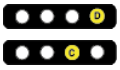
Week 4

Doelstellingen

Je bent in staat om te werken met de datatstructuur list.
Je bent in staat om een list aan functies door te geven of terug te krijgen.
Je plaatst alle oplossingen op Github.

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 4 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/Sx2c6GaC

Open hiervoor het project dat je vorige week op GitHub geplaatst hebt. Maak een submap voor week 3. 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.

Feedback labo week 3

Lees nog even de algemene opmerkingen uit de theorieles na. Je vindt ze op Leho terug.

Reminder uit vorig labo: via de sneltoets ' **Shift+Alt+F** ' kan je jouw code onmiddellijk volgens de PEP8-stijl formatteren.

Voor wie fan is van tips & trics bij het gebruik van Visual Studio Code:
https://code.visualstudio.com/docs/getstarted/tips-and-tricks

Oefeningen

Basisoefeningen lists

Vermeld in commentaar telkens de opgave!

Oef 01

Maak volgende lists aan:

- een list met de zes klasgroepen in MCT
- een list met de twee klasgroepen in MIT

Print beide lists af.

```
MCT heeft volgende klasgroepen in het eerste jaar: ['1MCT1', '1MCT2', '1MCT3', '1MCT4', '1MCT5', '1MCT6']
MIT heeft volgende klasgroepen in het eerste jaar: ['1MIT1', '1MIT2']
```

Kan je twee lists met elkaar optellen? Test uit door de som af te printen.

```
MIT heeft volgende klasgroepen in het eerste jaar: ['1MIT1', '1MIT2']
```

Oef 02

Maak een lege list 'nieuwe_list_getalllen' aan. Vul deze list op met getallen startend vanaf 1, met stapgrootte 13, tem 482. Doe nu het volgende:

- Print de list af.

```
[1, 14, 27, 40, 53, 66, 79, 92, 105, 118, 131, 144, 157, 170, 183, 196, 2
404, 417, 430, 443, 456, 469, 482]
```

- Print de list in omgekeerde volgorde af.

```
[482, 469, 456, 443, 430, 417, 404, 391, 378, 365, 35
05, 92, 79, 66, 53, 40, 27, 14, 1]
```

- Verwijder het eerste element (waarde 482) en print opnieuw de list af.

```
[469, 456, 443, 430, 417, 404, 391, 3
 261, 248, 235, 222, 209, 196, 183, 1
```

- Zoek de werkwijze om een specifiek element uit de list te verwijderen.

Oef 03

Maak een lege list ‘vrienden’ aan.
We laten de applicatie deze list dynamisch uitbreiden. Telkens wordt aan de gebruiker gevraagd om een nieuwe naam op te geven of een lege string. In dat laatste geval stopt de applicatie door de lijst met vrienden af te printen.

Voorbeeld van uitvoering:

```
Geef de naam van een vriend op, of sluit af met een leeg veld: Stijn
Geef de naam van een vriend op, of sluit af met een leeg veld: Bart
Geef de naam van een vriend op, of sluit af met een leeg veld: Kaat
Geef de naam van een vriend op, of sluit af met een leeg veld: Lies
Geef de naam van een vriend op, of sluit af met een leeg veld:
Uw vrienden zijn: ['Stijn', 'Bart', 'Kaat', 'Lies']
```

Oef 04

Test met een eenvoudige voorbeeld wat de vermenigvuldigingsoperator tussen een list en een natuurlijk getal betekent, alsook de += operator tussen twee lists.

Oef 05

Maak één list aan met de dagen van de week. Gebruik het printcommando (in één codelijn!) om volgende output af te printen:

- enkel de werkdagen van de week
- de weekenddagen van de week
- de onpare dagen van de week
- de pare dagen van de week

Tip: welke techniek zagen we vorige week om een deel uit een string op te halen?

```
['maandag', 'dinsdag', 'woensdag', 'donderdag', 'vrijdag']
['zaterdag', 'zondag']
['maandag', 'woensdag', 'vrijdag', 'zondag']
['dinsdag', 'donderdag', 'zaterdag']
```

Oef 06

Vraag aan de gebruiker een woord. Overloop deze string. Bewaar alle klinkers samen in één list, de medeklinkers in een andere list.

Print beide lists af. Hoe zorg je ervoor dat zowel hoofdletters als kleine letters in de list terechtkomen?

```
Geef een woord op? > Howest
De gevonden klinkers zijn : ['o', 'e']
De gevonden medeklinkers zijn: ['h', 'w', 's', 't']
```

Gebruik van list(s) als parameter van een functie

Oef 07

Maak volgende lists aan:

- een list met 4 gehele getallen
- een list met 5 decimale getallen
- een list met 3 strings

Maak nu één functie 'geef_info_list' die een string teruggeeft bestaande uit:

- de naam van de list
- elk element met zijn index

Roep deze functie voor de verschillende lists op.

```
Verzameling gehele getallen:
12 zit op positie 0
45 zit op positie 1
-9 zit op positie 2
-15 zit op positie 3
Verzameling decimale getallen:
12.23 zit op positie 0
45.1 zit op positie 1
9.478 zit op positie 2
15.125 zit op positie 3
-3.14 zit op positie 4
Verzameling strings:
Stijn zit op positie 0
Lies zit op positie 1
Henk zit op positie 2
```

Oef 08

Maak een functie 'kies_element' aan met als parameter een list. De functie kiest een willekeurig element uit de doorgegeven list en **geeft** deze **terug**. Test deze functie met

- een list met strings, nl. de verschillende maanden
- een list met getallen

Tip: Gebruik de documentatie van de module Random en zoek hoe je een willekeurige waarde uit een list kan opvragen. (Onder welke "groep" binnen de Data Types valt een list volgens de theorie les (zie schema))

Print telkens het gekozen element af.

```
De gekozen maand is september
Het gekozen getal is 200
```

Oef 09

Schrijf een functie 'max_en_min_uit_list' met als parameter een list. Deze functie zoekt uit de list het maximum en minimum op en **geeft** deze samen in een string **terug**. Test deze functie uit op een list met getallen en een list met woorden.

```
Uit [11, 52, 125, -89, 1245] is het max: 1245 en min: -89
Uit ['jan', 'feb', 'maa', 'apr', 'mei'] is het max: mei en min: apr
```

Oef 10

Schrijf een functie 'som_in_list' met als parameter een list van getallen die de totale som van de list getallen teruggeeft.

- Pas eerst de techniek van vorige week toe.
- Zoek nadien of er geen ingebouwde functie bestaat om de som te berekenen van de inhoud van een list.

Oef 11

Schrijf een functie ‘gemiddelde _in_list’ met als parameter een list van getallen die het gemiddelde van de getallen teruggeeft.

```
getallen = [12, 45, 465, 78, 1, 23, 89]
lege_lijst_getallen = []
```

```
Gemiddelde:  101.85714285714286
Gemiddelde:  oh dear, oh dear, oh dear, ....
```

Oef 12

Schrijf een functie ‘zijn_totaal_verschillend’ die 2 lists binnenkrijgt.
De functie geeft False terug indien er minimum één gemeenschappelijk element gevonden wordt.
True wordt pas terug gegeven als beide helemaal verschillend zijn.

```
List1: [4, 5, 6, 4]
List2: [89, 78, 4]
Er zit min 1 gemeenschappelijk element in list1 en list2
```

Oef 13

Schrijf een functie ‘tel_elementen_binnen_interval’ met drie parameters: een list, een minimum en een maximum. De functie telt hoeveel elementen uit de list binnen het interval [min, max] vallen en geeft dat aantal terug.

```
List1: [10, 20, 30, 40, 40, 40, 70, 80, 99]
Het aantal elementen tussen 40 en 100 bedraagt: 6
List2: ['a', 'b', 'c', 'd', 'e', 'f']
Het aantal elementen tussen a en e bedraagt: 5
```

Gebruik van lists als return-waarde van een functie

Oef 14

Schrijf een functie ‘geef_gemeenschappelijke_elementen’ die 2 lists binnenkrijgt.
De functie bepaalt de gemeenschappelijke elementen en geeft deze als een nieuwe list terug. Zorg ervoor dat er in de laatste list geen dubbels voorkomen. Zorg ook dat deze gesorteerd is. Print vervolgens alles af.

Test uit op twee lists van getallen en op twee lists van woorden.

```
List1: [78, 4, 5, 6, 4]
List2: [89, 78, 4]
Doorsnede: [4, 78]

List1: ['Tamara', 'Delfien', 'Elke', 'Marijn']
List2: ['Natasja', 'Mieke', 'Tamara', 'Elke', 'Carine']
Doorsnede: ['Elke', 'Tamara']
```

Oef 15

Schrijf een functie ‘verwijder_dubbels’ die een list als parameter heeft. Deze functie geeft een nieuwe list zonder duplicaten terug. Test uit door de beide lists af te printen.

```
['A', 89, 78, 4, 'A', 'test', 4]
Zonder dubbels: ['A', 89, 78, 4, 'test']
```

Oef 16

Schrijf een functie ‘geef_even_posities’ die een list als parameter binnenkrijgt. De functie maakt een nieuwe list aan, bestaande uit de elementen die op de even posities uit de parameter list staan.

```
De elementen uit [0, 10, 20, 30, 40, 50, 60, 70, 80, 90] op de even posities zijn: [0,
20, 40, 60, 80]
```

Oef 17

Schrijf een functie ‘kies_5_getallen’ die twee waarden (min en max) als parameter binnenkrijgt. De functie kiest 5 unieke getallen uit het interval [min,max], voegt deze toe aan een list en geeft uiteindelijk deze list terug.
Opgelet: er mogen geen dubbels in de teruggegeven lijst voorkomen.
Test voldoende uit.

```
Geef het minimum op:> 12
Geef het maximum op:> 86
De vijf geselecteerde getallen hiertussen zijn: [12, 17, 13, 14, 18]
```

Oef 18

Van deze oefening zijn er twee versies. Een niveau C oefening, die verdergaat in een niveau D oefening.

In een bedrijf werken arbeiders, bedienden en kaderpersoneel.

Schrijf een programma dat voor diverse werknemers, op basis van hun functie (arbeider, bediende of kader), het totale brutoloon berekent van de werknemersgroep binnen dezelfde functie.

Maw. per werknemer wordt de functie (A, B of K) en het brutoloon ingevoerd. Hou de ingevoerde gegevens in lists bij.

Let wel:

Als alle werknemers werden ingevoerd moeten volgende zaken worden getoond:

- het aantal arbeiders,
- het aantal bedienden,
- het aantal kaderleden,
- het totaal aantal werknemers,
- het totaal brutoloonbedrag.

```
Geef het aantal werknemers op:> 5
Geef de functie op (A: Arbeider, B: Bediende, K: Kaderpersoneel): > B
Geef het brutowedde op:> 2000
Geef de functie op (A: Arbeider, B: Bediende, K: Kaderpersoneel): > A
Geef het brutowedde op:> 1800
Geef de functie op (A: Arbeider, B: Bediende, K: Kaderpersoneel): > A
Geef het brutowedde op:> 1560
Geef de functie op (A: Arbeider, B: Bediende, K: Kaderpersoneel): > K
Geef het brutowedde op:> 3500
Geef de functie op (A: Arbeider, B: Bediende, K: Kaderpersoneel): > B
Geef het brutowedde op:> 2400

Overzicht:
A        1800€
A        1560€
B        2000€
B        2400€
K        3500€

Aantal werknemers: 5
Aantal arbeiders: 2
Aantal bedienden: 2
Aantal kaderpersoneel: 1
Totaal brutoloon: 11260€
```

Oef 18

De vorige oefening wordt complexer en gaat verder tot een niveau D oefening.

In een bedrijf werken arbeiders, bedienden en kaderpersoneel. Om hun firma van het faillissement te redden leveren arbeiders, bedienden en kaderpersoneel een percentage van hun brutoloon in. Voor arbeiders is dat 5 % van het brutoloon, voor bedienden is dat 8 % en het kaderpersoneel levert 10 % in.

Schrijf een programma dat voor diverse werknemers, op basis van hun functie (arbeider, bediende of kader) en hun brutoloon, berekent hoeveel het in te leveren bedrag bedraagt. Maw. per werknemer wordt de functie (A, B of K) en het brutoloon ingevoerd. Hou de ingevoerde gegevens in lists bij.

Let wel:

Als alle werknemers werden ingevoerd moeten volgende zaken worden getoond:

- het aantal arbeiders,
- het aantal bedienden,
- het aantal kaderleden,
- het totaal aantal werknemers,
- het totaal brutoloonbedrag (voor afhouden van het in te leveren bedrag),
- het totaal brutoloonbedrag (na afhouden van het in te leveren bedrag).
- De totale besparing voor het bedrijf

```
Geef het aantal werknemers op:> 5
Geef de functie op (A: Arbeider, B: Bediende, K: Kaderpersoneel): > B
Geef het brutowedde op:> 2000
Geef de functie op (A: Arbeider, B: Bediende, K: Kaderpersoneel): > A
Geef het brutowedde op:> 1800
Geef de functie op (A: Arbeider, B: Bediende, K: Kaderpersoneel): > A
Geef het brutowedde op:> 1560
Geef de functie op (A: Arbeider, B: Bediende, K: Kaderpersoneel): > K
Geef het brutowedde op:> 3500
Geef de functie op (A: Arbeider, B: Bediende, K: Kaderpersoneel): > B
Geef het brutowedde op:> 2400

Overzicht:

A        1800€ -> 1710.0€
A        1560€ -> 1482.0€
B        2000€ -> 1840.0€
B        2400€ -> 2208.0€
K        3500€ -> 3150.0€

Aantal werknemers: 5
Aantal arbeiders: 2
Aantal bedienden: 2
Aantal kaderpersoneel: 1
Totaal brutoloon: 11260€
Totaal brutoloon na de inlevering: 10390.0€
Totale inlevering: 870.0€
```

Uitbreiding – Eigen onderzoek

Onderzoek de functie ‘shuffle’ uit de module random.

Schrijf een functie ‘print_zinnen’ die 3 lists als paramaters heeft.
De functie genereert alle mogelijke zinnen bestaande uit 3 woorden waarbij elk woord telkens uit een resp. list komt.

Schrijf een functie ‘binary_search’ die een list en een te zoeken waarde als parameters binnenkrijgt. De functie doorzoekt de list op basis van het ‘binary-search’-algoritme.
Zoek even online op hoe dit algoritme precies werkt. Nadien wordt de positie van het element teruggegeven.

Oefeningen voor thuis

Thuis 1

Schrijf een functie ‘zijn gelijk’ die controleert of 2 lists dezelfde elementen bevatten.

```
List A: [10, 14, 2, 3, -10]
List B: [-10, 3, 2, 10, 14]
Gelijk? True
```

Thuis 2

In een autoverhuurbedrijf worden dagelijks verschillende wagens verhuurd. De huurprijs bestaat uit een vast gedeelte aangevuld met een variabel gedeelte (kost per afgelegde km)

Er zijn drie verschillende types wagens die men kan huren:

- Voor wagens van het type A: vast gedeelte bedraagt 25€/dag; variabel bedraagt 0,5€ per km
- Voor wagens van het type B: vast gedeelte bedraagt 35€/dag; variabel bedraagt 0,6€ per km
- Voor wagens van het type C: vast gedeelte bedraagt 45€/dag; variabel bedraagt 0,7€ per km

Jouw programma verzorgt de boekhouding van het autoverhuurbedrijf. Bij het terugbrengen van de auto geef je op:

- Over welk type het gaat
- Het aantal dagen dat de huurder de wagen benut heeft
- Het aantal gereden kilometers

Op het einde van de dag moeten volgende gegevens op het scherm getoond worden:

- het totale bedrag aan inkomsten
- het totale bedrag aan inkomsten van wagens van het type A,
- het totale bedrag aan inkomsten van wagens van het type B,
- het totale bedrag aan inkomsten van wagens van het type C,

- het totaal aantal wagens van het type A die verhuurd werden,
- het totaal aantal wagens van het type B die verhuurd werden,
- het totaal aantal wagens van het type C die verhuurd werden,

- het totaal aantal gereden km van de wagens van type A,
- het totaal aantal gereden km van de wagens van type B,
- het totaal aantal gereden km van de wagens van type C,