Lesbrief 4 Booleaanse expressies en meer met if

Booleaanse expressies

Eerder hebben we bij de `if`-expressies vergelijkingen gebruikt. Als ***dit*** waar is, dan doen we ***dat***! Deze vergelijkingen worden ook wel booleaanse expressies genoemd. Een boolaanse expressie is een vergelijking waarvan er alleen gezegd kan worden of deze waar is of niet. Het kan niet **misschien** waar zijn of half waar. Hier krijg je dan de Engelse woorden true (waar), en false (onwaar) van. In Python schrijf je waar als `True` en onwaar als `False`. Bij de if-expressies zou de booleaanse expressie 4>3, een true geven, en dan wordt de code in deze if-expressie uitgevoerd. Bij 4<3 zou er een false gegeven worden, en zou er niks gebeuren, of gaat de computer naar de volgende if-expressie. Kijk de volgende voorbeelden maar door, hier zie je een hele lijst met booleaanse expressies. Kijk maar of je snapt waarom het programma deze antwoorden terug geeft.

[Antwoorden uit een booleaanse vergelijking]

Als je een true terug krijgt bij deze booleaanse expressies, voert het programma alles af wat nog komt binnen de if-expressie, en anders stopt het en gaat deze verder. Later zullen we nog andere gevallen leren waarbij booleaanse expressies van belang zullen zijn.

Boolean is vernoemd naar de Engelse wiskundige George Boole (https://nl.wikipedia.org/wiki/George_Boole).

Opgave 1

Bedenk voordat je de komende booleaanse expressies zelf uit voert, of ze een true of false geven. Voer ze daarna uit.

1. 4 > 5
2. 4 != 4
3. 4 == 7 - 3
4. a = 3

   b = 6

   b - a == a
5. “hallo” == “hallo”

Zoals je hierboven kan zien, kan je ook booleaanse expressies uitvoeren met strings. Het komt erop neer dat alles dat voor de vergelijkings expressie hetzelfde is als erna, dat er dan true uit komt. Dit is handig voor als je input wilt vergelijken en wilt dat de gebruiker dit precies in typt zoals jij wilt!

Opgave 2
Schrijf een programma dat door middel van booleaanse waarden kijkt welke soorten fruit de gebruiker wilt, en vervolgens vraagt hoeveel de gebruiker er van zou willen. Geef twee verschillende soorten fruit als mogelijkheid.

Voorbeeld:

```
>>> Hallo, wilt u appels of peren?
>>> peren
>>> Hoeveel peren wilt u?
>>> 6
>>> Oke! Wij sturen u 6 peren!
```

Het if-statement uitbreiden met elif

Met de `if` opdracht kun je een onderscheid maken tussen **twee** verschillende mogelijkheden: of de if-voorwaarde is waar of hij is onwaar. Soms zijn er meer dan twee mogelijkheden en hebben we meer dan twee takken nodig. Een manier om een dergelijke berekening vorm te geven is een **gekoppelde voorwaarde**:

Als ik mijn schoen in mijn handen heb.
Trek ik hem aan.

of

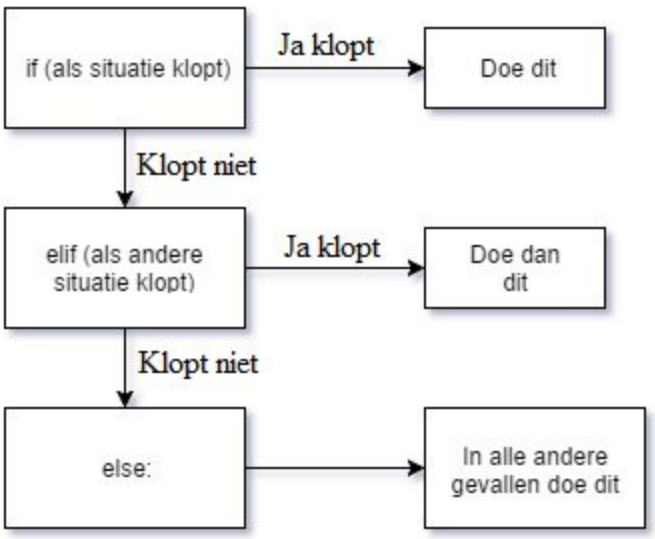
Als ik het koud heb.
Pak ik mijn jas.
En anders
Ga ik op de bank zitten.

Hier is dus een situatie waar je je schoen aantrekt, als je hem in je handen hebt.. Of ‘als je het koud hebt’, ‘dan pak je je jas’. En in alle andere gevallen ga je op de bank zitten. Je kunt dit soort situaties programmeren met een `elif`. We gaan dat uitleggen met een voorbeeld met getallen. Bekijk de volgende code:

```
x = 4
y = 3
z = 4
if (x == y):
      print ("x en y hebben dezelfde waarde")
elif (x == z):
      print ("x en z zijn wel hetzelfde")
else:
      print ("x en y zijn anders en x en z zijn ook anders!")
```

https://trinket.io/python3/04c4cf4ada

Hier zie je dus dat omdat x en y niet hetzelfde waren, ging hij naar de volgende vergelijking kijken (is x hetzelfde als z?). Omdat deze x en z wel gelijk waren werd de inhoud van deze vergelijking (de print functie) uitgevoerd. Hoe dit verloopt is te zien in de volgende figuur:



`elif` is een afkorting van "`else if`". Ook hier wordt precies één tak uitgevoerd. Er is geen limiet op het aantal `elif` instructies. Staat er een `else` opdracht, dan is dat het einde, maar deze *hoeft* er niet te staan.

```
keuze = input("Kies een letter (a,b,c)")
if keuze == "a":
      print("je hebt gekozen voor a")
elif keuze == "b":
      print("je hebt gekozen voor b")
elif keuze == "c":
      print("je hebt gekozen voor c")
else:
     print ("Je hebt voor een andere letter gekozen")
```

Elke voorwaarde wordt in volgorde gecontroleerd. Is de eerste onwaar, dan wordt de volgende gecontroleerd, enzovoorts. Is er één waar dan wordt de bijbehorende tak uitgevoerd en de instructie eindigt. Zelfs als er meer dan één voorwaarde waar zou zijn. Alleen de tak bij de eerste keer waar wordt uitgevoerd.

Dit alles wordt verder ook nog uitgelegd in het volgende filmpje: Filmpje: If elif else (en dit voorbeeld is gebruikt: Trinket: if elif else).

Opgave 3
Schrijf een programma waarin de gebruiker wordt gevraagd om een getal, en vergelijk die vervolgens met een ander getal die je er zelf in hebt gezet. Is het getal hetzelfde, vertel dit dan aan de gebruiker, en zeg anders dat het niet hetzelfde is.

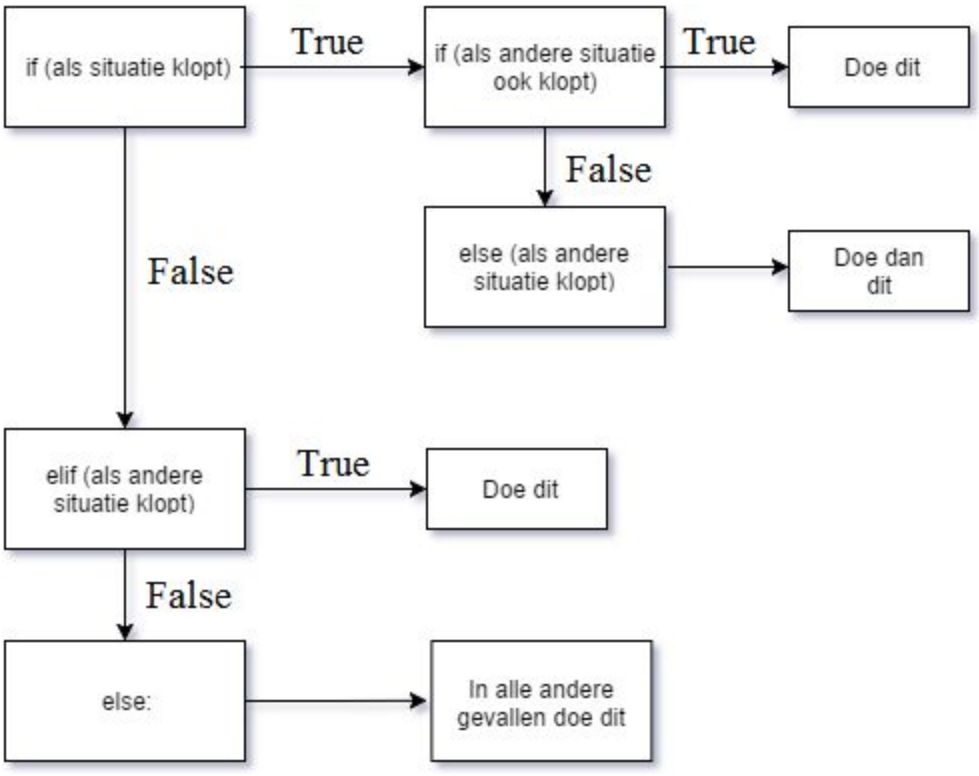
Opgave 4
Schrijf een programma waar de gebruiker wordt gevraagd van welk soort fruit hij/zij houdt. De keuze moet tussen twee soorten fruit zijn, en dit moet een toepasselijke reactie opbrengen. En als de gebruiker iets anders zegt moet dit ook opgevangen worden met een reactie.

Opgave 5
Schrijf een programma die een getal naar beneden afrond als de gebruiker “beneden” in typt, naar boven af rond als er “boven” wordt gegeven, en anders de wortel van het getal neemt.

***** Opgave 6 (ster opdrachten moet je sowieso inleveren!)**
Schrijf een programma dat reageert als een rekenmachine door middel van de `input()` functie. Zorg ervoor dat de eerste input de soort berekening wordt (optellen, aftrekken, delen of vermenigvuldigen) en de tweede en derde input de getallen zijn waarmee de berekening uitgevoerd wordt. Denk eraan hoe je bij bepaalde berekeningen de input en output wilt hebben in verband met informatietypes!

If-opdrachten in if-opdrachten

In de vorige les heb je geleerd dat over if-elif-else expressies, maarrrr wist je ook dat je if-opdrachten *in* if-opdrachten kan plaatsen?! Dit worden *geneste* if-opdrachten genoemd. Zo een geneste if ziet er schematisch uit als volgt:



In code kan dit er bijvoorbeeld uit zien als het volgende (Voorbeeld geneste if):

```
num1 = float(input("Geef een positief getal "))
num2 = float(input("Geef een hoger getal "))

if num1 >= 0:
    if num2 > num1:
        print("Je eerste getal is " + str(num2 - num1) + " hoger!")
    else:
        print("Je hebt een lager tweede getal ingevoerd!")
else:
    print("Je hebt geen positieve getallen ingevoerd!")
```

Als num groter dan of gelijk is aan 0, dan komt het programma in een tweede if. Deze tweede if wordt alleen uitgevoerd als de vergelijking `num >= 0` waar is (als deze booleaanse vergelijking dus true terug geeft!). De eerste else die er staat hoort bij de tweede if. Dit kun je zien aan de hand van de tabjes die zijn gebruikt. Het goed gebruiken van de tabs is hier dus ontzettend belangrijk! Mede omdat het een vereiste is van python zelf, maar ook omdat het je code een stuk leesbaarder maakt. Dit is altijd heel erg belangrijk!

Opdracht 7
Schrijf een programma die de gebruiker vraagt of hij een ‘lahmacun’ wil of een ‘borek’. Als de gebruiker kiest voor ‘lahmacun’, dan moet hij een tweede vraag krijgen: Wil hij er één *met* sambal of zonder sambal? Als de gebruiker er één met sambal wil dan zegt het programma "Alsjeblieft! Een lahmacun met sambal". Zo niet, dan zegt hij "Alsjeblieft! Een lahmacun zonder sambal". Zo ziet de werking van het programma eruit:

```
>>> Welkom bij de digitale kantine :)
>>> Maak een keuze (lahmacun of borek) lahmacun
>>> Wil je er ook sambal op? ja
>>> Alsjeblieft! Een lahmacun met sambal
```

***** Opdracht 8**
Breid nu het bovenstaande programma uit. Als de gebruiker borek wil, dan krijgt hij een tweede vraag: Met gehakt of kaas? Afhankelijk van zijn keuze krijgt hij een bericht te zien. Zo kan het eruit zien:

```
>>> Welkom bij de digitale kantine :)
>>> Maak een keuze (lahmacun of borek) borek
>>> Met gehakt of kaas gehakt
>>> Alsjeblieft! Een borek met gehakt!
```

Opdracht 9

Schrijf een programma waarbij je de gebruiker een vraag stelt waar alleen ja of nee op geantwoord mag worden (bij wat anders moet dit opgevangen worden!) en als er ja geantwoord wordt dan moet er een nieuwe vraag gesteld worden. Deze vragen mag je zelf bedenken, wees creatief!

Je kunt ook if-opdrachten in if-opdrachten in if-opdrachten plaatsen. In principe kun je if-opdrachten oneindig gaan nesten. Maar in de praktijk hangt het af van hoe slim je het programmeert en hoe moeilijk de keuzestructuur is.