Programmering Fortsättning 2010

Övning 2

Joachim von Hacht*

1 Polymorfism

1. Switch-satserna nedan betraktas som dålig kod. Förklara varför? Eliminera switch-satserna m.h.a. polymorfism. TIPS: Skapa gränssnitt med lämpliga metoder. Ersätt case-grenarna med klasser som implementerar gränssnittet.

```
package polymorphism;
import static java.lang.Math.PI;
public class Geometer {

    private static final int SQUARE = 0;
    private static final int RECTANGLE = 1;
    private static final int CIRCLE = 2;
    private double a, b, r;

    public double computeArea(int shape) {
        double area = 0;
        switch (shape) {
        case SQUARE: area = a * a; break;
        case RECTANGLE: area = a * b;break;
        case CIRCLE: area = PI * r * r;break;
        }
        return area;
    }

    public double computePerimeter(int shape) {
        double perimeter = 0;
        switch (shape) {
```

*Övningar med engelsk text hämtade från Sibylle Schupp.

```
            case SQUARE:perimeter = 4 * a;break;
            case RECTANGLE:perimeter = 2 * (a + b);break;
            case CIRCLE:perimeter = 2 * PI * r;break;
            }
            return perimeter;
        }
    }
```

2p

2 Override och overload

1. Vad kommer att skrivas ut? Varför? Kommer programmet att exekvera utan problem?

```
package overload;
public class Main {
   public static void main(String[] args) {
        Classifier c = new Classifier();
        Object[] o = new Object[] {
                new Integer(0), new String(), new Double(0)
        };
        for (int i = 0; i < o.length; i++) {
            c.classify(o[i]);
        }
    }
}
```

```
package overload;
public class Classifier {
   public void classify(Integer i) {
        System.out.println("It's an integer");
    }
    public void classify(String s) {
        System.out.println("It's a String");
    }
    public void classify(Object o) {
        System.out.println("Don't know, it's anything...");
    }
}
```

2p

2. Koden nedan innehåller ett fel vilket? Vilket starttillstånd kommer objektet a2 att ha (då felet eliminerats)?

```
package overload2;
public class A {
    int a = 0;
    double b = 0;
    String s = "Undef";
    public A( int a, double b, String s){
        this.a = a;
        this.b = b;
        this.s = s;
    }
    public A(int a, double b){
        this(a, b, "Undef");
    }
    public A(int a ){
        this(a, a);
    }
}
```

```
package overload2;
public class Main {
    public static void main(String[] args) {
        A a1 = new A();
        A a2 = new A(4);
    }
}
```

1p

3. Vad kommer att skrivas ut? Förklara varför!

```
package override;
public interface IA {
  public void doIt(Object o);
  public void doIt(B b);
}
```

```
package override;
public class B {
   public void sayIt() {
      System.out.println("I'm B");
   }
}
```

```
package override;
public class ImplA implements IA {
```

```
   public void doIt(Object o) {
      System.out.println("Hello");
   }
   public void doIt(B b) {
      b.sayIt();
   }
}

package override;
public class ImplA2 implements IA {
   public void doIt(Object o) {
      System.out.println("See you");
   }
   public void doIt(B b) {
      b.sayIt();
   }
}

package override;
public class Main {
   public static void main(String[] args) {
      IA ia = new ImplA();
      ImplA2 a2 = new ImplA2();
      B b = new B();
      Object o = b;
      ia.doIt(o);
      ia = a2;
      ia.doIt(o);
      ia.doIt(b);
   }
}
```

2p

3 Static

1. Fungerar följande? Om så, förklaring...?

```
package stat;
public class VerySpecial {
    public static void main(String[] args) {
        Math m = null;
        System.out.println(m.random());
```

```
    }
}
```

1p

2. Vad är problemet med följande? Förklara!

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

package stat.override;
public class ImplStatic implements IA {
    public static int doOther(){
        return 365;
    }
    public void doIt() {
      System.out.println("Gnhurv..");
    }
    public int doYetOther(int i){
        return 2*i;
    }
}
```

1p

4 Initiering

1. At first glance, one might think the program below would print the string "TheKing wears a size ...(years from 1930)...belt." Yet, initialization is tricky at times. So what happends and why?

```
package init;
import java.util.Calendar;
public class TheKing {
  public static final TheKing theKing = new TheKing();
  private final int beltSize;
  private static final int this_year = Calendar.getInstance()
      .get(Calendar.YEAR);

  private TheKing() {
    beltSize = this_year - 1930;
  }
```

```
  public int beltSize() {
    return beltSize;
  }

  public static void main(String[] args) {
    System.out.println("TheKing wears a size " +
        theKing.beltSize() + " belt.");
  }
}
```

1p