1 Voorwaardelijke uitvoering: `if ... elif ... else`

1.1 `if`-statement

In de uitvoering van een eenvoudig python programma, wordt iedere regel code gewoon uitgevoerd in de volgorde dat ze in het programma staan. Maar soms komt het voor dat je bepaalde code enkel wil uitvoeren als aan een bepaalde voorwaarde is voldaan. Bijvoorbeeld, je wil op het einde van een spel enkel de boodschap 'Je bent gewonnen!' tonen als de speler ook echt gewonnen heeft. In het andere geval wil je waarschijnlijk juist de boodschap 'Je bent verloren ...' tonen. Zo een voorwaarde koppelen aan de uitvoering van een stukje van de code kan je doen met een `if voorwaarde: ...`-statement. Het statement is opgebouwd als volgt:

```
# deze code wordt sowieso uitgevoerd
if voorwaarde:
    # code die enkel wordt uitgevoerd
    # als voorwaarde resulteert in True
    # anders wordt deze code overgeslagen
# deze code wordt ook terug altijd uitgevoerd
```

Hierin is `voorwaarde` een expressie tot een logische waarde (herinner je, dat is het type `bool`). Aleen en slechts alleen als de waarde van `voorwaarde` `True`is, dan wordt de code in het `if`-blok uitgevoerd. Deze voorwaarde kan van alles zijn:

- een vaste logische waarde:
  vb. `if True: ...` / `if False: ...` waarin het `if`-blok altijd / nooit wordt uitgevoerd.
- een variable zijn die een logische waarde bevat:
  vb. `if spelerIsGewonnen: print("Je bent gewonnen!")`.
- een expressie zijn die een logische waarde oplevert:
  vb. `if aantalPogingen < 3: print("Je bent een genie!")`.

Een `if`-blok begint zodra je een `if`-statement gebruikt hebt. Iedere regel code waarvan je wil dat het tot dit blok behoort, moet je inspringen. Een regel code is ingesprongen als er minstens één spatie of tab voor staat. Het blok loopt door tot aan de eerstvolgende regel code die niet meer ingesprongen is. Let op, al de regels code die tot hetzelfde blok behoren, moeten allen gelijk ingesprongen zijn. Je mag hiervoor geen spaties en tabs door elkaar gebruiken!

Dat is een hele boterham, he? Kan jij dat allemaal nog volgen? Zullen we eens kijken naar een voorbeeldje? Probeer je eens om te voorspellen wat de output gaat zijn van het volgende programma? Nadien kan je het programma kopiëren en uitvoeren in de interactive shell om te zien of je voorspelling juist was.

```
spelerIsGewonnen = True
aantalPogingen = 2

if spelerIsGewonnen:
    print("Je ben gewonnen!")

    if aantalPogingen < 3:
        # je kan een if-statement ook in
        # een ander if-statement plaatsen
        print("Je bent een genie!")

if not spelerIsGewonnen:
    print("Je bent verloren ...")

print("Einde van het spel")
```

EHBF - Eerste Hulp Bij Foutmeldingen

`IndentationError`

Je ziet deze error, wat nu?

Als je een `IndentationError` *krijgt, dan is er iets mis met de inspringing van je code. Je kan het volgende controleren:*

- *Heb je een* `if`*-statement gebruikt zonder bijhorend* `if`*-blok? Een* `if`*-statement moet minstens één regel code bevatten, anders krijg je deze error.*
- *Zijn alle regels code die tot hetzelfde blok behoren, allen gelijk ingesprongen? Je mag hiervoor geen spaties en tabs door elkaar gebruiken!*

1.2 `else`-statement

Net zoals dat je een blok code enkel kan laten uitvoeren als een bepaalde voorwaarde waar is, kan je een ander blok code enkel laten uitvoeren als diezelfde voorwaarde niet waar is. Dus je kan met eenzelfde voorwaarde er voor zorgen dat oftewel het ene blok code oftewel het andere blok code wordt uitgevoerd. Dit doen we met met het `else: ...`-statement. Het `else`-statement komt altijd na een `if`-statement en bevat een blok code dat enkel wordt uitgevoerd als de voorwaarde van het `if`-statement niet waar is. Bijvoorbeeld, nu kunnen we opnieuw op basis van de waarde van `spelerIsGewonnen` kiezen of we de boodschap 'Je bent gewonnen!' of 'Je bent verloren ...' tonen. Maar deze keer hebben we daar niet meer twee afzonderlijke `if`-statements voor nodig. We breiden het eerste `if`-statement uit met een `else`-statement op deze manier:

```
# deze code wordt sowieso uitgevoerd
if voorwaarde:
    # code die enkel wordt uitgevoerd
    # als voorwaarde resulteert in True
    # anders wordt deze code overgeslagen
else:
    # code die enkel wordt uitgevoerd
    # als voorwaarde resulteert in False
# deze code wordt ook terug altijd uitgevoerd
```

Laten we nog eens het voorbeeld van het spel bekijken. Maar deze keer gaan we maar één `if`*-statement gebruiken.*
Geeft de code de output die jij verwacht had?

```
spelerIsGewonnen = False
aantalPogingen = 7

if spelerIsGewonnen:
    print("Je ben gewonnen!")
else:
    print("Je bent verloren ...")
    if aantalPogingen < 10:
        print(
            "Maar je hebt het goed geprobeerd!"
        )

print("Einde van het spel")
```

1.3 `elif`-statement

Nu kunnen we al kiezen of we één blok code wel of niet willen uitvoeren door het `if`-statement te gebruiken. We kunnen ook al kiezen om eventueel een tweede blok code uit te voeren als het eerste blok code niet wordt uitgevoerd. Hiervoor hebben we het `else`-statement geïntroduceerd. Maar wat als we nu tussen meer dan twee blokken code moeten kunnen kiezen?

Bijvoorbeeld, we willen nog altijd de boodschap 'Je bent een genie!' tonen als de speler gewonnen heeft met minder dan 3 pogingen. En deze keer willen de boodschap 'Je hebt het goed gedaan!' tonen als de speler gewonnen heeft met minder dan 5 pogingen. En anders willen we de boodschap 'Je bent het goed geprobeerd!' tonen. Dit kunnen we doen met het `elif`-statement. Dit doen we als volgt:

```
# deze code wordt sowieso uitgevoerd
if voorwaarde:
    # code die enkel wordt uitgevoerd
    # als voorwaarde resulteert in True
    # anders wordt deze code overgeslagen
elif andereVoorwaarde:
    # code die enkel wordt uitgevoerd
    # als voorwaarde resulteert in False
    # en andereVoorwaarde resulteert in True
    # anders wordt deze code overgeslagen
else:
    # code die enkel wordt uitgevoerd
    # als beide, voorwaarde en andereVoorwaarde,
    # resulteren in False
# deze code wordt ook terug altijd uitgevoerd
```

Wist je dat je zoveel `elif`-statements kan toevoegen als je wil? Voor iedere bijkomend `elif`-statement geldt dat het bijhorend code blok enkel kan worden uitgevoerd als de voorwaarde van het `if`-statement en alle voorgaande `elif`-statements niet waar zijn.
Wist je ook dat je geen `else`-statement moet toevoegen als je dat niet wil?

Laten we voor een laatste keer nog eens het voorbeeld van het spel bekijken. Maar deze keer mag jij de code schrijven. Lukt het jou om het `if ... elif ... else`-statement te gebuiken om de juiste boodschap te tonen?

- *Als de speler gewonnen heeft met minder dan 3 pogingen, toon je 'Je bent een genie!'.*
- *Als de speler gewonnen heeft met minder dan 5 pogingen, toon je 'Je hebt het goed gedaan!'.*
- *Anders toon je 'Je hebt het goed geprobeerd!'.*

1.4 `if ... elif ... else`: samengevat

- `if voorwaarde:`
  Een `if`-statement laat je toe om een blok code enkel uit te voeren als een bepaalde voorwaarde resulteert tot `True`. Het `if`-statement is altijd het begin van een `if ... elif ... else`-statement.
- `elif andereVoorwaarde:`
  Na een `if`-statement kan je meerdere `elif`-statements toevoegen als je wil. Een `elif`-statement laat je toe om een blok code enkel uit te voeren als de bijhorende voorwaarde `True`is en alle voorgaande voorwaarden `False`zijn. Een `if`-statement hoeft niet altijd gevolgd te worden door een `elif`-statement als je dat niet nodig hebt.
- `else:`
  Tot slot laat het `else`-statement je toe om een blok code enkel uit te voeren als al de voorgaande voorwaardes allemaal `False`zijn. Dit zijn de voorwaardes van het `if`-statement

en alle eventuele `elif`-statements. Net zoals een `elif`-statement, moet je het `else`-statement ook niet altijd gebruiken. Maar als je het `else`-statement gebruikt hebt, dan moet het wel altijd het laatste statement zijn in een `if ... elif ... else`-statement.