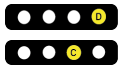
Week 2

Doelstellingen

Je bent in staat om te werken met controlestructuren.
Je bent in staat om eenvoudige functies in Python te schrijven.
Je bent in staat om functies met parameters uit te rusten en vervolgens deze functies op een correcte manier aan te roepen.
Je bent in staat om de scope van variabelen correct in te schatten.

Afspraken

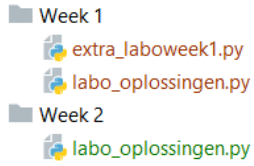
Eindniveau - oefeningen



Ben je een student MCT, dan beheers je de oefeningen tot moeilijkheidsgraad “D”
Ben je een student MIT, dan beheers je de oefeningen tot moeilijkheidsgraad “C”

GitHub

Alle oplossingen van week 2 dienen op Github geplaatst te worden.
Zie week 1 voor procedure.



Open hiervoor het project dat je vorige week op GitHub geplaatst hebt. Maak een submap voor week 2. Plaats hierin je oplossingen van deze week. Na elke oefening kan je een 'commit & push' doen zodat jouw versie op GitHub steeds aangepast wordt. Geef telkens een gepaste message mee.

Niet afgewerkte oefeningen werk je thuis verder af: voer regelmatig een 'push & commit'-opdracht uit zodat alle oplossingen op je github-repository beschikbaar zijn.

Bij een programmeertaal zoals Python onder de knie krijgen is veelvuldig oefenen essentieel en een noodzakelijke voorwaarde. Daarom vind je in elk labo-document nog twee extra onderdelen. Deze worden als volgt aangeduid.

Uitbreidingsoefeningen - eigen onderzoek

Dit onderdeel gaat verder dan de geziene leerstof van deze week. Vaak zijn de opdrachten net iets moeilijker dan hetgeen je in het labo deed. Je zal de Python documentation en Google nodig hebben voor dit onderzoek.

We motiveren iedereen om dit (thuis) iedere week voor te bereiden. Je onderzoekt in dit onderdeel een onderwerp die de volgende weken terugkomt in de theorie of het labo.

Oefeningen voor thuis

In dit onderdeel vind je analoge oefeningen zoals je reeds in het labo maakte.
Deze oefeningen hebben dezelfde moeilijkheidsgraden zoals in het labo. Het is pas door de oefeningen thuis “alleen” te maken dat je de leerstof zich eigen maakt. Loop je vast bij een oefening? Herbekijk de theorie, kijk of je een analoge oefening terugvindt die je maakte tijdens het labo. Lukt het nog steeds niet? Kom met je voorbereiding naar het monitoriaat!

Oefeningen

Gebruik van controlestructuren

Vermeld in commentaar telkens de opgave!

Oef 01

Vraag twee getallen aan de gebruiker. Controleer of deze gelijk zijn aan elkaar of verschillend. Print een gepaste boodschap af.

Oef 02

Vraag een niet-decimaal getal aan de gebruiker. Bepaal of het opgegeven getal even of oneven is. Print een gepaste boodschap af.

Oef 03

Vraag aan de gebruiker wat zijn geboortejaar is. Indien hij nog geen 18 is, print dan ook een gepaste melding af.

```
Wat is uw geboortejaar? 1977
Ok, u mag alcohol drinken.
```

```
Wat is uw geboortejaar? 2008
U bent nog geen 18!
Kom volgend jaar terug...
```

(Tip: via module datetime kan je now-functionaliteit gebruiken. Gebruik hiervan het jaartal)
(Extra: hou ook rekening met de geboortemaand en geboortedag om te verifiëren of iemand al 18 is.)

Oef 04

Maak een Python programma dat de leeftijd van een hond vertaalt naar een overeenkomstige leeftijd van een mens. Vraag eerst aan de gebruiker de leeftijd van zijn hond. Nadien print je een correcte boodschap af waarbij:

- Indien getal < 0, geef een foutmelding terug.
- Indien leeftijd = 1 → 14 mensenjaren
- Indien leeftijd = 2 → 22 mensenjaren
- Indien meer dan 2: mensenleeftijd = 22 + (jaren – 2) * 5

Voorbeeld:

```
Geef de leeftijd op van uw hond: 5
Deze leeftijd komt overeen met 37 mensenjaren
```

Oef 05

Vraag de decimale score (op 20) van een module aan de gebruiker. Print nadien af of hij/zij geslaagd is. Zorg ervoor dat de score correct afgerond wordt: indien het decimale gedeelte kleiner is dan 0,5 wordt er naar beneden afgerond. In het andere geval wordt er naar boven afgerond. Print ook de afgeronde score af.

(Tip: maak gebruik van de ceil/floor-functionaliteit uit de module math)

Bijvoorbeeld:

```
Geef uw score op:  9.2
Helaas, volgende keer beter!
```

```
Geef uw score op:  9.8
U bent geslaagd!
```

Oef 06

Controleer of Python bij het vergelijken van 2 strings al dan niet hoofdlettergevoelig is.

```
Geef een eerst woord: Howest
Geef een tweede woord: howest
De woorden Howest en howest zijn niet gelijk
```

Oef 07

Vraag aan de gebruiker twee woorden op. Ga na of deze aan elkaar gelijk zijn (zonder rekening te houden met kleine letters of hoofdletters)

```
Geef een eerst woord: Howest
Geef een tweede woord: howest
De woorden Howest en howest zijn gelijk
```

Werken met functies

Oef 08

Schrijf een functie ***print_welkom*** die een string als parameter heeft. Deze string stelt de naam voor. Print **in** de functie een welkomsbericht af waarin de naam gebruikt wordt. Deze functie zal dus **geen returnwaarde** hebben.

Test de methode met verschillende namen.


```
Wat is je voornaam? Sarah
Welkom Sarah
```

Oef 09

Schrijf een functie **maak_verwelkoming_klas** die twee strings als parameters heeft: naam en klasgroep. Zorg dat de parameter groep een defaultwaarde '1MIT1' krijgt.

Print het welkomstbericht, waarin naam & klasgroep vermeld staan, **buiten** de functie. Deze functie heeft met andere woorden **wel** een **returnwaarde**.

Het voordeel van deze werkwijze is dat je later kan beslissen wat je met de uitkomst van de functie doet. De functie werkt volledig autonoom van de in- of output. Deze werkwijze heeft de voorkeur.

Test de functie voldoende (zowel met 1 als 2 argumenten)

Oef 10

Schrijf een functie ***vergelijking*** met 4 parameters (a,b,c,d) die getallen voorstellen. De laatste 2 parameters hebben 0 als default-waarde. De functie geeft het resultaat van volgende berekening terug:
a – b + c – d.

- Roep de functie aan door 4 getallen door te geven.
- Roep de functie aan met dezelfde 4 getallen maar in andere volgorde (gebruik de parameternamen)
- Roep de functie aan met twee getallen.

Oef 11

Schrijf een functie ***toon_max*** die 3 getallen binnenkrijgt. De functie geeft het maximum terug.

Oef 12

Schrijf een functie ***vertaal_maandnummer_naar_str*** die een maandnummer binnenkrijgt als parameter. Controleer in de functie of het getal tussen 1 en 12 ligt. Geef nadien de corresponderende maand terug. Indien buiten het interval, geef je een foutboodschap terug. Test de functie met meerdere maandnummers.

Nadien test je je functie door volgende input aan de gebruiker te vragen.

```
Geef een maandnummer: 9
De overeenkomstige maand is -> september

Geef een maandnummer: 13
De overeenkomstige maand is -> onbekende maand
```

Oef 13

Schrijf volgende twee functies:

- Eerste functie: ***geef_celsius*** krijgt een temperatuur in Fahrenheit binnen.
  De overeenkomstige temperatuur in Celsius wordt berekend en teruggegeven.
  De formule is: **temp = (t_in_fahrenheit - 32) * 5 / 9**

- Tweede functie ***geef_fahrenheit*** krijgt een temperatuur in Celsius binnen.
  De overeenkomstige temperatuur in Fahrenheit wordt berekend en teruggegeven.
  De formule is: **temp = (temp_in_celsius * 9 / 5) + 32**

Vraag aan de gebruiker welke temperatuureenheid hij gebruikt. Vraag vervolgens de temperatuur. Bereken de overeenkomstige temperatuur en print deze af.

```
Welke eenheid gebruikt u? [C: Celsius, F: Fahrenheit] C
Geef een temperatuur in celsius op: 123.45
De overeenkomstige temperatuur in fahrenheit bedraagt 254.21
```

Uitbreiding – Eigen onderzoek

Zoek een manier om, in Python, te controleren of de gebruiker wel degelijk een getal opgeeft wanneer dat gevraagd wordt. Het input-commando geeft immers standaard een string terug. Wanneer de ingetikte waarde geen getal is, toon dan een foutmelding aan de gebruiker. In het andere geval wordt het dubbel van het ingevoerd getal afgeprint.

Zoek even uit wat men met een ‘recursieve’ functie bedoelt. Pas deze techniek toe om de faculteit van een getal te berekenen.

Oefeningen voor thuis

Thuis 1

Gebruik jouw kassa-oplossing van vorige week.

```
*** welkom bij het kassa systeem ***
Hoeveel broeken werden er gekocht? 2
Hoeveel T-shirts werden er gekocht? 3
Hoeveel vesten werden er gekocht? 4
Totaal te betalen:
604.87
```

Verplaatst nu de berekening in een **functie *Kassa*** dat het kassasysteem nabootst. De functie geeft terug wat het totaal te betalen bedrag is.

- Wat zijn de parameters van deze functie? Maw welke waarden veranderen binnen de berekening?
- Wat geeft deze functie terug? Datatype?
- Zorg dat je binnen de functie geef input of print gebruikt!

Test uit!

Thuis 2

Maak een functie ***exclusief_naar_inclusief***. Deze ontvangt twee parameters *bedrag* en *btw*. Het resultaat is het bedrag inclusief btw.

Je zal de functie als volgt oproepen. Vervolledig de code door zelf de functie ***exclusief_naar_inclusief*** te schrijven.

```
excl_btw = float(input("Hoeveel bedraagt het bedrag exclusief btw? "))
btw_percentage = float(input("Wat is het btw percentage? "))
incl_btw = exclusief_naar_inclusief(excl_btw, btw_percentage)
print(f"Het inclusief bedrag dat je moet betalen is: {incl_btw}")
```

Het output zal het volgende zijn.

```
Hoeveel bedraagt het bedrag exclusief btw? 200
Wat is het btw percentage? 1.06
Het inclusief bedrag dat je moet betalen is: 212
```

Thuis 3

Schrijf een functie **toon_boodschap** die - afhankelijk van de het uur - volgende boodschap teruggeeft.

Voorwaarde	Resultaat
Tussen 7.00 u en 11.59 u , dus uur is >=7 en <12	"Goede morgen, Karel"
Tussen 12.00 u en 12.59 u , dus uur is ….	"Joepie het is middag, Karel"
Tussen 13.00 u en 16.59 u	"Karel, goede namiddag"
Tussen 17.00 u en 20.59 u	"Karel, goede avond"
Tussen 21.00 u en 6.59 u	"Slaapwel Karel"

Je zal de functie als volgt oproepen. Vervolledig de code door zelf de functie ***toon_boodschap*** te schrijven.

```
uur = int(input("Hoe laat is?"))
voornaam = input("Wat is je voornaam?")
verwelkoming = toon_boodschap(uur, voornaam)
print(verwelkoming)
```

Het resultaat zal het volgende zijn.

```
Hoe laat is? 15
Wat is je voornaam? Sarah
Sarah, goede namiddag
```

Thuis 4

Uitbreiding: kan je de vorige thuis oefening 3 combineren met hetgeen je aanleerde in de labo oefening 3? Op deze manier moet je het uur niet aan de gebruiker vragen, maar gebruikt je de huidige tijd van de computerklok. Pas de eerste lijn aan.
Je zal dus **niets** **binnen** de functie aanpassen! Maar enkel bij de oproep.

```
# uur = ## Zorg dat je hier het tijdsstip (uur) van de computerklok 100% Zo 11:05 inleest.
voornaam = input("Wat is je voornaam?")
verwelkoming = toon_boodschap(uur, naam)
print(verwelkoming)
```

Het resultaat zal het volgende zijn.

```
Wat is je voornaam? Louis
Louis, goede morgen
```

Thuis 5

Schrijf een functie ***stopafstand*** die de remafstand van een voertuig berekent. De formule hiervoor vind je hieronder. (Bepaal hieruit zelf welke parameters deze functie moet hebben).
Test voldoende uit.

stopafstand = reactieweg + remweg

= snelheid * reactietijd + (snelheid)² / (2 * remvertraging)

Hou rekening met:

- De snelheid dient omgezet te worden in m/s.
  De omzetting van km/u naar m/s gebeurt door een deling van 3,6
  Bv: 50 km/u → 50/3,6 →13,89 m/s

- De reactieweg is de afstand afgelegd tussen het moment van waarneming en het moment van de reactie (het remmen).

- De remweg is de afstand om bij een bepaalde snelheid volledig tot stilstand te komen.

De snelheid van het voertuig die tijdens het remmen per seconde afneemt, noemt men de remvertraging. Dit wordt aangeduid met m/sec². Voor een personenauto bedraagt dit bij een droog wegdek 8 m/sec², bij een natwegdek 5 m/sec².