Listen, Ein- und Ausgaben

Claas de Boer, Tilman Hinnerichs

18.11.2020

Python-Grundlagen

Letzte Woche: Zusammenfassung

- rechnen in der Shell
- Indentation/Einrückung
- Kommentare
- Vergleichen von Werten
- Boolesche Werte (True, False, and, or)
- Kontrollstrukturen
  - if, else, elif
  - while und continue

Gliederung

Intermezzo: Listen

Was sind Listen?

Wie alltägliche Listen. In Python:

- `[]`, bzw. `[1,2,3]`
- Methoden: `insert()`, `append()`, `pop()`, `remove()`, `index()`
- siehe auch `help([])`

Wie verwende ich Listen

```
a = [1,5,3,2] # create new list
a.sort() # a is now sorted
print (a[3]) # 4th element of sorted list -> 5
a[3] = 7 # sets last/fourth(!) element to 7
highest = a.pop() # removes last element (7) and returns it
highest += 1
a.insert(0, highest) # add 8 at the start
print (a) # -> [8,1,2,3]
```

For-Schleife

For-Schleife

Klassische Zählschleife gibt es nicht in Python

↪ `for` durchläuft Listen

```
1    a = [1,2,3]
2    for elem in a:
3        print (elem)
4
```

Wie zählen?

```
1    for index in range(10):
2        print (index)
3        # prints numbers 0 to 9
4
```

```
1    a = [1,2,3]
2    for index, elem in enumerate(a):
3        print (index, elem)
4
```

Einige Aufgaben 3

Schreibe ein Python Programm, welches

1. alle Zahlen von 1 bis 20 addiert
2. alle Zahlen einer Liste aufsummiert
3. alle geraden Zahlen von 1 bis 20 ausgibt
4. eine gegebene Liste in ihrer Reihenfolge invertiert

Input

Problem

```
1    number = input('Gib eine Zahl ein: ')
2    result = 5 + number
3    print('Deine Zahl plus 5 ergibt: ' + result)
4
```

Problem: Fehlerhafte Typen

```
1    number = input('Gib eine Zahl ein: ')
2    result = 5 + number # Fehler: number ist ein String!
3    print('Deine Zahl plus 5 ergibt: ' + result)
4
```

Typen

Arbeiten mit Typen

- dynamische Typisierung
- erlaubte Operationen
- Typumwandlung (Casting)

Rückblick: Typübersicht

Name	Funktion
`int`	Ganzzahl ”beliebiger”Größe
`float`	Kommazahl ”beliebiger”Größe
`str`	Zeichenkette
`bool`	Wahrheitswert (`True`, `False`)
`list`	gewöhnliche Liste
`tuple`	unveränderbares n-Tupel
`set`	(mathematische) Menge von Objekten
`dict`	Hash-Map

Grundlagen: Typen

- Typbestimmung: `type(...)`
- Casting: `int('42')`, `float('10.3')`, etc.
- potentielle Fehlerquelle

```
a = int('42')
text = input('Deine Eingabe: ')

b = int(text) # kann einen Fehler werfen

3 + 6.3 # => 9.3
int(3 + 6.3) # => 9

def harm_folge(n):
    if type(n) == int:
        return 1/n

```

Ausgabe

Konsolenausgabe

Ausgabe über `print(...)`

```
1    print('Wundervoller Text')
2    print('Noch', 'mehr', 'wundervoller', 'Text')
3
4    day = input('Welcher Tag ist heute?')
5    time = input('Wie spaet ist es?')
6    print('Es ist ' + day + ', um ' + time + '.')
7
```

Konsolenausgabe

Ausgabe über `print(...)`

```
1     print('Wundervoller Text')
2     print('Noch', 'mehr', 'wundervoller', 'Text')
3
4     day = input('Welcher Tag ist heute?')
5     time = input('Wie spaet ist es?')
6     print('Es ist ' + day + ', um ' + time + '.')
7
8     # oder schoener:
9     print('Es ist {}, um {}.'.format(day, time))
10
```

Stringformatierung

String können mit `'...'.format(...)` formatiert werden:

```
1   number = 3
2   item = 'Katzen'
3   place = 'APB'
4
5   # per Reihenfolge
6   'Es gibt {} {} im {}!'.format(number, item, place)
7
8   # per Name
9   'Es gibt {amount} {things} im {where}!'.format(where=place,
    amount=10, things=item)
10
11  # per Reihenfolge und Name
12  'Es gibt {} {things} im {}!'.format(number, place, things='
    Koalas')
13
```

Aufgaben

Aufgaben

1. Fragt den Nutzer nach Vor- und Nachnamen und grüßt ihn!
2. Lest zwei Zahlen ein und berechnet die Summe!
3. Lest eine (vom Nutzer vorher festzulegende) Anzahl an Zahlen ein. Berechnet die Summe (Maximum, Mittelwert, ...) und gebt sie zusammen mit der Liste der Zahlen aus!
4. Modifiziert euer Programm so, dass der Nutzer nach jeder Zahl gefragt wird, ob er weitere Zahlen eingeben möchte!
5. Wählt eine beliebige natürliche Zahl. Lasst den Nutzer eure Zahl raten. Wenn er falsch geraten hat, gebt einen Hinweis ($>$, $<$)!