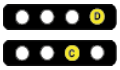
Week 3

Doelstellingen

Je bent in staat om te werken met lusstructuren.
Je bent in staat om te werken met de range-functie om een lijst van getallen te genereren.
Je bent in staat om verschillende string-functies te combineren.
Je kan werken met de module random.
Je bent in staat om eenvoudige functies met parameters in Python te schrijven.
Je bent in staat om de beschikbare debug-tools in je editor te gebruiken.

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 3 dienen op Github geplaatst te worden. Volg hiervoor de procedure uitgelegd op Leho.

Om je repository in onze Github-classroom aan te maken, klik je op volgende link:
https://classroom.github.com/a/7CHIhj3f

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

Naming conventions in Python

Zoals in theorieles vermeld, levert Python een aantal richtlijnen (afspraken) rond de te hanteren programmeerstijl. Deze kan je hier terugvinden:

https://www.python.org/dev/peps/pep-0008/

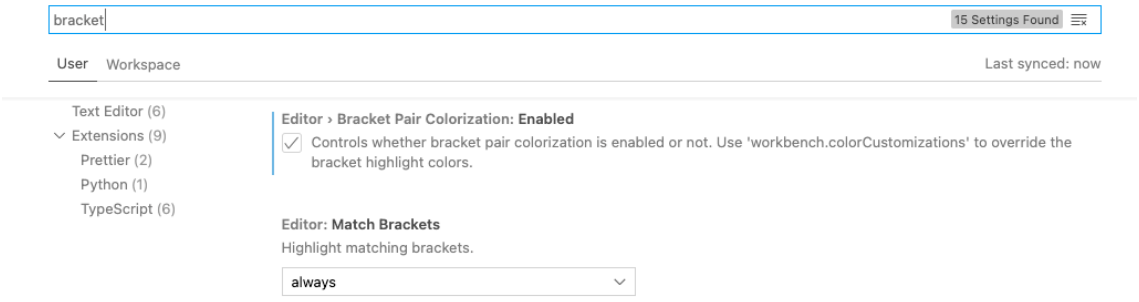
Zoek even op wat de afspraken zijn voor variabelenamen, functienamen,… De linter 'pylint' – die we in week 01 installeerde - kan jouw code hierop controleren. Probeer even op jouw oplossing uit.

Wist je dat de sneltoets ' **Shift+Alt+F** ' jouw code onmiddellijk formateert volgens de PEP8-stijl?

(Sneltoetsen kunnen in VS code steeds gewijzigd worden via het menu-item 'File' -> 'Preferences' -> 'Keyboard shortcuts')

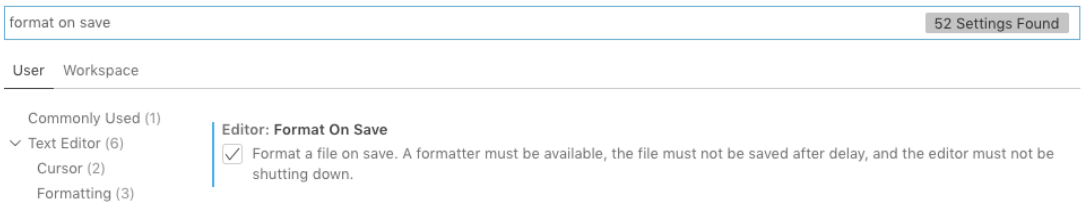
Tip van de docenten

Je kan de haakjes (die bij elkaar horen) binnen jouw code een kleurtje geven.



Je kan je code ook automatisch "formateren" als je een bestand bewaart.

Via 'Settings' -> 'Editor' -> 'Format on Save' stel je in dat je de code onmiddellijk correct opmaakt.



Vanaf nu moet je enkel **ctr+s** uitvoeren, en je code wordt onmiddellijk correct opgemaakt. Hierdoor maak je minder snel fouten en zal je code overzichtelijk blijven.

Gebruik van lusstructuren

Vermeld in commentaar telkens de opgave!

Oef 01

Print de som van de eerste 100 getallen. Gebruik een for-lus.

```
Sum of 1 until 100: 5050
```

Oef 02

Print de lijst van getallen af tussen 20 tem 50. Gebruik een for-lus.

```
20
21
22
23
24
25
26
27
```

Oef 03

Print alle even getallen tussen 4 en 100 **onder** elkaar af. Gebruik een for-lus.

```
4
6
8
10
12
```

Oef 04

Print alle oneven getallen tussen 10 en 129 **naast** elkaar af. Gebruik een for-lus.

```
11, 13, 15, 17, 19, 21, 23, 25, 27, 29, 31, 33, 35, 37, 39, 41, 43, 45, 4
```

Oef 05

Analoog als vorige oefening, maar nu aftellend van 99 tem 0, met stapgrootte van -3.

```
99
96
93
90
87
84
```

Oef 06

Vraag aan de gebruiker volgende 2 getallen op:

- een startwaarde
- een stopwaarde

Print een lijst van alle getallen tussen de twee opgegeven grenzen (grenzen inclusief) die deelbaar door 7 maar geen veelvoud van 5 zijn.

```
Geef een startwaarde op:> 203
Geef een stopwaarde op:> 508
De gezochte getallen tussen 203 en 508 zijn:
203
217
224
231
238
252
259
```

Oef 07

Vraag aan de gebruiker volgende 3 getallen op:

- een startwaarde
- een positieve stapgrootte
- het gewenste aantal af te printen getallen

Schrijf een functie 'print_lijst_getallen' die deze 3 getallen als parameter binnen krijgt. De functie print een lijst, met het gewenste aantal getallen, af waarbij het eerste getal gelijk is aan de startwaarde en de getallen met de stapgrootte vergroten.De functie heeft **geen** return waarde. Het afprinten gebeurt IN de functie.

```
Geef een startwaarde op: >10
Geef een positieve stapgrootte op: >6
Hoeveel getallen moeten er afgeprint worden? >5
De lijst met getallen is:
10
16
22
28
34
```

Oef 08

Print de som van de eerste 100 getallen. Gebruik hiervoor een while-lus.

```
Opgave 8
Sum of 1 until 100: 5050
```

Oef 09

Laat je applicatie een getal kiezen tussen 1 en 20. Laat vervolgens de gebruiker naar het getal raden. Bij iedere poging krijgt hij feedback: "kleiner" of "groter". De applicatie stopt pas als het getal geraden is. Je toont hoeveel maal de gebruiker gokte.
Tip: maak uit de module 'random' gebruik van de functionaliteit 'random':
https://docs.python.org/3/library/random.html

```
Doe een gokje tussen 1 en 20:> 10
Getal te groot
Doe een gokje tussen 1 en 20:> 5
Getal te klein
Doe een gokje tussen 1 en 20:> 7
Getal te groot
Doe een gokje tussen 1 en 20:> 6
Proficiat. Gevonden!
        U gokte: 4 maal
```

Oef 10

Maak een applicatie dat aan de gebruiker een aantal in te voeren producten vraagt. Vervolgens loop je dat aantal af en vraag je telkens bij elk product op:

- De categorie op: de gebruiker kiest tussen "Groente", "Fruit", "Drank".
- De kostprijs van het product

Print op het einde per categorie de totale kost op, alsook de gemiddelde prijs per categorie af. We gebruiken hiervoor volgende functie:

- Functie 'geef_ticket_per_categorie' met als parameters de categorienaam, totaal en aantal_producten: deze functie geeft een ticket (string) met alle info van de categorie terug. Roep deze functie voor elke categorie aan.

Voorbeeld:

```
Geef het aantal producten op dat u wenst in te voeren:> 5
Wat is de categorie? [G: Groente, F: Fruit, D: Drank]> D
Wat is de kostprijs van het product?> 10
Wat is de categorie? [G: Groente, F: Fruit, D: Drank]> D
Wat is de kostprijs van het product?> 20
Wat is de categorie? [G: Groente, F: Fruit, D: Drank]> G
Wat is de kostprijs van het product?> 12.6
```

```
Wat is de categorie? [G: Groente, F: Fruit, D: Drank]> F
Wat is de kostprijs van het product?> 4
Wat is de categorie? [G: Groente, F: Fruit, D: Drank]> G
Wat is de kostprijs van het product?> 8.7

2 producten zitten in de categorie Groenten
Totale kostprijs: 21.3
Gemiddelde prijs per product: 10.65

1 producten zitten in de categorie Fruit
Totale kostprijs: 4.0
Gemiddelde prijs per product: 4.00

2 producten zitten in de categorie Drank
Totale kostprijs: 30.0
Gemiddelde prijs per product: 15.00
```

Oef 11

Vraag aan de gebruiker twee jaartallen op. Print de schrikkeljaren tussen deze twee jaartallen af.

Opmerking: maak eerst een functie '**is_schrikkeljaar**' waarbij getest wordt of een jaartal al dan niet een schrikkeljaar is. https://www.am-pm.nl/schrikkeljaar/

Roep dan deze functie op voor elk jaartal dat tussen de opgegeven grenzen ligt.

Gebruik van String

Oef 12

Vraag aan de gebruiker een woord, print vervolgens elk karakter onder elkaar af.

```
Geef een woord op?> appel
a
p
p
e
l
```

Oef 13

Schrijf een functie 'swap' die twee strings binnenkrijgt. De functie stelt één nieuwe string op waarbij de twee letters van elk woord worden omgewisseld en beide nieuwe woorden door een spatie gescheiden worden.

Het resultaat van de functie is de nieuwe string.

Voorbeeld: "abc" en "xyz" → "xyc abz"

Oef 14

Vraag aan de gebruiker zijn naam, voornaam en geboortedatum (formaat: dd-mm-yyyy) op. Genereer hiermee een paswoord door:

- de eerste 3 karakters van de ingegeven familienaam (in kleine letters en zonder de eventuele spaties mee te nemen)
- de eerste 2 karakters van de voornaam (in hoofdletters en ook zonder spaties)
- 4 cijfers (mmyy) uit de geboortedatum.

Maak hiervoor een afzonderlijke functie 'genereer_paswoord'. Welke parameters gebruik je, wat zal de return waarde zijn?

Voorbeeld:

```
Geef uw naam op: walcarius
Geef uw voornaam op: stijn
Geef uw geboortedatum op (dd-mm-yyyy): 30-08-2016
walST0816
```

Oef 15

Vraag aan de gebruiker zijn/haar howest-e-mailadres op. Haal hieruit de naam & voornaam en print deze af. Voor de eenvoud gaan we ervan uit dat de voornaam uit één deel bestaat. Zorg ervoor dat de eerste letter van de naam & voornaam met een hoofdletter afgeprint wordt.

```
Geef een correct howest-emailadres op:> stijn.walcarius@howest.be
De familienaam is Walcarius en de voornaam is Stijn.


Geef een correct howest-emailadres op:> charlotte.de.haene@howest.be
De voornaam is Charlotte
De naam is De Haene
```

Oef 16

Maak een functie 'genereer_paswoord_bis' met als parameters minimum_lengte en maximum_lengte. De functie genereert een willekeurig paswoord bestaande uit een combinatie van kleine en hoofdletters, én waarvan de lengte van het genereerde paswoord tussen beide grenzen valt.

Tip: maak gebruik van string.ascii_letters uit de klasse string

Voorbeeld:

```
Geef de minimumlengte van het paswoord op: 5
Geef de maximumlengte van het paswoord op: 10
gekozen lengte: 7
Uw paswoord is: NNZQaEd
```

Uitbreiding – Eigen onderzoek

Een while-lus kan ook (optioneel) uitgerust worden met een else-gedeelte. Zoek even op hoe dit precies werkt. Maak zelf een kleine demo. Wat is hier de betekenis van?

Oefeningen voor thuis

Thuis 1

Vraag aan de gebruiker een woord op. Overloop elk karakter in het woord en tel hierbij het aantal klinkers en medeklinkers. Print nadien beide aantallen af.

```
Geef een woord op? > appel
Aantal klinkers : 2
Aantal medeklinkers : 3
```

Thuis 2

Vraag aan de gebruiker een woord op. Overloop elk karakter in het woord en schrap alle klinkers.

```
Geef een woord op? > appelboom
de string met geschrapte klinkers is *pp*lb**m
```

Thuis 3

Vraag aan de gebruiker twee jaartallen op. Print de schrikkeljaren tussen deze twee jaartallen af. Hergebruik hiervoor defunctie '**is_schrikkeljaar**' waarbij getest wordt of een jaartal al dan niet een schrikkeljaar is.

```
Geef het startjaar op? >1993
Geef het eindjaar op? >2005
1996 is een schrikkeljaar
2000 is een schrikkeljaar
2004 is een schrikkeljaar
```

Thuis 4

Schrijf de functie **controleer_emailadres** die controleert of het gaat om een geldig studenten e-mailadres (voornaam.naam@student.howest.be).

Op wat kan je allemaal controleren?