Übungszettel 2b - Python

Aufgabe 1: Windchill Temperatur

Der Mensch empfindet Temperaturen häufig anders als sie tatsächlich sind. Grund hierfür ist vor allem der Wind, aber auch andere Rahmenbedingungen.

Es gibt für die Berrechnung der gefühlten Temperatur zwei verschiedene Berechnungsmöglichkeiten, eine veraltete und eine aktuelle.

$$W_{alt} = 33 + (0,478 + 0,237 \cdot \sqrt{v} - 0,0124 \cdot v) \cdot (T - 33)$$

$$W_{neu} = 13,12 + 0,6215 \cdot T - 11,37 \cdot v^{0,16} + 0,3965 \cdot T \cdot v^{0,16}$$

Dabei bezeichnet W die Windchill-Temperatur in °C, T die reale Temperatur in °C und v die Windgeschwindigkeit in km/h.

Implementiere beide Varianten für die Windchill-Temperatur. Dabei sollen folgende Schritte ausgeführt werden:

- Lies die tatsächliche Temperatur und Windgeschwindigkeit anhand einer Benutzereingabe ein
- Gib dem Benutzer einen Hinweis, in welchem Format Eingaben gemacht werden müssen!
- Überprüfe mit Hilfe einer if-Abfrage, ob die Eingabe des Nutzers als `int` interpretiert werden kann.
- Berechne die Werte der Windchill-Temperatur (alt und neu)
- Gib diese Werte (sinnvoll) aus.

Solution: Hier geht's darum, dass man übt mit den Operatoren umzugehen. Natürlich gibt es die Funktion `sqrt` im Modul `math`, soweit sind wir in der Vorlesung aber noch nicht. Natürlich dürfen die Leute es trotzdem benutzen, falls sie es schon kennen oder herausfinden, wie man das importieren und benutzen kann.

Für alle anderen wäre es super, wenn ihnen einfällt, dass $\sqrt{a} = a^{1/2}$ ist.

Aufpassen muss man, dass Kommazahlen in python mit Dezimalpunkt eingegeben werden müssen und mit sinvoller Ausgabe ist gemeint, dass man sich Gedanken darüber macht, wie viele Nachkommastellen man wohl bei der Temperaturausgabe braucht, wenn die Eingabe ohne Nachkommastelle erfolgen muss (`isdigit()` gibt False zurück, wenn ein Punkt oder Komma in der Eingabe auftaucht).
Wenn Leute `try... except` kennen, können sie das natürlich auch eleganter machen und auch Gleitkommaeingaben akzeptieren.

```
####################
# Windchill-Temperatur berechnen
####################
print('''Dieses Programm berechnet die gefühlte Temperatur nach alter und
    neuer
Berechnung. Gemessene Temperatur und Windgeschwindigkeit müssen ohne
Nachkommastellen eingegeben werden.''')
```

```
7
8  while True:
9      T_string=input("aktuelle Temperatur in Celsius: ")
10     if T_string.isdigit():
11         T_int=int(T_string)
12     else:
13         print("Bitte starten Sie das Programm erneut und geben Sie
               ausschliesslich ganze Zahlen ein")
14         break
15     v_string=input("Windgeschwindigkeit in km/h: ")
16     if v_string.isdigit():
17         v_int=int(v_string)
18     else:
19         print("Bitte starten Sie das Programm erneut und geben Sie
               ausschliesslich ganze Zahlen ein")
20         break
21     W_alt=33+(0.478+0.237*(v_int**(1/2))-0.0124 * v_int)*(T_int-33)
22     W_neu=13.12+0.6215 * T_int - 11.37 * (v_int**0.16) + 0.3965 * T_int * (
           v_int**0.16)
23
24     print("Windchill (alt) in Celsius:", round(W_alt,0))
25     print("Windchill (neu) in Celsius:", round(W_neu,0))
26     break
```

Aufgabe 2: Zahlentrick

Es gibt einen Zahlentrick, der folgendermaßen funktioniert:
Man bittet jemanden, sich eine einstellige Zahl auszudenken. Dann soll er zu dieser Zahl noch einmal dieselbe Zahl addieren. Danach soll die 10 hinzuaddiert werden und durch 2 geteilt werden. Schließlich wird noch die gedachte Zahl abgezogen.
Egal, welche Zahl ausgedacht wurde - das Ergebnis ist immer 5.
Um dies zu überprüfen, schreibe ein Programm, das eine einstellige Zahl als Eingabe bekommt und führe genau diese Rechenschritte mit dieser Zahl aus. Gib das Ergebnis aus.

Solution:

```
1  #!/usr/bin/env python
2  # encoding: utf-8
3  """
4  Zahlentrick.py
5
6  Created by Linda Luy on 2010-09-26.
7  """
8
9  zahl = int(input("Bitte gib eine einstellige Zahl ein: "))
10 loesung = (((zahl+zahl)+10)/2)-zahl
11 print("Die Loesung ist ",loesung ,".")
```

Aufgabe 3: Chiffrierung

Zeichen (Buchstaben, Ziffern, Interpunktionszeichen, etc) werden rechnerintern mithilfe einer Tabelle verwaltet, die jedem Zeichen eine Zahl zuordnet. Eine Möglichkeit der Textverschlüsselung ist, für jedes Zeichen des Textes das um n Tabelleneinträge verschobene Zeichen zu schreiben. Falls beim Verschieben das Ende der Tabelle erricht wird, so wird beim Anfang weitergemacht. (Der `%`-Operator ist hier hilfreich).

Beispiel
Bei einem Verschiebungsoffset von $n = 1$ wird z. B. aus 'HAL' der String 'IBM'

(a) Schreibe ein Programm, welches den Benutzer zunächst auffordert einen Wert einzugeben, der den Verschiebungsoffset n festlegt. Dann soll der Benutzer aufgefordert werden den zu verschlüsselnden Text einzugeben. Nach Eingabe des Textes soll der chiffrierte Text auf dem Bildschirm ausgegeben werden.

(b) Schreibe ein zweites Programm, das einen chiffrierten Text als Eingabe erwartet und den dechiffrierten Text auf dem Monitor ausgibt.

Solution: Die Lösungen arbeiten alle mit einer Tabelle von 256 Zeichen. Es wird aber in Python wohl Unicode verwendet. Ich habe nicht herausgefunden, wie viele Zeichen es gibt. So geht es aber, es wird halt nur mit den ersten 256 Zeichen gearbeitet. Sollte für die meisten eingegebenen Texte aber ausreichen.

Built-in Funktionen `ord()` und `char()` sind im Skript S.18 erwähnt.

(a)
```
##############################
# Verschlüsselungsprogramm
##############################

n = int(input('Geben Sie eine Schrittweite für die Verschlüsselung an: '))
text = input('Geben Sie einen Text ein, der verschlüsselt werden soll: ')
ausgabe = '' # string variable für chiffrierten Text
for item in text:
    new = (ord(item)+n)%256 # (Ordnungszahl des Zeichens + n) mod 256 (ASCII-Zeichen) = neue Ordnungszahl
    ausgabe = ausgabe + chr(new) # Neues Zeichen in Ausgabestring schreiben
print('Der chiffrierte Text lautet: ', ausgabe)
```

(b)
```
#####################################
# Verschlüsselungsprogramm
#   Dechiffrierung
#####################################

n = int(input('Was war die Verschlüsselungszahl? '))

text = input('Geben Sie einen Text ein der entschlüsselt werden soll: ')
ausgabe = '' # string Variable für dechiffrierten Text
for item in text:
    new = (ord(item)+(256-n))%256 # Verschlüsselung rückgängig machen
    ausgabe = ausgabe + chr(new) # Neues Zeichen in Ausgabestring schreiben
print('Der dechiffrierte Text lautet: ', ausgabe)
```

Aufgabe 4: Schleifen

(a) Schreibe ein Programm, das alle ungeraden Zahlen von 1 bis 100 ausgibt.

1. Mithilfe einer for-Schleife
2. Mithilfe einer while-Schleife

Solution:

```
def ungerade_for_range():
    for a in range(1, 100 + 1, 2):
        print(a)

```

```python
def ungerade_for_mod():
    for a in range(1, 100):
        if (a % 2) == 1:
            print(a)

def ungerade_while():
    a = 1
    while(a <= 100):
        print(a)
        a += 2
```

(b) Schreibe ein Programm, das den Benutzer auffordert 5 Zahlen einzugeben, und dann den Mittelwert der Zahlen berechnet und ausgibt.
Verwende dabei eine Schleife.

Solution:

```python
s=0
for a in range(0,5):
    s+=int(input("Gib eine Zahl ein:"))

mean=s/5

print("Der Mittelwert ist: " + str(mean))
```

(c) **schwierig** Schreibe ein Programm, das den Mittelwert beliebig vieler Zahlen berechnet und ausgibt.
Tipp: Überlege dir, woran das Programm erkennen soll, dass alle Zahlen eingegeben sind und der Mittelwert nun berechnet werden soll.

Solution: Es gibt zwei Ansätze. Entweder ich frage den Benutzer erst wieviele Zahlen eingegeben werden sollen und speichere den Wert in einer Variablen. Oder mit while TRUE und der Benutzer muss ein Abbruchzeichen eingeben.

```python
stop=int(input("Wieviel Zahlen sollen eingegeben werden? "))
s=0
for i in range(0,stop):
    s+=int(input("Gib eine Zahl ein: "))
mean=s/stop
print("Der Mittelwert ist: " + str(mean))
```

```python
print("Wenn Sie die Eingabe beenden wollen, geben Sie \"0\" ein.")
s=0
count=0
while(True):
    tmp=int(input("Gib eine Zahl ein: "))
    if (tmp==0):
        break
    s+=tmp
    count+=1
mean=s/count
print("Der Mittelwert ist: " + str(mean))
```

(d) Schreibe ein Programm, das die Zahlen von 1 bis $n = 4$ paarweise miteinander multipliziert und das Ergebnis ausgibt.
Die Ausgabe könnte z. B. so aussehen:

```
1 x 1 = 1
```

```
1 x 2 = 2
1 x 3 = 3
...
4 x 4 = 16
```

Solution:

```python
def static_mul_table():
    for a in range(1, 5):
        for b in range(1, 5):
            print(str(a) + " x " + str(b) + " = " + str(a * b))
```

(e) Erweitere Dein Programm um eine Abfrage nach der Zahl n, bis zu der die Produkte berechnet werden sollen (d.h. n soll nun nicht mehr auf 4 festgelegt sein).

Solution:

```python
def adv_mul_table():
    print("Will output: (1 x 1)  to (n x n)")
    biggest = input("n = ")
    biggest += 1 # for the range()-function
    for a in range(1, biggest):
        for b in range(1, biggest):
            print(str(a) + " x " + str(b) + " = " + str(a * b))
```

(f) Schreibe ein Programm, das eine Multiplikationstabelle erzeugt und ausgibt.
Hinweis: Eine *Multiplikationstabelle* ist eine Tabelle mit natürlichen Zeilen- und Spaltenindizes, die in Feld i, j die Zahl $i \cdot j$ enthält.

Solution:

```python
def multiplicationstabelle_simple(i,j):
    """A simple implementation"""
    for y in range(1, i + 1):
        for x in range(1, j + 1):
            print(y * x, end = ', '), # single product
        print() # new line

def multiplicationstabelle_nice(i, j):
    """An implementation with nice formatting"""
    print("    ", end = "") # for a better look
    for x in range(1, j + 1):
        print("%3d"  % x, end = '  ') # draw a column-id
    print()# end the line
    for y in range(1, i + 1):
        print("%2d" % y, end = '  ') # draw a row-id
        for x in range(1, j + 1):
            print("%3d" % (y * x), end = '  ') # draw a product
        print ()# end the line
```

Aufgabe 5: Programmieraufgaben

(a) Schreibe ein Programm, welches ein Passwort verlangt. Gibt der Benutzer das richtige Passwort ein, soll das Programm „Authentifizierung erfolgreich“ ausgeben. Wird das Passwort jedoch dreimal hintereinander falsch eingegeben, soll die Ausgabe „Zugriff verweigert“ lauten.

(b) Schreibe ein Programm, welches eine Zahl als Eingabe anfordert und alle Primzahlen bis zu dieser Zahl ausgibt. Eventuell ist es hilfreich, sich über das Thema „Primfaktorzerlegung“ und den Modulo- (%-) Operator zu informieren.

(c) Schreibe ein Programm, welches den Benutzer auffordert, sich eine Zahl zwischen 1 und 100 auszudenken. Das Programm soll dann sukzessive Fragen der Form „Ist die Zahl größer/kleiner als. . . “ stellen und so die ausgedachte Zahl mit möglichst wenigen Fragen erraten. Eventuell ist es hilfreich, das Thema „Binärsuche“ zu recherchieren.

Solution:

(a)
```
####################################
# Programm zur Authentifizierung
####################################

passwort = 'vorkurs15/16'
korektur = False
count = 0
for i in range(0,3):
    count += 1
    pw = input('Bitte geben Sie das Passwort ein: ')
    if(pw == passwort):
        print('Authentifizierung erfolgreich')
        break
    elif(count==3):
        print('Zugriff verweigert')
        break
    else:
        print('neuer Versuch')
```

(b) Da kommt einem gleich das Sieb des Eratosthenes in den Kopf. Allerdings kennen die Teilnehmer zu diesem Zeitpunkt noch keine Listen, also ist es schwierig, eine dynamische Anzahl von Werten zu verwalten. Daher die naive Methode für jede Zahl n auszuprobieren, ob sich unter den Zahlen $2, \ldots (n-1)$ ein Teiler findet. Diese Methode kann man schneller machen, wenn man sich überlegt, dass man eigentlich nur die Zahlen $2, \ldots, \sqrt{n}$ als mögliche Teiler testen muss.

```
####################################
# Programm gibt alle Primzahlen bis zur
#  angegebenen Grenze n aus.
####################################

while True:
    grenze = input('Bitte geben Sie eine Zahl als Obergrenze an. ')
    if (grenze.isdigit()): # überprüfen, ob Eingabe eine Ganzzahl ist
          break
    else:
        print('Sie müssen einen ganzzahligen Wert eingeben')
limit = int(grenze) # umwandeln in integer-Wert
if (limit<2): # Obergrenze < 2
    print('Es gibt keine Primzahl die kleiner ist als ', limit)
for i in range (2,limit+1): # für alle Zahlen von 2 bis Obergrenze
    prim = True # zunächst nehmen wir an, dass sie prim ist
    for k in range(2, int(i**(1/2))+1): #für alle Zahlen von 2 bis sqrt
        (i)
        if (i%k==0): # falls i durch k teilbar ist
            prim = False # ist sie nicht prim
            break # wir können aufhören zu testen
    if (prim == True): # falls i prim ist
        print(i) # ausgeben
```

(c)
```
####################################
# Programm rät die Zahl,
#  die sich der Benutzer denkt
```

```
4  ######################################
5
6  print('''Denken Sie sich eine ganze  Zahl zwischen 1 und 100.
7  Ich werde versuchen die Zahl in möglichst wenigen Schritten
8  zu erraten. Sie müssen mir helfen, indem Sie mir sagen,
9  ob die von mir geratene Zahl zu grœs oder zu klein ist.''')
10
11 korrekt = False # Kontrollvariable für while-Schleife
12 low = 1 # niedrigstmögliche Zahl
13 high = 100 # höchstmögliche Zahl
14 dazu = 0 #hilfsvariable
15 while not korrekt: # solange die Zahl nicht erraten wurde
16     if (high > (low + 1)): #falls es noch mindestens 3 mögliche Zahlen
           gibt
17         rate = ((high -low)//2) + dazu #rate die Zahl in der Mitte
18         a = input('ist die Zahl ' + str(rate) + '? (y/n) ')
19         if(a == 'n'): #geratenen Zahl ist falsch
20             b = input('ist die Zahl kleiner? (y/n) ')
21             if(b =='n'):
22                 low = rate + 1 #alle Zahlen <= geratene verwerfen
23                 dazu = rate
24             else: # geratene Zahl zu grœs
25                 high = rate - 1 # alle Zahlen >= geratene verwerfen
26                 dazu = low
27         else: # Die Zahl wurde erraten
28             korrekt = True
29     else: # es sind nur noch 2 mögliche Zahlen übrig
30         c = input('ist die Zahl ' + str(low) + '? (y/n) ')
31         if(c == 'n'): # diese war's nicht
32             print('Die Zahl muss ' + str(high) + ' sein')
33             korrekt = True
34         else:
35             korrekt = True
```

Viel Erfolg!