technische universität dortmund

fakultät für informatik

Elias Kuthe
Roman Kalkreuth
Bernd Zey

Dortmund, den 24. Januar 2020

Praktikum zur Vorlesung Einführung in die Programmierung WS 19/20

Blatt 13

Es können 30 Punkte erreicht werden.

Allgemeine Hinweise

1. Bitte lesen Sie vor der Bearbeitung **alle** Aufgaben sorgfältig durch! Dies erspart Ihnen unnötige Arbeit und somit auch Zeit!
2. Lassen Sie sich fertiggestellte Aufgaben bitte möglichst **frühzeitig** testieren. In der letzten halben Stunde vor Schluss werden bei diesem Blatt nur noch **zwei** Teil-Aufgaben testiert!
3. Wir akzeptieren ein Testat nur, wenn die Lösung eigenständig auf Anhieb erklärt werden kann. Andernfalls müssen wir die entsprechende Teilaufgabe mit 0 Punkten bewerten.
4. Bei diesem Blatt gibt es keine Einschränkungen bezüglich inkludierten Headern.

Aufgabe: Klassen, Funktionen, Vektoren, Ausnahmen (30 Punkte)

a) Legen Sie ein neues Projekt `Aufgabe_13` an und fügen Sie eine Klasse `Student` hinzu. Ein Student hat einen Namen (Typ `string`) und eine Matrikelnummer (`unsigned int`); die Attribute sollen natürlich privat sein. Fügen Sie einen Konstruktor zum Setzen und get-Methoden zum Lesen der Attribute hinzu.

_____ (4)

b) Stellen Sie sicher, dass die get-Methoden (und die Methode aus **e)**) der Klasse `Student` auch auf einem konstanten Objekt aufrufbar sind.

_____ (2)

c) Fügen Sie Ihrem Projekt eine `.cpp` mit der `main`-Funktion hinzu. Erzeugen Sie einen `std::vector`, in dem Studenten gespeichert werden können. Fügen Sie 5 Studenten mit unterschiedlichen Namen und Matrikelnummern hinzu.

_____ (4)

d) Durchlaufen Sie den Vektor mit Hilfe eines Iterators und geben Sie die Studenten mit Ihrem Namen und der Matrikelnummer untereinander aus.

_____ (3)

e) Fügen Sie eine Methode hinzu, um Objekte vom Typ `Student` direkt mit `cout` ausgeben zu können. Beispielsweise soll

```
Student s("Max Mustermann", 555);
cout << "Student: " << s << endl;
```

folgendes ausgeben:

```
Student: Name Max Mustermann, Matrikelnummer 555
```

Durchlaufen Sie den Vektor wie in **d)** mit Hilfe eines Iterators und geben Sie die Studenten mit Hilfe dieser Methode aus.

_____ (4)

f) Sortieren Sie den Studenten-Vektor mit `std::sort` anhand der Matrikelnummern und geben Sie den Vektor anschließend aus.
Damit `std::sort` Objekte vom Typ `Student` vergleichen kann, ist es am einfachsten, den `<`-Operator für die Klasse zu implementieren. Fügen Sie daher die Funktion `bool operator<(Student const & s2) const` zur Klasse `Student` hinzu. Diese Funktion soll `true` zurück geben, wenn die eigene Matrikelnummer kleiner als die Matrikelnummer von `s2` ist.

_____ (5)

g) Schreiben Sie eine bool'sche Funktion `duplikat`, welche als Parameter den Studenten-Vektor als Referenz übergeben bekommt.
Die Funktion `duplikat` soll `false` zurückgeben, wenn alle Matrikelnummern unterschiedlich sind und `true` im anderen Fall. Gehen Sie bei dieser Funktion wie folgt vor. Für einen leeren Vektor oder einen Vektor der Größe eins wird direkt `false` zurück gegeben. Ansonsten durchlaufen Sie – wiederum mit Hilfe eines Iterators – den Vektor und vergleichen Sie den aktuellen Studenten mit dem darauf Folgenden (beachten Sie hierbei die Vektor-Grenzen). Falls diese zwei Matrikelnummern übereinstimmen geben Sie `true` zurück. Wird der Vektor komplett durchlaufen und kein Duplikat gefunden, dann geben Sie `false` zurück.
Testen Sie Ihre Implementierung auf einem Duplikat-freien Vektor, und nachdem Sie ein Duplikat hinzugefügt haben, indem Sie die Funktion in der `main`-Funktion aufrufen.

_____ (5)

h) Die Funktion `duplikat` funktioniert nur, wenn der Vektor sortiert ist. Ändern Sie die Funktion daher dementsprechend ab, so dass im Durchlauf zusätzlich überprüft wird, ob der Vektor nach Matrikelnummern aufsteigend sortiert ist. Wenn dies nicht der Fall ist, dann soll in der Funktion eine Exception geworfen werden. Definieren Sie hierfür eine geeignete Klasse.
Passen Sie zudem die `main`-Funktion an, so dass eine mögliche Exception gefangen wird. In dem Fall soll außerdem eine Fehlerausgabe angezeigt werden.
Testen Sie den Mechanismus indem Sie die Funktion auf einem sortierten und einem unsortierten Vektor aufrufen.

_____ (3)