CHALMERS TEKNISKA HÖGSKOLA
Datavetenskap TDA547

Tentamen i Objektorienterad programmering

Fredagen 15 mars 2013, 14.00 – 18.00.
Förslag till lösningar.

1. Lösning till (a), (b) och (d):

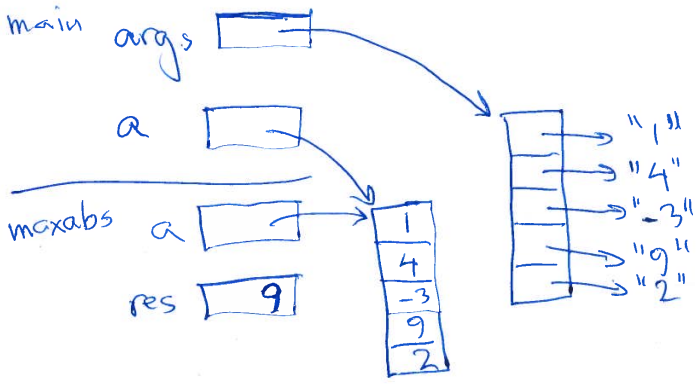
```
public class Uppgift1 {

    public static int maxabs(int[] a) {
        int res = Math.abs(a[0]);
        for (int i=0; i<a.length; i++)
            res = Math.max(res, Math.abs(a[i]));
        return res;
    }

    public static double[] scale(int[] a) {
        double s = maxabs(a);
        double[] res = new double[a.length];
        for (int i=0; i<res.length; i++)
            res[i] = a[i]/s;
        return res;
    }

    public static void main(String[] args) {
        int[] a = new int[args.length];
        for (int i=0; i<a.length; i++)
            a[i] = Integer.parseInt(args[i]);
        System.out.println(maxabs(a));
    }
}
```

Lösning till (c) (obs: skissen kan se annorlunda ut beroende på hur man löst (a) och (b)):



2. (a) `public class Course {`

```
    private String name;
    private int credits;

    public Course(String name, int credits) {
        this.name = name;
        this.credits = credits;
    }

        public String getName() {return name;}

        public int getCredits() {return credits;}
    }
```

(b)
```
public class Student {
        private String name;
        private Course[] passed;
        private int nrPassed;

        public Student(String name, int max) {
            this.name = name;
            passed = new Course[max];
            nrPassed = 0;
        }

        public String getName() {return name;}

        public void add (Course c) {
            passed[nrPassed] = c;
            nrPassed++;
        }

        public int getNrPassed() {return nrPassed;}

        public Course getCourse(int i) {return passed[i];}

        public int totalCredits() {
            int res = 0;
            for (int i=0; i < nrPassed; i++)
                res = res + passed[i].getCredits();
            return res;
        }
    }
```

3.
```
import java.io.*;
import java.util.Scanner;

public class Uppgift3 {

    private static void printHead(Scanner in) {
        int lines = 0;
        while (lines < 10 && in.hasNextLine()) {
            System.out.println(in.nextLine());
            lines++;
```

```
        }
    }

    public static void main(String[] args) throws FileNotFoundException {
        for (int i=0; i<args.length; i++) {
            if (args.length>1)
                System.out.println("==> " + args[i] + " <==");
            printHead(new Scanner(new File(args[i])));
        }
    }
}
```

4. ```
   public class Uppgift4 {

       public void blockpixelate(int[][] image, int x, int y, int side) {
           int avg = 0;
           for (int i=0; i<side; i++)
               for (int j=0; j<side; j++)
                   avg = avg + image[x+i][y+j];
           avg = avg/(side*side);
           for (int i=0; i<side; i++)
               for (int j=0; j<side; j++)
                   image[x+i][y+j] = avg;
       }

       public void pixelate(int[][] image, int side)  {
           for (int x=0; x<image.length/side;  x++)
               for (int y=0; y<image[0].length/side; y++)
                   blockpixelate(image,x*side,y*side,side);
       }
   }
   ```

   Ovanstående lösning antar att hela blocket ligger inom bilden; det är lätt att lägga till test som undersöker detta.

5. ```
   import java.io.*;

   public class FindFile {

       public static void find(File f, String name) {
           if (f.getName().equals(name))
               System.out.println(f.getPath());
           else {
               File[] fs = f.listFiles();
               if (fs!=null)
                   for (int i=0; i<fs.length; i++)
                       find(fs[i],name);
           }
       }

       public static void main(String[] args) {
           find(new File("."),args[0]);
       }
   ```

```
}
```