

Einführung in die Programmierung
WS 2018/19

Übungsblatt 12: Klassenhierarchien, Exceptions, UML-Diagramme

Besprechung: 28.01.-01.02.2019

Aufgabe 12-1 *Überschreiben, Überladen, Verdecken*

Betrachten Sie die folgenden Klassen:

```
1 public class A {  
2     public int i = 11;  
3     public A(int i) {                // Signatur: A(I)  
4         super();  
5         this.i = i;  
6     }  
7     public A(float f) {              // Signatur: A(F)  
8         this((int)(f + 1));  
9     }  
10 }
```

```
1 public class B extends A {  
2     public float f = 31;  
3     public B(double d) {              // Signatur: B(D)  
4         this((float)(d - 1));  
5     }  
6     public B (float f) {              // Signatur: B(F)  
7         super(f);  
8         this.f = f;  
9     }  
10 }
```

```
1 public class Main {  
2     public static void main (String [] args) {  
3         A a1 = new A(31);              // (K1)  
4         System.out.println("A.i: " + a1.i);  
5         A a2 = new A(31.9999f);         // (K2)  
6         System.out.println("A.i: " + a2.i);  
7         A ab = new B(3.5);              // (K3)  
8         System.out.println("A.i: " + ((A) ab).i);  
9         System.out.println("B.i: " + ((B) ab).i);  
10        B b = new B(8);                  // (K4)  
11        System.out.println("A.i: " + ((A) b).i);  
12        System.out.println("B.i: " + ((B) b).i);  
13    }  
14 }
```

Geben Sie an, welche Konstruktoren- und Methodenaufrufe stattfinden. Benutzen Sie nach Möglichkeit keinen Computer, sondern nutzen Sie Ihr Wissen zur Objektorientierung aus der Vorlesung. Verwenden Sie die als Kommentare gegebenen Funktionssignaturen. Begründen Sie alle Antworten kurz.

- (a) Beschreiben Sie in maximal 3 Sätzen den Unterschied zwischen Überschreiben, Überladen und Verdecken.
- (b) Geben Sie an, welche Konstruktoren in welcher Reihenfolge an den mit (K1)-(K4) bezeichneten Stellen in Klasse `C` benutzt werden. Achtung: Wie Sie wissen, hat jede Klasse außer `Object` eine Oberklasse. Geben Sie außerdem an, welche Attribute mit welchen Werten belegt werden und welche Werte durch die `println`-Anweisungen ausgegeben werden.

Lösungsvorschlag:

```
K1: Object() A(I)           => A.i: 31
K2: Object() A(I) A(F)      => A.i: 32
K3: Object() A(I) A(F) B(F) B(D) => A.i: 3      B.i: 3
K4: Object() A(I) A(F) B(F) => A.i: 9      B.i: 9
```

a) Überschreiben kann nur in einer erbbenden Klasse geschehen, wobei eine signaturgleiche Methode wie aus der Superklasse neu erstellt wird. Überladen bedeutet, dass es mehrere Methoden (in der selben Klasse) gibt, die den gleichen Namen haben, aber eine andere Parameterliste. Verdecken/-Verstecken tritt auf, wenn ein Attribut oder eine Methode aus der Superklasse in der erbbenden Klasse neu definiert wird.

Aufgabe 12-2 *Quartett*

In dieser Aufgabe modellieren Sie die Erstellung eines Quartetts. Ihnen steht bereits eine `main`-Methode zur Verfügung. Wie Sie sicherlich wissen, besteht ein Quartett aus maximal 4 Musikern. Implementieren Sie die Klasse `Musiker`, die maximal 4 Musiker unterstützt. Nutzen Sie dazu die Möglichkeiten der Datenkapselung. Wichtig ist hierbei, dass keine Musikerobjekte über die ersten 4 hinaus erstellt werden dürfen. Das heißt, dass auch keine andere Klasse den Konstruktor aufrufen können darf. Wenn das Maximum an Musikern erreicht ist, soll eine Exception geworfen werden, die eine entsprechende Meldung ausgibt.

Aufgabe 12-3 *Dungeons, Beasts and Swords (DBS)*

Modellieren Sie ein rudimentäres Spiel, in dem Menschen und Zwerge gemeinsam gegen Orks und Goblins kämpfen. Ein Kampf wird von n Einheiten geführt, die gegeneinander kämpfen. Jede Einheit hat Lebenspunkte und die Einheit stirbt, falls diese auf einen Wert kleiner oder gleich null fallen. Zu Beginn startet jede Einheit mit 20 Lebenspunkten. Unterschiedliche Klassen haben spezielle Kampfeigenschaften.

Das gesamte Programm liegt als Sourcefiles zur Verfügung. Erstellen Sie aus dem Codegerüst ein UML-Diagramm, das den Code konzeptuell beschreibt. Eine solche Spezifikation ermöglicht jemandem ohne Java-Kenntnisse, das Programm eventuell in eine andere Sprache zu portieren.