Övning 6.

Innehåll

På denna övning behandlars trådar och aktiva objekt.

Uppgift 1

För att söka efter en viss sträng i en textfil kan klassen FileSearcher användas.

```
import java.io.*;
public class FileSearcher {
    private String filename;
    private boolean foundIt;

    public FileSearcher(String filename) {
        this.filename = filename;
    }//konstruktor

    public void search(String pattern) throws Exception {
        BufferedReader br = new BufferedReader(new FileReader(filename));
        String line = br.readLine();
        while(line != null && line.indexOf(pattern) == -1)
            line = br.readLine();
        foundIt = line != null;
        br.close();
    }//search

    public String getFileName() {
        return filename;
    }// getFileName

    public boolean found() {
        return foundIt;
    }//found
}//FileSearcher
```

När en instans av klassen skapas ges namnet på den fil man vill söka i och när metoden search anropas ges strängen man söker efter.

Din uppgift är att skriva ett program i vilken man parallellt söker i flera filer efter samma sträng. Strängern som eftersökes och filerna som skall sökas igenom ges om argument till programmet. Klassen FileSearcher skall användas i din lösning (utan förändringar).

Uppgift 2

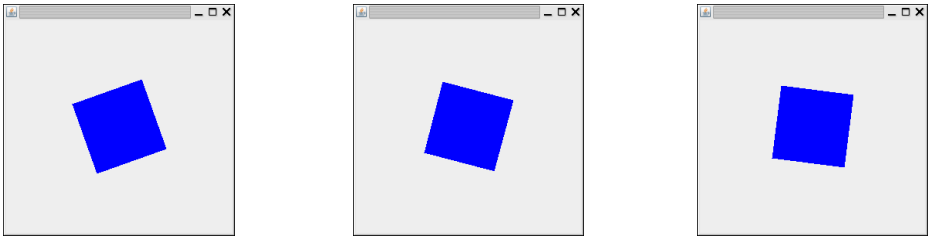
Betrakta klassen Color nedan. Klassen avbildar färger som en trippel (röd, grön, blå) enlig RGB-modellen, där varje färg kodas som ett heltalsvärde mellan 0 och 255.

```
public class Color {
    private int red;
    private int green;
    private int blue;

    public Color(int rred, int green, int blue) {
        if (!isRBGColor(red, green, blue)) {
            throw new IllegalArgumentException();
        }
        this.red = red;
        this.green = green;
        this.blue = blue;
    }//konstruktor

    public void setColor(int red, int green, int blue) {
        if (!isRBGColor(red, green, blue)) {
            throw new IllegalArgumentException();
        }
        this.red = red;
        this.green = green;
        this.blue = blue;
    }//setColor

    public int[] getColor() {
        int[] retVal = new int[3];
        retVal[0] = red;
        retVal[1] = green;
        retVal[2] = blue;
        return retVal;
    }//getColor

    public void invert() {
        red = 255 - red;
        green = 255 - green;
        blue = 255 - blue;
    }//invert

    private static boolean isRBGColor(int red, int green, int blue) {
        return (0 <= red && red <= 255) && ( 0 <= green && green <= 255 )
                &&  ( 0<= blue && blue <= 255);
   }//isRBGColor
}//Color
```

Klassen Color är inte *thread-safe*, dvs om flera trådar samtidigt utför operationer på en instans av Color kan inkonsistens uppstå. Gör klassen *thread-safe*.

Uppgift 3

På övning 4 skrev vi ett program som skapade ett fönster som innehöll en panel i vilken en kvadrat ritades ut som roterade åt höger (enligt figuren nedan). För att få en rörlig bild nyttjade vi ett Timer-objekt. Er uppgift är att lösa samma uppgift, men nu genom att implementera gränssnittet Runnable istället för att använda ett Timer-objekt. Vidare skall ni lägga till en knapp med vilken det skall vara möjligt att ändra åt vilket håll kvadraten roterar.



```
import java.awt.*;
import java.awt.event.*;
import javax.swing.*;
public class RotatingSquare extends JPanel  implements ActionListener{
    private int vinkel= 0;
    private javax.swing.Timer t = new javax.swing.Timer(100, this);
    public RotatingSquare() {
        t.restart();
    }//konstruktor
    public void paintComponent(Graphics pen) {
        super.paintComponent(pen);
        int x0 = getWidth()/2;
        int y0 = getHeight()/2;
        int radie = Math.min(getWidth(), getHeight())/4;
        Polygon p = new Polygon();
        for (int i = 0; i <= 4; i = i + 1) {
            int x1 = x0 + (int) (radie*Math.cos(Math.toRadians(vinkel + i*90)));
            int y1 = y0 + (int) (radie*Math.sin(Math.toRadians(vinkel + i*90)));
            p.addPoint(x1,y1);
        }
        pen.setColor(Color.blue);
        pen.fillPolygon(p);
    }//paintComponent

    public void actionPerformed(ActionEvent e) {
        vinkel = (vinkel + 1) % 360;
        repaint();
    }//actionPerformed

    public static void main(String[] args){
        JFrame w  = new JFrame();
        RotatingSquare rs = new RotatingSquare();
        w.add(rs);
        w.setSize(300, 300);
        window.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        w.setVisible(true);
    }//main
}// RotatingSquare
```

Uppgift 4

På en viss resturang har man tre personer som sköter om disken. En av personerna tvättar disken och de två andra torkar disken (för gästerna lyckligtvis efter att den blivit tvättad). Den som diskar ställer den tvättade disken i ett ställ från vilket de som torkar hämtar disken. Stället rymmer 3 tre enheter.

Din uppgift är att göra en enkel simulering över hur diskningen av 100 enheter går till. Varje enhet modelleras med ett heltal. Anta att det i genomsnitt tar 4 gånger så lång tid att torka en enhet som att tvätta den.