TECHNISCHE UNIVERSITÄT MÜNCHEN
FAKULTÄT FÜR INFORMATIK

Lehrstuhl für Sprachen und Beschreibungsstrukturen
Praktikum Grundlagen der Programmierung
Prof. Dr. Helmut Seidl

SS 2008
Aufgabenblatt 5
19. Mai 2008

Abgabe: 27.-29. Mai 2008 beim jeweiligen Tutor

Praktikum Grundlagen der Programmierung

Themen: Klassen und Objekte (Kapitel 10), Listen (Kapitel 11.1)

Wichtiger Hinweis: Es wird dringend angeraten, die oben genannten Kapitel im Skript *vor* dem Praktium *selbständig* zu erarbeiten, um Unklarheiten, insbesondere im Bezug auf die Hausaufgaben, rechtzeitig mit dem Tutor abklären zu können.

Aufgabe 27 (Ü) Bruchrechnen

Implementieren Sie eine Klasse `Rational` zur Repräsentation rationaler Zahlen. Definieren Sie Objekt-Methoden für folgende Operationen:

a) Addition: `Rational add(Rational r)`
b) Subtraktion: `Rational sub(Rational r)`
c) Multiplikation: `Rational mul(Rational r)`
d) Division: `Rational div(Rational r)`
e) Dezimalwert: `double decimal()`
   Der Aufruf `r.decimal()` soll den dezimalen Wert von `r` berechnen und zurückliefern.
f) Vergleich mit einer anderen Bruchzahl: `int compareTo(Rational r)`
   Der Aufruf `r1.compareTo(r2)` soll -1 zurückliefern falls $x_1 < x_2$, 0 falls $x_1 = x_2$ und schließlich 1 falls $x_1 > x_2$. Dabei sollen x_1 und x_2 die Werte sein, die durch die Objekte `r1` und `r2` repräsentiert werden.
g) Duplizieren: `Rational clone()` Der Aufruf `r.clone()` soll eine Kopie von `r` erstellen und zurückliefern.

Anmerkung: Die Bruchzahl soll stets in gekürzter Form gespeichert werden.

Aufgabe 28 (Ü) Testen auf Gleichheit

Java bietet zum Testen auf Gleichheit den Operator == und die für alle Objekte definierte öffentliche Methode **`boolean`** `equals(Object obj)` an.

```
int a = 1;
int b = 1;
String c = new String("ab");
String d = new String("a" + "b");
```

a) Gegeben sei folgendes Code-Fragment:
   - Welche Vergleiche sind möglich? Welche Ergebnisse liefern diese Vergleiche? Schreiben Sie ein kleines Testprogramm, um dies herauszufinden.
   - Wie unterscheiden sich `int`-Variablen von `String`-Variablen?
   - Was unterscheidet den Operator == von der Methode `equals()`?
b) Implementieren Sie eine Klasse `Person` mit Attributen für Vor- und Nachname in der eine öffentliche Methode **`boolean`** `equals(Person p)` sinnvoll definiert ist.

Aufgabe 29 (H) Parkhaus (10 Punkte)

Ziel dieser Aufgabe ist es, ein Parkhaus zu simulieren.

a) Implementieren Sie die Klasse `Auto` mit den privaten Attributen `marke` und `kennzeichen`, sowie den beiden Methoden `String getMarke()` und `String getKennzeichen()` und einem sinnvollen Konstruktur.

b) Erweitern Sie die Klasse `Auto` um ein privates Attribut **`int`** `fahrgestellnummer` und eine öffentliche Methode **`int`** `getFahrgestellnummer()`. Die Fahrgestellnummer soll bei jedem neu erzeugten `Auto`-Objekt um eins erhöht werden.

c) Implementieren Sie die Klasse `Parkhaus`, die ein Parkhaus mit 20 Parkplätzen repräsentiert.

d) Erweitern Sie die Klasse `Parkhaus` um die Methode **`int`** `parken(Auto a)`, bei der das Auto `a` den ersten freien Parkplatz des Parkhauses belegt. Die Methode soll dabei die Nummer des belegten Parkplatzes zurückliefern. Wenn das Parkhaus voll ist, soll `-1` zurückgegeben werden.

e) Erweitern Sie die Klasse `Parkhaus` um die Methode **`int`** `suchen(String k)`, die mit Hilfe des Kennzeichens ein Auto im Parkhaus sucht. Ist das Auto mit dem Kennzeichen `k` im Parkhaus, soll die Methode die Nummer des Parkplatzes ausgeben, ansonsten `-1`.

f) Erweitern Sie die Klasse `Parkhaus` um die Methode `Auto wegfahren(String k)`, das den vom Auto mit dem Kennzeichen `k` belegten Parkplatz wieder freigibt und das Auto zurückgibt. Falls das Auto mit Kennzeichen `k` nicht im Parkhaus ist, soll **`null`** zurückgegeben werden.

g) Testen Sie beide Klassen jeweils mit einer geeigneten `main`-Methoden. Simulieren Sie dabei auch die Interaktion zwischen beiden Klassen.

Aufgabe 30 (H) Listen und Mengen (10 Punkte)

Ziel dieser Aufgabe ist es, eine Klasse `Menge` zur Repräsentation von Mengen von `Strings` zu entwickeln. Implementieren Sie die Klasse `Menge` mit Hilfe einer **verketteten Liste** (siehe Vorlesungsskript), in der jedes Element der repräsentierten Menge **genau einmal** vorkommt. Die Klasse `Menge` soll folgende Methoden bereitstellen:

a) Einen Konstruktor zur Erzeugung einer **leeren** Menge.

b) Eine Methode **`boolean`** `istElement(String s)`, die überprüft, ob der String `s` in der Menge enthalten ist.

c) Eine Methode **`void`** `hinzufuegen(String s)`, die den String `s` zu der Menge hinzufügt.

d) Eine Methode `String toString()`, die eine String-Darstellung der Menge erzeugt und zurückliefert.

e) Eine Methode `Menge vereinige(Menge m)`, die die Vereinigung der Menge **`this`** und `m` zurückliefert.

f) Eine Methode `Menge schnitt(Menge m)`, die den Durchschnitt der Menge **`this`** und `m` zurückliefert.

g) Eine Methode **`int`** `size()`, welche die Größe der Liste zurückliefert.

Hinweis: Die Verwendung statischer Member kann sehr hilfreich sein!