Programmiermethodik
LV-Nr.: 703017-_
Sommersemester 2013
Übungstest
3.6.2013
Dauer: 60 Minuten

Name: _____
Matrikelnummer: _____

Lösungen

Dieser Test enthält 10 Seiten (inklusive Deckblatt) und 4 Probleme. Bitte überprüfen Sie die Seitenzahl. Notieren Sie Name und Matrikelnummer auf dem Deckblatt.

Es sind *keine* Unterlagen erlaubt!

Sollte Ihnen der Platz für Antworten zu knapp werden, verwenden Sie bitte die Rückseite.

Bitte tragen Sie *nichts* in die Tabelle rechts ein!

Problem	Punkte	Erzielte Punkte
1	5	
2	10	
3	10	
4	15	
Total:	40	

1. (5 Punkte)

```
1  import java.io.*;
2
3  class Foo {
4
5    public static void main(String [] args) {
6      Foo p = new Foo();
7
8      boolean result = p.isOrdered(new int[]{1,2,4,5,9});
9      System.out.println(result);
10     result = isOrdered(new int[]{1,2,4,5,4});
11     System.out.println(result);
12     int sum = p.multiply(5,5);
13   }
14
15   static boolean isOrdered(int...xs) {
16     int temp = Integer.MIN_VALUE;
17     for(int x : xs) {
18         if( x >= temp) { temp = x; }
19         else return false;
20     }
21       return true;
22   }
23
24
25   long multiply(int x, int y) {
26     long result = x * y;
27     return result;
28   }
29
30   double multiply(double x, double y) {
31     double result = x * y;
32     return result;
33   }
34 }
```

Listing 1: Foo.java

2. (10 Punkte) Kisten, Kisten, Kisten.

Implementieren Sie ein Verwaltungssystem, das Gegenstände in Boxen verwalten kann. Eine Box kann dabei eine beliebige Anzahl an Gegenständen und kleineren Boxen beinhalten. Implementieren Sie zwei Klassen `Box` und `Item`, die das gegebene Interface `IBox` implementieren. Die Methode `getWeight` gibt das Gewicht einer Kiste bzw. eines Items zurück. Das Gewicht der Items wird explizit gespeichert, das Gewicht einer Kiste berechnet (wobei die Kiste selbst nichts wiegt). Die Methode `getMaxWeight` gibt das Gewicht des schwersten Items zurück. Implementieren Sie weiters die Methode `getItemsHeavierThan`, die alle Items in einer Kiste zurückliefert, die mehr wiegen als der Grenzwert der übergeben wird.

```
import java.util.Collection;

public interface IBox {

  public double getWeight();

  public double getMaxWeigth();

  public Collection<Item> getItemsHeavierThan(double weight);

}
```

Listing 2: IBox.java

```
import java.util.Collection;
import java.util.HashSet;


public class Box implements IBox {

  private Collection<IBox> contained = new HashSet<IBox>();

  public double getWeight() {
    double weight = 0;
    for (IBox element : contained) {
      weight += element.getWeight();
    }
    return weight;
  }

  public double getMaxWeigth() {
    double max = 0;
    for (IBox element : contained) {
      if (max < element.getMaxWeigth()) max = element.getMaxWeigth();
    }
    return max;
  }

  public Collection<Item> getItemsHeavierThan(double weight) {
    HashSet<Item> ret = new HashSet<Item>();
    for (IBox element : contained) {
      ret.addAll(element.getItemsHeavierThan(weight));
    }
    return ret;
  }
```

```
32
34 }
```

Listing 3: Box.java

```
import java.util.Collection;
2 import java.util.HashSet;

4
public class Item implements IBox {
6
  private double weight;
8
  public Item(double weight) {
10   this.weight = weight;
  }
12
  public double getWeight() {
14   return this.weight;
  }
16
  public double getMaxWeigth() {
18   return this.weight;
  }
20
  public Collection<Item> getItemsHeavierThan(double weight) {
22   HashSet<Item> ret = new HashSet<Item>();
    if (this.weight > weight) ret.add(this);
24   return ret;
  }
26
}
```

Listing 4: Item.java

3. (10 Punkte) Überladen, Überschreiben, ...

   Betrachten Sie das Programm `Talk.java` - was gibt dieses Programm aus? Notieren Sie bitte jeweils die Zeilennummer und die entsprechende Ausgabe. Sie können ihre Antworten auch abkürzen (etwa Samantha: Carry wird zu S:C).

   17 Samantha: Carry

   18 Samantha: Miranda

   19 Miranda: Carry

   20 Samantha: Carry

   21 Miranda: Carry

   22 Samantha: Miranda

   23 Miranda: Miranda

   24 Samantha: Samantha

   25 Miranda: Miranda

   26 Exception in thread "main"java.lang.ClassCastException: Carry cannot be cast to Samantha at Talk.main(Talk.java:26)

4. Generics

   (a) (5 Punkte)

```
1  class Pair {
2
3    Object getValue() { return value; }
4
5    void setKey(Object key) { this.key = key; }
6
7    Object key;
8    Object value;
9  }
10
11 class Test {
12
13   void foo(Pair p) {
14     int v = (Integer)p.getValue();
15     p.setKey(String.valueOf(v));
16   }
17 }
```

Listing 5: Pair.java

(b) (5 Punkte) Betrachten Sie das Progamm aus Listing 6.

```
1  public class Wildcard {
2
3      //Note that we couldn't declare the 2D list as List<Object>
4
5       public static double foldLeft(List<?> list, int initialValue) {
6          Double sum = (double)initialValue;
7          for(Object e : list) {
8              switch(initialValue) {
9                  case 0: sum += (Double)e;
10                 case 1: sum *= (Double)e;
11             }
12
13         }
14         return sum;
15     }
16
17     public static void main(String [] args) {
18         List<Number> list = new LinkedList();
19         list.add(3);
20         list.add(null);
21         list.add(5.5);
22         System.out.println(foldLeft(list, 0));
23     }
24 }
```

Listing 6: Wildcard.java

- Warum der Kommentar? Warum kann `list` nicht als Liste von `Object` deklariert werden?
  Aufgrund des Aufrufs

```
foldLeft(list, 0)
```

- Unter der Annahme, das Programm kompiliert, welches Problem könnte auftreten?
  foldLeft(...) könnte eine Liste mit nicht–numerischen Typen erhalten und wirft folglich bei der Berechnung eine Exception.
- Wie müsste der Typ von `list` abgeändert werden, damit keine Probleme auftreten können?

```
1 public static double foldLeft(List<? extends Number> list, int
      initialValue)
```

(c) (5 Punkte) Gegeben Sei das Programm aus Listing 7.

```
1  interface A { }
2
3  class B implements A { }
4
5  class C extends B { }
6
7
8  class Carpet<V extends A, B> {
9
10   public Carpet() {
11
12   }
13
14   public <X extends V> Carpet<?, ? extends V> method() {
15     // insert code here
16   }
17 }
```

Listing 7: Generics.java

Welche(s) der folgenden Statements kann/können in Zeile 15 eingefügt werden? Begründen Sie Ihre Antwort, insbesondere, warum ein Statement nicht erlaubt ist!

A `return new Carpet<V, X>();`

B `return new Carpet<X, V>();`

C `return new Carpet<A, B>();`

D `return new Carpet<B, C>();`

E `return new Carpet<A, C>();`