3IB: Programmieren 3, Studienleistung 2

Aufgabe 1: (Vererbung)

Erzeugen Sie ein Beispiel, welches den folgenden Fehler produziert: *TypeError: Cannot create a consistent method resolution order (MRO) for bases . . ..*

Geben Sie den dazugehörigen Quellcode an.

Die Abgabe der Aufgabe 1 erfolgt als .py mit dem Namen s2_a1.py

Aufgabe 2: (Regular Expressions)

1. Gegeben ist der folgende Satz: ”If the the problem is textual, use the the re module”. Verwenden Sie reguläre Ausdrücke um den offensichtlichen Fehler des doppelten ”the ” zu beheben. Geben Sie den richtigen Satz aus.

Die Abgabe der Aufgabe 2 erfolgt als .py mit dem Namen s2_a2.py

Aufgabe 3: (Regular Expressions und JSON)

1. Gegeben ist die Datei Personen.txt mit dem folgenden Aufbau: *Name, Adresse, Geburtsdatum, Rufnummer*. Die Einträge der Namen weisen einen unterschiedlichen Aufbau auf. Vereinheitlichen Sie die Einträge mithilfe von regulären Ausdrücken, so dass

   a) der Namen dem Aufbau: Titel Vorname ggfs. Zweitname und Nachname entspricht. Die Anrede entfällt aufgrund genderneutraler Anrede.
   b) die Adresse in Straße und Hausnummer, sowie PLZ und Wohnort aufgeteilt ist.
   c) das Geburtsdatum der deutschen Notation dd.mm.yyyy entspricht.

   Speichern Sie die Personen.txt unter dem Namen PersonenNeu.json als JSON Datei nach folgendem Format ab:

```
[{
'Index': 499,
'Titel': ['Ing.'],
'Vorname': ['Birgid'],
'Zweitname': [None],
'Nachname': ['Oestrovsky'],
'Geburtsdatum': ['03.08.2021'],
'Straße': ['Jasmina-Schacht-Weg '],
'Hausnummer': ['969'],
'PLZ': ['14667'],
'Wohnort': ['Sigmaringen']
},...
]
```

Die Abgabe der Aufgabe 3 erfolgt als .py mit dem Namen s2_a3.py. Die Abgabe der .json Datei ist nicht erforderlich, da diese durch den Quellcode erzeugt werden können muss.

Aufgabe 4: (Regular Expressions, Exceptions und Unit)

In den Vereinigten Staaten von Amerika und Kanada bestehen die Telefonnummern aus zehn Ziffern, die in der Regel in eine dreistellige Ortsvorwahl, eine dreistellige Amtskennziffer und eine vierstellige Vorwahl aufgeteilt sind. Den Telefonnummern kann die Landesvorwahl +1 vorangestellt werden, muss aber nicht. In der Praxis gibt es jedoch viele Möglichkeiten, eine Rufnummer zu formatieren, z. B. (NNN) NNN-NNNN, NNN-NNN-NNNN, NNN NNN-NNNN, NNN.NNN.NNNN, und NNN NNN NNNN, um nur einige zu nennen. Sofern die Landesvorwahl nicht verwendet wird, darf kein + vorhanden sein.

In dieser Übung besteht Ihre Aufgabe darin, einen Telefonnummern-Normalisierer zu erstellen, der jedes der Formate annimmt und eine normalisierte Telefonnummer 1-NNN-NNN-NNNN zurückgibt. Die folgenden sind alle möglichen und somit gültige Telefonnummern:

- +1 223-456-7890,
- 1-223-456-7890,
- +1 223 456-7890,
- (223) 456-7890,
- 1 223 456 7890,
- 223.456.7890.

Schreiben Sie ein Python-Programm, dass mittels regulärer Ausdrücke die Eingabe von Telefonnummern auf Gültigkeit prüft und die Telefonnummer ggfs. auf das Format 1-NNN-NNN-NNNN normalisiert.

a) Eine Telefonnummer ist gültig, wenn die erste Ziffer der Vorwahl und der Ortsvorwahl den Wert 2-9 beinhaltet. Weiterhin kann die zweite Ziffer einer Vorwahl nicht 9 sein. Verwenden Sie diese Informationen, um die Eingabe zu überprüfen, und geben Sie eine ValueError Exception mit der Meldung „Ungültige Telefonnummer“ zurück, wenn die Nummer ungültig ist.

b) Schreiben Sie für Ihren Telefonnummern-Normalisierer einen Unit-Test mit folgenden Testdaten:

```
test_numbers = ["+1 223-456-7890",
"1-223-456-7890",
"+1 223 456-7890",
"(223) 456-7890",
"1 223 456 7890",
"223.456.7890",
"1-989-111-2222"]
```

Verwenden Sie mindestens ein `asserEqual` und ein `assertRaises`.

Die Abgabe der Aufgabe 4a erfolgt als .py mit dem Namen s2_a4_a.py.
Die Abgabe der Aufgabe 4b erfolgt als .py mit dem Namen s2_a4_b.py.

Abgabehinweise:

☐ Erzeugen Sie in dem Git der Hochschule Mannheim (Gitty: `https://gitty.informatik.hs-mannheim.de`) ein Git-Repository für die Studienleistung 2 (z.B. /pr3/skriptspr/team12/s2).

☐ Geben Sie bei der Abgabemöglichkeit *Studienleistung 2* in dem Moodle Kurs Programmieren 3 - Wintersemester 22/23 in dem Bereich Skriptsprachen sowohl den Git-Link, als auch die Vornamen, Namen und Matrikelnummern der drei Gruppenmitglieder sowie die Teamnummer an und laden Sie genau 5 Dateien gepackt als `.zip` in Moodle hoch.

☐ Erzeugen Sie nur eine Abgabe für die drei Teammitglieder.

☐ Laden Sie genau 5 Dateien mit der folgenden Bezeichnung in Ihr Git-Repository:

1. s2_a1.py
2. s2_a2.py
3. s2_a3.py
4. s2_a4_a.py
5. s2_a4_b.py

☐ Abgabe erfolgt bis 20.01.2023, 23:59:59 MESZ.