Dr. M. Grabert
Johannes Mayer

Objektorientierte Programmierung mit C++ (WS 2001/2002)

Abgabetermin: 20. Dezember 2001

10 Studenten, wollt ihr ewig leben? (10 Punkte)

Schreiben Sie analog zur Vorlesung Klassen *Student*, *BioStudent* und *WimaStudent*. Die beiden letzten Klassen werden direkt von der Klasse *Student* abgeleitet. Zum Test soll es mindestens eine virtuelle Methode in *Student* geben, die von den beiden abgeleiteten Klassen überschrieben wird. (Zum Beispiel die Methode `printInfo()` aus der Vorlesung.) Soweit gibt's ja noch nichts neues!

Nun sollen die Klasse *Student* und die beiden abgeleiteten Klassen die Möglichkeit bieten, sich in einen Stream zu schreiben und aus einem Stream den gespeicherten Zustand wieder herzustellen. Dazu schreiben Sie zwei Methoden `store()` und `restore()`, die dafür zuständig sind. Außerdem soll es einen Konstruktor geben, der als Parameter einen Input-Stream erhält und aus diesem den Zustand des Objekts (mittels `restore()`) wieder herstellt. (Objekte, die ihren Zustand dauerhaft speichern können und aus dem gespeicherten Zustand wieder hergestellt werden können, heißen *persistente Objekte*.)

Nun brauchen wir noch einen Verwalter. Dazu schreiben Sie eine eigene Klasse, die Objekte vom Typ *Student* (oder einer abgeleiteten Klasse) verwaltet. Diese soll beim Konstruktor den Namen der Datenbank-Datei erhalten und alle Studenten aus der Datenbank einlesen. Dabei soll jeweils der richtige Typ wieder hergestellt werden, also ein gespeicherter Student soll auch wieder ein Student sein, ein Bio-Student wieder ein Bio-Student, und gleiches gilt für einen Wima. Sie sollen also zwei private Methoden schreiben, die alle verwalteten Studenten in die Datenbank schreiben und alle Studenten aus der Datenbank lesen (wobei der Typ der Objekte erhalten bleiben soll, d. h. Student bleibt Student, Bio-Student bleibt Bio-Student und Wima bleibt Wima). Außerdem schreiben Sie öffentliche Methoden zum Anlegen, Ersetzen, Löschen und Finden eines Studenten (über seinen Login-Namen). Außerdem schreiben Sie noch eine Methode mit der man einen Vektor der Login-Namen aller verwalteten Studenten erhält.

Und last but not least: Was bringen die schönsten Klassen, wenn wir sie nicht verwenden. Schreiben Sie also eine kleine Anwendung, die die vorgestellten Klassen verwendet und folgendes kann: Ausgabe der Liste aller verwalteten Studenten (mit und ohne individuelle Ausgabe `printInfo()` oder einer anderen virtuellen Methode), Anlegen/Ersetzen eines Studenten und Löschen eines Studenten. Diese kann so ähnlich aufgebaut sein wie der Mail-Reader Emilia.

P.S. Die Liste der Studente, die vor dem Beenden des Testprogramms angezeigt werden kann, soll bei einem erneuten Programmstart wieder genauso ausschauen! Auch die Typen der Objekte, die ja angezeit werden können (indirekt via `printInfo()` oder so), sollen noch immer gleich sein. Sonst hat sich wo ein Fehler eingeschlichen. ;-)

Viel Spass und Erfolg!