\so19079\source\repos\ConsoleAppLes3\bin\D...  
Geef het merk: Volvo  
Volvo 66, bouwjaar: 1975, aantal deuren: 2, type: sedan  
Volvo 265, bouwjaar: 1980, aantal deuren: 5, type: wagon  
Volvo 340, bouwjaar: 1982, aantal deuren: 5, type: hatchback  
_
```

- B. Breidt het programma verder uit zodat je kan blijven zoeken totdat er “q” wordt ingegeven.



```
C:\Users\so19079\source\repos\ConsoleAppLes3\bin\De...
Geef het merk (type q om te stoppen): Fiat
Fiat Panda, bouwjaar: 1981, aantal deuren: 3, type: hatchback

Geef het merk (type q om te stoppen): BMW
BMW 732i, bouwjaar: 1979, aantal deuren: 3, type: sedan
BMW 525, bouwjaar: 1974, aantal deuren: 4, type: sedan

Geef het merk (type q om te stoppen):
```

Kan je er voor zorgen dat als de gebruiker enter drukt zonder iets in te geven al de wagens worden getoond?

De informatie van de verschillende wagens staat nu verspreid over meerdere arrays. Dat is niet erg handig. Binnenkort gaan we betere manieren zien om de data te groeperen zodat bijeenhorende data (zoals merk, bouwjaar, deuren en type) per wagen netjes bijeen komen te staan.

Groeperen kan met multi-dimensionele arrays, lists, tuples of objecten.

Logische operatoren

Met de logische operatoren *en* en *of* kan je de if verder uitbreiden.

&&

De dubbele ampersand staat voor *en*. Als je hier twee voorwaarden mee koppelt moeten ze beiden waar zijn om heel de test waar te maken.

Hiermee kan je bijvoorbeeld testen of een getal binnen bepaalde grenzen valt.

```
if(bouwjaar >= 1970 && bouwjaar <= 1980) {
    // doe iets
}
```

Welke boodschap zal er hieronder worden getoond?

```
string merk = "BMW";
string type = "sedan";
int bouwjaar = 1979;

if(merk == "BMW" && type != "sedan" && bouwjaar <= 1979)
{
    Console.WriteLine("Auto gevonden!");
}
else
```

```
{  
    Console.WriteLine("Auto niet gevonden :(");  
}
```

||

Het dubbele “pipe”-teken staat voor de logische *of*. Als je hier twee voorwaarden mee koppelt is het voldoende dat één van de twee deelvoorwaarden waar is om heel de test waar te maken.

Welke boodschap zal hier getoond worden?

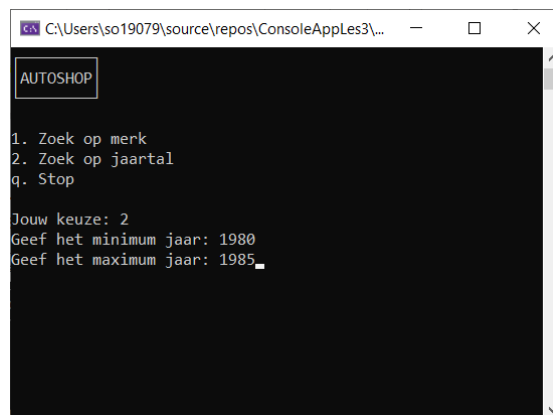
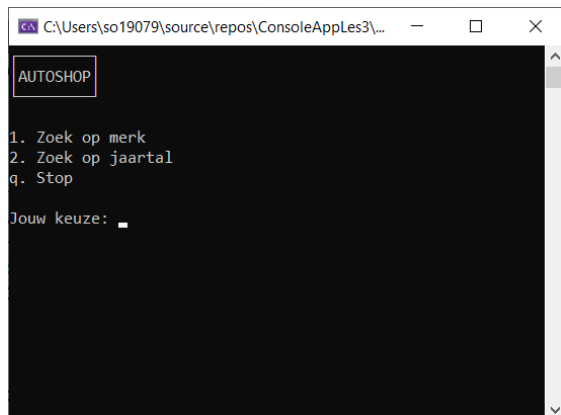
```
string vorm = "cirkel";  
  
if(vorm == "driehoek" || vorm == "rechthoek")  
{  
    Console.WriteLine("Hoekig");  
}
```

Oefening: auto's filteren op jaartal

Breidt het programma van de CLASSIC AUTOSHOP verder uit met een menu en zodat de gebruiker een extra keuze krijgt: de gebruiker kan een minimum bouwjaar en een maximum bouwjaar opgeven. De wagens die binnen deze grenzen zitten worden getoond.

TIPS:

- Om je programma overzichtelijk te houden is het best om met methodes te werken.
- Je kan de lijsten meegeven als argumenten aan de methodes.
- Als je lang op iets vastzit vraag dan hulp aan de leraar.



```
C:\Users\so19079\source\repos\ConsoleAppLes3\bin\Deb...  
AUTOSHOP  
Gevonden:  
Fiat Panda, bouwjaar: 1981, aantal deuren: 3, type: hatchback  
Volvo 265, bouwjaar: 1980, aantal deuren: 5, type: wagon  
Volvo 340, bouwjaar: 1982, aantal deuren: 5, type: hatchback  
  
1. Zoek op merk  
2. Zoek op jaartal  
q. Stop  
  
Jouw keuze: _
```

