PROTRANET

Python - Seminar

Übungsaufgaben

Dr.sc.nat. Michael J.M. Wagner, Protranet*

Revision 316

*michael.wagner@protranet.de

Inhaltsverzeichnis

1 Einführung

2 Verfügbare IDEs

Aufgabe:

- Prüfen Sie die installierten Python-Versionen, indem Sie in der Shell folgende Kommandos absetzen:

```
$ python -V
$ python3 -V
```

- Starten Sie den Python-Interpreter und führen Sie „Übung u_grundrechenarten"[1] aus.

Aufgabe:

- Legen Sie in der Entwicklungsumgebung ein neues Projekt an und fügen Sie eine Datei `HalloWelt.py` hinzu.
- Geben Sie in dieser Datei den Code für die Ausgabe ein.
- Führen Sie die Datei auf die drei oben beschriebenen Weisen aus.

  Stellen Sie in Eclipse `/usr/bin/python3` als Standardinterpreter ein.

3 Typen

3.1 Zeichenketten[2]

Aufgabe: Schreiben Sie im Modul `Medienverwaltung.py` eine Funktion `isSignatureInFile(signatur)`, die

- die Datei `medien.csv` Zeile für Zeile ausliest,
- jede Zeile prüft, ob sie mit der übergebenen Signatur beginnt,
- als Ergebnis einen bool'schen Wert zurückgibt: `True`, falls die Signatur bereits in der Datei vorhanden ist, sonst `False`.
- Falls die Eingabedatei nicht vorhanden ist, soll dies auch zu „Signatur nicht vorhanden" (`False`) führen.
- Überlegen Sie sich für das Hauptprogramm eine Benutzerführung, mit der Sie zwischen den angebotenen Funktionen wählen können.

[1] Theis: S. 25f.
[2] Theis: Kap. 4.2

3.2 Listen

Aufgabe:

- Ergänzen Sie das Modul `Medienverwaltung` um die Funktion `getMediaList()`, die eine Liste von Medien zurückgibt:
  - Legen Sie eine leere Liste an.
  - Lesen Sie wie in `isSignatureInFile()` die Datei Zeile für Zeile.
  - Zelegen Sie jede Zeile mit `split()`.
  - Erzeugen Sie mit `createMedium()` eine Instanz.
  - Fügen Sie die Instanz der Liste hinzu.
  - Nach dem Einlesen der Datei geben Sie die Liste zurück.
- Erweitern Sie das Hauptptogramm um den Menupunkt *Alle Medien anzeigen*. Darin soll
  - `getMediaList()` aufgerufen werden,
  - über die Liste iteriert und die Medien ausgegeben werden.

3.3 Verzeichnisse und Mengen

Aufgabe:

Ergänzen Sie die Bibliotheksanwendung. Um die Datei nicht jedes mal auf's Neue lesen zu müssen, sollen die Daten in einem Verzeichnis abgelegt werden. Dazu bekommt die Medienverwaltung jetzt interne Daten, wird vom Funktionsmodul zur Klasse.

- Wandeln Sie die Medienverwaltung in eine Klasse um:
  - Schreiben Sie eine Methode `load()`, die die Datei einliest und ein internes Verzeichnis füllt. Als Schlüssel soll dabei die Signatur dienen, als Wert eine Buch- oder CD-Instanz. Rufen Sie `getMediaList()`, iterieren Sie über die Liste und füllen Sie das Verzeichnis.
  - Der Konstruktor der Klasse ruft die `load()`-Methode auf.
  - Ergänzen Sie die Medienverwaltung im die Funktion `getMedium(signatur)`, die im Verzeichnis nach dem entsprechenden Medium schaut. Falls eines gefunden wird, wird es zurückgegeben. Falls nicht, wird `null` zurückgegeben.
  - `addMedium()` muss nun nach erfolgreichem Ändern der Datei `load()` aufrufen, damit der interne Datenbestand aktuell bleibt.
- Passen Sie das Hauptprogramm entsprechend an:
  - Beim Start des Hauptprogramms soll eine Instanz von `Medienverwaltung` angelegt werden.

- `addMedium()` und `getMediaList()` sind nun Methoden dieser Instanz.
- Ergänzen Sie das Hauptprogramm um den Menupunkt *Ausleihe*.
- Wird *Ausleihe* gewählt, wird nach einer Signatur gefragt und über Funktion `getMedium()` das Medium aus dem Datenbestand geholt. Falls es gefunden wird, wird es ausgegeben, falls nicht, kommt eine Fehlermeldung.

4 Operatoren

5 Elementare Anweisungen

Variablen

Aufgabe:

Mit den nächsten Übungen soll Schritt für Schritt eine Bücherverwaltung für eine Bücherei erstellt werden. Eine Bücherei arbeitet mit *Medien* (Bücher, Zeitschriften, CD, ...). Ein Medium sei durch folgende sechs Werte charaterisiert: Signatur, Autor, Titel, Typ, Seitenzahl, Spieldauer Führen Sie folgende Schritte aus:

- Legen Sie ein neues Projekt *Bucherei* an, darin das Programm `Bucherei.py`.
- Belegen Sie über `input`-Anweisungen sechs Variablen für die Eigenschaften eines Mediums.
- Geben Sie die sechs Variablen schön formatiert aus.

Verzweigungen

Aufgabe:

Ergänzen Sie die Büchereianwendung mit Prüfungen auf die eingegebenen Werte:

- Signatur, Autor, Titel dürfen nicht leer sein.
- Typ darf nur die Werte `B` (Buch) oder `C` (CD) annehmen.
- Signatur und Spieldauer müssen ganzzahlig sein. Das prüft das System, wenn wir es in den Datentypen `int` umwandeln.
- Bei einer Fehleingabe brechen Sie das Programm ab.

Schleifen

Aufgabe:

Ergänzen Sie die Büchereianwendung:

- Wiederholen Sie Eingabe so lange, bis alles richtig ist. Definieren Sie dazu eine Variable `allesok`, die Sie zu Beginn der Schleife auf `True` setzen.
- Falls eine Prüfung fehl schägt, wird sie auf `False` gesetzt.
- Die Schleife wird verlassen, wenn die Variable nach allen Prüfungen immer noch auf `True` steht.

6 Grundlagen von Funktionen

Aufgabe:

Ergänzen Sie die Büchereianwendung:

- Legen Sie ein Modul `InOut.py` an.
- Packen Sie darin Eingabe und Prüfung in eine Funktion `eingabeMedium()`, die die sechs Medienparameter zurückgibt.
- Packen Sie darin die Ausgabe in eine Funktion `ausgabeMedium()`, die die sechs Medienparameter übernimmt und ausgibt.

7 Grundlagen von Modulen und Packages

Datei-Operationen

Das Öffnen einer Datei erfolgt mit:

```
with open(<Dateiname>[, <modus>]) as datei_objekt:
```

Anmerkung: Allokiert ein Objekt Resourcen und implementiert das Objekt eine `__exit__`-Methode zur Freigabe der Resourcen, kann durch Verwendung des `with` gesichert werden, dass mit Verlassen des Blocks auf jeden Fall die Resourcen freigegeben werden. Im Fall des `open` ist damit kein `close` mehr nötig.

`<modus>`: `r` (default), `w`, `a`[3]

[3] Theis: S. 259.

Für die zeilenweise Verarbeitung einer Datei empfiehlt sich folgende Formulierung:

```
with open("readfile.txt") as d:

   # Ausgabe und Addition aller Elemente
   sum = 0
   for line in d:
      print(line, end="")
      sum += float(line)

# Ausgabe der Summe
print("Summe:", sum)
```

Das Schreiben einer Zeile erfolgt mit:

```
d.write( string_var )
```

Alternativ kann auch die Funktion `print` verwendet werden, wenn der benannte Parameter `file` versorgt wird. `print` hat den Vorteil, dass

- mehere Parameter übergeben werden können,
- mit dem benannten Parameter `sep` bestimmt werden kann, welches Zeichen zwischen den übergebenen (normalen) Parametern eingefügt werden soll,
- am Zeilenende standardmäßig ein Zeilenumbruch angefügt wird.

```
with open("writefile.txt", "w") as d:
   print(file=d,"Zeile zum Schreiben")
```

Aufgabe:

Mit den nächsten Übungen soll Schritt für Schritt eine Bücherverwaltung für eine Bücherei erstellt werden. Führen Sie folgende Schritte aus:

- Legen Sie die Datei `Medienverwaltung.py` an.
- Schreiben Sie eine Funktion `addMedium`, die sechs Parameter nimmt:

  `addMedium(signatur, autor, titel, typ, seitenzahl, spieldauer)`
- `addMedium` soll die Daten an die Datei `medien.csv` kommasepariert anhängen.
- Rufen Sie die Prozedur aus dem Hauptprogramm (ggf. mit konstanten Werten) auf.

8 Klassen und Objekte

8.1 Klassen

Ergänzen Sie die Bibliotheksanwendung um Benutzer.

- Erstellen Sie eine Klasse `Benutzer` mit einem Konstruktor, der vier Parameter nimmt: Vorname, Nachname, Straße, Ort. In den objektorientierten Sprachen werden Klassen üblicherweise in gleichnamige Dateien gelegt. In Python wird in Python von dieser Regel abgewichen. Verwenden Sie `BenutzerModule.py` als Dateinamen.
- Ergänzen Sie das Hauptprogramm um eine Auswahl, ob ein Medium oder ein Benutzer angelegt werden soll.
- Ergänzen Sie `InOut.py` um folgende Funktionen:
  - `eingabeBenutzer()` liest die vier Parameter von der Eingabe, legt in `Benutzer`-Objekt an und gibt es zurück.
  - `ausgabeBenutzer()` nimmt ein `Benutzer`-Objekt als Parameter und gibt es aus.
- Ergänzen Sie das Hauptprogramm um eine Auswahl, ob ein Medium oder ein Benutzer angelegt werden soll und verwenden Sie die genannten Funktionen.

8.2 Vererbung

Aufgabe:

Ergänzen Sie die Bibliotheksanwendung:

- Legen Sie das Modul `MediumModul.py` an:
  - Leiten Sie von der Klasse `MediumBase` die Klassen `Buch` und `CD` ab.
  - Verteilen Sie die Medium-Attribute sinnvoll auf die Klassen.
  - Ergänzen Sie jede Klasse um eine Methode `__str__`, die den Datensatz ausgibt.
  - Schreiben Sie eine Klassenmethode `createMedium`, das die sechs Medienparameter übernimmt und je nach Typ ein `Buch` oder eine `CD` zurück gibt.
- Ändern Sie das Anlegen im Hauptprogramm so ab, dass nach erfolgreichem Hinzufügen eines Mediums über `createMedium` eine Instanz angelegt wird und diese ausgeben wird.
- `ausgabeMedium()` soll nun statt den sechs Parametern nur eine Mediuminstanz übernehmen.

9 Exceptions

Aufgabe:

Ergänzen Sie die Medienverwaltung:

- In der Funktion `addMedium` um den Aufruf von `isSignatureInFile`. Falls die Signatur schon vorhanden ist, geben Sie einen entsprechenden Integer-Fehlercode an das Hauptprogramm zurück. Definieren Sie dazu entsprechende Konstanten (was leider vom Python nicht wirklich unterstützt wird).
- Werten Sie im Hauptprogramm die Fehlercodes aus.
- Prüfen Sie in `isSignatureInFile`, ob auch jeder gelesene Datensatz aus sechs Bestandteilen besteht (mit `split` in eine Liste zerlegen und mit `len` die Anzahl der Elemente bestimmen). Falls ein ungültiger Datensatz gelesen wird, werfen Sie eine Exception.
- Fangen Sie die Exception im Hauptprogramm.

10 Python-Bibliotheken

11 Datenbankanbindung

Aufgabe:

Ergänzen Sie die Büchereianwendung:

- Legen Sie ein Modul `Benutzerverwaltung.py` an, das
  - im Konstruktor prüft, ob die DB vorhanden ist, wenn nein, eine Tabelle `Benutzer` anlegt. Diese Tabelle habe zusätzlich zu den vier Datenfeldern ein Feld `id MEDIUMINT NOT NULL AUTO_INCREMENT`,
  - die Verbindung zur Datenbank in einer Membervariable speichert,
  - eine Funktion `addUser` implementiert, das ein `Benutzer`-Objekt übernimmt und der Datenbank hinzufügt.
  - eine Funktion `getUser(id)`, das den entsprechenden Benutzer in der Datenbank sucht, ein `Benutzer`-Objekt anlegt und dieses zurückgibt. Wird keine Benutzer gefunden, wird `null` zurückgegeben.
- Ergänzen Sie die entsprechende Funktion im Hauptprogramm.
- Ergänzen Sie die Funktion *Ausleihe*:
  - Falls das Medium gefunden wurde, soll eine Benutzernummer abgefragt werden.
  - Mit der Benutzernummer wird über `getUser()` der Benutzer geladen.

– Falls auch der Benutzer erfolgreich geladen werden konnte, wird ein entsprechender Ausgabetext gemacht, falls nicht eine Fehlermeldung.

12 GUI Programmierung

Aufgabe:

Erstellen Sie eine GUI, die das Hauptprogramm `Bucherei.py` ersetzt. Beginnen Sie mit dem Geschäftsvorfall *Ausleihe*.

Achten Sie darauf, dass die Verwalterklassen nur einmal instanziert wird.

13 XML

14 Quellen

Theis, Thomas Einstieg in Python. Ideal für Programmiereinsteiger, 5. Auflage, 2017