**1. Frage: Instanziierungen**

Gegeben seien die folgenden Klassen:

public class Kaefig<T> {  
 private T einTier;  
 public void setTier(T x) {  
 einTier = x;  
 }  
 public T getTier() {  
 return einTier;  
 }

}  
   
public class Tier{ }  
public class Hund extends Tier { }  
public class Vogel extends Tier { }

Beschreiben Sie was mit dem folgenden Code geschieht. Die Möglichkeiten sind

* er übersetzt nicht
* er übersetzt mit Warnungen
* er erzeugt Fehler während der Ausführung
* er übersetzt und läuft ohne Probleme

**1.1**

Kaefig<Tier> zwinger = new Kaefig<Hund>();

**1.2**

Kaefig<Vogel> voliere = new Kaefig<Tier>();

**1.3**

Kaefig<?> voliere = new Kaefig<Vogel>();  
voliere.setTier(new Vogel());

**1.4**

Kaefig voliere = new Kaefig();  
voliere.setTier(new Vogel());

**2. Umwandeln einer nicht generischen Implementierung in eine generische Implementierung**

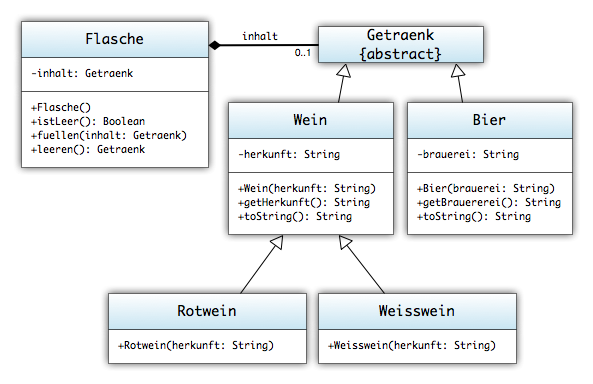
In diesem Beispiel werden Flaschen mit verschiedenen Getränken gefüllt und entleert. Das Befüllen und Entleeren erfolgt in der *main()* Methode der Klasse *Flasche*.

Die vorliegende Implementierung erlaubt das Befüllen der Flaschen f1 und f2 mit beliebigen Getränken.

Aufgabe:

Ändern Sie die Implementierung der Klasse *Flasche* in eine generische Klasse die für unterschiedliche Getränke parametrisiert werden kann.

* passen Sie alle Methoden der Klasse *Flasche* so an das sie mit einem parametrisierten Getränk befüllt werden können (Bier oder Wein)
* Ändern Sie die *main()* Methode derart, dass f1 nur mit Bier befüllt werden kann und f2 nur mit Wein.



**2.1 Klasse *Flasche***

Die Klasse dient als Hauptprogramm. In dieser Übung muss ausschließlich die Methoden dieser Klasse und das Hauptprogramm angepasst werden.

package Kurs2.Generics;

public class Flasche {

Getraenk inhalt = null;

public boolean istLeer() {  
 return (inhalt == null);  
 }

public void fuellen(Getraenk g) {  
 inhalt = g;  
 }

public Getraenk leeren() {  
 Getraenk result = inhalt;  
 inhalt = null;  
 return result;  
 }

public static void main(String[] varargs) {  
 // in generischer Implementierung soll  
 // f1 nur für Bier dienen  
 Flasche f1 = new Flasche();  
 f1.fuellen(new Bier("DHBW-Bräu"));  
 System.out.println("f1 geleert mit " + f1.leeren());  
 f1 = new Flasche();  
 f1.fuellen(new Bier("DHBW-Export"));  
 System.out.println("f1 geleert mit " + f1.leeren());  
   
 // In der generischen Implementierung soll f2 nur für  
 // Weinflaschen dienen  
 Flasche f2;  
 f2 = new Flasche();  
 f2.fuellen(new Weisswein("Pfalz"));  
 System.out.println("f2 geleert mit " + f2.leeren());  
   
 f2 = new Flasche();  
 f2.fuellen(new Rotwein("Bordeaux"));  
 System.out.println("f2 geleert mit " + f2.leeren());  
 }  
}

**2.2 Klasse *Getraenk***

package Kurs2.Generics;

public abstract class Getraenk {

}

**2.3 Klasse *Bier***

package Kurs2.Generics;

public class Bier extends Getraenk {  
 private String brauerei;  
   
 public Bier(String b) { brauerei = b;}  
 public String getBrauererei() {  
 return brauerei;  
 }  
 public String toString() {return "Bier von " + brauerei;}  
}

**2.4 Klasse *Wein***

package Kurs2.Generics;

public class Wein extends Getraenk {  
 private String herkunft;

public String getHerkunft() {  
 return herkunft;  
 }

public String toString(){ return ("Wein aus " + herkunft);}

public Wein (String origin) {  
 herkunft = origin;  
 }

}

**2.5 Klasse Weisswein**

package Kurs2.Generics;

public class Weisswein extends Wein {  
 public Weisswein(String h) {super(h);}

}

**2.6 Klasse Rotwein**

package Kurs2.Generics;

public class Rotwein extends Wein {  
 public Rotwein(String h) {super(h);}

}

**3. Typprüfungen**

Welche Zeilen in der *main()* Methode werden vom Übersetzer nicht übersetzt?

Markieren Sie die Zeilen und nennen Sie den Fehler:

package Kurs2.Generics;

public class KoordinateTest<T extends Number> {

public T x;  
 public T y;

public KoordinateTest(T xp, T yp) {  
 x = xp;  
 y = yp;  
 }

public static void main(String[] args) {  
 KoordinateTest<Double> k11, k12;  
 KoordinateTest<Integer> k21, k22;  
 Koordinate<String> k31;  
 KoordinateTest<Number> k41, k42;  
   
 k11 = new KoordinateTest<Float>(2.2d, 3.3d);  
 k12 = new KoordinateTest<Double>(2.2d, 3.3d);  
 k21 = new KoordinateTest<Integer>(2, 3);  
 k31 = new Koordinate<String>("11","22");  
 k41 = new KoordinateTest<Number>(2l, 3l);  
   
 k41 = new KoordinateTest<Number>(4.4d, 5.5f);  
 k11 = new Koordinate<Double>(3.3f,9.9d);  
   
 KoordinateTest<? super Double> k99;  
 k99 = k11;  
 k99 = k41;  
 k99 = k31;  
   
 k11 = k12;  
 k12 = k21;  
 KoordinateTest k55 = new KoordinateTest<Number>(7.7f, 8.8f);  
 KoordinateTest k66 = new KoordinateTest(7.7f, 8.8f);  
 }  
}