Prof. Dr. A. Weber
Dr. Hassan Errami

Universität Bonn
Institut für Informatik II
06.06.2019

Sommersemester 2019

Übungen zur Vorlesung
Objektorientierte Softwareentwicklung (BA-INF-024)
Aufgabenblatt 8 (20 Punkte)

Zu bearbeiten bis: 14.06.2019

In der Woche vom 24.6 bis 27.6 findet die zweite Online-Aufgabe statt.

Aufgabe 1 *(Generics - 4*1=4 Punkte)*

Gegeben seien die folgenden Klassen:

```
public class TierKaefig<E> {
    private E insasse;
    public void setInsasse(E x){
        insasse= x;
    }
    public E getInsasse(){
        return insasse;
    }
}
public class Tier {}
public class Hund extends Tier {}
public class Katze extends Tier {}
```

Überlegen Sie sich, ob die Java-Code-Auschnitte

```
a) TierKaefig<Tier> kaefig= new TierKaefig<Katze>();
```

```
b) TierKaefig<Hund> kaefig= new TierKaefig<Tier>();
```

```
c) TierKaefig<?> kaefig= new TierKaefig<Katze>();
   kaefig.setInsasse(new Katze());
```

```
d) TierKaefig kaefig = new TierKaefig();
   kaefig.setInsasse(new Hund());
```

- nicht compilierbar sind,
- mit einer Warnung wegen mangelnder Typsicherheit compilierbar sind,
- einen Laufzeitfehler erzeugen oder
- ohne Probleme lauffähig sind.

Aufgabe 2 *(Generics - 6 Punkte)*

Gegeben sei der folgende Quelltext:

```
class R { }
class E extends R { }
class B extends R { }
class G extends B { }
class U { }
public class Foo {
  public static <T> T bar(T x, T y) {
    return x;
  }
  public static void main(String[] args) {
```

```
    Object o;
    R r = new R();
    E e = new E();
    B b = new B();
    G g = new G();
    U u = new U();
    R[] x;
    E[] y;

    // Stelle 1
  }
}
```

Akzeptiert der Java-Compiler die folgenden Anweisungen jeweils eingefügt an der Stelle 1? Begründen Sie Ihre Antwort mit einem Satz.

- r = bar(r,b);
- r = bar(b,r);
- r = bar(e,g);
- r = bar(b,g);
- e = bar(r,b);
- u = bar(u,r);
- o = bar(e,u);
- b = bar(e,g);
- x = bar(new E[2], new B[4]);
- y = bar(new E[4], new B[11]);
- o = bar(new G[8], new U[5]);

Aufgabe 3 *(Stack - 1+3+1+1+4=10 Punkte)*

Sie sollen nachfolgend eine Klasse programmieren, die die „Stack"-Datenstruktur für Strings zur Verfügung stellt. Die Verwendung einer bestehenden Stack-Klasse u.ä. ist nicht erlaubt.

a) Programmieren Sie ein Java-Interface, welches von List ableitet und in dem die folgenden üblichen Stack-Methoden enthalten sind: push, pop.

b) Programmieren Sie Ihre Stack-Klasse, welches Ihr Interface implementiert. Sie müssen nur essentielle Methoden implementieren.

c) Betrachten Sie Sonderfälle bzw. Randbedingungen. Welche könnten das sein? Sorgen Sie dafür, dass Ihr Programm auch dann noch gemäß einer intuitiven Erwartung funktioniert, falls diese Fälle eintreten.

d) Programmieren Sie das Hauptprogramm, welches Ihre Stack-Klasse testet. Pushen oder Poppen Sie in der folgenden Reihenfolge und geben Sie nach jedem den Stack verändernden Schritt den kompletten Stack auf der Konsole aus. *Push „Apfel", Push „Orange", Pop, Push „Erdbeere", Push „Kiwi", Pop, Pop, Pop, Pop*

e) Machen Sie Ihren Stack nun generisch, d.h. ändern Sie nun ihren Code derart, dass der Stack für beliebige Typen instanziert werden kann. Testen Sie Ihre Implementierung, indem Sie einen Stack für die Verwaltung von Integers instanzieren und einige Zahlen einfügen und abrufen.