Lesbrief 3 Bibliotheken, vergelijking en keuzes

Bibliotheken

Soms wil je bepaalde berekeningen doen of bepaalde functies gebruiken die heel veel tijd zouden kosten om zelf allemaal te moeten schrijven. Python heeft hier een handigheid voor. Namelijk libraries! “Library” is een Engels voor bibliotheek en je kan het in principe ook zien als bibliotheek. Het is een bibliotheek van functies die iemand anders voor je heeft geschreven om het leven allemaal wat makkelijker te maken. Door deze libraries in te laden (ook wel importeren genoemd) kan je dus gebruik maken van functies die buiten de normale functies van python vallen (normale functies zoals de `input()` functie of `print()` functie).

Toen je net begon met python heb je een tijd met `turtle` gewerkt toch? Nou dit was dus ook een library! Vandaar dat je elke keer boven je programma `import turtle` moest zetten. Hierdoor had je toegang had tot bijvoorbeeld de `turtle.forward()`, `turtle.right()` en `turtle.goto()` functies. Naast de importbibliotheek `turtle`, heeft Python ook meerdere bibliotheken waar je gebruik van zou kunnen maken. Je gaat nu kennis mee gaat maken met de bibliotheek `math`. Deze bibliotheek bevat net als turtle allemaal functies maar dan met betrekking tot wiskundige berekeningen die je zou willen uitvoeren. Hieronder een voorbeeld:

‘math’ komt van het Engelse woord ‘mathematics’. Dit betekent wiskunde.

```
import math

print (math.floor(4.5)) # rond een getal af naar beneden
print (math.ceil(4.5)) # rond een getal af naar boven
```

‘ceil’ komt van het Engelse woord ‘ceiling’. Dit betekent plafond en daarom rond je naar boven af.

‘floor’ betekent vloer en je rond dan ook af naar beneden

Probeer maar eens uit: Voorbeelden uit de math library

Je kunt de uitkomst van een functie ook in een variabele plaatsen:

```
import math

x = math.floor(4.5)
print (x)
```

Met deze twee functies kan je al interessante dingen doen! Doe de volgende paar opdrachten maar eens:

Opgave 1
Stel je bent met een groepje van x mensen op Amsterdam centraal en je moet met de taxi naar het Leidseplein, maar per taxi mogen er maar 4 mee. Schrijf een programma die de input x in neemt en berekent hoeveel taxi's je minimaal nodig hebt om alle mensen mee te kunnen nemen!

Bedenk welke functie het beste zou passen, de `floor()` of de `ceil()` functie?

Verder hebben we nog de `sqrt()` functie. Deze functie geeft de wortel terug van x. Dus `math.sqrt(9)` zal dan 3 terug geven want 3 keer 3 is 9. Of de `pow()` functie die de macht van een getal terug geeft (`math.pow(3, 2)` zal dan 3*3=9 terug geven!).

'sqrt' is een afkorting van 'square root' en dat betekent wortel nemen.

'pow' komt van 'power' en betekent 'tot de macht'.

Opgave 2
Schrijf een programma dat een getal vraagt aan de gebruiker, en hier vervolgens de wortel van terug geeft.

Opgave 3
Breid het vorige programma uit door het antwoord naar boven af te ronden! En print hierna ook het type van het antwoord. (Denk aan de vorige lesbrief)

Opgave 4
Maak een programma dat de gebruiker vraagt om een getal. Bereken dan het kwadraat van dat getal en print dat uit. Maak gebruik van de functie `math.pow()`. Dus:

```
>>> Geef een getal:
>>> 12
>>> Het kwadraat van 12 is 144
```

Wat als?!?

Een van de handigste eigenschappen van een computer is dat deze in staat is om logische stappen te maken. Je kan een programma heel makkelijk sturen door een "als / dan" situatie te creëren. Bijvoorbeeld:

Als ik mijn schoen in mijn handen heb.
Trek ik hem aan.
Als ik het koud heb.
Pak ik mijn jas.

Dit soort situaties zijn heel handig, omdat je dan direct invloed hebt op oorzaak / gevolg. Bij de voorgaande voorbeelden was de oorzaak (wat moet er aan de hand zijn) steeds **dik** gedrukt, en het gevolg *cursief* (wat ge je dan doen?). Als de dik gedrukte situatie gebeurt, wordt de cursieve opdracht uitgevoerd. Bij programmeren doen we dit met een `if`-statement. Het if-statement wordt als volgt gebruikt:

```
x = 3
y = 3
if (x == y):
      print ("x en y hebben dezelfde waarde")
```

Als x en y hetzelfde zijn

In het voorgaande voorbeeld zien we dat x de waarde 3 toegewezen krijgt en y ook een 3. Na de vergelijking `if (x == y):` wordt de printopdracht uitgevoerd. Het dubbele is-teken (==) betekent hier 'is hetzelfde als'. Als x en y dus niet hetzelfde waren geweest was de printfunctie na de `if` nooit uitgevoerd.

```
x = 4
y = 3
if (x == y):
      print ("x en y hebben dezelfde waarde")
```

Als x en y anders zijn

Nu zie je dus dat er niks geprint wordt. Dit kan je opvangen dat een `else:` statement te gebruiken. Hierbij zeg je "als x gelijk is aan y, voer de komende opdracht uit. En anders (x is **niet** gelijk aan y), doe het volgende". Dit werkt als volgt:

```
x = 4
y = 3

if (x == y):
      print ("x en y hebben dezelfde waarde")
else:
      print ("x en y zijn anders!")
```

'if' betekent letterlijk 'als'
'else' betekent letterlijk 'anders'

Opvangen met else

Op deze manier kan je dus alle situaties opvangen die je niet in je `if` gedeelte beschreven hebt. Maar soms zijn er meerdere situaties die van belang zijn, en waar op een bepaalde manier mee om gegaan moet worden. Zo kan je zoals bij het eerste voorbeeld met de dik gedrukte en cursieve zinnen voor alle twee de voorwaarde checken.

Nog meer vergelijkingen met if-else

Vaak wil je een opdracht pas uitvoeren als deze voldoet aan een bepaalde voorwaarde. Pas als de voorwaarde klopt, dan moet de opdracht uitgevoerd worden. Dit kun je realiseren door de `if`-opdracht te gebruiken die we net hebben geleerd. Zie eens het volgende voorbeeld:

```
if 3 < 4:
  print ("Hoi")
```

Na de `if` volgt een **vergelijking**. Een vergelijking is altijd *waar* of *onwaar*. Hier zien we na de drie een '<'-teken, wat "kleiner dan" betekent. Als drie kleiner is dan vier, print Hoi. Als de vergelijking waar is, dan worden de opdrachten die bij de `if` horen uitgevoerd. Alle opdrachten die zijn ingesprongen (met een tab!) na de `if` horen bij de `if`. Nog een voorbeeld dat dit illustreert:

3 < 4	klopt want 3 is kleiner dan 4
5.1 < 2	klopt niet want 5.1 is **niet** kleiner dan 2
10 > 2	klopt wel want 10 is groter dan 2
getal < 2	kunnen we alleen weten als we weten wat 'getal' is

Zoals gezegd is 3 < 4 een wiskundige vergelijking waarmee je kijkt of het eerste getal kleiner is dan het tweede getal. De bovenstaande code voert nu alle drie de print-opdrachten uit. Maar wat zou er gebeuren als je i.p.v. `3 < 4` de vergelijking `3 > 4` zou schrijven? De > betekent het tegenovergestelde van <, dus dit is dan "groter dan". Dan zou alleen de laatste print-opdracht uitgevoerd worden!

Kijk zelf maar!

Je kunt de if-opdracht ook uitbreiden met een else. Als de vergelijking na de if onwaar is, dan zal de else-tak uitgevoerd worden. Zie maar eens het volgende programma: if / else

```
if 4 < 3:
  print ("hoi")
else:
  print ("doei")
```

In het bovenstaande voorbeeld wordt alleen de `else`-tak uitgevoerd. Dit, omdat de vergelijking na de `if` *onwaar* is.

Naast de > en < heb je nog meer vergelijkingsoperatoren. Hieronder een tabel van alle vergelijkingsoperatoren en de bijbehorende betekenis:

Vergelijking	Betekenis
x != y	x is niet gelijk aan y
x > y	x is groter dan y
x < y	x is kleiner dan y
x >= y	x is groter dan of gelijk aan y
x <= y	x is kleiner dan of gelijk aan y
x == y	x is gelijk aan y

De bovenstaande uitleg over de `if-else` opdrachten wordt aan de hand van een voorbeeld ook uitgelegd in een video op youtube. Zie:

Filmpje met uitleg over de if-else opdrachten

Opgave 5

Schrijf een programma dat de gebruiker om een getal vraagt. Als het getal groter is dan tien, schrijf dan naar het scherm 'Het getal is groter dan tien'.

Opgave 6

Schrijf een programma dat de gebruiker vraagt hoe oud hij is. Als hij ouder is dan 18, dan zegt het programma: Gefeliciteerd, je bent volwassen!
Het programma wordt **altijd** afgesloten met de melding: “Bedankt voor je deelname!”.

Opgave 7

Schrijf een programma dat de gebruiker vraagt om het huidige jaar en om zijn geboortejaar. Als het geboortejaar na het huidige jaar ligt, schrijf dan naar het scherm 'Je bent nog niet geboren.'. Schrijf anders naar het scherm hoe oud de gebruiker is (aan het eind van het jaar).

Opgave 8

Schrijf een programma waar de gebruiker een temperatuur moet ingeven. Als het onder nul is moet het programma zeggen "De temperatuur is onder nul.". En anders dan moet het "De temperatuur is boven nul." zeggen.

Opgave 9

Met 5-letter LINGO is het de bedoeling dat een gebruiker een woord invoert dat precies 5 karakters bevat. Zo niet, dan moet het programma aangeven dat het ingegeven woord incorrect is. Maak een programma (functie) die de gebruiker om een woord vraagt, vervolgens berekent hoeveel letters erin zitten. Als er meer of minder dan 5 letters inzitten, dan geeft hij een foutmelding. Zo niet, dan zegt het programma: “Goed zo, je hebt een 5 letter woord ingevoerd. (TIP: gebruik de functie `len()`, als je dan bijvoorbeeld `woord_len = len(“broek”)` komt er 5 in `woord_len` te staan)

Uitdaging: zorg dat je programma alleen letters accepteert (en niet bijvoorbeeld ‘+’ of ‘:’.