Lösningsförslag till tentamen 150417

Uppgift 1

1) c
2) c
3) a
4) b
5) a
6) c
7) b
8) b
9) b
10) a

Uppgift 2

Public-key encryption bygger på att man har två nycklar, en publik och en privat. Man måste således ha båda nycklarna för att kunna avkoda meddelandet.

Public-key encryption är en kryptering som används för att på ett säkert sätt kunna skicka och ta emot information över Internet. Den fungerar på så sätt att det finns två slags nycklar, en publik och en privat.

Den publika nyckeln används för att koda ett meddelande medan den privata nyckeln krävs för att kunna avkoda meddelandet.

Den publika nyckeln finns tillgänglig för vem som helst som kan tänkas vilja skicka ett meddelande till en viss person, medan det endast är mottagaren som har tillgång till den privata nyckeln. Även om flera personer har den publika nyckeln och vet hur meddelandet är kodat kan ingen utom mottagaren avkoda meddelandet. På så sätt kan man med hjälp av public-key encryption säkert skicka information över Internet utan att någon obehörig kommer åt den.

Uppgift 3

Overflow och trunkering uppkommer på grund av att en dator lagra numeriska värden i ett begränsat minnesutrymme.

Antag att vi lagrar heltal i 8-bitar på tvåkomplementsform. Det största heltal som då kan lagras är 127, vilket representeras som bitmönstret

```
01111111
```

Om vi nu utför operationen 127 + 1 kommer vi att få overflow och resultatet vi erhåller blir -128!

```
01111111
      +1
11111111
```

Trunkering är en form av avrundning som inträffar på grund av att minnesutrymmet är begränsat. Betrakta beräkningen

```
1/3 + 1/3 + 1/3
```

Om vi har 4 decimala siffror att lagra resultaten i erhåller vi

```
0.3333 + 0.3333 + 0.3333 = 0.9999
```

Med obegränsat antal decimaler hade vi fått resultatet 1.0.

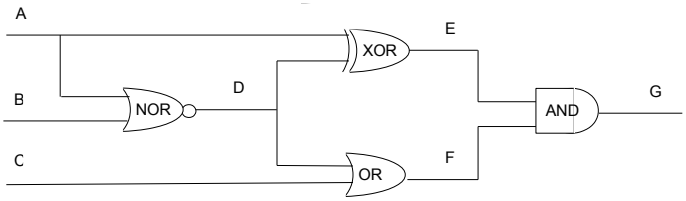
Uppgift 4

a) 9 jämförelser utförs.

b) 2 jämförelser behövs, nämligen med talen 1039 och 9876.

c) Nej! Om vi t.ex. söker efter talet 5 i den givna sekvensen behöver sekventiell sökning endast 1 jämförelse, medan binärsökning behöver 3 jämförelser (med talen 1039, 167 och 5).

Uppgift 5



A	B	C	D	E	F	G
0	0	0	1	1	1	1
0	0	1	1	1	1	1
0	1	0	0	0	0	0
0	1	1	0	0	1	0
1	0	0	0	1	0	0
1	0	1	0	1	1	0
1	1	0	0	1	0	0
1	1	1	0	1	1	1

Uppgift 6

Tabellen som erhålls får utseendet:

Hemmalag	**Bortalag**
Malmö FF	IFK Göteborg

Uppgift 7

a) Utskriften blir

```
2  3  4  5
2  3  4  5
```

b) När metoden grow anropas uppdateras inte instansvariabeln area! Felet kan korrigeras antingen genom att förändra metoden grow:

```
public void grow() {
    sideLength = 2 * sideLength;
    area = sideLength*sideLength;
}//grow
```

Man kan dock diskutera om instansvariabeln area överhuvudtaget skall finnas i klassen. Alternativet är att istället låta metoden getArea beräkna genom att använd instansvariabeln sideLength:

```
public class Square {
    private int sideLength;

    public Square(int initiallength) {
        sideLength = initiallength;
    }//constructor

    public int getArea() {
        return sideLength * sideLength;
    }//getArea

    public void grow() {
        sideLength = 2 * sideLength;
    }//grow
}//Square
```

c) Utskriften blir

```
2  3  4  5
```

d) Utskriften blir:

```
2
```

Metoden

```
methodUnknown(String str, char c)
```

räknar och returnerar antalet förekomster av tecknet c i strängen str.

Uppgift 8

```
import javax.swing.*;
import java.util.*;
public class Body{
    public static void main(String args[]) {
        boolean done = false;
        while (!done) {
            String indata = JOptionPane.showInputDialog("Ge vikt i kilo och längd i meter: ");
            if (indata == null) {
                done = true;
            }
            else {
                Scanner sc = new Scanner(indata);
                double kilo = sc.nextDouble();
                double meter= sc.nextDouble();
                if (kilo < 0 || meter < 0) {
                    JOptionPane.showMessageDialog(null, "Ogiltig indata!");
                }
                else {
                    JOptionPane.showMessageDialog(null, String.format("BMI- värdet är %.2f", bmi(kilo,meter)));
                }
            }
        }
    }//main

    public static double bmi(double kilo, double meter) {
        return kilo/(meter*meter);
    }//bmi
}//Body
```

Uppgift 9

```
public class FotballTeam {
    private String name;
    private int nrOfMatches;
    private int scoredGoals;
    private int concededGoals;
    private int points;

    public FotballTeam(String name) {
        this.name = name;
    }//constructor

    public String getName() {
        return name;
    }//getName

    public int getNrOfMatches() {
        return nrOfMatches;
    }//getNrOfMatches

    public int getScoredGoals() {
        return scoredGoals;
    }//getScoredGoals

    public int getConcededGoals() {
        return concededGoals;
    }//getConcededGoals

    public int getPoints() {
        return points;
    }//getPoints

    public void registerMatch(int scoredGoals, int concededGoals) {
        nrOfMatches = nrOfMatches + 1;
        this.scoredGoals = this. scoredGoals + scoredGoals;
        this.concededGoals = this.concededGoals + concededGoals;
        if ( scoredGoals > concededGoals)
            points = points + 3;
        else if (scoredGoals == scoredGoals)
            points = points + 1;
    }//registerMatch

    public int compareTo(FotballTeam otherTeam) {
        if (points > otherTeam.points )
            return 1;
        else if ( points < otherTeam.points)
            return -1;
        else if ((scoredGoals - concededGoals) > (otherTeam.scoredGoals - otherTeam.concededGoals))
            return 1;
        else if ( (scoredGoals - concededGoals) < (otherTeam.scoredGoals - otherTeam.concededGoals))
            return -1;
        else if (scoredGoals > otherTeam.scoredGoals )
            return 1;
        else if (scoredGoals < otherTeam.scoredGoals )
            return -1;
        else
            return 0;
    }//compareTo

    public String toString() {
        return name + "      " + nrOfMatches + "     " + scoredGoals + " - " + concededGoals
                + "    " + points;
    }//toString
}// FotballTeam
```

Uppgift 10

a)

```
public static ArrayList<FotballTeam> losers(ArrayList<FotballTeam> teams) {
    ArrayList<FotballTeam> res = new ArrayList<FotballTeam>();
    for (int i = 0; i < teams.size(); i = i + 1) {
        FotballTeam aTeam = teams.get(i);
        if (aTeam.getScoredGoals ()  - aTeam.getConcededGoals () < 0) {
            res.add(aTeam);
        }
    }
    return res;
}//losers
```

Alternativ lösning med förenklad for-sats:

```
public static ArrayList<FotballTeam> losers(ArrayList<FotballTeam> teams) {
    ArrayList<FotballTeam> res = new ArrayList<FotballTeam>();
    for (FotballTeam aTeam : teams) {
        if (aTeam.getScoredGoals ()  - aTeam.getConcededGoals () < 0) {
            res.add(aTeam);
        }
    }
    return res;
}//losers
```

b)

```
public int[] limitAmplitud(int[] samples, int limit) {
    int[] newSamples = new int[samples.length];
    for (int i = 0; i < samples.length; i = i  + 1) {
        if (samples[i] > limit)
            newSamples[i] = limit;
        else if (samples[i] < -limit)
            newSamples[i] = -limit;
        else
            newSamples[i] = samples[i]
    }
    return newSamples;
}// limitAmplitud
```

c)

```
public static int[][][] montage(int[][][] foreground, int[][][] background) {
    int[][][] newSample = new int[foreground.length][foreground[0].length][3];
    for (int row = 0; row < newSample.length; row = row + 1) {
        for (int col = 0; col < newSample[x].length; col = col + 1) {
            if (isGreen(row, col, foreground)) {
                for (int i = 0; i < 3; i = i + 1) {
                    newSample[row][col][i] = background[row][col][i];
                }
            }
            else {
                for (int i = 0; i < 3; i = i + 1) {
                    newSample[x][y][i] = foreground[x][y][i];
                }
            }
        }
    }
    return newSample;
}//montage

private static boolean isGreen(int i, int j, int[][][] image) {
    return image[i][j][0] == 0 && image[i][j][1] == 255 && image[i][j][2] == 0 ;
}//isGreen
```

Uppgift 11

a)

```
import java.awt.*;
import javax.swing.*;
import java.util.*;
public class Flag extends JPanel {
    private Color c1, c2, c3;
    public Flag() {
        setBackground(Color.WHITE);
        setForeground(Color.WHITE);
    }//konstruktor

    public void paintComponent(Graphics pen) {
        super.paintComponent(pen);
        pen.setColor(c1);
        pen.fillRect(0, 0, getWidth() / 3, getHeight());
        pen.setColor(c2);
        pen.fillRect(getWidth() / 3, 0, getWidth()/3, getHeight());
        pen.setColor(c3);
        pen.fillRect(2*getWidth() / 3, 0, getWidth()/3, getHeight());
    }//paintComponent

    public void setColors(Color c1, Color c2, Color c3) {
        this.c1 = c1;
        this.c2 = c2;
        this.c3 = c3;
    }//setColor
}//Flag
```

b)

```
import java.awt.*;
import javax.swing.*;
import java.awt.event.*;
public class ShowFlags extends JFrame implements ActionListener {
    private JButton frankrike = new JButton("Frankrike");
    private JButton italien = new JButton("Italien");
    private JButton belgien = new JButton("Belgien");
    private Flag theFlag = new Flag();
    public ShowFlags() {
        JPanel buttonPanel = new JPanel();
        buttonPanel.setLayout(new GridLayout(1,3, 10, 10));
        buttonPanel.add(frankrike);
        buttonPanel.add(italien);
        buttonPanel.add(belgien);
        frankrike.addActionListener(this);
        italien.addActionListener(this);
        belgien.addActionListener(this);
        setLayout(new BorderLayout());
        add("Center", theFlag);
        add("South", buttonPanel);
        setDefaultCloseOperation(EXIT_ON_CLOSE);
    }//konstruktor

    public void actionPerformed(ActionEvent e) {
        if (e.getSource() == frankrike)
            theFlag.setColors(Color.BLUE,Color.WHITE,Color.RED);
        else if (e.getSource() == italien)
            theFlag.setColors(Color.GREEN,Color.WHITE,Color.RED);
        else if (e.getSource() == belgien)
            theFlag.setColors(Color.BLACK,Color.YELLOW,Color.RED);
        theFlag.repaint();
    }//actionPerformed

    public static void main(String[] args) {
        ShowFlags f = new ShowFlags();
        f.setSize(325, 225);
        f.setVisible(true);
    }//main
}//ShowFlags
```