Fachhochschule Ingolstadt WS 2007/2008
Studiengang Elektro- und Informationstechnik/
Studiengang Informatik

Prüfung Praktische Informatik 2
Prüfung Softwareentwicklung 2
Prüfung Grundlagen der Programmierung 2

Aufgabensteller: Prof. Glavina, Prof. Grauschopf, Prof. Hahndel, Prof. Schmidt
Prüfungsdauer: 90 Minuten
Hilfsmittel: keine

Aufgabe 1: Klassen und Methoden (ca. 25 %)

Die folgenden Klassen bilden eine Teilmenge einer Notenverwaltung. Die dabei genutzte Exception-Klasse `IllegalArgumentException` ist im Paket `java.lang` vordefiniert und von `RuntimeException` abgeleitet.

a) Schreiben Sie eine Klasse `Bewertung`, die als private Attribute eine Matrikelnummer und eine Punktezahl hat. Für die Klasse soll ein Konstruktor definiert werden, der die Attribute mit per Parameter übergebenen Werten belegt. Falls der Parameter für die Punktezahl nicht zwischen 0 und 100 liegt, so soll eine `IllegalArgumentException` ausgelöst werden. Weiter sollen für die beiden Attribute Getter definiert werden.

Eine Klasse `Notenschlüssel` hat die Attribute

```
private final static int PUNKTE[] = {0, 41, 47, 53,
               59, 65, 71, 77, 83, 89, 95};
     private final static double NOTE[] = {5.0, 4.0, 3.7, 3.3,
               3.0, 2.7, 2.3, 2.0, 1.7, 1.3, 1.0};
```

`PUNKTE[i]` ist die Minimalzahl von Punkten, die zum Erreichen von Note `NOTE[i]` nötig ist.
Die Note `NOTE[0]` zählt als „nicht bestanden“, alle anderen als „bestanden“.

b) Schreiben Sie zu `Notenschlüssel` eine statische Methode `bestanden` mit Rückgabetyp `boolean`, die zu einer als Parameter übergebenen Bewertung ermittelt, ob die Punktezahl zum Bestehen ausreicht.

c) Schreiben Sie zu `Notenschlüssel` eine statische Methode `note`, die zu einer als Parameter übergebenen Bewertung die zugehörige Note berechnet und zurückgibt.

d) Schreiben Sie eine Klasse `Prüfung`, die als Attribute den Namen der Prüfung und eine Liste von Bewertungen hat (Klassendeklaration und Attribute reichen). Die Liste müssen Sie in den folgenden Teilaufgaben elementweise durchlaufen können. Wählen Sie dazu eine geeignete Datenstruktur.

e) Schreiben Sie für die Klasse `Prüfung` eine Methode `einzelNote`, die als Parameter eine Matrikelnummer erhält. Die Methode durchsucht die Liste der Bewertungen nach dieser Matrikelnummer. Findet sich für die Matrikelnummer keine Bewertung, so wird eine `IllegalArgumentException` ausgelöst. Sonst berechnet man für die Bewertung mit Hilfe des Notenschlüssels die Prüfungsnote und gibt sie zurück.

f) Schreiben Sie zu Klasse `Prüfung` eine Methode `durchschnittBestanden`, die die Durchschnittsnote aller als bestanden gewerteten Einzelleistungen ermittelt.

Aufgabe 2: Vererbung und Polymorphie (ca. 25 %)

a) Schreiben Sie eine abstrakte Klasse `Nahrungsmittel` mit drei Attributen `gewicht, fett` und `kcal`. `gewicht` ist das Gesamtgewicht in Gramm, `fett` das Gewicht des enthaltenen Fettes und `kcal` der KCal-Wert für 100 Gramm dieses Lebensmittels.

   Ausserdem verfügt die Klasse über einen geeigneten Konstruktor, der alle Attribute belegt und eine Methode `getKcal`, die anhand des Gesamtgewichtes den Gesamtwert der KCal berechnet

b) Leiten Sie von der Klasse `Nahrungsmitel` folgenden drei Klassen ab und implementieren Sie diese vollständig mit folgenden Eigenschaften:
   - Eine Klasse `Semmel` mit 50 Gramm Gewicht, 1 Gramm Fett und 250 Kcal/ 100 gr
   - Eine Klasse `Kassler` mit einem Fettgehalt von 1/6 des Gesamtgewichtes und 260 KCal pro 100 Gr.
   - Eine Klasse `Sauerkraut` mit 28/10000 Fettanteil und 15 KCal pro 100 Gramm

c) Schreiben Sie eine Klasse `Gang`. Diese enthält in einer geeigneten, privaten Datenstruktur eine Liste von Nahrungsmitteln. Ausserdem verfügt sie über eine Methode `add`, um den Zutaten ein neues Nahrungsmittel hinzufügen zu können. Eine Methode `getKcal` berechnet den Wert der Kcal des kompletten Ganges, indem die einzelnen Kcal-Werte der Zutaten aufaddiert werden.

d) Schreiben Sie eine Anwendungsklasse mit einer `main`-Methode, die folgendes leistet:

   Sie erzeugt einen hauptgang vom Typ `Gang`, fügt diesem 2 Semmeln, ein Kassler mit einem Gewicht von 140 Gramm und 200 Gramm Sauerkraut hinzu und druckt anschliessend die berechneten Kcal auf der Konsole aus.

Aufgabe 3: Modellierung (ca. 25 %)

Die Grundbausteine der Materie sind die Elementarteilchen. Diese können wir – in sehr stark vereinfachter Form – durch Angabe ihrer elektrischen Ladung und ihres Drehimpulses, des sogenannten „Spins“, modellieren.

Wir wollen die Teilchen wie folgt verwenden können:

```
Teilchen elektron = new Teilchen(- 1.0, 0.5);
Teilchen up       = new Teilchen(+ 2.0 / 3.0, 0.5);
Teilchen down     = new Teilchen(- 1.0 / 3.0, 0.5);
Teilchen photon   = new Teilchen(0.0, 1.0);
```

Dabei stellt der erste Parameter die Ladung, der zweite den Spin des Teilchens dar.
Die Parameter sind Gleitpunktzahlen, weil Ladung und Spin bei den Elementarteilchen als Vielfache von 1/2 und 1/3 vorkommen. Das up-Quark hat z.B. die Ladung +2/3 und den Spin 1/2, das down-Quark die Ladung -1/3 und den Spin 1/2.

a) Definieren Sie die Klasse `Teilchen` mit einem Getter für den Spin.

b) Definieren und initialisieren Sie eine Variable `strange`, die ein strange-Quark mit der Ladung -1/3 und dem Spin 0 darstellen soll.

c) Fügen Sie der Klasse `Teilchen` eine Methode `antiteilchen` hinzu, die ein neues `Teilchen` zurückgibt, dessen Ladung gerade das entgegengesetzte Vorzeichen hat wie das aktuelle `Teilchen`. Die Methode soll man wie folgt aufrufen können:

```
Teilchen antidown = down.antiteilchen();
```

Wir wollen jetzt Teilchensysteme modellieren, die aus mehreren Elementarteilchen zusammengesetzt sind. Baryonen bestehen z.B. aus 3 Elementarteilchen, Mesonen nur aus 2. Dafür bietet sich ein dynamisches Array als Datenstruktur an.

d) Definieren Sie eine Klasse `Teilchensystem` mit dem Attribut

```
private Teilchen[] teilchen;
```

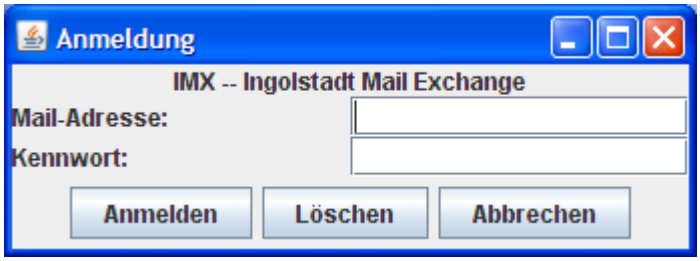
und initalisieren Sie dieses Attribut im Konstruktor.

e) Definieren Sie ein Proton, bestehend aus einem down- und zwei up-Quarks, und ein Pion, bestehend aus einem up-Quark und einem down-Antiquark. (Verwenden Sie die vorstehend definierten Variablen `up`, `down` und `antidown`.)

f) Bosonen sind Teilchensysteme mit *ganzzahligem* Gesamtspin, z.B. 0, 1 oder 2. Teilchensysteme mit *halbzahligem* Spin – z.B. 1/2 oder 3/2 – sind keine Bosonen. Der Gesamtspin ist die Summe der Spins aller Teilchen eines Teilchensystems.
Fügen Sie der Klasse `Teilchensystem` eine Methode `istBoson` hinzu, die `true` liefert, falls das `Teilchensystem` ein Boson ist, sonst `false`.

Aufgabe 4: Java-GUI (ca. 25 %)

Bei der Anmeldung von Benutzern eines Mail-Dienst soll eine interaktive Oberfläche gemäß folgender Abbildung zum Einsatz kommen.



a) Skizzieren Sie die Hierarchie der verschachtelten Layout-Manager des obigen Fensters; aus der Skizze sollten die von Ihnen gewählten Layout-Manager und die darin angeordneten GUI-Elemente ablesbar sein.

Der Benutzer gibt seine Mail-Adresse und sein Kennwort ein und drückt dann "Anmelden"; für unsere Zwecke soll hier statt einer Weitergabe der Daten an den Mail-Server nur eine Ausgabe an der Konsole gedruckt werden:

- falls die Mail-Adresse ein `@`-Zeichen enthält, werden die eingegebene Mail-Adresse und das Kennwort ausgegeben, in der Form "`Anmeldung von Mueller@imx.de/relleum`".
- ansonsten wird eine Fehlermeldung ausgegeben, etwa "`Ungültige Mail-Adresse: mueller.imx.de`".

Bei "Löschen" sollen die beiden Eingabefelder geleert werden (z.B. für eine Neueingabe). Bei "Abbrechen" wird das Programm beendet.

b) Schreiben Sie ein vollständiges Java-Programm, so dass das vom Programm erzeugte Fenster der obigen Abbildung entspricht und auch das Verhalten des Programms mit der obigen Beschreibung übereinstimmt.