Serie 10 - Weitere Datenstrukturen

> **Programmieren für Naturwissenschaften FS 2023**
> Gruppe: Sofia Kessler, Florian Mohaupt, Lukas Batschelet

Aufgabe 1: 2D-Listen

Aufgabe

Gegeben ist folgendes Code-Fragment mit zwei 2D-Listen:

```
students = [["Joe", "-"],
            ["Kim", "-"],
            ["Sam", "-"]]
grades = [[5.0, 5.5, 4.5],
          [4.5, 4.0, 5.0],
          [3.5, 5.0, 6.0]]

compute_final_grades(students, grades)
print(students)
```

Schreiben Sie eine Funktion `compute_final_grades`, welche in obigem Code-Fragment aufgerufen wird. Die Funktion soll die Durchschnittsnote einer Studentin oder eines Studenten mit Hilfe der Liste `grades` berechnen und diese an zweiter Position in der entsprechenden Liste in `students` speichern. Die Noten der Studierenden befinden sich in der entsprechenden Zeile in der Liste grades. Zum Beispiel erzielte Joe die Noten 5.0, 5.5 und 4.5.

Formatieren Sie das Resultat vorher so, dass genau zwei Nachkommastellen angezeigt werden.

Die Ausgabe sollte folgendermassen aussehen:

```
[['Joe', '5.00'], ['Kim', '4.50'], ['Sam', '4.83']]
```

Tipp: Lagern Sie die Berechnung der Durchschnittsnote in eine Hilfsfunktion `compute_avg` aus.

Mögliche Lösung

```
import statistics

def compute_final_grades(students, grades):
    for i in range(len(students)):
        mean_student = statistics.mean(grades[i])
        students[i][1] = round(mean_student, 2)
    return students

students = [["Joe", "-"],
            ["Kim", "-"],
            ["Sam", "-"]]
grades =   [[5.0, 5.5, 4.5],
            [4.5, 4.0, 5.0],
            [3.5, 5.0, 6.0]]

compute_final_grades(students, grades)
print(students)
```

Aufgabe 2:

Aufgabe

Drei Vorlesungen *V1*, *V2* und *V3* werden von einer Gruppe von Studierenden besucht. Die folgende Tabelle zeigt die Belegung der Vorlesungen:

Name	V1	V2	V3
Sarah	x	x	x
Jasmine	x	x	
Dominique		x	x
Stefan			x
Uda	x		x
Nina	x		x
Berfin	x	x	
Marius	x		x

Schreiben Sie eine Funktion `get_lectures`, welche als Parameter ein Name entgegennimmt und die Vorlesungen zurückgibt, welche dieser Studierende besucht. Die Funktion soll die Datenstruktur `set` verwenden. Führen Sie die Funktion mit drei Namen aus und geben Sie die gefundenen Vorlesungen mithilfe von `print` aus

Zusatz: Geben Sie mithilfe der Datenstruktur `set` die Menge der Studierenden aus, welche sowohl V1 als auch V2 besuchen.

Mögliche Lösung

```
student_lectures = {
    "Sarah":        {"V1", "V2", "V3"},
    "Jasmine":      {"V1", "V2"      },
    "Dominique":    {      "V2", "V3"},
    "Stefan":       {            "V3"},
    "Uda":          {"V1",       "V3"},
    "Nina":         {"V1",       "V3"},
    "Berfin":       {"V1", "V2"      },
    "Marius":       {"V1",       "V3"},
    }

def get_lectures(name):
    return student_lectures.get(name, set())

def get_students_lectures(V1, V2):
    students = set()
    for name, lectures in student_lectures.items():
        if V1 in lectures and V2 in lectures:
            students.add(name)
    return students

print("Studierende die V1 und V2 besuchen:", get_students_lectures("V1", "V2"))

another = "y"

while another == "y":
    name = input("Gebgen Sie den Namen einer Studierenden Person ein: ")
    print(get_lectures(name))
    another = input("Wollen Sie einen weiteren Namen eingeben? (y/n)")
```

Aufgabe 3: Wörterbuch

Aufgabe

Definieren Sie ein Wörterbuch, das folgendermassen definiert ist (**Tipp**: Schreiben Sie eine for-Schleife):

i	*i³*
100	1'000'000
101	1'030'301
102	1'061'208
...	...
10'000	1'000'000'000'000

Ihr Programm soll dem Benutzer oder der Benutzerin danach eine Abfrage nach folgendem Muster ermöglichen:

```
Ganze Zahl eingeben: 4563
Kubikzahl Ihrer Eingabe: 9500608154
```

Mögliche Lösung

```python
import math

cubes = {}

for i in range(100, 10001):
    cubes[i] = math.pow(i, 3)

zahl = int(input("Ganze Zahl eingeben: "))

print("Kubikzahl ihrer Zahl: {:'}".format(cubes.get(zahl)))
```