Vorlesung „Java I“ SS 04 — 4. Übung (17. 5. 2004)

`http://www.coli.uni-sb.de/cl/courses/java-04/`

1 Geometrische Figuren (3 Punkte)

Definiere eine Klasse `GeoFigur`, die eine der beiden Rechteck-Klassen von letzter Woche verallgemeinert und mindestens die folgenden öffentlichen Methoden implementiert:

- `void draw()`: Zeichne die Figur als Zeichengrafik auf den Bildschirm.
- `int inhalt()`: Berechne den Flächeninhalt der Figur.
- `void test()`: Zeichne die Figur und gib ihren Flächeninhalt auf dem Bildschirm aus.

Die Methoden `draw()` und `inhalt()` sollen dabei zunächst Dummy-Implementierungen sein, da sie für allgemeine geometrische Formen ja so nicht definiert sind.

Implementiere dann eine der Rechteck-Klassen sowie eine Klasse `Dreieck` als abgeleitete Klassen von `GeoFigur` und schreibe ein Hauptprogramm, das zwei Variablen vom Typ `GeoFigur` mit einem Dreieck und einem Rechteck belegt und sie testet. Vermeide möglichst, Code zu duplizieren.

2 Härtetest (3 Punkte)

Lade dir das Programm `http://www.coli.uni-sb.de/cl/courses/java-04/uebungen/Ueb4.java` herunter. Was gibt dieses Programm aus? Erkläre im Detail, warum.

3 Listen (3+1 Punkte)

(a) Definiere (ohne Verwendung von Klassen aus `java.util`) eine Klase `IntList`, die Listen von `int`-Werten darstellt. Leite dazu von einer allgemeinen Klasse `IntList` zwei Klassen `IntNil` (leere Liste) und `IntCons` (Paar aus einem `int`-Wert und einer `IntList`) ab. Die Klasse `IntList` soll zumindest folgende öffentliche Methoden haben:

   - `int first()`: Gib das erste Element der Liste zurück.
   - `IntList rest()`: Gib den Rest der Liste (ohne das erste Element) zurück.
   - `int size()`: Wie viele Elemente hat diese Liste?

(b) Definiere eine statische Methode `IntList parse(String str)` in der Klasse `IntList`, die einen durch Kommata separierten String von `int`-Literalen als Argument nimmt und eine neue `IntList` zurückgibt, die die Zahlen in dieser Reihenfolge enthält.

(c) Schreibe ein Hauptprogramm, das auf der Kommandozeile eine Komma-separierte Liste von Zahlen entgegennimmt, daraus ein `IntList`-Objekt macht und dann die Liste mit einer Schleife ausgibt.

(d) Für den Bonuspunkt definiere zusätzlich eine Methode `IntList map(UnaryFunction op)` in der Klasse `IntList`. Diese Methode soll die `map`-Operation aus funktionalen Programmiersprachen implementieren, d.h. jedes Element der Liste wird in eine einstellige Funktion (z.B. $f(x) = 2 \cdot x$) eingesetzt und die Liste der Ergebnisse zurückgegeben. Die Methode gibt eine neu erzeugte Liste zurück und ändert die Original-Liste nicht.

Schreibe dazu eine Klasse `UnaryFunction`, die eine öffentliche Methode `int compute(int arg)` hat, die die Funktion auf dem Wert `arg` ausrechnet. Implementiere dann eine Klasse `MultMit`, deren `compute`-Methode ihr Argument mit einer konstanten Zahl multipliziert, die im Konstruktor übergeben wird. Schreibe damit ein Hauptprogramm, das aus einer Liste die Liste der doppelten Werte berechnet.

4 Mitarbeiter-Liste (3+1 Punkte)

Für ein Programm zur Personalverwaltung soll eine Klasse `Person` erstellt werden, die Angestellte einer Universität darstellt. Es gibt zwei Sorten von Personen: Professoren und wissenschaftliche Mitarbeiter. Jede Person hat einen Namen, eine Telefonnummer und eine Personalnummer; Professoren haben zusätzlich eine Sekretärin, Mitarbeiter sind einem Professor zugeordnet.

Schreibe entsprechende Klassen, die die beiden Personaltypen darstellen. Achte darauf, dass die Klassen nützliche Konstruktoren haben. Insbesondere sollen die Konstruktoren dafür sorgen, dass jede Person eine global eindeutige Personalnummer erhält.

Schreibe Methoden zum Zugriff auf die (als private Felder implementierten) Daten jeder Person sowie eine Methode `void print()`, die die Daten der Person geeignet auf dem Bildschirm ausgibt. Schreibe ein Hauptprogramm, das einige Beispielpersonen anlegt und ausgibt. Beispieldaten findest du z.B. auf der Coli-Webseite unter "People".

Für den Bonuspunkt implementiere eine Klasse `PersonList` analog zu `IntList` aus Aufgabe 3, die jetzt Listen von Personen darstellt. Lege die Beispieldaten in einem `PersonList`-Objekt ab und gib sie mit einer Schleife wieder aus.

Abgabe bis 24. 5. 2004, 9 Uhr

(java-uebungen@coli.uni-sb.de)