Start to Program: Python, Les 7

Datatype: Lijsten – lists

Een handig datatype in programmeren zijn de lijsten. Hierin kan je informatie groeperen. Je gaat hierin andere datatypes, die bij elkaar horen, opslaan.
Denk hierbij aan bookmarks in je browser of liedjes in je bibliotheek. Dit kan gaan van puur andere datatypes tot variabelen en zelfs functies.

De items in een lijst zijn doorgaans evenwaardig. Zoals de namen van de studenten. Niet bepaald extra informatie. Daarvoor gebruik je een ander soort opsomming.

Deze lijst sla je op in een variabele.

```
bloemen = ['roos', 'tulp', 'margriet', 'paasbloem', 'lelie']

print(bloemen)
```

indexering

Net zoals bij strings heeft elk element in de lijst een indexnummer.
LET OP! Deze indexering begint ook bij 0. Dus een lijst met 4 elementen heeft als hoogste indexering...

Een bepaald element op een gekozen indexnummer oproepen doe je zo:

```
print(bloemen[2])
```

Als je een niet bestaand indexnummer aanspreekt zal er een error verschijnen:

```
IndexError: list index out of range
```

Net zoals bij strings kan je ook achteraan beginnen tellen als je negatieve getallen invult.

```
print(bloemen[-2])
```

Opdracht: maak een lijst met de dagen van de week.
Loop door deze lijst en print de dagen van de week 1 voor 1.
Loop ook omgekeerde door de lijst.

Je moet niet steeds heel de lijst gebruiken. Je kan ook een deel van de lijst tussen bepaalde indexeringen opvragen.

```
print(bloemen[1:4]) => tussen bepaalde indexnummers
print(bloemen[:4]) => tot aan een bepaald indexnummer
print(bloemen[1:]) => vanaf een bepaald indexnummer
```

Door een derde waarde toe te voegen kan je de interval van je stappen bepalen.

```
print(bloemen[1:4:2])
```

Elementen aanpassen en verwijderen

Stel dat je een element wil vervangen, omdat er een fout in geslopen is of het een steeds veranderlijke lijst is. Dan kan je in plaats van de variabele met de lijst aan te spreken rechtstreeks het element oproepen en hier een andere waarde aan toe kennen.

Op deze manier kan je ook een lege lijst aanvullen door een loop te gebruiken.

```
bloemen[2] = 'vergeetmenietje'
```

Het element op indexnummer 2 zal nu vervangen zijn door je nieuwe waarde.

Je kan ook een element uit de lijst gaan verwijderen. Hiervoor heb je een specifiek statement nodig.

```
del bloemen[2]
        of
del bloemen[1:3]
```

Zo kan je 1 of meerder elementen tegelijk verwijderen.

Operatoren

Wiskundige operatoren kan je gebruiken om lijsten samen te voegen of om een lijst meerdere malen af te printen.

Opdracht: Maak 2 lijsten. De eerste bestaat uit de weekenddagen. De 2de lijst bevat de dagen van de week.

Print deze als 1 lijst af door ze bij elkaar op te tellen.

Lijst-methodes

Er zijn een aantal methodes die je rechtstreeks kan toepassen op lijsten.

Elementen toevoegen

Lijst elementen toevoegen kan op verschillende manieren gebeuren.

```
lijst_var.append(element) => achteraan toevoegen
lijst_var.insert(plaats, element) => bepaalde plaats toevoegen
lijst_var.extend(andere lijst) => lijsten achteraan toevoegen
```

specifiek element verwijderen

Je weet al een manier om een element op een bepaalde index te verwijderen. Stel dat je de waarde weet van het element, maar niet de indexering dan kan je het volgende toepassen.

```
lijst_var.remove(element)
```

Deze methode is echter 'stil'. Als je zeker wil zijn dat het juiste element verwijderd wordt kan heb je nog de volgende methode.

```
print(bloemen.pop(3))
print(bloemen)
```

Dit zal in de eerste opdracht het element verwijderen op de 3de index, maar het ook nog printen. De 2de print zal dan de lijst zonder 'paasbloem' weergeven.

Indexering opvragen

Als je de waarde van een element in de lijst kent kan je de indexnummer gaan opvragen.

```
lijst_var.index(element)
```

lijst kopiëren

Een kopie van een lijst maken en in een nieuwe variabele opslaan.

```
nieuwe_lijst_var = lijst_var.copy()
```

lijst omkeren

Net zoals een string de lijsts omkeren.

```
lijst_var.reverse()
```

elementen tellen

Hoeveel keer komt een element voor in de lijst?

```
lijst_var.count(element)
```

elementen sorteren

Als je een lijst met alleen numbers hebt kan je deze oplopend sorteren.

```
lijst_var.sort()
```

lijst wissen

Alle waarden uit een lijst verwijderen.

```
lijst_var.clear()
```

For-loop en lijsten

Je kan met een for-loop rechtstreeks het element gebruiken in de 'teller' van de loop.
Je gaat niet door de lengte van de lijst, maar door de elementen.

```
for i in lijst_var:
    #code
```

i bevat hier dus ineens de waarde van elk element en niet de indexnummer.
Dan gebruik je:

```
for i in range(len(lijst_var)):
    #code
```

opdrachten:

1. nummer = [10,20,30,20,10,50,60,40,80,50,40]
   Maak een nieuwe lijst waar alle dubbele waarden uit gefilterd zijn.
   Gebruik een de logische operator 'not' in de for-loop

2. Maak gebruik van if om te controleren of een lijst leeg is.

3. Kopieer een lijst met weekdagen in een nieuwe lijst.

4. 'De kat krabt de krollen van de trap'
   Maak een lijst met woorden die langer zijn dan 3 letters.
   Tip!: splits de lange string eerst in stukken bij elke spatie.

5. kleuren = [0,1,2,3,4,5,6,7,9]
   Vul een lijst met elementen waarvan de INDEX deelbaar is door 3.

6. ['abc', 'xyz', 'aba', '1221', '22', 'cedc', 'aa']
   Maak een nieuwe lijst met de elementen die langer zijn dan 2 tekens
   en het eerste en laatste teken gelijk zijn.