Dirk Praetorius
Michele Ruggeri

Wintersemester 2014/15
16.12.2014

Übungen zur Vorlesung
Einführung in das Programmieren für TM

Serie 12

Aufgabe 12.1. Implementieren Sie die Zugriffsfunktionen der Klasse Bruch, welche durch

```
class Bruch {
   long zaehler;
   unsigned long nenner;
public:
   Bruch();
   Bruch(long zaehler, unsigned long nenner);
   setZahler(long z);
   setNenner(unsigned long n);
   double getWert();
};
```

gegeben ist. Die Methode `getWert` soll dabei den Dezimalwert des Bruchs zurückgeben. Achten Sie bei den `set`-Funktionen auf eventuelle Sicherheitsabfragen. Implementieren Sie auch den Konstruktor `Bruch()`, welcher den Zähler auf 0 und den Nenner auf 1 setzen soll. Speichern Sie den Source-Code unter `bruch.cpp` in das Verzeichnis `serie12`.

Aufgabe 12.2. Schreiben Sie eine Klasse `Stoppuhr` welche zur Simulation einer Stoppuhr dienen soll. Die Stoppuhr bestehe dabei aus zwei Knöpfen. Wird der erste Knopf gedrückt, so soll die Zeitmessung gestartet werden. Wird dieser Knopf nochmals gedrückt, wird die Zeitmessung gestoppt. Der zweite Knopf dient dazu die Zeit wieder zurückzusetzen. Schreiben Sie dazu die Methoden `pushButtonStartStop` und `pushButtonReset`. Implementieren Sie weiters eine Methode, welche die verstrichene Zeit im Format `hh:mm:ss.xx` ausgibt (Beträgt die gemessene Zeit also zwei Minuten so soll `00:02:00.00` ausgegeben werden). Sie können diese Stoppuhr nun dazu verwenden Zeitmessungen durchzuführen. Speichern Sie den Source-Code unter `stoppuhr.{hpp,cpp}` in das Verzeichnis `serie12`.
Hinweis: Verwenden Sie den Datentyp `clock_t` und die Funktion `clock()` aus der Bibliothek `time.h`. Vermutlich ist es auch sinnvoll eine Variable `isRunning` vom Typ `bool` einzuführen. Bei Betätigen des ersten Knopfes wird diese Variable entweder von `false` auf `true` gesetzt oder umgekehrt.

Aufgabe 12.3. Schreiben Sie eine Klasse `University`. Diese soll neben den Feldern `numStudents`, `city` und `name` die Methoden `graduate` und `newStudent` haben. Wird `graduate` aufgerufen, so verringert sich die Anzahl der Studenten um 1, wohingegen `newStudent` die Anzahl um 1 erhöht. Alle Datenfelder sollen als `private` deklariert sein. Sie müssen sich also zusätzlich `get`- und `set`-Methoden schreiben. Speichern Sie den Source-Code unter `University.{hpp,cpp}` in das Verzeichnis `serie12`.

Aufgabe 12.4. Erstellen Sie eine Klasse `Name` welche zwei String-Variablen `vorname` und `nachname` enthält. Implementieren Sie auch die Zugriffsfunktion `setName`, welche einen String übernimmt, diesen in Vor- und Nachname aufteilt und in die entsprechenden Variablen abspeichert. Beachten Sie auch, dass mehrere Vornamen vorhanden sein können! Schreiben Sie weiters eine Methode `printName` die Vor- und Nachname am Bildschirm ausgibt. Bei mehreren Vornamen soll, beginnend ab dem zweiten Vornamen, jeder weitere Vorname mit dem Anfangsbuchstaben und einem Punkt abgekürzt werden! Beim Namen `Max Maxi Mustermann` soll also `Max M. Mustermann` am Bildschirm erscheinen. Speichern Sie den Source-Code unter `name.{hpp,cpp}` in das Verzeichnis `serie12`.

Aufgabe 12.5. Als Kunde einer Bank kann man auch mehrere Sparkonten besitzen. Erstellen Sie eine Klasse `Kunde` welche eine Liste von Sparkonten beinhaltet. Weiters soll die Klasse auch ein Objekt der Klasse `Name` aus Aufgabe 12.4 enthalten. Implementieren Sie Methoden zum Hinzufügen und Löschen von Konten. Schreiben Sie dann eine Methode die das Guthaben auf allen vorhandenen Sparkonten

berechnet. Überlegen Sie welche Funktionen noch sinnvoll wären. Speichern Sie den Source-Code unter `kunde.{hpp,cpp}` in das Verzeichnis `serie12`.

Aufgabe 12.6. Erstellen Sie eine Klasse `SparKonto` mit den Variablen `kontonummer`, `guthaben` und `zinssatz`. Ferner sollen noch die `get` und `set` Funktionen für die Variablen `zinssatz` und `kontonummer` implementiert werden. Um das Guthaben zu ändern schreiben Sie die Methoden `abheben` und `einzahlen`. Beachten Sie, dass Sie bei einem Sparkonto nicht ins Minus gehen können. Der Zinssatz und die Kontonummer dürfen natürlich auch nicht negativ werden. Schließlich implementiere man noch die Methode `berechneGuthaben`. Speichern Sie den Source-Code unter `sparkonto.{hpp,cpp}` in das Verzeichnis `serie12`.

Aufgabe 12.7. Schreiben Sie ein `Makefile` für die aktuelle Übungsserie. Dieses soll zumindest folgende Dinge umfassen:

- Erzeugen von Programmen aller von Ihnen gelösten Aufgaben.
- Das Generieren einer Bibliothek und deren Verwendung in einem Programm.

Aufgabe 12.8. Was gibt folgender Code am Bildschirm aus und warum?

```
#include <iostream>
#include <string>
using namespace std;

class T1 {
   string t1;
public:
   T1(string val) { cout << "Ich bin Konstruktor von " << val << endl; t1=val; }
   T1() { cout << "Ich bin Konstruktor von default" << endl; t1="default"; }
   ~T1() { cout << "Ich bin Destruktor von " << t1 << endl; }
};

int main() {
   T1 bert("bert");
   T1 bob;
   T1 def("bob");
   return 0;
}
```