METHODES
- waardegerelateerde functies -

→ Elk datatype heeft zijn eigen set methodes.

→ Het datatype ‘list’ heeft bijvoorbeeld handige methodes voor het vinden, toevoegen, verwijderen, … van waarden in een lijst.

```
index()          remove()
append()         sort()
insert()         reverse()
```

METHODES

.index()

→ Lijstwaarden beschikken over een index() methode waaraan een waarde toegekend kan worden.

→ Als de toegekende waarde zich in de lijst bevindt, wordt de index ervan geretourneerd.

```
>>> spam = ['hallo', 'hoi', 'howdy', 'hey']
>>> spam.index('hallo')
0

>>> spam.index('hey')
3
```

METHODES

.index()

→ Als de toegekende waarde zich niet in de lijst bevindt, wordt een error geretourneerd.

```
>>> spam = ['hallo', 'hoi', 'howdy', 'hey']
>>> spam.index('kat')
Traceback (most recent call last):
  File "<pyshell#31>", line 1, in <module>
    spam.index('kat')
ValueError: 'kat' is not in list
```

METHODES

.index()

→ Als de waarde zich meer dan één keer in de lijst bevindt, wordt enkel de eerste index die met deze waarde overeenstemt geretourneerd.

```
>>> spam = ['hallo', 'hoi', 'howdy', 'hey', 'hallo']
>>> spam.index('hallo')
0
```

METHODES

→ Aan de hand van de append() methode kan een waarde toegevoegd worden aan het einde van een bestaande lijst.

```
>>> huisdieren = ['Soetkin', 'Nala']
>>> huisdieren.append('Woef')
>>> huisdieren
['Soetkin', 'Nala', 'Woef']
```

METHODES

.insert()

→ Aan de hand van de insert() methode kan een waarde toegevoegd worden aan een bestaande lijst op een zelf gekozen index.

```
>>> huisdieren = ['Soetkin', 'Nala']
>>> huisdieren.insert(1, 'Woef')
>>> huisdieren
['Soetkin', 'Woef', 'Nala']
```

METHODES

→ Aan de hand van de remove() methode kan een waarde verwijderd worden uit een bestaande lijst.

```
>>> spam = ['hallo', 'hoi', 'howdy', 'hey']
>>> spam.remove('hallo')
>>> spam
['hoi', 'howdy', 'hey']
```

METHODES

→ Als de toegekende waarde zich niet in de lijst bevindt, wordt een error geretourneerd.

```
>>> spam = ['hallo', 'hoi', 'howdy', 'hey']
>>> spam.remove('kat')
>>> spam
Traceback (most recent call last):
  File "<pyshell#11>", line 1, in <module>
    spam.remove('kat')
ValueError: list.remove(x): x not in list
```

METHODES

→ Als de waarde zich meer dan één keer in de lijst bevindt, wordt enkel de eerste waarde die overeenstemt verwijderd.

```
>>> spam = ['hallo', 'hoi', 'howdy', 'hey', 'hallo']
>>> spam.remove('hallo')
>>> spam
['hoi', 'howdy', 'hey', 'hallo']
```

METHODES

→ Aan de hand van de sort() methode kunnen waarden binnen een bestaande lijst opnieuw geordend worden.

→ Als de waarden binnen de lijst enkel getallen zijn (i.e. floats en/of integers), worden deze van klein naar groot geordend.

```
>>> getallen = [2, -1, 3.14, 7, -2.1]
>>> getallen.sort()
>>> getallen
[-2.1, -1, 2, 3.14, 7]
```

METHODES

→ Als de waarden binnen de lijst enkel strings zijn , worden deze alfabetisch geordend.

```
>>> dieren = ['Soetkin', 'Nala', 'Woef']
>>> dieren.sort()
>>> dieren
['Nala', 'Soetkin', 'Woef']
```

```
>>> letters = ['A', 'b', 'a', 'B', 'c']
>>> letters.sort()
>>> letters
['A', 'B', 'a', 'b', 'c']
```

Merk op dat ‘alfabetisch ordenen’ niet letterlijk genomen mag worden. Eerst worden alle strings die met een hoofdletter beginnen geordend, pas daarna die met een kleine letter.

METHODES

→ Om strings ‘echt’ alfabetisch te ordenen, voegen we het argument `key=str.lower` toe.

```
>>> aanwezigen = ['mier', 'aap', 'Anna', 'Mark', 'leeuw']
>>> aanwezigen.sort(key=str.lower)
>>> aanwezigen
['aap', 'Anna', 'leeuw', 'Mark', 'mier']
```

METHODES

→ Omdat Python geen string-waarden kan vergelijken met getallen, kunnen lijsten die zowel strings als integers en/of floats omvatten niet geordend worden.

```
>>> verzameling = ['mier', 'aap', 1, 3.14, 'leeuw']
>>> verzameling.sort()
>>> verzameling
Traceback (most recent call last):
  File "<pyshell#70>", line 1, in <module>
    verzameling.sort()
TypeError: '<' not supported between instances of 'str' and 'int'
```

METHODES

→ Aan de hand van de reverse() methode kan de volgorde van de waarden binnen een bestaande lijst omgekeerd worden.

```
>>> getallen = [2, -1, 3.14, 7, -2.1]
>>> getallen.reverse()
>>> getallen
[-2.1, 7, 3.14, -1, 2]
```

```
>>> dieren = ['Soetkin', 'Nala', 'Woef']
>>> dieren.reverse()
>>> dieren
['Woef', 'Nala', 'Soetkin']
```