Skriptsprachenorientierte Programmiertechnik

Die Aufgaben sind in der Programmiersprache Python zu lösen.
Die Verwendung von Funktionen aus Bibliotheken ist nur für re, datetime, date und time erlaubt.

Die üblichen Einrückregeln müssen beachtet werden.

Aufgabe 1: Funktionen (9 Punkte)

a) Schreiben Sie in Python eine Funktion, die einen Startwert und einen Endwert übergeben bekommt und die Summe der Zahlen zwischen Startwert und Endwert berechnet und zurückliefert.
(4 Punkte)

```
def teilSumme(start,end):
    summe =0
    for i in range(start,end+1):
        summe +=i
    return summe
```

b) Schreiben Sie eine Funktion BinToDez, die einen String als Parameter bekommt (der nur 1 und 0 enthält) und die entsprechende Dezimalzahl zurückliefert.
Beispiel:
$10110 = 1*2^4 + 0*2^3 + 1*2^2 + 1*2^1 + 0*2^0$
(5 Punkte)

```
def binToDez(s):
    dez=0
    for z in s:
        dez = dez*2 + int(z)
    return dez
```

Aufgabe 2: Listen und Funktionen (11 Punkte)

a) Folgende Liste ist gegeben:
zahlen =[-1,5,-10,0,2,3,4,-2,20]

Geben Sie die Anweisungen an um eine Liste der positiven Zahlen absteigend sortiert zu erzeugen.
(3 Punkte)

```
neuZ =[]
for i in zahlen:
    if i>0:
```

```
        neuZ.append(i)
neuZ = sorted(neuZ)
neuZ.reverse()
```

b) Schreiben Sie eine Funktion, die aus zwei Listen (Parameter der Funktion) eine neue Liste macht, die die gemeinsamen Elemente der beiden Listen enthält und zurückliefert. (4 Punkte)

```
def Vereinige(list1,list2):
    neuList=[]
    for i1 in list1:
        if i1 in list2:
            neuList.append(i1)
    return neuList
```

c) Schreiben Sie eine Funktion, die alle Vorkommen eines Wertes (Parameter) aus einer Liste (Parameter) entfernt und die geänderte Liste zurückliefert.
(4 Punkte)

```
def entferne(liste,item):
    neuListe=[]
    for x in liste:
        if x != item:
            neuListe.append(x)
    return neuListe
# alternative Lösung
def removeWert(liste,wert):
    while wert in liste:
        liste.remove(wert)
    return liste
```

Aufgabe 3 (10 Punkte)

a) Ein Dictionary, mit Namen und zugehörigem Gehalt ist gegeben:
z.B.
gehalt = {'Maier':1800, 'Schmidt': 2500, 'Mustermann':3100,‘Schmidt‘:3500}

Geben Sie die Anweisungen für folgende Funktionen

- Geben Sie die Namen im Dictionary alphabetisch sortiert aus

```
print(sorted(gehalt.keys())
```

- Fügen Sie Musterfrau mit dem Gehalt 2900 hinzu.

```
gehalt['Musterfrau'] = 2900
```

- Löschen Sie den Mitarbeiter Mustermann

```
del gehalt['Mustermann']
```

- Lesen Sie den Namen eines Mitarbeiters ein und geben Sie dann seinen Gehalt aus

```
name=input()
print(gehalt[name])
```

- Geben Sie die Namen aller Mitarbeiter aus, deren Gehalt größer als aus als ein Wert (Variable limit) ist, der über Konsole eingelesen wird. Das Einlesen von Konsole können Sie als gegeben annehmen.

```
for k,v in gehalt.items():
   if v > limit:
        print(k)
```

(6 Punkte)

b) Im folgenden Dictionary sind zu jedem Fach erreichte Noten gespeichert.

Noten = {"Mathe":[1.7,1.3,2.0], "Deutsch":[2.3,3.0], "Biologie":[2.3]}

- Geben Sie die Anweisungen an um zu jedem Fach die Durchschnittsnote auszugeben
- Fügen Sie ein weiteres Fach, z.B. Englisch, mit zwei Noten hinzu, aber nur, wenn das Fach noch nicht enthalten ist. Falls es schon enthalten ist, machen Sie nichts.(4 Punkte)

```
for k,v in Noten:
     S =0
     for i in v:
               S +=i
     print(k,”Notendurchschnitt: “,s/len(v))

if “English” not in Noten:
     Noten[“English”] = [2,3]
```

Aufgabe 4 Objektorientierung (10 Punkte)

a) Entwerfen Sie eine Klasse Clock zur Verwaltung von Uhrzeiten mit den privaten Daten hour und minute.
Implementieren Sie folgende Methoden:

- Init-Methode
- Tick: erhöht die Anzahl der Minuten um 1

Legen Sie ein Objekt der Klasse Clock an.
(5 Punkte)

```
class Clock(object):
    def __init__(self,hour,minute):
        self.__hour = hour
        self.__minute = minute
    def Tick(self):
        self.__minute = self.__minute+1

 cl1 = Clock(2,35)
```

b) Überschreiben Sie in der Klasse Clock die Funktion str
(2 Punkte)

```
def __str__(self):
        return str(self.__hour)+":"+str(self.__minute)
```

c) Leiten Sie von der Klasse Clock eine Klasse ClockWithSeconds ab
(3 Punkte)

- Sehen zusätzlich ein Element für die Sekunden vor
- Geben Sie init unter Verwendung von init der Basisklasse an

```
class ClockWithSeconds(Clock):
    def __init__(self,hour,minute,second):
          super().__init__(hour,minute)
          self.__second = second
```

Aufgabe 5 (12 Punkte)

a) Geben Sie die regulären Ausdrücke für folgende Zeichenfolgen

-ein Plus(+) oder ein (-) Zeichen gefolgt von mindestens einer Ziffer
[+-][0-9]+

- eine Zeile, in der nur das Wort „Test“ steht.
^Test$

(2 Punkte)

b) Ergänzen Sie in der folgenden Tabelle jede Zeile für den Match True oder False:

RegEx	Untersuchte Strings	Match?
al.e al+e all?e	Alle Voegel fliegen hoch Finale am Sonntag Walter liebt Waerme.	**false** **true** **false**
A[lt]?e al[lt]?e a[l]*e	Alle Voegel fliegen hoch Finale am Sonntag Walter liebt Waerme.	**false** **true** **true**

(6 Punkte)

c) In der Variablen text ist eine Zeichenkette gespeichert.

Geben Sie die Anweisungen zu folgenden Aufgabenstellung an:

- Extrahieren Sie alle Zeichen, die in runden Klammern stehen.
- Ersetzen Sie alle (durch [und) durch]

(4 Punkte)

```
x = re.findall(r"\((\w+)\)",text)
if x != None:
    text =re.sub(r"\(","[",text)
    text =re.sub(r"\)","]",text)
```

Aufgabe 6: Dateien, Listen, Funktionen (8 Punkte)

Gegeben ist folgender Python-Programmcode, bei dem aus einer csv-Datei Daten gelesen werden. Die Datei hat folgenden Aufbau:
10;Bahnhof;7,2;12,4
15;Bahnhof; 10,4;22,5
10;Flughafen;8,3;13,1
15;Flughafen;11,8;23,2
16;Bahnhof;11,8;25,6

Die einzelnen Spalten haben folgende Bedeutung:
Spalte1: Tag
Spalte2:Mess-Ort
Spalte3: Minimaltemperatur
Spalte4:Maximaltemperatur

```
import csv
try:
   with open("Wetter.csv") as file:
      reader = csv.reader(file,delimiter=";")
      # Spaltenüberschrift lesen
      header = next(reader)
      wetter=[]
      for row in reader:
         wetter.append(row)
except:
   print ("cannot open file")
```

Schreiben Sie eine Funktion MaxTemperatur, die einen Tag und eine Liste mit den Daten aus der csv-Datei übergeben bekommt und die Maximaltemperatur für diesen Tag bestimmt und zurückliefert.
(8 Punkte)

```
def MaximalTemp(liste,tag):
    maxT = -273
    for item in liste:
        if item[0] == tag:
            temp = item[3].replace(',','.')
            if float(temp) > maxT:
                maxT = temp
    return maxT
```