LU20.A01 - Bibliotheksverwaltung

Realisieren Sie die verschiedenen Programmteile der Applikation „library“ gemäss den Beschreibungen.

Allgemeine Infos

Vorlage

Im Vorlage-Repository sind alle Funktionen als rudimentäre Stubs realisiert. Diese Stubs liefern, soweit nötig, einen statischen Returnwert. Die Logik der Methode „main()“ ist bereits vollständig umgesetzt.

Vorgehen

Wie immer beim Programmieren gilt:

1. Wählen Sie einen Programmteil aus, dessen Vorgänger korrekt funktioniert.
2. Codieren Sie den neuen Programmteil.
3. Testen Sie den neuen Programmteil,
    1. Falls die Tests nicht erfolgreich sind, gehen Sie zurück zu Schritt 2.
4. Führen Sie alle Tests mit `pytest -vv` aus.
5. Falls mehr Tests erfolgreich sind als zuvor: Commit & Push.

Nr	Programmteil	Vorgänger	Tests
1	Dataclass - Rental	-	rental_test.py > `test_attribute_assignments`
2	cost	1	rental_test.py > `test_cost_no_overdue` rental_test.py > `test_cost_with_overdue`
3	read_rental	1,7,8	library_test.py > `test_read_rental`
4	init_books	1	library_test.py > `test_init_books`
5	add_rental	1, 3, 4	library_test.py > `test_add_rental_empty_books` library_test.py > `test_add_rental_existing_books` library_test.py > `test_add_rental_invalid_book_name` library_test.py > `test_add_rental_multiple_rentals`
6	show_balance	1, 2	library_test.py > `test_show_balance`
7	read_int	-	library_test.py > `test_read_int_valid_input` library_test.py > `test_read_int_invalid_input`
8	read_date	-	library_test.py > `test_read_date_valid_input` library_test.py > `test_read_date_invalid_input`

Testing

Die einzelnen Programmteile (Funktionen, Attribute, Property) werden mittels Unit Tests überprüft. Die Testfunktionen finden Sie in `library_test.py` und `rental_test.py`. Für jeden erfolgreichen

Test erhalten Sie Punkte.

Beschreibungen

Der Code stellt ein terminalbasiertes Bibliotheksverwaltungssystem zur Verwaltung von Buchausleihen dar. Es wird mit einer Reihe von Büchern („Herr der Ringe“-Trilogie) initialisiert und ermöglicht es den Benutzern, Ausleihdatensätze hinzuzufügen, indem sie Buchnamen und Ausleihdetails (Ausleih- und Rückgabedaten, Anzahl der Ausleihtage) eingeben. Das System berechnet und zeigt die Mietkosten an, einschließlich der Strafen für überfällige Rückgaben. Die Benutzer interagieren über ein einfaches Textmenü, in dem sie Optionen zum Hinzufügen von Vermietungen, zum Anzeigen von Salden oder zum Beenden des Programms auswählen. Die Schnittstelle besteht hauptsächlich aus dem Lesen von Benutzereingaben und dem Ausdrucken von Informationen direkt im Terminal.

Fürs Testen:

Es gibt nur 3 Bücher, `L0TR 1`,`L0TR 2`,`L0TR 3`

Modul rental.py

Dataclass - Rental

- Datum der Ausleihe: Datum
- Rückgabedatum: Datum
- Anzahl inkludierte Miettage: Ganzzahl

@property cost()

Argumente	self
Rückgabewerte	Kosten des Ausleihens (Dezimalzahl)

Logik

Die Funktion berechnet die Kosten der Buchausleihe. Die Kosten für die Buchausleihe sind normalerweise CHF 4.50 für die gesamte Zeitdauer der `Anzahl Miettage`. Wird diese Frist jedoch überschritten wird eine Strafgebühr von CHF 3.35 pro angefangene 24 Stunden berechnet. Die effektive Anzahl Miettage wird berechnet aus `Rückgabedatum - Datum der Ausleihe`

Der Returnwert wird auf 2 Nachkommastellen gerundet: `return round(Ausleihgebür,2)`

Modul library.py

main()

Argumente	keine
Rückgabewerte	keine

Logik

Die Logik dieser Funktion ist bereits **vollständig umgesetzt**. Beachten Sie die Argumente und Returnwerte der jeweiligen Funktionen.

read_rental()

Argumente	keine
Rückgabewerte	`Rental`-Objekt

Logik

Der Benutzer wird aufgefordert, die Angaben zur Ausleihung einzugeben:

- Datum der Ausleihe
- Anzahl Miettage
- Rückgabedatum

Verwenden Sie für das Einlesen der Werte die passende Einleser-Funktion `read_int()` oder `read_date()`. Die Benutzereingaben werden in ein neues Rental-Objekt gespeichert und dieses wird von der Funktion zurückgegeben.

init_books()

Argumente	keine
Rückgabewerte	Dictionary mit Büchern

Logik

Die Funktion erstellt einen neuen Dictionary. Es werden drei Elemente in das Dictionary eingefügt:

Schlüssel	**Inhalt (Wert)**
L0TR 1	Leere Liste [ ]
L0TR 2	Leere Liste [ ]
L0TR 3	Leere Liste [ ]

add_rental()

Argumente	Dictionary mit Büchern und Ausleihungen
Rückgabewerte	keiner

Logik

Der Benutzer wird aufgefordert, einen Buchnamen einzugeben. Die Funktion wählt den Eintrag im Dictionary mit dem Buchnamen als Schlüssel. Gibt es das Buch nicht, wird nochmals nach dem Buchnamen gefragt. Anschliessend wird `read_rental()` aufgerufen um ein `Rental-Objekt` zu

erhalten.

Dieses `Rental-Objekt` wird beim gewählten Dictionary-Element am Ende der Liste mit den Buchungen eingefügt. Anschliessend wird der Benutzer gefragt, ob er die Erfassung beenden will. Solange der Benutzer **y** eingibt, wird der gesamte Ablauf wiederholt.

show_balance()

Argumente	Dictionary mit Büchern und Ausleihungen
Rückgabewerte	keiner

Logik

Die Funktion gibt für jedes Buch …

- … den Buchtitel aus.
- … für jede Buchung (Ausleihe) des Buchs …
    - … das Datum der Miete und die Kosten dieser Miete aus.
- … das Total aller Buchungen aus.

Beachten Sie die Darstellung gemäss diesem Beispiel:

```
Statement for LOTR 1
  - 05.01.2023: CHF 4.50
  - 06.03.2023: CHF 7.85
Total: CHF 12.35
Statement for LOTR 2
  - 06.01.2023: CHF 4.50
  - 23.02.2023: CHF 4.50
Total: CHF 9.00
Statement for LOTR 3
  - 06.01.2023: CHF 4.50
  - 23.02.2023: CHF 14.55
Total: CHF 19.05
```

read_int()

Argumente	prompt: String
	minimum: Integer (Optional)
	maximum: Integer (Optional)
Rückgabewerte	Benutzereingabe als Ganzzahl

Logik

Die Funktion fordert den Benutzer zur Eingabe einer Ganzzahl von `minimum` bis `maximum` auf. Falls die Eingabe keine gültige Ganzzahl ist, wird …

- … die Meldung „Please, enter a whole number!“ angezeigt.
- … der Benutzer zur erneuten Eingabe aufgefordert.

Falls die Eingabe zu klein / zu gross ist, wird …

- … die Meldung „Please, enter a number greater than or equal to `minimum`“ angezeigt.
- … die Meldung „Please, enter a number less than or equal to `maximum`“ angezeigt.
- … der Benutzer zur erneuten Eingabe aufgefordert.

read_date()

Argumente	prompt: String
Rückgabewerte	Benutzereingabe als Datum/Uhrzeit-Objekt

Logik

Die Funktion fordert den Benutzer zur Eingabe eines Zeitpunkts auf. Die Eingabe muss im Format `d(d).m(m).jjjj` erfolgen, z.B. 15.7.2023 Falls die Eingabe kein gültiger Zeitpunkt ist, wird …

- … die Meldung „Please enter a valid date.“ angezeigt.
- … der Benutzer zur erneuten Eingabe aufgefordert.

⇒ *GitHub Repo für externe Besucher*

GitHub Repository https://github.com/templates-python/m319-lu20-a01-library

*Lernende am BZZ **müssen** den Link zum GitHub Classroom Assignment verwenden*

m310-LU20

Kevin Maurizi, Marcel Suter

From:
https://wiki.bzz.ch/ - **BZZ - Modulwiki**

Permanent link:
https://wiki.bzz.ch/de/modul/m319/learningunits/lu20/aufgaben/bibliothek

Last update: **2025/06/23 07:45**