Übungszettel 6a - Python OOP

Falls du heute lieber noch Aufgaben von den vorherigen Übungszetteln bearbeiten möchtest, darfst du das natürlich auch gerne tun.

Aufgabe 1: Objekte erzeugen

Hinweis: Die Datei kann auf der Vorkurshomepage heruntergeladen werden
Gegeben ist folgende Python-Klasse:

```
class Person:

    # Attribute (Eigenschaften)
    name="noch kein Name"
    age=0

    # Methoden (um Attribute ggf. zu aendern)
    def setName(self,x):
        self.name = x

    def setAge(self,x):
        self.age = x

    def haveBirthday(self):
        self.age = self.age + 1

    def talk(self):
        print("Ich heisse", self.name, "und ich bin", self.age, "Jahre alt")
```

(a) Speichere zuerst den Code in einer Datei namens `Person.py` und führe diese in der interaktiven python-Shell (z. B in IDLE) aus. Du kannst den Code abtippen, oder von der Vorkurshomepage herunterladen.

(b) Führe unmittelbar nach Teil (a) folgendes in der python-Shell aus:

1. Erzeuge ein Objekt der Klasse Person. Speichere dabei das Objekt in einer Variablen *p*.
2. Setze den Namen des Objekts auf *Alice*.
3. Setze das Alter des Objekts auf 22.
4. Lasse die Person sprechen. Was ist die Ausgabe?
5. Lasse die Person ihren Geburtstag feiern. Überprüfe mit der Methode *talk()*, ob das Alter um eins erhöht wurde.

(c) Ändere die Datei `Person.py` wie folgt:

1. Erweitere die Klasse um ein Attribut *eyeColor*.
2. Schreibe eine Methode *setEyeColor()*, mit der du die Augenfarbe setzen kannst.
3. Passe die Methode *talk()* so an, dass die Person auch ihre Augenfarbe nennt.
4. Speichere die Datei und öffne sie wieder.

(d) Erzeuge nun unmittelbar nach (c) eine Person namens Bob. Bob soll 20 Jahre alt sein und blaue Augen haben. Lasse Bob sprechen.

Aufgabe 2: Entwurf einer Klasse

In dieser Aufgabe sollst du nun eine Klasse selbst schreiben.

(a) Schreibe eine Klasse namens `Bestellung` mit:

1. Attributen: `speise, getraenk`
2. Methoden: `setSpeise(), setGetraenk(), bestellungVorlesen()`

Speichere die Klasse in einer Datei namens `Bestellung.py` ab und führe diese in der interaktiven python-Shell aus.

(b) Führe unmittelbar nach Teil (a) folgendes in der python-Shell aus.

1. Erzeuge ein Objekt der Klasse `Bestellung`. Speichere das Objekt in einer Variablen `B`.
2. Du möchtest Pizza und Cola bestellen. Verwende die entsprechenden set-Methoden.
3. Überprüfe mit der Methode `bestellungVorlesen()`, ob deine Bestellung richtig war.

(c) Um diese Bestellung zu dokumentieren (und um diese morgen wieder auszuführen), schreibe ein Programm, das unter Verwendung von `Bestellung.py` die Bestellung durchführt.
Hinweis: Benutze `import`.

Aufgabe 3: Konstruktor verwenden

Meistens möchten wir beim Erzeugen eines Objekts auch gleich die Werte einiger Attribute festlegen. Dies können wir mit dem Konstruktor `__init__()` erreichen. Beachte die *zwei* Unterstriche vor und nach init.

```
class Box:

    # Konstruktor (zum Initialisieren der Attribute)
    def __init__(self,x,y,z,f):
        self.laenge = x
        self.breite = y
        self.hoehe = z
        self.farbe = f
        print("Du hast gerade eine Box erzeugt.")
```

(a) Speichere die Klasse in einer Datei namens `Box.py` ab und führe diese in der interaktiven python-Shell aus.

(b) Erzeuge unmittelbar nach (a) ein Objekt der Klasse Box. Das Objekt soll die Länge 3, Breite 2 und Höhe 5 besitzen und rot sein. Verwende die python-Shell.

(c) Erweitere die Klasse um eine Methode *bestimmeVolumen()*. Diese Methode soll das Volumen ausgeben.

(d) Erweitere die Klasse um eine Methode *bestimmeFarbe()*. Diese Methode soll die Farbe der Box ausgeben. Speichere die Datei und führe sie aus.

(e) Erzeuge nun mit der erweiterten Klasse die Box aus (b). Lasse dann das Volumen und die Farbe deiner Box im Terminal ausgeben.

Viel Erfolg!