



## IT Essentials

# Hoofdstuk 4

## Iteraties

### **DE HOGESCHOOL MET HET NETWERK**

Hogeschool PXL – Elfde-Liniestraat 24 – B-3500 Hasselt  
[www.pxl.be](http://www.pxl.be) - [www.pxl.be/facebook](https://www.pxl.be/facebook)



# Inhoud



1. Inleiding
2. For loop
  1. Met getallenreeks als collectie
  2. Met een string als collectie
  3. Met een variabele collectie
  4. Met een handmatige collectie
3. While loop
  1. Een eerste voorbeeld
  2. Een tweede voorbeeld
  3. Onder controle van de gebruiker
  4. Eindeloze while loop
4. Geneste loop



## 4.1 Inleiding

- opdracht 4.1: In een klas zitten 5 studenten. Bereken de gemiddelde leeftijd van deze studenten. Hoe?
- opdracht 4.2: Zelfde opgave maar met 28 studenten. Hoe?
- je hebt nood aan een efficiënte manier om opdrachten die herhaaldelijk voorkomen, uit te voeren → iteraties (herhalingen)





## 4.2 For loops

- syntax = 

```
for <variabele> in <collectie>:  
    <acties>
```
- lees als: voer de <acties> zo vaak uit als er waarden in de <collectie> zijn
- indentatie → zeer belangrijk (zie verder)
- collectie?
  - getallenreeks, string, variabele of handmatige collectie



## 4.2.1 For loop met getallenreeks als <collectie>

- wordt vaak gebruikt als je op voorhand weet hoe vaak de loop doorlopen zal worden
- functie `range()`: genereert reeks van getallen
  - `range(5)`
  - `range(3, 7)`
  - `range(4, 11, 3)` 
- opmerking
  - stapgrootte negatief = aftellen 
  - default beginparameter = 0
  - default stapgrootte = 1

## 4.2.1 For loop met getallenreeks als <collectie>

- opdracht 4.3: Wat is de output van volgend Python programma?

```
for i in range(4):  
    print(i)
```

```
for j in range(1, 5):  
    print(j)
```

```
for k in range(7, 18, 3):  
    print(k)
```

```
for l in range(8, 5, -2):  
    print(l)  
    print(l*1)  
print(l)
```

```
for m in range(8, 4):  
    print(m)
```



## 4.2.1 For loop met getallenreeks als <collectie>

- opdracht 4.4: Schrijf een programma om volgende output te genereren.

```
1
De waarde van het kwadraat van 1 is 1
3
De waarde van het kwadraat van 3 is 9
5
De waarde van het kwadraat van 5 is 25

De waarde van i na afloop van de for-lus is 5
```

## 4.2.1 For loop met getallenreeks als <collectie>

- opmerking:
  - Je moet de lusvariabele nergens expliciet verhogen. De for – loop handelt dit automatisch af. Telkens de loop terugkeert bij de regel met de for, wordt het volgende item uit de collectie gepakt.
  - Lusvariabele behoudt na afloop van de lus de laatste waarde.



## 4.2.1 For loop met getallenreeks als <collectie>

- opdracht 4.5: Schrijf een programma dat de getallen van 400 t.e.m. 350 afdrukt te beginnen met 400.
- opdracht 4.6: Schrijf een programma dat alle veelvouden van 7 (kleiner dan 200) afdrukt.
- opdracht 4.7: Schrijf een programma dat alle getallen tussen -10 en +10 afdrukt. Voeg bij de positieve getallen het “+”teken. (0 heeft geen teken).
- opdracht 4.8: Schrijf een programma dat alle getallen tussen 0 en 10000 afdrukt die deelbaar zijn door 6 en door 28.
- opdracht 4.9: Programmeer opgave 4.2

## 4.2.2 For loop met een string als <collectie>

- string = collectie van tekens
- "banaan" =  
collectie van "b" "a" "n" "a" "a" "n"
- volgorde is van belang



## 4.2.2 For loop met een string als <collectie>

- opdracht 4.10: Wat is de output van volgend programma?

```
for i in "python":  
    print(i)  
  
for i in "python is tof":  
    print(i, end="")  
print()  
  
print()  
for i in "python":  
    print(i)  
    print(i, end="")
```

## 4.2.3 For loop met een variabele <collectie>

- opdracht 4.11 a: Wat is de output van volgend Python programma?

```
fruit = "banaan"
for letter in fruit:
    print(letter)
print("Klaar")
```

- opdracht 4.11 b: Wat als de inhoud van de variabele gewijzigd wordt?

```
fruit = "banaan"
for letter in fruit:
    print(letter)
    if letter == "n":
        fruit = "mango"
print("Klaar")
```



## 4.2.4 For loop met een handmatige <collectie>

- opdracht 4.12: Wat is de output van volgend programma?

```
for x in (10, 100, 1000, 10000):  
    print(x)
```

```
for x in (2, " appels", " en ", 1, " peer ") :  
    print(x, end="")
```



## 4.3 While loop

- = iteratiestructuur met test vooraan
- syntax: 

```
while <boolean expressie>:  
    <acties>
```
- lees als: zolang de <boolean expressie> True is, worden de <acties> uitgevoerd



## 4.3.1 While loop: een eerste voorbeeld

- opdracht 4.13 a: Wat is de output van  
volgend programma?


```
a = 2
while a < 5:
    print(a)
    a = a + 1
```

- opdracht 4.13 b: Wat is de output van volgend  
programma?

```
b = 4
while b < 8:
    print(b)
    b = b + 1
```

- indentatie → zeer belangrijk

## 4.3.2 While loop: een tweede voorbeeld

- opdracht 4.14: maak opdracht 4.2 maar mbv een while loop 
- Merk op: variabelen initialiseren



### 4.3.3 While loop onder controle van de gebruiker

- opdracht 4.15: Schrijf een programma waarbij de gebruiker achtereenvolgens getallen ingeeft totdat hij een getal ingeeft dat deelbaar is door 5. Als een getal ingegeven is, wordt dit getal alleen geprint indien het deelbaar is door 2. Dan print je “Getal x is deelbaar door 2”. Na afloop print het programma “Einde”
- Merk op: op voorhand is niet altijd geweten hoe vaak de loop doorlopen zal worden



## 4.3.4 Eindeloze while loop

- opdracht 4.16: Volgend programma print alle getallen ( $\geq 1$ ) waarvoor het kwadraat kleiner is dan 10: Wat is er fout?

```
c = 1
while c * c < 10:
    print(c)
```



- Merk op: oneindige loop → wijzig altijd <boolean conditie> in de while loop

wijziging wordt meestal als laatste statement van de while loop gedaan

## 4.3 While loop

- opdracht 4.17: Schrijf een programma dat een getal inleest en dan het getal en het kwadraat van dat getal afdruckt tot de gebruiker als getal 0 ingeeft.
- opdracht 4.18: Maak een programma dat de gebruiker vraagt een getal in te geven tussen 0 en 10. Indien de gebruiker een ongeldig getal ingeeft, herhaal je de vraag.



## 4.4 Geneste loops

- opdracht 4.19: We schrijven een programma om volgende vermenigvuldigingstabel op te stellen.

1	2	3	4	5	6	7	8	9	10
2	4	6	8	10	12	14	16	18	20
...	...	...	...	...					
5	10	15	20	25	30	35	40	45	50

## 4.4 Geneste loops

- opdracht 4.20: Wat is de output van onderstaand programma:

```
for i in range(5, 2, -1):  
    for j in range(2, 6, 2):  
        print(i, " ", j)
```
- opdracht 4.21: Schrijf een programma om volgende output te bekomen:

```
a  
a a  
a a a  
a a a a  
a a a a a
```