Herhaling

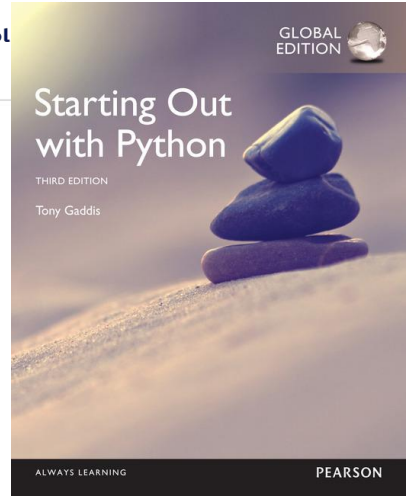


BIN-OWE1

Studiewijzer

Les	Onderwerp	
	Algemeen	**Python**
1	Linux	
2	Git	Input/output
3	Pseudocode	If/elif/else Booleans
4		For loop
5		Lists and tuples Files (CSV bestanden)
6	Flowchart	Functies
7	Git in teams	Strings CSV bestanden

Studiemateriaal



- Boek: "*Starting Out with Python, third edition*"
- Reader: "*Linux voor Bio-informatici*"
- Onderwijs Online
- Kom je er niet uit? → Google

HAN

git

Doelstellingen

- Aan het eind van deze les kan je
  - Gebruik maken van 2D lijsten en deze parsen

- Aan het eind van deze week begrijp je
  - Hoe 2D lijsten werken en hoe 2D indices werken

Inhoud

- For-loop
- While-loop
- Indices
- Lijsten

For loop

- For loop herhaalt voor een gegeven aantal waardes
- Is dus een herhaalstructuur met een zogenaamde count controlled loop
- Vooraf wordt dus bepaald hoe vaak de loop doorlopen wordt

Voorbeeld

```
for getal in [1,2,3,4]:
  print(getal)
```

Range functie

- De range functie is een handige functie om te combineren met de for loop

```
range (10)
range (4, 10)
range (4, 10, 2)
```

```
for i in range(4,10):
    print(i)
```

Wat print deze loop?

Loopen over tekst

- We kunnen ook loopen over tekst

```
for c in “Hello World!”:
  print (c)
```

```
H
e
l
l
o...
```

Loopen over lijsten

- En ook nog over lijsten

```
for w in [“Hello”, “World”, “!”]:
  print (w)
```

En wat zou deze printen?

Inhoud

- For-loop
- While-loop
- Indices
- Lijsten

While loop

- While loop is een herhaalstructuur met zogenaamde conditionele controle

- Zolang een bepaalde voorwaarde (conditie) geldig is, blijft de loop doorgaan

Voorbeeld

```
doorgaan = “ja”
while doorgaan == ”ja”:
    doorgaan = input(“Doorgaan? (ja/nee): “)
```

Pretest loop

- De while loop is een zogenaamde pretest loop

- Ofwel, van te voren test je de conditie
  - Is deze waar: doe de loop
  - Is deze onwaar: sla de loop over

Voorbeeld

```
doorgaan = “ja”
while doorgaan == ”ja”:
    doorgaan = input(“Doorgaan? (ja/nee): “)
```

Inhoud

- For-loop
- While-loop
- Indices
- Lijsten

Index syntax

```
<object>[<index>]
<object>[start:stop]
```

- Enkel element:

```
string[1]
```

- Slice:

```
string[2:4]
```

Positieve en Negatieve indices

- Positieve indices (tellen van voor naar achter, begin bij 0)

```
[2]  [:2]  [2:]  [2:4]
```

- Negatieve indices (tellen van achter naar voor, begin bij -1)

```
[-1]  [:-1] [-1:] [-4:-1]
```

Index	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**
Karakter	H	e	l	l	o		w	o	r	l	d	!

Index	**-12**	**-11**	**-10**	**-9**	**-8**	**-7**	**-6**	**-5**	**-4**	**-3**	**-2**	**-1**
Karakter	H	e	l	l	o		w	o	r	l	d	!

Inhoud

- For-loop
- While-loop
- Indices
- Lijsten

Lists

- Lijsten zijn reeksen met waardes
- Met lijsten is het mogelijk om een aantal bij elkaar horende items op te slaan
- Voorbeelden:
  - `lijst = [1,2,4,7]`
  - `namen = [“Piet","Jan","Kees"]`
  - `dieren = ["aap","konijn","vos"]`
  - `dingen = [87, “iPhone 5", "konijn", 99, 2]`
- Deze items noemen we ook wel elementen

Elementen in een list

- Lijsten hebben indices

```
lijst = [“aap”, ”konijn”, ”vos”]
```

Index	**0**	**1**	**2**
Karakter	aap	konijn	vos

Slicen van een list

- Omdat een lijst indices heeft kan ik ook een slice opvragen

```
lijst = [“aap”, ”konijn”, ”hond”, “mus”]
```

Index	**0**	**1**	**2**	**3**
Karakter	aap	konijn	hond	mus

```
print(dieren[2:4])
[‘hond’,’mus’]
```

Methodes van lists

- Ik mag vanalles aan een lijst vragen

```
>>> lijst = ["apen","noten","miezen"]
>>> lijst.
```

append
count
extend
index
insert
pop
remove
reverse
sort

Appenden

```
lijst = [“aap”, ”konijn”, ”hond”]
```

Index	0	1	2	3
Karakter	aap	konijn	hond	mus

```
lijst.append(“mus”)
```

Komt element voor in lijst

- Ik kan controleren of iets voorkomt in een lijst, of de elementen van een lijst bekijken. Dit kan ik ook voor strings doen.

```
for element in lijst:
    print(element)

dier = "konijn"
if dier in lijst:
    print("Komt voor")
```

Twee dimensionale lists

- Net als dat we for loops kunnen nesten, kunnen we lijsten ook nesten
- Dit noemen we een k dimensionale lijst (waarbij k dus het aantal nests is)
- Voorbeeld van een 2 dimensionale lijst:

```
l = [[1,  2], [3, 4]]
print (l[0])
[1,2]
print (l[1][0])
3
```

Loopen over 2D list

```
l = [["1","2"],["3","4"],["5","6"]]
for element in l:
    print(element[0])
```

```
1
3
5
```

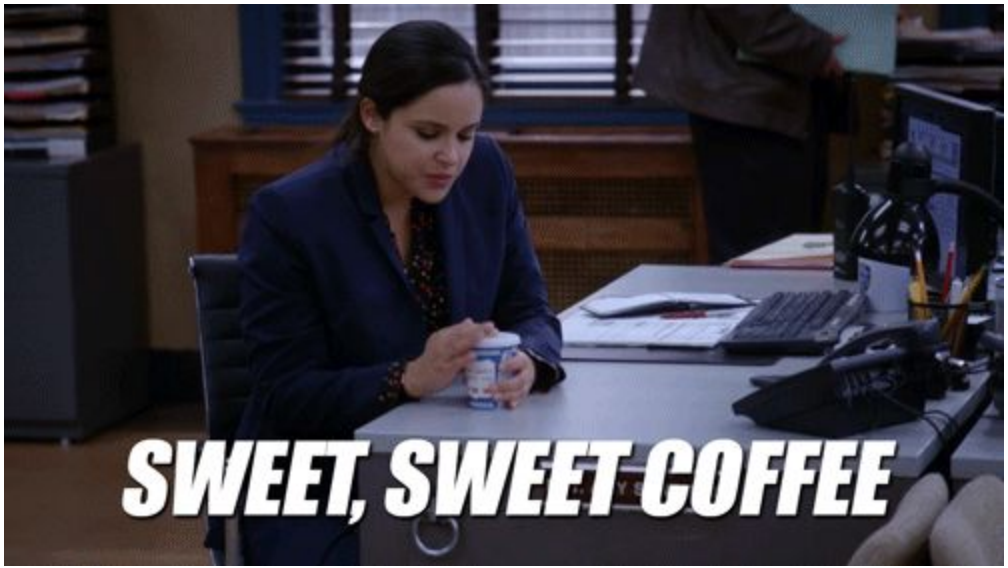
Voorbeeld Uitwerken

- We hebben een aantal lijsten met gegevens
- Omzetten naar een 2D lijst
- Gegevens ophalen uit deze 2D lijst

- Welke vorm van de lijst is het handigste?
  - Iedere sublijst is een kolom
  - Iedere sublijst is een rij

Laatste les van het blok

- Vrijdag 27-10 of Maandag 30-10
- Voorbeeld Python gedeelte Thematoets maken


SWEET, SWEET COFFEE

Opdracht

- Ga naar OnderwijsOnline
- Maak Afvinkopdracht 7
- Bij problemen:
  - Google
  - Klasgenoten
  - Docent
- Lever deze in bij de praktijkdocent

Verantwoording

- In deze uitgave is géén auteursrechtelijk beschermd werk opgenomen
- Alle teksten © Martijn van der Bruggen/Esther Kok/HAN tenzij expliciet externe bronnen zijn aangegeven
- Screenshots op basis van eigen werk auteur en/of vernoemde sites en/of fair use
- Eventuele images zijn opgenomen met vermelding van bron