Übung 01: Bibliotheksverwaltung

Abgabetermin: 12. 3. 2015, 8:15

Name: _____ **Matrikelnummer**: _____

Informatik: ☐ G1 (Prähofer) ☐ G2 (Prähofer) ☐ G3 (Grimmer) ☐ G4 (Grimmer)

WIN: ☐ G1 (Khalil) ☐ G2 (Kusel) ☐ G3 (Kusel)

Aufgabe	Punkte	abzugeben schriftlich	abzugeben elektronisch	korr.	Punkte
Übung 1	24	Prosabeschreibung, Java-Programm, Ausgabe eines Spielablaufs	Java-Programm	☐	

Library

In dieser Übung soll eine Bibliotheksverwaltungssoftware (`Library`) implementiert werden, die es erlaubt, eingeschriebene *Personen*, *Bücher*, sowie *Ausleihen* zu verwalten. Bitte beachten Sie die folgenden Detailanforderungen.

Personen

Für die Personenverwaltung soll möglich sein, neue Personen einzuschreiben, sowie bestehende über eine ID, welche bei der Einschreibung der Person vom System vergeben wurde, abzufragen. Neu einzuschreibende Personen sind gekennzeichnet durch einen *Vornamen*, einen *Nachnamen* sowie eine *Adresse*. Bitte beachten Sie, dass alle drei Angaben verpflichtend sind, um eine neue Person erfolgreich einschreiben zu können.

Bücher

Für die Bücherverwaltung soll es möglich sein, neue Bücher in die Bibliothek aufzunehmen, bestehende über eine ID, welche bei der Aufnahme eines Buches vom System vergeben wurde, abzufragen, sowie alle Bücher, die einen bestimmten Text im Titel enthalten, abzufragen. Ein neu aufzunehmendes Buch ist gekennzeichnet durch einen *Titel*, sowie den zukünftigen *Standort* in der Bibliothek. Bitte beachten Sie, dass wiederum beide Angaben verpflichtend sind, um ein Buch erfolgreich in die Bibliothek aufnehmen zu können.

Ausleihen

Für die Ausleihenverwaltung soll es möglich sein, neue Ausleihen zu erstellen, bestehende zu verlängern bzw. bei Rückgabe eines Buches zu löschen. Eine Ausleihe ist gekennzeichnet durch die *ausleihende Person*, das *ausgeliehene Buch*, sowie das *Fälligkeitsdatum* der Rückgabe. Dieses soll mit 14 Tage nach Erstellung der Ausleihe initialisiert werden. Wird eine Ausleihe verlängert, so soll das alte Fälligkeitsdatum um 7 Tage verschoben werden.

Darüber hinaus soll es auch möglich sein, alle ausgeliehenen Bücher einer bestimmten Person abzufragen, sowie alle Ausleihen zu berechnen, die überfällig sind, d.h., deren Rückgabedaten vor dem aktuellen Datum liegen. Schließlich soll man auch noch abfragen können, ob ein bestimmtes Buch aktuell verfügbar ist.

Hinweise: Um mit Datums zu arbeiten, können Sie die Klassen `java.util.Date` und `java.text.DateFormat` wie folgt verwenden:

- `Date` speichert Datum und Zeit in der Form von Millisekunden.
- Mit Konstruktor `Date(long date)` erzeugt man ein Date-Objekt, wobei `date` Millisekunden sind. Mit Konstruktor `Date()` wird ein `Date`-Objekt zur aktuellen Zeit erzeugt.
- Mit `long getTime()` erhält von einem `Date`-Objekt die Zeit in Millisekunden.
- Durch Addition und Subtraktion der entsprechenden Millisekunden, kann man aus einem Datum ein Datum in der Zukunft oder Vergangenheit erzeugen.
- Mit Aufruf `DateFormat.getInstance().parse(dateString)` kann man aus einem String `dateString` ein `Date`-Objekt erzeugen.

- Mit Aufruf `DateFormat.getDateInstance().format(dateObj)` kann man aus einem `Date`-Objekt `dateObj` einen formatierten Ausgabestring erzeugen.

Vorgehensweise

Schritt 1) Öffentliche Schnittstelle

Gestalten Sie die öffentliche Schnittstelle (API) für die Bibliotheksverwaltung. Überlegen Sie genau, welche Klassen öffentlich sichtbar sind und die Signaturen der öffentlichen Methoden. Dokumentieren Sie die öffentliche Schnittstelle in einem eigenen Dokument. Dieses Dokument ist schriftlich abzugeben.

Schritt 2) Implementierung des Datenmodells + Schnittstelle

Implementieren Sie das Datenmodell für die Bibliotheksverwaltung. Personen, Bücher, wie auch Ausleihen sollen in einfach verketteten Listen verwaltet werden, d.h., verwenden Sie bitte keine von Java vorgegebenen Collections-Klassen. Wird eine Objektmenge zurückgegeben (z.B. bei der Abfrage aller überfälligen Ausleihen), so soll diese Menge als Array retourniert werden. Ziel dieser Aufgabe ist es nicht eine möglichst effiziente Implementierung zu erstellen – achten Sie vorrangig auf eine saubere Schnittstelle, d.h., achten Sie bei der Implementierung besonders auf die richtige Verwendung der Modifier `public`, `package`, `protected`, `private` und `final`.

Schritt 3) Benutzerschnittstelle

Die Klasse `Application` realisiert die textbasierte Benutzerschnittstelle und wird Ihnen zur Verfügung gestellt (siehe Vorgabe). Ergänzen Sie diese Klasse an den markierten Stellen, indem Sie entsprechende Aufrufe Ihrer Implementierung einfügen. Verwenden Sie die Klasse, um Ihr System interaktiv zu testen.

<u>Abzugeben sind:</u>

- Kurze Prosabeschreibung der öffentlichen Schnittstelle
- Java-Programm
- Repräsentativer Testlauf