Tag 4a - Python

Aufgabe 1: Listen

Mache dich im Python-Interpreter mit dem Umgang mit Listen vertraut. Kapitel 2.6 im Skript (S.40) enthält nützliche Informationen.

(a) Lege eine Liste mit Namen "zahlen" an, die mindestens 10 Elemente enthält.
(b) Gib das 7. Element der Liste aus.
(c) Gib das 2.-8. Element der Liste als Liste aus.
(d) Gib das 2.-8. Element der Liste als einzelne Elemente aus.
(e) Füge die Zahlen 15, 23 und 95 zur Liste hinzu.
(f) Lösche das 3. Element der Liste
(g) Lösche die Zahl 23 aus der Liste

Aufgabe 2: Funktionen

Schreibe eine Funktion, die eine Liste von Zahlen entgegennimmt und

(a) Das kleinste Elemement der Liste ausgibt
(b) Das größte Element der Liste ausgibt
(c) Alle Elemente der Liste ausgibt, die durch 5 teilbar sind.
(d) Alle Elemente der Liste ausgibt, die durch eine beliebige Zahl, die der Funktion übergeben wird, teilbar sind.

Verwende nicht die eingebauten Funktionen `min` oder `max`.

Aufgabe 3: Namensräume und Shadowing

Folgender Programmcode ist gegeben:

```
1 x = "foo"
2 def x():
3     x = "bar"
4     print(x)
5 print(x)
```

(a) Was wird das Programm ausgeben?
  a) Das Programm terminiert mit einem Fehler.
  b) Das Programm gibt zuerst "bar", dann "foo" aus.
  c) Das Programm gibt etwas wie "<function x at 0x100495578 >" aus.
  d) Das Programm gibt nur "foo" aus.
(b) Wie wäre die Ausgabe, wenn die Funktion einen anderen Namen als "x" hätte?

Aufgabe 4: Dateien lesen und schreiben

Erstelle mit einem Editor eine .txt-Datei, wie folgt:

- Gib eine Zahl ein.
- Wechsel mit „Enter“ in die nächste Zeile und gib noch eine Zahl ein,

Speichere diese Datei als eingabe.txt ab.
Erstelle nun ein Programm, welches die beiden Zahlen aus der Datei ausliest und eine Grundrechenoperation auf diesen ausführt (z.B. die beiden Zahlen addiert). Das Ergebnis dieser Operation soll in einer Datei ausgabe.txt abgespeichert werden.

Aufgabe 5: Rekursion und Interation

Schreibe eine Funktion, die eine natürliche Zahl n als Parameter bekommt und das Produkt aller natürlichen Zahlen von 1 bis einschließlich n ausgibt.

(a) Implementiere diese Funktion iterativ

(b) Implementiere diese Funktion rekursiv

Hinweis: Das Beispiel der Summen-Funktion im Skript (S.60) könnte hilfreich sein.

Aufgabe 6: Karnickel

Die Fibonacci-Folge ist eine unendliche Folge natürlicher Zahlen, die folgendermaßen definert ist:

$$fib(n) := \begin{cases} 1, & \text{falls } n = 1 \text{ oder } n = 2 \\ fib(n-1) + fib(n-2), & \text{sonst.} \end{cases}$$

(a) Implementiere eine Funktion `fib(n)`, die die n-te Fibonacci-Zahl rekursiv berechnet.

(b) Implementiere die Funktion iterativ.

(c) Vergleiche wie lange die iterative und die rekursive Funktion benötigt um `fib(35)`, `fib(40)` oder `fib(45)` zu berechnen.
Hinweis: mit der Tastenkombination Strg + c kann ein laufendes Programm im Python-Interpreter abgebrochen werden.

Viel Erfolg!