Tag 4b - Python

Aufgabe 1: Geschachtelte Funktionen

Schreibe eine Funktion, die zwei Integer addiert und dann eine andere Funktion aufruft, in der das Ergebnis der ersten Funktion mit 2 multipliziert wird.

Aufgabe 2: Programmanalyse

(a) Was macht das folgende Programm? Beschreibe die einzelnen Schritte.
Hinweis: Am PC ausprobieren.

```
def programm1():
    liste = [97, 43, 24, 2, 7, 12]
    print(liste)
    listLength = len(liste)
    for a in range(0, listLength - 1):
        lastUnprocessedValueIndex = listLength - 1 - a
        for b in range(0, lastUnprocessedValueIndex):
            if liste[b] > liste[b + 1]:
                liste[b], liste[b + 1] = liste[b + 1], liste[b]
    print(liste)
```

(b) Modifiziere dieses Programm, sodass die Zahlen rückwärts sortiert werden.

(c) Es ergibt wenig Sinn, eine fest definierte Liste immer wieder aufs Neue zu sortieren. Schreibe daher ein Programm, mit dem der Benutzer eine Liste zur Laufzeit eingeben kann.
Beachte hierbei:

1. Gibt der Benutzer eine Zahl ein, wird diese in die Liste eingetragen
2. Wird ein leerer String eingegeben, wird die Eingabe beendet
3. Wird etwas anderes eingegeben, teile dem Benutzer mit, dass nur Zahlen eingegeben werden dürfen und ignoriere die Eingabe

Hinweis: Eine do-while-Schleife kann man in Python mithilfe von `while True` und `break` imitieren

(d) Kombiniere die beiden Programme, sodass der Benutzer eine Liste eingibt und diese anschließend per Bubble-Sort sortiert wird.

Aufgabe 3: Selectionsort

Um eine Liste mit n Zahlen zu sortieren, arbeitet der Selectionsort-Algorithmus in $(n-1)$ Phasen. Zunächt bestimmt er die kleinste Zahl der gesamten Liste und vertauscht diese mit der Zahl, die an erster Position der Liste steht. Dann bestimmt Slectionsort die kleinste Zahl der letzten $(n-1)$ Elemente der Liste und vertauscht diese mit der Zahl an Posititon 2, usw., bis die gesamte Liste durchgearbeitet ist.

(a) Schreibe eine Funktion, die den Index der kleinsten Zahl einer ihr ihr übergebenen Liste findet, und diesen zurückgibt.

(b) Schreibe eine Funktion `selsort(<list>)`, die eine unosrtierte Liste von Zahlen entgegennimmt und eine sortierte Liste zurückgibt.

(c) Schreibe ein Programm, das Zahlen aus einer Datei liest, diese mit der Funktion `selsort( )` sortiert und wieder in die Datei zurückschreibt.

Aufgabe 4: Rekursion und Iteration

(a) Eine positive ganze Zahl a ist durch eine positive ganze Zahl b ganzzahlig ohne Rest teilbar, wenn sich b mehrmals von a subtrahieren lässt, sodass das Ergebnis schließlich 0 ergibt. Implementiere eine Funktion, die überprüft, ob a durch b teilbar ist. Verwende an mathematischen Operationen ausschließlich die Subtraktion!
Schreibe sowohl eine iterative als auch eine rekursive Variante.

(b) Ein Palindrom ist eine Zeichenkette, die von vorne und von hinten gleich, d.h. symmetrisch ist. 'otto', 'anna', 'abcba', '666' und 'X' sind beispielsweise Palindrome.
Implementiere eine Funktion, die als Parameter einen String erwartet, und überprüft, ob dieser String ein Palindrom darstellt. Der Rückgabewert der Funktion soll `True` bzw. `False` lauten.
Schreibe sowohl eine iterative als auch eine rekursive Variante.
Hinweis:

- Unterscheiden sich bereits das erste und das letzte Zeichen, kann kein Palindrom vorliegen.
- Gleichen sich das erste und das letzte Zeichen...

Aufgabe 5: Collatz-Funktion

Die Collatz-Funktion ist mathematisch wie folgt definiert:

$$c(n) := \begin{cases} c(n/2), & \text{falls n gerade ist} \\ c(3n+1), & \text{falls n ungerade ist} \end{cases}$$

(a) Implementiere die Collatz-Funktion aus der Vorlesung als rekursive Python-Funktion. Die Funktion soll einen beliebigen Startwert als Parameter erhalten, bei jedem Aufruf zunächst die aktuelle Zahl ausgeben, und terminieren, sobald das erste Mal die Zahl 1 in der Folge auftaucht.
Zur Erinnerung, die Collatz-Funktion ist folgendermaßen definiert:

$$c(n) := \begin{cases} c(n/2), & \text{falls n gerade ist} \\ c(3n+1), & \text{falls n ungerade ist} \end{cases}$$

(b) Erweitere die Funktion so, dass die Anzahl der Schritte bis zum ersten Auftauchen der Zahl 1 ausgegeben wird.

(c) Führe die Funktion für die Startwerte $1, 2, 3, \ldots, 1000$ aus und speichere jeweils die Anzahl der Schritte in einer Liste.

(d) Werte die in der Liste gespeicherten Schrittzahlen aus. Was ist die durchschnittliche Schrittzahl? Bei welchen Startwerten wurde die maximale Schrittzahl erreicht, bei welchen die minimale Schrittzahl? Woran liegt das?

(e) Implementiere die Collatz-Funktion nun iterativ statt rekursiv.

(f) **(Achtung, schwierig!)** Finde einen Startwert $n \in \mathbb{N}$, sodass die Collatz-Folge niemals die Zahl 1 erreicht. Was bedeutet das für den Startwert? Merke Dir die Zahl und setze Dich umgehend mit Deinem Tutor in Verbindung.

Aufgabe 6: Programmieraufgaben

In dieser Aufgabe ist eine einfache Relation in einem Python-Programm umzusetzen, die die Liebesbeziehungen zwischen verschiedenen Personen festhält. Eine Person x liebt eine Person y, wenn (x, y) in der Relation der Liebesbeziehungen enthalten ist.

Die Relation können in Python beispielsweise in Form einer Liste von Tupeln festgehalten werden (Siehe auch [1]).

(a) Schlage in einem Lexikon Deiner Wahl folgende Begriffe nach: *Symmetrie (bei Relationen), Reflexivität, Transitivität.* Überlege Dir, ob die hier vorgestellte Relation der Liebesbeziehungen symmetrisch, reflexiv und/oder transitiv ist.

(b) Schreibe eine Funktion *verliebt_sich*, die die Namen des/der Verliebten und der/des Geliebten als Argumente übergeben bekommt und diese in die Liebesrelation einfügt.

(c) Schreibe eine Funktion *wen_liebt*, die einen Namen des/der Verliebten als Argument übergeben bekommt, und eine Liste der Personen, die diese Person liebt, zurückgibt.

(d) Schreibe eine Funktion *liebespaare*, die eine Liste aller Liebespaare zurückliefert. Ein Liebespaar liegt vor, wenn sowohl (x,y) als auch (y,x) in der Liebesrelation vorhanden sind.

Viel Erfolg!

[1] http://docs.python.org/library/datatypes.html