Prof. Dr. A. Weber
Dr. Hassan Errami

Universität Bonn
Institut für Informatik II
08.05.2019

Sommersemester 2019

Übungen zur Vorlesung
Objektorientierte Softwareentwicklung (BA-INF-024)
Aufgabenblatt 5 (20 Punkte)

Zu bearbeiten bis: 17.05.2019

Aufgabe 1 *(Kapselung in Java - 2+2=4 Punkte)*

Betrachten Sie die folgende Java-Klasse.

```
class Angestellter {
    String vorname;
    String nachname;
    int alter;
    int gehalt;
}
```

a) Wenden Sie das Prinzip der Kapselung auf alle Eigenschaften der Klasse an. Die Eigenschaften sollen ausschließlich während der Initialisierung gesetzt werden können und sonst nur lesbar sein. Programmieren Sie zudem je eine Methode, welche den Angestellten um ein Jahr altern lässt resp. das Gehalt des Angestellten um einen bestimmten Betrag erhöht.

b) Programmieren Sie eine Klasse *Praktikant*, welche von *Angestellter* ableitet. Das Gehalt von Praktikanten kann nicht erhöht werden und liegt konstant bei 400 Euro. Treffen Sie entsprechende Vorkehrungen.

Aufgabe 2 *(Vererbung und Objektverwaltung - 3+3=6 Punkte)*

a) Zur Verwaltung verschiedener Körper im Raum $\mathbb{R}^3$ möchten wir eine entsprechende Klassenhierarchie konstruieren und implementieren.

- Es gibt Kugeln und Quader.
- Jeder Körper befindet sich an einer Position $p \in \mathbb{R}^3$
- Der Mittelpunkt der Kugel ist p, bei dem Quader ist p die linke, untere, vordere Ecke.
- Die Körper sollen verschoben werden können.
- Die Körper sollen um einen Faktor $a \in \mathbb{R}$ skaliert werden können.
- Für jeden Körper soll das Volumen berechnet werden können.
- Stellen Sie zur Ausgabe der Parameter (Art, Volumen, Position) eine Methode zu verfügung.

b) Organisieren Sie mehrere Körper mit Hilfe einer Liste (java.util.ArrayList<E>). Fügen Sie hierzu mehrere beliebige Körper in die Liste ein. Anschließend soll die Liste durchlaufen werden und für jeden Körper die Ausgabefunktion aufgerufen werden.

Aufgabe 3 *(Sortieren durch Auswahl - 4 Punkte)*

Schreiben Sie eine Methode, die ein Array mit double-Variablen sortiert. Die Methode soll ein Array übergeben bekommen und dieses dann durch geschickte Wahl des Elements sortieren. Dabei wird immer das kleinste Element im übergebenen Array rausgesucht und in das Neue, an der ersten freien Stelle eingefügt.

Aufgabe 4 *(Ausnahmebehandlung - 3*2=6 Punkte)*

Betrachten Sie das folgende Programm `TestTrace`, welches die Methode `methodA()` der Klasse `CallEg` aufruft:

```
class CallEg {
  public void methodA() throws ArithmeticException { }
  public void methodB() throws ArithmeticException { }
  public void methodC() throws ArithmeticException { }
}
public class TestTrace {
  public static void main(String[] args) {
    CallEg eg = new CallEg(); // use default constructor
    try {
      eg.methodA();
    } catch (ArithmeticException oops) {
      oops.printStackTrace();
    }
  }
}
```

a) Der `catch{}`-Block des Hauptprogramms gibt den *Stacktrace* der abgefangenen Ausnahme aus. Ergänzen Sie `methodA()` so, dass eine Division durch Null auftritt. Betrachten Sie die Ausgabe und verfolgen Sie die angegebenen Aufrufe im Stack.

b) Verschieben Sie die Division aus `methodA()` nach `methodC()`. Ändern Sie den Code so, dass `methodA()` die Methode `methodB()` aufruft, welche wiederum `methodC()` aufruft. Führen Sie das Programm aus und beobachten Sie, wie sich die Ausgabe ändert.

c) Ändern Sie den Code aus Aufgabenteil b) so, dass `methodB()` von `methodA()` innerhalb eines `try{}`-Blocks aufgerufen wird, ebenso soll `methodC()` von `methodB()` in einem `try{}`-Block aufgerufen werden. Setzen Sie auch die Division in `methodC()` in einen `try{}`-Block. Setzen Sie hinter jeden `try{}`-Block ein `catch{}`, welches die Ausnahme fängt, einen *Stacktrace* ausgibt und die Ausnahme an seinen Aufrufer wirft. Kommentieren Sie die Ausgabe.