Programmering Fortsättning 2010

Övning 1

Joachim von Hacht

För att lösa problemen kan man naturligtvis testa koden själv. Boken är en källa, en annan källa är Java Language Specification.

1 Referenser

1. Avgör för varje kommentar med frågetecken i koden om uttrycker är sant eller falskt[1].

```
package refs;
public class Refs {
    private void checkRefs() {
        String s1 = "True";
        String s2 = "True";
        // s1 == s2 ?

        String s3 = new String("False");
        String s4 = new String("False");
        //s3 == s4?

        String s5 = "True";
        String s6 = "Tr" + "ue";
        //s5 == s6?
        String s7 = "False";
        String sx = "F";
        String s8 = sx + "alse";
        //s7 == s8 ?

        Integer i = new Integer(2);
        Integer j = new Integer(2);
```

[1] En grundregel är att alltid undvika språkets mörka hörn, men detta är en kurs så...

```
        // i >= j ?
        // i <= j ?
        // i == j ?

        Integer j1 = 127; //Bizarre
        Integer j2 = 127;
        // j1==j2 ?

        Integer k1 = 128;
        Integer k2 = 128;
        // k1 == k2?

    }
}
```

1p

2. Betrakta klasserna SimpleSwapper, ValueHolderSwapper, ValueHolderSwapper2 och ValueHolder och avgör vad som kommer att skrivas ut i Main (main-metoden). Förklara (rita gärna variabler och referenser)!

```
package refval;
class SimpleSwapper {
    public void swap(Integer x, Integer y) {
        Integer temp = x;
        x = y;
        y = temp;
    }
}
```

```
package refval;
class ValueHolderSwapper {
    public void swap(ValueHolder v1, ValueHolder v2) {
        Integer tmp = v1.i;
        v1.i = v2.i;
        v2.i = tmp;
    }
}
```

```
package refval;
class ValueHolderSwapper2 {
    public void swap(ValueHolder v1, ValueHolder v2) {
        v1 = new ValueHolder(v2.i);
        v2 = new ValueHolder(v1.i);
    }
}
```

```
package refval;
public class ValueHolder {
    public Integer i;
    public ValueHolder(Integer i) {
        this.i = i;
    }
}
```

cd

```
package refval;
public class Main {
    public static void main(String[] args) {
        SimpleSwapper ss = new SimpleSwapper();
        ValueHolderSwapper vhs = new ValueHolderSwapper();
        ValueHolderSwapper2 vhs2 = new ValueHolderSwapper2();

        int a = 1;
        int b = 2;
        ss.swap(a, b);
        System.out.println("a= " + a + " b= " + b);

        Integer c = new Integer(1);
        Integer d = new Integer(2);
        ss.swap(c, d);
        System.out.println("c= " + c + " d= " + d);

        ValueHolder v1 = new ValueHolder(a);
        ValueHolder v2 = new ValueHolder(b);
        vhs.swap(v1, v2);
        System.out.println("a= " + v1.i + " b= " + v2.i);

        vhs2.swap(v1, v2);
        System.out.println("a= " + v1.i + " b= " + v2.i);
    }
}
```

2p

3. Lägg till en klass Box så att koden nedan kan kompileras och exekveras. Vad kommer att skrivas ut? Förklara! Ändra så att utskriften blir "bättre" (ni tolkar).

```
package doublebox;
public class DoubleBox {
    private Box innerBox = new Box();
```

```
    public void setValue(int v) {
        innerBox.setValue(v);
    }
    public int getValue() {
        return innerBox.getValue();
    }
    public DoubleBox copy() {
        DoubleBox newValue = new DoubleBox();
        newValue.innerBox = innerBox;
        return newValue;
    }
    public static void main(String[] args) {
        DoubleBox a = new DoubleBox();
        a.setValue(18);
        DoubleBox b = a.copy();
        a.setValue(23);
        System.out.println("b is " + b.getValue());
    }
}
```

2p

2 Enum

1. Har vi gjort något fel i koden nedan (i main-metoden)?

```
package _enum;

public enum Season {
  WINTER, SPRING, SUMMER, FALL;
}
```

```
package _enum;

public class Main {
  public static void main(String[] args) {
    Season s1 = Season.FALL;
    Season s2 = Season.FALL;
    if (s1 == s2) {
      System.out.println("Same season");
    }
    if (s1.equals(s2)) {
      System.out.println("Same season");
    }
```

```
  }
}
```

1p

3 Paket och inkapsling

1. Vad sägs om följande kvalificerade klassnamn?

   ```
   edu.gu.hajo.myapp.class.ExceptionHandler
   ```

   1p

2. Koden nedan har två problem vilka (OBS! Paketen)?

```
package pkg;
import pkg.sub.Impl;
public class Client {
    private Impl impl = new Impl();
    int veryInternalData;
    public Client(int i){
        veryInternalData = impl.square(i);
    }
    public int getI(){
        return veryInternalData;
    }
}

package pkg.sub;
class Impl {
    public int square(int i){
        return i*i;
    }
}
```

1p

4 Gränssnitt

1. Kommer följande att fungera. Om ej, varför (fundera)?

a)

```
public interface IA {
   public int doIt();
   public double doIt();
}
```

b)

```
public interface IA {
   public void doIt() throws Exception;
}

public class ImplA implements IA{
   public void doIt(){
     :
   }
}
```

c)

```
public interface IA {
   public void doIt();
}

public class ImplA implements IA{
   public void doIt() throws Exception{
     :
   }
}
```

d)

```
public interface IA {
   public abstract void doIt();
   public abstract int doOther();
   public abstract int doYetOther(int i);
}

public interface IB {
  public abstract void doIt();
  public abstract void doOther();
  public abstract int doYetOther(String i);
}

public class ImplAB implements IA, IB {
    :
}
```

e)

```
public interface IA {
  public void doIt();
  public void doIt(int i);
}
```

```
public class ImplA implements IA {
  public void doIt() {
    System.out.println("Doing it");}
  public void doIt(int i) {
    System.out.println("Doing it int" + i);}
  public void doIt(String s) {
    System.out.println("Doing it string " + s); }
}
```

2p