PROG1-Python Dauer: 210 Minuten Abgabe am Ecampus 09.11.2020

Name: *Matrikelnr.:*

Alle Python3-Code Dateien müssen am Ecampus als ZIP hochgeladen werden. ***Letzte Abgabe 12:40 Uhr!***

1. Qualitätskontrolle von Produkten *[35 Punkte]*

Sie können für dieses Beispiel das beigelegte Grundgerüst `product_quality.py` verwenden!

5P • Erstellen Sie die Klasse `Produkt` mit den folgenden Attributen, wobei die Wert-Zuweisung per Instanzaufruf (bei Objekterstellung) übergeben oder direkt im Konstruktor durchgeführt werden kann:

ID	1-50 (eindeutige ID - darf nur einmal vorkommen)
Quality	20-100% (Wert wird via Zufallsfunktion zugewiesen)
Vulnerable	Yes|No oder 1|0 (Wert wird via Zufallsfunktion zugewiesen)
category	["IT-Equipment", "Mobile", "Flight-Control", "Industrial-Part"] Kategorie, welche zufällig gewählt wird; Tipp: random.choice()[1]
cost	1.000,00€ – 3.000,00€ (zufälliger float Wert); Tipp: random.uniform(start, end)

5P • Erzeugen Sie 15 Produkte und speichern Sie diese in einer Klassenliste ab

8P ○ Entfernen Sie die Produkte welche qualitativ minderwertig sind (**unter 65% Qualität**) und vulnerable sind (**vulnerable = yes|1|True**)

5P • Sortieren Sie die Liste mit den übrig gebliebenen Produkten und geben Sie davon die 5 Produkte mit der niedrigsten Qualität aus.

○ Es kann passieren, dass weniger als 5 Produkte übrig bleiben und somit z.B. nur 2 Produkte ausgegeben werden. Falls jedoch kein Produkt mehr über ist – überprüfen Sie dies und geben entsprechend eine Meldung aus!

10P • Geben Sie anschließend alle übrig gebliebenen Produkte aus, welche den Kategorien IT-Equipment oder Industrial-Part zugeteilt worden sind, aber mindestens 2.200€ wert sind.

○ Es kann passieren, dass kein Produkt den Kriterien entspricht, geben Sie dafür eine entsprechende Meldung aus – z.B. “Kein Produkt mit den Kriterien vorhanden“.

2P • Abschließend geben Sie die Anzahl der verbliebenen Produkte aus.

Möglicher Ablauf:

1. Klasse erstellen; Produkte erzeugen und in eine Liste speichern
2. Die Produktliste durchgehen und alle Produkte welche den oben genannten Kriterien entsprechen aus der Liste entfernen
3. Übrig gebliebene Produktliste nach der Qualität aufsteigend (0-100%) sortieren und jene 5 mit niedrigster Qualität ausgeben
4. Produktliste durchgehen und alle Produkte welche in der Kategorie IT-Equipment oder Industrial-Part sind sowie >= 2200€ wert sind ausgeben
5. Größe der aktuellen Produktliste ausgeben

[1] https://docs.python.org/3/library/random.html#random.choice

Möglicher Output:

```
removing product with ID 2
removing product with ID 13
removing product with ID 11
removing product with ID 14
removing product with ID 4
removing product with ID 6


Produkte mit niedrigster Qualität:
ID: 12 Quality: 72.08%   Vulernable: False  Category: IT-Equipment Cost: 1227.14€
ID: 8 Quality: 72.33%   Vulernable: False  Category: IT-Equipment Cost: 2616.07€
ID: 7 Quality: 90.83%   Vulernable: False  Category: Industrial-Part  Cost: 2406.71€


Produkte in Kategorie IT-Equipment oder Industrial-Part und mit Kosten von >= 2000€
ID: 8 Quality: 72.33%   Vulernable: False  Category: IT-Equipment Cost: 2616.07€
ID: 7 Quality: 90.83%   Vulernable: False  Category: Industrial-Part  Cost: 2406.71€

Verbleibende Produkte:
Amount: 9
```

2. Kontaktdaten verwalten *[35 Punkte]*

Sie können für dieses Beispiel das beigelegte Grundgerüst `contacts_client.py` verwenden!

Gegeben ist eine Datei mit Kontaktdaten (contacts.json[2]). Alle Personen sollen eingelesen werden und pro Person ein Klassenobjekt mit den dementsprechenden Attributen erzeugt werden, siehe Grundgerüst.

5P • Normalisieren Sie alle Attribute (String in Klein/Großschreibung umwandeln). Berechnen Sie zusätzlich das Alter der einzelnen Kontakte (nur nach Jahr). Speichern Sie alle Kontakte in eine Liste.

5P • Löschen Sie alle Kontakte aus der Liste welche noch minderjährig sind (**Alter < 18 Jahre**)

10P • Ordnen Sie die übrig gebliebenen Kontaktliste nach **Alter** in absteigender Reihenfolge. Geben Sie die Anzahl der übrigen Kontakte aus.

15P • Abschließend sollen alle Kontakte die in Österreich wohnen und einen VIP-Status haben (**vip=true**) via `Sockets` an den beiliegenden Server geschickt werden.

a. Schreiben Sie alle Kontakte mit den angeführten Kriterien in eine neue Liste (`austriaVIPList`)
b. Übertragen Sie das Listenobjekt zum Server (Server vorher starten)
c. Kontrollieren Sie ob der Server das Objekt empfängt und richtig ausgibt

Kleiner Auszug aus der Kontaktdatei:

```
{
  "contacts": [
    {
      "id": 1,
      "fname": "Gary",
      "lname": "Ortiz",
      "country": "Austria",
      "birthday": "2000-05-16",
      "vip": false
    },
    {
      "id": 2,
      "fname": "Albert",
      "lname": "Williamson",
      "country": "China",
      "birthday": "2001-03-11",
      "vip": true
    },
     …
  ]
}
```

[2] Am Ecampus unter Prüfungsunterlagen zu finden

Möglicher Ablauf:

1. Grundgerüst verwenden
2. Datei öffnen und Json-Daten einlesen und speichern
3. Klassenattribute zuweisen; Produkte erzeugen und in eine Liste speichern
4. Die Kontaktliste durchgehen und alle Kontakte welche den oben genannten Kriterien entsprechen aus der Liste entfernen (Alter < 18)
5. Übrig gebliebene Produktliste nach Alter absteigend sortieren und Anzahl der übrigen Kontakte ausgeben
6. Kontaktliste durchgehen und alle Kontakte welche als Land Austria haben und einen VIP-Status (True) haben, in austriaVIPList schreiben
7. Socket erstellen und als Objektname s speichern; vordefinierten send-Aufruf auskommentieren
8. Server.py starten und danach contacts_client.py starten und Ergebnis mit möglichen Output kontrollieren

Möglicher Output:

contacts_client.py

```
Removing person with ID 8 - age: 5
Removing person with ID 10 - age: 17
Removing person with ID 23 - age: 12

Anzahl ohne minderjährige Kontakte 22

Österreicher mit VIP-Status:
20: Juanos Evans    1976-07-09          43 Jahre  VIP: True  aus Austria
7: Michael Rice     1980-12-06          39 Jahre  VIP: True  aus Austria
12: Lori Edwards    1989-02-05          31 Jahre  VIP: True  aus Austria
14: Craig Göschl    1992-01-20          28 Jahre  VIP: True  aus Austria

Verbindung zu Server aufbauen ..
Liste an Server senden
sent!
```

Server.py

```
Waiting for Connection ...
Client verbunden

20: Juanos Evans    1976-07-09          43 Jahre  VIP: True  aus Austria
7: Michael Rice     1980-12-06          39 Jahre  VIP: True  aus Austria
12: Lori Edwards    1989-02-05          31 Jahre  VIP: True  aus Austria
14: Craig Göschl    1992-01-20          28 Jahre  VIP: True  aus Austria

well done
```

3. Kontaktdaten verwalten *[30 Punkte]*

Nehmen Sie ihr Lotto Beispiel aus U2Bsp7 oder das Beispiel LV_Tag10 (06.11.) > Übungen > U2Bsp7.py oder U2Bsp7_extended.py als Vorlage/Grundgerüst.

Programmieren Sie das Beispiel wie folgt um:

15P
- Eine Ziehung kostet 1.5€ ihr Guthaben beträgt 3500€
- Gewinne:
  - 3er 5.5€
  - 4er: 55€
  - 5er: 2500€
  - 6er: 1000000 € (1 Million)

5P Lassen Sie ihr Programm solange durchlaufen bis Sie pleite sind (kein Geld mehr haben). Geben Sie abschließend alle Gewinne als Statistik und die Anzahl der durchgeführten Ziehungen aus.

10P Es soll pro Ziehung immer ihr aktuelles Guthaben ausgerechnet werden (Bitte keine Ausgabe pro Ziehung), solange bis kein Guthaben mehr vorhanden ist, d.h. Überprüfen ob Ziehung noch möglich ist (Guthaben >= 1.5€)

Möglicher Output:

```
0er 2046
1er 1204
2er 512
3er 4
4er 0
5er 0
6er 0
Pleite nach 3766 Ziehungen.
```