Fachhochschule Ingolstadt WS 2005/2006
Studiengang Elektro- und Informationstechnik
Studiengang Informatik

Prüfung Softwareentwicklung 2
Prüfung Grundlagen Informatik 2[1]
Prüfung Praktische Informatik 2

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

Aufgabe 1: (Containerklassen und Exceptions, ca. 24%)

In Ihrer Freizeit betätigen Sie sich gerne als DJ. All Ihre Musik-CD's sind fein säuberlich nummeriert. Um Ihre Auftritte besser planen zu können, programmieren Sie eine CD-Verwaltung.

a) Eine CD hat

- eine Nummer,
- einen Namen,
- einen Interpreten, und eine
- Titelliste. Ein einzelner Titel wiederum hat
    - eine Nummer (nämlich das wievielte Stück der Titel auf der CD ist)
    - einen Namen,
    - eine Länge,
    - und (wir brauchen dies später) eine Referenz auf die CD, auf der der Titel drauf ist.

Modellieren Sie dies über Klassen. In Java sollten Sie korrekt parametrisierte Instanzen von `java.util.LinkedList` oder `java.util.ArrayList` verwenden. (Falls Sie C++ verwenden, nutzen Sie analog entweder die Klasse `list` oder `vector`.) Konstruktoren und `get...`-Methoden (z.B. `getName()`) müssen nicht definiert werden und können im weiteren als gegeben angenommen werden.

b) Mit einer grafischen Oberfläche (nicht Bestandteil dieser Aufgabe) können Sie eine Play-Liste erstellen. Eine Play-Liste ist einfach eine Liste von Titeln, die Sie hintereinander spielen wollen. Außerdem hat die Play-Liste ein konstantes Attribut, das die maximale Gesamtlänge aller Stücke auf eine Stunde begrenzt, und ein Attribut, das die aktuelle Gesamtlänge aller Stücke angibt. Schreiben Sie eine Klasse für eine Play-Liste. Ein Konstruktor muss nicht definiert werden, kann aber im weiteren als gegeben angenommen werden.

1 Falls Sie die Prüfung als Wiederholungsprüfung für „Softwareentwicklung 2“ oder „Grundlagen Informatik 2“ schreiben, können Sie die Aufgaben 1 bis 3 wahlweise in C++ oder Java programmieren.

c) Die Play-Liste soll eine Methode zum Hinzufügen eines Titels haben. Zwei Bedingungen sollen dabei zur Auslösung einer `PLException` führen:

i) Wenn der selbe Titel zum zweiten mal in eine Play-Liste eingefügt wird.
ii) Wenn die Gesamtlänge aller Stücke die Maximallänge übersteigt.

Schreiben Sie die Methode und die Exception-Klasse.

d) Die Play-Liste soll die Methode `public String toString()` haben, die eine String-Repräsentation der Play-Liste erzeugt. Der Rückgabe-String soll für jeden Titel eine Zeile mit folgenden Daten enthalten:

- Nummer der CD
- Name des Interpreten
- Nummer des Titels auf der CD
- Name des Titels

Schreiben Sie die Methode.

Aufgabe 2 (Vererbung und Polymorphie, 27%)

Für das Warenangebot eines Supermarkts sollen Klassen modelliert werden. Wir beschränken uns hier auf die folgenden drei Warengruppen mit den jeweiligen Attributen

- Fleisch: Gewicht (in Gramm), Kilopreis (in Cent)
- Gebäck: Frische-Faktor (0 bis 100, 100 bedeutet 100% frisch; bei abnehmender Frische wird der Verkaufspreis des Produkts proportional reduziert), Stück-Preis (in Cent, Referenzpreis für ganz frisches Produkt)
- Getränk: Inhalt (in Centiliter), Liter-Preis (in Cent), Pfand-Preis (in Cent)

Ausserdem sollen alle Waren eine Artikel-Nummer und eine Bezeichnung haben, sowie eine Methode `getPreis()`, die den Preis (in Cent) des Produkts aus den Attributen berechnet.

a) Skizzieren Sie für dieses Warenangebot ein Klassendiagramm inklusive der Namen für Attribute und Methoden. Vermeiden Sie unnötige Mehrfachdefinitionen.

b) Geben Sie die vollständigen Definitionen der Klassen gemäß Teilaufgabe a) inklusive geeigneter Konstruktoren an.

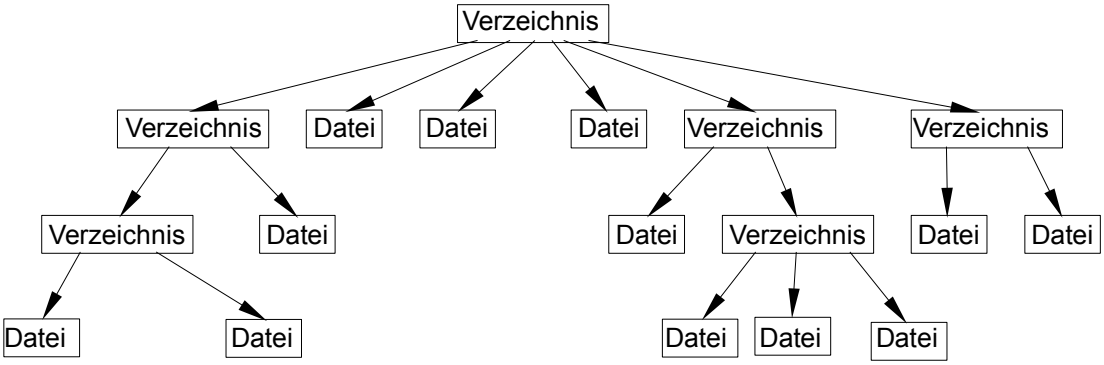
c) Schreiben Sie eine Anwendung mit einer Methode `main`, die in einer Reihungsvariablen `angebot` zunächst die folgenden Produkte anlegt (das soll dem Angebot des Ladens entsprechen):

- "Schinken" (ein Fleischprodukt, Artikelnr = 1022), 200 g, 29 Euro/kg
- "Brezel" (ein Gebäck, Artikelnr = 1390), 50% frisch, 45 Cent (Referenzpreis)
- "Burpsicola" (ein Getränk, Artikelnr = 2525), 0.3 l, 5.29 Euro/l, 16 Cent Flaschenpfand

Dann soll über die Reihung iteriert und dabei der Wert (=Summe aller Verkaufspreise) des Ladenangebots festgestellt werden. Geben Sie den Angebotswert am Bildschirm in der Form "Der Gesamtwert aller Waren beträgt ... Cent." aus.

Aufgabe 3 (Rekursive Datenstrukturen, ca 24%)

Ein einfaches Dateisystem kann als Verzeichnis aufgefasst werden, das als Einträge sowohl Dateien als auch weitere Verzeichnisse enthalten kann.



Zur Vereinfachung wird angenommen, dass ein Verzeichnis maximal 1000 Einträge haben kann.

- Sowohl Dateien als auch Verzeichnisse verfügen über eine Bezeichnung, einen Eigentümer und eine Berechtigung, die eine Beschreibung der Zugriffsrechte enthält. Diese Attribute können über Zeichenketten modelliert werden.

- Jedes Verzeichnis verfügt über eine Reihung mit Einträgen und ein Attribut `anzahlEinträge`, dass die Anzahl dieser Einträge enthält.

- Jede Datei verfügt über die Angabe der Dateigröße (Anzahl der Bytes als ganzzahligen Wert).

a) Entwerfen und implementieren Sie eine geeignete Klassenhierarchie für dieses Dateisystem. Konstruktoren und `get...`-Methoden (z.B. `getBerechtigung()`) müssen nicht definiert werden und können im weiteren als gegeben angenommen werden.

b) Ergänzen Sie die Klasse `Verzeichnis` um die Methode `dateiAnlegen`. Diese Methode erhält als Parameter die Bezeichnung, den Eigentümer, die Berechtigung der Datei und die Dateigröße.

c) Deklarieren Sie in allen Klassen Ihrer Klassenhierarchie die Methode

```
Datei findeDatei(String dateiName)
```

Diese Methode soll in der jeweiligen Klasse nach einer Datei mit dem angegebenen Namen suchen und diese, falls gefunden, zurückgeben; andernfalls soll eine null-Referenz zurückgegeben werden.

Implementieren Sie diese Methode in der Klasse `Verzeichnis` und, falls notwendig, in weiteren Klassen. Die Methode `findeDatei` der Klasse `Verzeichnis` soll all ihre Dateien und Unterverzeichnisse durchsuchen, um die angegebene Datei zu finden; die erste Datei mit dem gesuchten Namen soll zurückgegeben werden.

Aufgabe 4 (GUI in Java, ca 25%)

Schreiben Sie ein vollständiges Java-Programm, welches ein Fenster mit Hilfe der Methode `setLocation(x, y)` aus der Klasse `Frame` an verschiedene Stellen des Bildschirms plaziert. Das Fenster habe folgende Elemente:

- Der Titel in der oberen Fensterleiste soll "Move it!" heißen (`setTitle`).
- Das Fenster soll anfangs mit der linken oberen Ecke auf die Koordinaten (500, 300) gesetzt werden.
- Innerhalb des Fensters soll der Text "Bitte nach links oder rechts schieben" zu lesen sein.
- Unter diesem Text sollten drei Buttons nebeneinander stehen, mit den Beschriftungen "<", "Ende" und ">".
- Wenn man den Button "<" anklickt, sollte das Fenster um 1 Pixel nach links verschoben werden.
- Wenn man den Button ">" anklickt, sollte das Fenster um 1 Pixel nach rechts verschoben werden.
- Wenn man den Button "Ende" anklickt, sollte das Programm beendet werden.

Hinweis: Zur Klasse `ActionEvent` gehört die Methode `getActionCommand;` diese Methode liefert die Beschriftung des Buttons zurück, der das `ActionEvent` verursacht hat.

(Ende des Aufgabentextes)