Aufgabenblatt 1: Python Grundlagen

Bearbeitungszeit: 2 Wochen
Abgabe in Canvas

Aufgabe 1 (Operatoren)
Starten Sie Python mit der interaktiven Konsole python3. Experimentieren Sie mit einigen Vorlesungsbeispielen. Evaluieren Sie die folgenden Ausdrücke und machen Sie sich klar, was warum passiert.

```
2 / 5
2 // 5
3 ** 4
int("42")
float("42")
"Hallo"
print("Hallo" + " Welt")
(i, j) = (1, 2)
i
i,j
i,j = 2,3
i,j
True/2
range(10)
list(range(1,1000,100))
s = "ham"
"eggs " + 2
"ham " "and " "eggs"
s * 5
s[:0]
s[0][0][0]
("x",)[0]
"eggs"[2]
("x", "y")[1]
(1,2) + (3+4, 5)
(1,2) + (3)
(1,2) + (3,)
lis = [1,2,3] + [4,5,6]
lis[:]
lis[:0]
lis[-2]
lis[-2:]
([1,2,3] + [4,5,6])[2:4]
(lis[2],lis[3])
len(lis)
m = list("hallo"); m
m.reverse(); m
```

```
m.sort(); m
m.index("o")
{"a":1, "b":2}["b"]
d = {"x":1, "y":2, "z":3}
d["w"] = 0
d["x"] + d["w"]
d[(1,2,3)] = 4
d
d.keys()
list(d.keys())
list(d.values())
(1,2,3) in d
4 in range(10)
"dio" in "He's an idiot, but he's ok"
0 or "" or [] or () or {} or None or "Ende"
x, y = 1, 2
x, y = y, x
```

Aufgabe 2 (Heron)
Schreiben Sie ein Programm heron, um die Wurzel einer Gleitkommazahl anzunähern mit der Methode von Heron. Die Methode berechnet iterativ eine numerische $\sqrt{x}$ Sequenz a_i , die x annähert mit

$$a_0 = \frac{1+x}{2} \qquad\qquad a_{i+1} = \frac{a_i + \frac{x}{a_i}}{2}$$

Wir berechnen diese Sequenz bis zu einem Ergebniswert a_{i+1}. Wir stoppen, sobald sich a_i und a_{i+1} nur noch um einen Wert kleiner als ein gegebenes ϵ unterscheiden.

Wir übergeben die Werte von x und ϵ über die Kommandozeile in dieser Reihenfolge. Falls ϵ nicht angegeben ist verwenden Sie 10^{-6} , Vorgabe fürx ist 10.0. Starten Sie das Programm auch auf der Konsole.

Aufgabe 3
Lesen Sie in der Dokumentation nach was die eingebaute Funktion `dir` macht. Welche Methoden haben Listen? Probieren Sie sie aus.

Aufgabe 4
Erstellen Sie ein Dictionary `de2en` für die Übersetzung der Zahlen von eins, zwei und drei ins englische. Verwenden Sie als Schlüssel sowohl die Zahlen als auch die Ziffern. Welche Methoden haben Dictionaries? Probieren Sie sie aus.

Aufgabe 5
Schreiben Sie folgende for-Schleife als while-Schleife um.

```
lis = [1, 2, 3]
for ele in lis:
print(ele)
```

Schreiben Sie folgende while-Schleife als for-Schleife um.

```
dic = {1: "eins", 2: "zwei", 3: "drei"}
lis = list(dic.keys())
idx = 0
while idx < len(lis):
key = lis[idx]
print(key, dic[key])
idx = idx + 1
```

Schreiben Sie die Programme so kompakt wie möglich.

Aufgabe 6 (Bilder bearbeiten)
Schreiben Sie ein Programm `imagemanip`, das Graustufenbilder manipuliert. Sie verwenden die Python-Bibliothek PIL.Image (Hinweis 1), um einfach Graustufenbilder im PGM-Format einzulesen, zu manipulieren und zu schreiben. Es soll möglich sein Bilder aufzuhellen, zu spreizen und zu binarisieren. Das Programm akzeptiert immer vier Parameter: Die Methode, eine Zahl, die Eingabedatei und die Ausgabedatei. Zum Beispiel wendet

```
python3 imagemanip.py gamma 1.4 bilder/a.pgm out.pgm
```

die Methode gamma mit Parameter 1.4 auf die Bilddatei a.pgm an und erzeugt die Bilddatei out.pgm. Im Folgenden steht v für den Helligkeitswert eines Pixels und V (= 255) für den maximal zulässigen Helligkeitswert.

Folgende Methoden sollen unterstützt werden:

- heller, Parameter $-100 \leq p \leq 100$, int: Der Helligkeitswert v jedes Pixels wird um p Prozent des maximalen Helligkeitswerts erhöht: $v \leftarrow v + p\% \cdot V$
- gamma, Parameter $0.0 \leq \gamma \leq 10.0$, float: Der Helligkeitswert v jedes Pixels wird auf den Bereich $[0, 1]$ normiert mit γ (Gamma) potenziert: $v \leftarrow \lfloor V \cdot (\frac{v}{V})^{\gamma} \rfloor$
- binarisieren, Parameter $0 \leq s \leq V$, int: Wenn der Helligkeitswert v jedes Pixels grösSer oder gleich s ist wird er auf V gesetzt, ansonsten auf 0: $v \leftarrow \lceil (sgn(v - s) + 1)/2 \rceil$. Im manipulierten Bild gibt es also nur noch schwarze oder weiSSe Pixel.

Hinweis 1. Wir verwenden die Python Image Library (PIL), um Bilder einzulesen und zu speichern wie folgt:

```
import PIL.Image as Image
img = Image.open("lena.pgm")
cols, rows = img.size # 512, 512
v = img.getpixel((cols-1, rows-1)) # hole Wert eines Pixel (0..255)
img.putpixel((0,0), v) # setze Pixel von rechts unten nach links oben
img.save("out.pgm")
```

Aufgabe 7
Das Soundex-Verfahren wurde in den USA verwendet, um Volkszählungsda- ten nach ähnlich klingenden Nachnamen (gleicher Soundex-Wert) zu gruppieren und so schreibfehlertolerant nach englischen Wörtern zu suchen. Der Soundex-Wert eines Wortes berechnet sich wie in folgender Tabelle dargestellt, dabei ist der erste Buchstabe des Soundex-Werts der erste Buchstabe des Worts.

Buchstabe	BFPV	CGJKQSXZ	DT	L	MN	R	AEIOUWYH
Ziffer	1	2	3	4	5	6	-

Jeder nachfolgende Buchstabe wird (unabhängig von GroSS- oder Kleinschreibung) schritt- weise anhand voriger Tabelle in eine Ziffer umgewandelt oder ignoriert. Falls die Soundex- Ziffer gleich der vorhergehenden Soundex-Ziffer ist, dann wird sie ignoriert. Der Soundex- Wert besteht aus maximal 6 Zeichen, weitere Zeichen werden ignoriert. Wenn der Soundex- Wert weniger als 6 Zeichen hat, dann wird der Soundex-Wert mit der Ziffer 0 aufgefüllt.

Schreiben Sie Python-Programm `soundex`, das beliebig viele Worte als Kommandozeilen- parameter entgegen nimmt und den Soundex-Wert jedes Wortes berechnet. Beispiel:

```
> ./soundex.py soundex soundeggs flurbel
1 soundex : s53200
2 soundeggs : s53200
3 flurbel : f46140
```

Schreiben Sie ein Programm `similar`, das ein Wort und einen (optionalen) Dateinamen erwartet. Aus der Datei werden alle Worte eingelesen und die ausgegeben, die nach dem Soundex-Verfahren ähnlich zu dem ersten Argument sind. Testen Sie das Programm mit der Datei `/usr/share/dict/words`, was auch der Vorgabewert ist. Berücksichtigen Sie nur Worte, die ausschlieSSlich aus ASCII-Buchstaben bestehen.

Aufgabe 8 (List-Comprehension)
Schreiben Sie List-Comprehension-Ausdrücke zur Berechnung von:

- Geradzahlige Kubikzahlen der Zahlen 1 bis 10
- Alle Teiler einer Zahl z auSSer 1 und z (testen Sie mit 123, 12345, 123456)
- Alle Primzahlen zwischen 10000 und 10100

Schreiben Sie die obigen Ausdrücke nochmals, aber nur unter Verwendung der funktionalen Primitiven (`map, filter, reduce`) ohne List-Comprehension.

Aufgabe 9
Ramanujan wurde von einmal Hardy besucht, der erwähnte, dass er ein Taxi mit der Nummer 1729 genommen hatte. Ramanujan antwortete, dass die 1729 bemerkenswert sei, da es die kleinste Zahl ist, die in zwei verschiedenen Arten als Summe von zwei Kubikzahlen dargestellt werden kann.

$$1729 = 1^3 + 12^3 = 9^3 + 10^3$$

Schreiben Sie einen Python-Ausdruck, der Ihnen hilft diese Aussage nachzuvollziehen. Experimentieren Sie schrittweise mit List-Comprehension. Suchen Sie in allen Kombinationen von Zahlen bis zu einer Grenze nach vier paarweise verschiedenen Zahlen, von denen die Summe der Kubikzahlen von je zwei gleich ist. Erhöhen Sie die Grenze (interaktiv) immer weiter, bis die Liste der 4-Tupel nicht leer ist, geben Sie davon das Minimum aus.