Universität Augsburg
Institut für Informatik

Übung zur Vorlesung Informatik 2 SS 2018
Fakultät für Angewandte Informatik
Lehrprofessur für Informatik
Prof. Dr. Lorenz, Marius Brendle, Johannes Metzger 18.04.2018

Übungsblatt 2

Abgabe: 25.04.2018, 12:00 Uhr (Postkasten der Veranstaltung und E-Mail an Tutor)

- Dieses Übungsblatt muss im Team abgegeben werden (Einzelabgaben sind nicht erlaubt!).
- Bitte zur Angabe von Namen, Übungsgruppe und Teamnummer das **Deckblatt** verwenden!
- Die **Zeitangaben** geben zur Orientierung an, wie viel Zeit für eine Aufgabe später in der Klausur vorgesehen wäre; gehen Sie davon aus, dass Sie zum jetzigen Zeitpunkt wesentlich länger brauchen und die angegebene Zeit erst nach ausreichender Übung erreichen.

* leichte Aufgabe / ** mittelschwere Aufgabe / *** schwere Aufgabe

Aufgabe 5 * *(In der Programmklasse API-Klassen benutzen, 10 Minuten)*

In den folgenden Teilaufgaben sollen Sie jeweils eine Programmklasse implementieren. Benennen Sie die Programmklasse jeweils nach der Nummer der Aufgabe.
In allen Aufgaben ist mit Ausgabe die Ausgabe auf Kommandozeile (= Standardausgabe) gemeint.

a) (*, Klasse `String`, Klasse `Character`, alte Klausuraufgabe, 5 Minuten)
Schreiben Sie ein übersetzbares Java-Programm, das genau zwei Kommandozeilenparameter erwartet. Der erste Parameter muss `-o` entsprechen. Der zweite Parameter muss mit einer Ziffer beginnen. Falls die übergebenen Kommandozeilenparameter dieser Form entsprechen, soll das Programm `ok` ausgeben, sonst `nicht ok`.

b) (*, Klasse `StringBuilder`, 5 Minuten)
Schreiben Sie ein Java-Programm mit folgender Funktionalität unter ausschließlicher Benutzung von Ausgabeoperationen und von Operationen der Klasse `StringBuilder`:

- Zuerst soll ein `StringBuilder`-Objekt `s` mit leerem Inhalt erzeugt werden.
- Dann sollen nacheinander alle übergebenen Kommandozeilenparameter an den jeweils aktuellen Inhalt von `s` angehängt werden.
- Am Ende soll die Länge des Inhalts von `s` ausgegeben werden.

Aufgabe 6 ** *(In der Programmklasse API-Klassen benutzen, 12 Minuten)*

In den folgenden Teilaufgaben sollen Sie jeweils eine Programmklasse implementieren. Benennen Sie die Programmklasse jeweils nach der Nummer der Aufgabe.
In allen Aufgaben ist mit Ausgabe die Ausgabe auf Kommandozeile (= Standardausgabe) gemeint.
Achten Sie darauf, alle Ausnahmen, die bei Operationsaufrufen auftreten können, abzufangen. Benutzen Sie dabei im `catch`-Block jeweils die genau passende Ausnahmeklasse. Der `try`-Block soll genau die von einer evtl. auftretenden Ausnahme betroffenen Anweisungen enthalten. Die Fehlerbehandlung soll jeweils in der Ausgabe der Fehlermeldung der Ausnahme bestehen.

a) (**, Klasse `Double`, 5 Minuten)
Schreiben Sie ein Java-Programm mit folgender Funktionalität unter ausschließlicher Benutzung von Ausgabeoperationen und Operationen der Klasse `Double`:

- Es soll abbrechen, wenn nicht genau zwei Kommandozeilenparameter übergeben wurden.
- Es sollen beide Kommandozeilenparameter in `double`-Werte umgewandelt werden.
- Am Ende sollen die beiden `double`-Werte multipliziert und das Ergebnis der Multiplikation ausgegeben werden.

b) (**, Klasse `ArrayList`, Klasse `Math`, 7 Minuten)
Schreiben Sie ein Java-Programm mit folgender Funktionalität unter ausschließlicher Benutzung von Ausgabeoperationen und von Operationen der Klassen `ArrayList` und `Math`:

- Es soll ein `ArrayList<Double>`-Objekt `list` mit leerem Inhalt erzeugt werden.
- Dann soll eine zufällige ganze Zahl `n` zwischen `1` und `100000` (jeweils einschließlich) bestimmt werden.
- Danach sollen `n` zufällige `double`-Werte zwischen `0.0` (einschließlich) und `1.0` (ausschließlich) bestimmt, zu jedem dieser `double`-Werte ein `Double`-Objekt mit dem `double`-Wert als Inhalt erzeugt und diese `Double`-Objekte dem `ArrayList<Double>`-Objekt hinzugefügt werden.
- Am Ende soll das `ArrayList<Double>`-Objekt durchlaufen und das arithmetische Mittel aller gespeicherten `double`-Werte berechnet und ausgegeben werden.

Aufgabe 7 ** *(Einfache Programmklasse zu einer vorgegebenen Klasse implementieren, 16 Minuten)*

a) (**, 8 Minuten)
Betrachten Sie die als Download im Digicampus vorgegebene Klasse `Hobby` zur Verwaltung von Hobbys mit folgenden öffentlichen Verwaltungsoperationen und Konstruktoren:

- `public Hobby(String name) throws DataFormatException`: Konstruktor
- `public void setPriority(int priority) throws DataFormatException`: Setzen der Priorität eines Hobbys

(i) (3 Minuten)
Welche Attribute hat die Klasse `Hobby` und welche Datenstrukturinvarianten erfüllen diese Attribute?

(ii) (alte Klausuraufgabe, 5 Minuten)
Implementieren Sie unter Benutzung dieser Verwaltungsfunktionen und Konstruktoren eine Programmklasse mit folgender Funktionalität:

- Es soll ein neues Hobby mit dem Namen `Lesen` und ohne Angabe einer Priorität angelegt werden.
- Danach erhöhen Sie die Priorität dieses Hobbys um 1.
- Bei Fehlern soll das Programm mit passender ausführlicher Fehlermeldung über den Standardfehlerstrom abbrechen.

b) (**, 8 Minuten)
Betrachten Sie die als Download im Digicampus vorgegebene Klasse `Matrix` zur Verwaltung von Matrizen mit folgenden öffentlichen Verwaltungsoperationen und Konstruktoren:

- `public Matrix(int rowDimension, int columnDimension) throws DataFormatException`: Konstruktor
- `public void setValue(int row, int column, int value)`: Setzt den Wert `value` in Zeile `row` und Spalte `column`

(i) (3 Minuten)
Welche Attribute hat die Klasse `Matrix` und welche Datenstrukturinvarianten erfüllen diese Attribute?

(ii) (alte Klausuraufgabe, 5 Minuten)
Implementieren Sie unter Benutzung dieser Verwaltungsfunktionen und Konstruktoren eine Programmklasse mit folgender Funktionalität:

- Es soll eine Matrix mit zwei Zeilen und drei Spalten angelegt werden
- Falls das Anlegen erfolgreich war, sollen alle Matrixeinträge auf 1 gesetzt werden
- Bei Fehlern soll das Programm jeweils mit passender ausführlicher Fehlermeldung über den Standardfehlerstrom abbrechen.

Aufgabe 8 ** *(Einfache Klassen implementieren, 10 Minuten)*

Implementieren Sie nach folgenden Vorgaben **komplett** eine Klasse `Immobilie` zur Verwaltung von **Immobilien** für einen Anwendung zum Inserieren von Immobilien:

- Für eine Immobilie soll ein Preis, eine ganzzahlige Wohnfläche und die Höhe der Maklerprovision gespeichert werden. Die Werte für Preis und Wohnfläche müssen positiv sein. Die Maklerprovision soll für jede Immobilie eine Höhe von $3,45$ Prozent haben.
- Stellen Sie sicher, dass alle geforderten Datenstrukturinvarianten berücksichtigt werden.
- Implementieren Sie für alle Attribute, soweit sinnvoll, zugehörige `get-`, `set-` und `check-`Funktionen.
- Implementieren Sie einen Konstruktor, mit dem Preis und Wohnfläche gesetzt werden.
- Bei Übergabe ungültiger Werte soll eine `DataFormatException` mit geeigenter ausführlicher Fehlermeldung erzeugt und weitergereicht werden.

Ihnen steht eine Programmklasse zum Testen Ihrer Implementierung zum Download zur Verfügung.