Prof. Dr. A. Weber
Dr. Hassan Errami

Universität Bonn
Institut für Informatik II
24.04.2019

Sommersemester 2019

Übungen zur Vorlesung
Objektorientierte Softwareentwicklung (BA-INF-024)
Aufgabenblatt 3 (20 Punkte)

Zu bearbeiten bis: 03.05.2019

Aufgabe 1 *(Statisches und Dynamisches Binden - 8*0.5 = 4 Punkte)*

Gegeben seien folgende Java-Klassen:

```
class C1 {
    static int s=1;
    void f1(){
        System.out.println("C1::f1");
    }
    void f2(){
        System.out.println("C1::f2");
    }
}


class C2 extends C1 {
    static int s=2;
    void f1(){
        System.out.println("C2::f1");
    }
    void f3(){
        System.out.println("C2::f3");
    }
}

public class Test {
    public static void main (String[] args) {
        C1 a=new C2();
        C2 b=new C2();
        // Stelle 1
    }
}
```

Geben Sie an, was das Ausgabeergebnis von folgenden Codefragmenten ist, wenn sie an `Stelle 1` eingefügt werden. Neben der Ausgabe - die auch leer sein kann und in diesem Fall durch $\emptyset$ gekennzeichnet werden soll - sind auch *Kompilierfehler* bzw. *Laufzeitfehler* mögliche Antworten.

1. `((C2) a).f1();`
2. `((C1) b).f1();`
3. `System.out.println(a.s);`
4. `b.f1();`
5. `b.f2();`
6. `b.f3();`
7. `C1 c = new C1(); c.f3();`

8. `C1 c = new C1(); ((C2) c).f1();`

Aufgabe 2 *(Cloning - 2+3+3+2=10 Punkte)*

Diese Aufgabe beschäftigt sich mit dem Klonen von Schaf-Objekten. Unter dem Klon O_k eines Objektes O versteht man ein unabhängiges Objekt, welches den gleichen Typ wie O hat und dessen interner Zustand mit dem Zustand von O übereinstimmt. Damit gilt also: O_k ist gleich O, aber nicht identisch. Gegeben seien die Klassen *Sheep.java* und *Fur.java*:

```
public class Sheep {
    public String name;
    public Fur fur;

    public Sheep(String name, Fur fur) {
        this.name = name;
        this.fur = fur;
    }

    public void shear() {
        fur.length = 0;
    }

    public String toString() {
        return "Name: " + name + " " + fur.toString();
    }

    public Sheep clone() {
        return this;
    }
}

public class Fur {
    public int length;

    public Fur(int length) {
        this.length = length;
    }

    public String toString() {
        return "Felllaenge: " + length;
    }
}
```

a) Schreiben Sie eine Klasse *Main.java* mit einer *main*-Methode, in der ein neues Schaf-Objekt mit einer Felllänge von 10cm erstellt werden soll. Dieses Schaf soll unter Zuhilfenahme der *clone*-Methode geklont werden. Geben Sie die Attribute beider Schafe auf der Konsole unter Benutzung der *toString*-Methode aus. Ändern Sie nun den Namen des Klons, scheren es mit der *shear*-Methode und geben die Attribute beider Schafe erneut aus. Was fällt Ihnen auf?

b) Recherchieren Sie, was man unter deep- und shallow cloning versteht. Implementieren Sie dann zwei weitere Methoden *shallowclone()* und *deepclone()*, deren Funktionalität die Durchführung eines shallow- und eines deep clones sein soll. Testen Sie das Beispiel aus Aufgabe 1 mit beiden *clone*-Methoden erneut. Unterscheiden sich die Ausgaben? Wenn ja: Weshalb und was ist der Unterschied? **Hinweis:** Beachten Sie, dass Sie eventuell ebenfalls die Klasse *Fur* verändern müssen!

c) Geben Sie eine korrekte Implementierung für die *equals*-Methode der *Sheep*-Klasse an. Zwei Objekte sind gleich, wenn ihr interner Zustand, d.h. alle Instanzvariablen, gleich sind. Um zu überprüfen, ob ein Objekt O vom Typ T ist, verwenden Sie folgende Abfrage:

```
if (O instanceof T) // ...
```

d) Recherchieren Sie was für eine Bedeutung das Interface Cloneable in Java hat.

Aufgabe 3 *(Bindungen - 3 Punkte)*

Konstruieren Sie ein Interface Playable, das eine Methode void song() zu verfügung stellt und bei Aufruf eine Zeile eines Liedes ausgibt.
Schreiben Sie eine Klasse Kassette, die das Interface implementiert und 'Alle meine Entchen...' ausgibt.
Schreiben Sie eine weitere Klasse CD, die auch das Interface implementiert, aber bei Aufruf von song() 'O Tannenbaum...' ausgibt.
Schrieben Sie eine dritte Klasse Stereoanlage, die eine Methode play(Playable p) hat, die ein Interface Playable übergeben bekommt und dann die Methode song() aufruft.

Sei nun folgender Code in der Main-Methode gegeben:

```
Stereoanlage s=new Stereoanlage();
Playable pl=new CD();
s.play(pl);
pl=new Kassette();
s.play(pl);
```

Warum kommen unterschiedliche Ausgaben raus, obwohl Zeile 3 und 5 identisch sind?

Aufgabe 4 *(Singleton - 2+1 Punkte)*

a) Entwickeln Sie eine Klasse Singleton, von der immer nur eine Instanz erzeugt werden darf. Dies wird erreicht, indem der Konstruktor „private“ deklariert wird, und eine Instanz der Klasse nur über eine öffentliche Methode erhältlich ist. Die einzige Instanz der Klasse wird in einem privaten Datenfeld gehalten. Wenn dieses noch nicht initialisiert ist, also den Wert null enthält, konstruiert diese Methode mit dem privaten Konstruktor die einzige Instanz. Das Singleton in ein bekanntes Entwurfsmuster. In was für Fällen kann es Verwendung finden?

b) Es gibt eine Variante der Klasse Singleton, bei der die Instanz der Klasse nicht erst beim Aufruf der öffentlichen Methode konstruiert wird, sondern schon vorher. Schreiben Sie ein weitere Klasse Singleton2 die dies realisiert.