Collections

Claas de Boer, Tilman Hinnerichs

26.11.2020 und 03.12.2020

Python-Grundlagen

Was zuletzt geschah...

- Datentypen: `int, str, list, dict, …`
- Komperatoren: `==, !=, …`
- Kontrollstrukturen: `if, elif, else`
- Schleifen: `while, for`
- Funktionen: `return, def f(x): …`

Was zuletzt geschah... 2

- Objektorientiertes programmieren: `class, __init__, get`
- modellieren mit python (Wie man richtig würfelt)

Gliederung

Quiz zu Typen

Name	Funktion	iterierbar	anwendbare Funktionen	Methoden
`int`				
`float`				
`str`				
`bool`				
`list`				
`tuple`				
`set`				
`dict`				

Auflösung

Name	Funktion	iterierbar	anwendbare Funktionen	Methoden
`int`	Ganzzahl	no	+, float	`to_bytes`
`float`	Kommazahl	no	+	`__round__`
`str`	Zeichenkette	no	+, `len`	`lower()`
`bool`	Wahrheitswert	no	`int`	...
`list`	gewöhnliche Liste	yes	`len`	`append`
`tuple`	unveränderbares n-Tupel	yes	`len`	`count`
`set`	(mathematische) Menge	yes	`len`	`pop, add, update`
`dict`	Hash-Map	yes	`len`	`update, pop`

Einige Aufgaben 7

Schreibe eine Python Klasse/ein Python Programm, welche

1. ein Rechteck darstellt, dabei Höhe und Breite einliest, get und set, und eine Methode zur Berechnung der Fläche hat
2. eine/-n VolleyballspielerIn darstellt (name, Nummer)
3. welches eine Volleyballmannschaft darstellt (Mannschaft, Geld)
4. eine Funktion, welche Transfers zwischen zwei Mannschaften ausführt
5. zwei Mannschaften gegeneinander spielen lässt und ein zufälliges Ergebnis ausführt (Optional: Das Ergebnis nach Geld gewichten)
6. eine Saison von 6 Mannschaften mit Hin- und Rückrunde simuliert

Built-In functions

Bekannte Built-In Funktionen

- `int, float, str`
- `len, range, ...`

Built-In functions

Bekannte Built-In Funktionen

- `int, float, str`
- `len, range, ...`

Neue und spannende Built-In Funktionen:

- `enumerate`
- `map`
- `lambda`
- `zip`
- `filter`

enumerate

`enumerate` nimmt iterable und gibt tuple von Index und Element zurück

```
for i in range(len(my_list)-1):
    ele = my_list[i]
    f(ele, i%dim1)

for i, ele in enumerate(L):
    next\_ele = my\_list[i+1]
    f(ele, i%dim1)
```

map und lambda

```
1 def f(a,b):
2     return a%b + 42
3
4 f = lambda a,b: a%b + 42
```

map und lambda

```
1 def f(a,b):
2     return a%b + 42
3
4 f = lambda a,b: a%b + 42
```

```
1 # map(func, iterable) -> Funktion auf alle Elemente
2
3 #use the list() function to display a readable version of the
      result:
4 list(map(lambda a: a+5, list_of_numbers))
```

filter und zip

```
1  ages = [5, 12, 17, 18, 24, 32]
2
3  def myFunc(x):
4      if x < 18:
5          return False
6      else:
7          return True
8
9  adults = filter(myFunc, ages)
10
11 for x in adults:
12     print(x)
```

filter und zip

```
ages = [5, 12, 17, 18, 24, 32]

def myFunc(x):
    if x < 18:
        return False
    else:
        return True

adults = filter(myFunc, ages)

for x in adults:
    print(x)
```

```
>>> a = ("John", "Charles", "Mike")
>>> b = ("Jenny", "Christy", "Monica", "Vicky")

>>> x = zip(a, b)

#use the list() function to display a readable version of the
    result:

>>> list(x)
```

- `enumerate`: ele -> i, ele
- `map`: map(func, iter) -> func(i for i in iter)
- `zip`: zip(iter1, iter2) -> iter of tuples
- `filter`: filter(func, iter) -> [iter for i in iter if func(i)]

Einige Aufgaben 8

Schreibe ein Python Programm, welches

1. eine jedes Element einer gegebenen Liste verdoppelt
2. alle geraden Elemente eine gegebenen Liste von Zahlen zurückgibt
3. 1-dimensionale Listen von Koordinaten nimmt und diese zu 3D-Koordinaten zusammenfügt
4. 3 gleichlange Listen von Zahlen Elementweise addiert
5. die Unterschiede zweier gegebener Listen ausgibt