7. Übung **Informatik B** **SS 08**

Klaus Kriegel **Abgabe: 10.06.2008, 12:00 Uhr**

Aufgabe 1: **Vererbung** $(3 + 2 + 4$ Punkte$)$

Ziel ist die Implementierung eines einfachen Weckers als Erweiterung einer Uhr.

a) Definieren Sie eine Klasse `Clock` mit den Attributen `hour` und `minute` zur Darstellung der Zeiten von `00:00` bis `23:59`. Die Attribute sollen als `private` deklariert sein sollen und entsprechende `public get`–Funktionen und `protected set`–Funktionen haben. Die `void`–Methode `next()` soll die Uhrzeit um eine Minute weiterstellen.

b) Definieren Sie eine Unterklasse `AlarmClock` mit den Attributen `alHour` und `alMinute` zur Darstellung der Weckzeit analog wie oben. Dazu gibt es eine Variable `alarm` vom Typ `boolean`, die beim Wert `true` genau dann den Wecker auslöst, wenn Uhrzeit und Weckzeit gleich sind. Das soll durch eine Methode boolean checkAlarm() realisiert werden.

c) Implementieren Sie eine Testklasse, mit der folgenden Funktionalität: Man übergibt auf der Kommandozeile eine Uhrzeit, eine Weckzeit und einen weiteren positiven `int`–Wert, der angibt, wieviele Minuten seit Stellen der Uhr vergangen sind. Als Ausgabe sollte der String
`Es ist hh:mm Uhr, der Weckalarm wurde nicht/ wurde vor x Minuten ausgelöst`
erscheinen. Wenden Sie dazu die Methoden aus a) und b) an. Die Uhrzeit sollte mit jeweils zweistelligen Zahlen angegeben werden.

Aufgabe 2: **Klasse für Graphen** $(2 + 3 + 3 + 3$ Punkte$)$

a) Entwerfen Sie eine Klasse `Graph` mit einem Attribut `n` für die Anzahl der Knoten und einem Feld `adj` vom Typ `boolean[][]` für die Adjazenzmatrix des Graphen, wobei $V = \{0, 1, 2, \ldots, n-1\}$ angenommen wird.
Implementieren Sie einen Konstruktor `Graph(int k, int[][] adjList)`, der für einen Graphen mit k Knoten und der vorgegebenen Adjazenzliste die entsprechende Adjazenzmatrix erzeugt. Sie koennen voraussetzen, dass die gegebene Adjazenzliste passend ist, d.h. aus genau `k` Feldern von Typ `int[]` besteht in denen nur Zahlen aus dem Bereich von 0 bis $k-1$ auftreten.

b) Implementieren Sie mit einer Funktion `int[][] convert()` die Umkehrabbildung zum Konstruktor aus a).

c) Implementieren Sie eine Methode `boolean checkSimpl()`, die überprüft, ob das Objekt einen schlichten Graphen repräsentiert, und Methoden `void insertEdge(int i, int j)`, `void deleteEdge(int i, int j)` zum Einfügen und Löschen einer Kante zwischen den Knoten i und j. Geben Sie eine entsprechende Nachricht aus, wenn eine bereits vorhanden Kante eingefügt bzw. eine nicht vorhandene Kante gelöscht werden soll.

d) Schreiben Sie eine `main`–Methode mit der Ihre Klasse am Beispiel eines "Teilerfremd"-Graphen getestet wird. Dabei soll für eine Eingabe n der Graph auf $V = \{0, 1, \ldots, n-1\}$ konstruiert werden, in dem i und j genau dann benachbart sind, wenn $ggT(i, j) = 1$ ist.