Python – Übung 3

1 Ausnahmebehandlungen

1.1 Fehler identifizieren und abfangen

Gegeben seien folgende Variablen:

```
d = {"flower": "lily", "color": "turquoise", "fish": "clownfish"}
t = (2020, 3, 4)
p = "1001.90 CHF"
```

☞ Welche Fehler werden bei den untenstehenden Codezeilen geworfen? Fangen Sie diese Fehler mit einer geeigneten `try-except`-Struktur ab.

```
d["dish"]
tup[0]
t[3]
float(p)
c = t + p
t[0] = 2021
```

Gegeben sei der folgende Python Code:

```
my_list = [0, 1, 2, 3, "a", "b", "c"]
for value in my_list:
    r_value = 1/value
    print(f"Reziproker Wert von {value} ist {r_value}")
```

☞ Überlegen Sie sich, welche Fehler auftreten können und fangen Sie diese mit der entsprechenden `try-except`-Struktur ab.

1.2 Exception werfen

Jemand hat bereits folgende Funktion implementiert:

```
def student(name, id, visited_subjects):
    print(f"Name: {name}")
    print(f"ID: {id:d}")
    print(f"Visited:")
    for subject in visited_subjects:
        print(f" - {subject}")
```

☞ Erweitern Sie die obige Funktion, so dass entprechende Exceptions geworfen werden, falls die Funktion mit falschen Argumenten aufgerufen wird:

- Das Argument `name` soll ein nicht-leerer String sein, sonst wird ein `ValueError` mit einer entsprechenden Fehlermeldung geworfen.

- Das Argument `id` soll eine nicht-negative Ganzzahl sein, sonst wird ein `ValueError` mit einer entsprechenden Fehlermeldung geworfen.
- Das Argument `visited_subjects` soll vom Typ `list` oder `tuple` sein, sonst wird ein `TypeError` geworfen. In der Fehlermeldung soll der Typ des falschen Arguments genannt werden und auf die erwarteten Typen hingewiesen werden.

2 Stringformatierung

Gegeben sind folgende Variablen:

```
var1 = 256
var2 = 7.98765
var3 = 2800031
var4 = 0.0000000089
var5 = "NEWS"
length = 2.35
```

☞ Nutzen Sie die passenden Formatierungsregeln, um die im folgenden gezeigten Outputs zu erlangen. Verwenden Sie dafür passende f-Strings.

```
0x0100
7.9877
2,800,031.00
8.9e-09
+++NEWS+++
length = 2.350
```

3 Telefonbuch

Gegeben sind folgende Telefonbucheinträge:

```
"""Maria;Meier;Genf;022 548 95 26
Peter;Müller;Basel;061 498 56 18
Patrick;Huber;Chur;081 445 66 58
Anna;Schmid;St. Gallen;071 216 88 68
Markus;Keller;Winterthur;052 354 87 98
Sandra;Weber;Bern;031 985 46 48
Patrick;Huber;Chur;081 445 66 58
Monika;Schneider;Solothurn;032 468 98 05
Bruno;Fischer;Luzern;041 225 56 85
Silvia;Brunner;Zürich;044 123 65 47
Sandra;Weber;Bern;031 985 46 48
"""
```

☞ Schreiben Sie ein Python-Programm, welches aus diesem String eine Liste mit Zweiertupel erstellt, die jeweils aus dem Vornamen und der Telefonnummer bestehen, z.B.:

```
[("Maria", "022 548 95 26"),
 ("Peter", "061 498 56 18"),
 ("Patrick", "081 445 66 58"),
 ("Anna", "071 216 88 68"),
 ("Markus", "052 354 87 98"),
 ("Sandra", "031 985 46 48"),
 ("Patrick", "081 445 66 58"),
 ("Monika", "032 468 98 05"),
 ("Bruno", "041 225 56 85"),
 ("Silvia", "044 123 65 47"),
 ("Sandra", "031 985 46 48")]
```

☞ Erweitern Sie das Python-Programm, so dass die Liste keine Duplikate mehr enthält, aber die ursprüngliche Reihenfolge der Einträge beibehalten wird, z.B.:

```
[("Maria", "022 548 95 26"),
 ("Peter", "061 498 56 18"),
 ("Patrick", "081 445 66 58"),
 ("Anna", "071 216 88 68"),
 ("Markus", "052 354 87 98"),
 ("Sandra", "031 985 46 48"),
 ("Monika", "032 468 98 05"),
 ("Bruno", "041 225 56 85"),
 ("Silvia", "044 123 65 47")]
```

4 Buchstaben zählen

☞ Erweitern Sie das mitgelieferte Programm `character_count.py`, so dass die Häufigkeit der einzelnen Buchstaben im Text ermittelt wird. Die Gross- und Kleinschreibung soll dabei ignoriert werden. Das Resultat soll wie folgt formatiert ausgegeben werden:

```
A: 58
...
Z: 11
Ä: 2
Ö: 4
Ü: 8
```

5 Wörter zählen

☞ Erweitern Sie das mitgelieferte Programm `word_count.py`, so dass die Anzahl der Wörter im Text ermittelt und das Resultat wie folgt formatiert ausgegeben wird.

```
The text contains 179 words.
```

Hinweis: Das `string`-Modul beinhaltet nützliche Konstanten, z.B. `string.punctuation`.

6 String bereinigen

Das mitgelieferte Python-Programm `string_bereinigen.py` enthält ein Text, welcher unnötige Leerzeichen und Zeilenumbrüche beinhaltet.

☞ Erweitern Sie das Programm, so dass mehrere aufeinanderfolgende Leerzeichen und/oder Zeilenumbrüche im Text mit einfachen Leerzeichen ersetzt werden. Im Text sollen also keine Zeilenumbrüche mehr vorhanden sein und die Wörter sollen jeweils nur von einem einzelnen Leerzeichen getrennt werden.

Hinweis: Benutzen Sie dafür die eingebauten Stringmethoden.

7 Hex Dump

In Python bestehen `str`-Objekte aus einer Reihe von Unicode-Zeichen, d.h. ein String kann auch Sonderzeichen beinhalten. Die Zeichen werden im `str`-Objekt intern als binäre Bytes abgespeichert, wobei gewisse Unicode-Zeichen auch mehrere Bytes benötigen. Will man nun die Zeichenkette eines `str`-Objekts in ein Array von Bytes konvertieren, kann man dies mit der `.encode()`-Methode tun.

☞ Implementieren Sie die Funktion `hexdump(text, encoding="utf-8")`, welche den String `text` in den Datentyp Bytes umwandelt und als sog. Hex-Dump ausgibt. Benutzen Sie die .encode()-Methode des Stringobjekts für die Umwandlung.

Die Funktion gibt den zweistelligen, hexadezimalen Wert jedes Bytes aus, wobei folgende Formatierung beachtet werden muss:

- In jeder Zeile werden 16 Bytes aufgelistet. Die letzte Zeile beinhaltet den Rest und darf entsprechend kürzer sein.
- Jede Zeile beginnt mit dem (mindestens) vierstelligen, hexadezimalen Index des ersten Bytes in der Zeile gefolgt von einem Doppelpunkt.
- Die Hex-Werte werden mit einem Leerzeichen getrennt.
  Hinweis: Benutzen Sie dafür die `.join()`-Methode des Datentyps `str`.

Hier ein Beispiel:

```
>>> text = """Python ist eine universelle,
üblicherweise interpretierte,
höhere Programmiersprache.
"""
>>> hexdump(text)
0000: 50 79 74 68 6F 6E 20 69 73 74 20 65 69 6E 65 20
0010: 75 6E 69 76 65 72 73 65 6C 6C 65 2C 0A C3 BC 62
0020: 6C 69 63 68 65 72 77 65 69 73 65 20 69 6E 74 65
0030: 72 70 72 65 74 69 65 72 74 65 2C 0A 68 C3 B6 68
0040: 65 72 65 20 50 72 6F 67 72 61 6D 6D 69 65 72 73
0050: 70 72 61 63 68 65 2E 0A
```