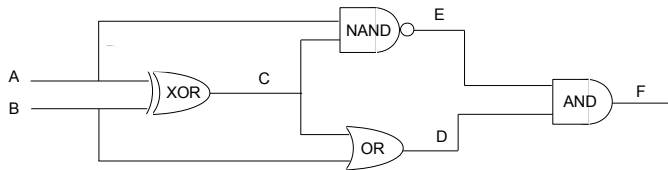


Lösningsförslag till tentamen 170314

Uppgift 1

- a) Falskt. Talområdet för ett 8-bitars tal på tvåkomplementsform är -128 till 127.
- b) Sant.
- c) Falskt, HTTP är ett kommunikationsprotokoll som används för att överföra webbsidor. Simple Mail Transfer Protocol (SMTP) är det kommunikationsprotokoll som vanligtvis används för att leverera elektronisk post.
- d) Falskt. Algoritmen har tidskomplexiteten $\Theta(n^3)$.
- e) Sant.
- f) Sant.
- g) Falskt. Man eftersträvar att eliminera all redundans.
- h) Sant.
- i) Falskt. White-box-testing (som också kallas glass-box-testing) används under programutvecklingsfasen.
- j) Sant.

Uppgift 2



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

Uppgift 3

Primärminne även kallat arbetsminne, används för att lagra olika sorters data under tiden det används och centralprocessorn har direkt åtkomst av all denna information. Sekundärminnet måste lagras i primärminnet innan centralprocessorn har tillgång till dess information till skillnad från primärminnet. Data och information i primärminnet lagras bara en kortare stund och försvinner då datorn stängs av. Primärminnet är således inget permanent minne. Vilket den därför kallas för ett flyktigt minne och när den fortsätts lagras utan att strömmen är påslagen kallas för ett icke-flyktigt minne. Vill man spara data en längre period utnyttjar man sekundärminne.

Uppgift 4

En *stack* är en datastruktur i vilken element kan sättas in och tas ut. En stack har ett LIFO- beteende, dvs endast det element som senast sattes in i stacken är nåbart och är således det enda elementet som kan tas bort från stacken.

En *kö* är en datastruktur i vilken element kan sättas in och tas ut. En kö har ett FIFO- beteende, dvs endast det element som först sattes in i kön är nåbart och är således det enda elementet som kan tas bort från kön.

Uppgift 5

Följande tabell erhålls:

Name	School
Klas Stålnacke	Hagaskolan
Clara Skoog	Stjärnaby skolan

Uppgift 6

Med *software ecosystem* menas allt det som krävs för att mjukvarusystem skall fungera, utvecklas och bli framgångsrik. Ett software ecosystem består bl.a av en mjukvaruplattform, av interna och externa utvecklare, en serviceorganisation, intressesgrupper, mm.

Uppgift 7

a)

i) Utskriften blir:

1000
1111

ii) Metoden `mystery` översätter ett positivt heltal på decimalform till binärform

b) För att jämföra om två strängar är lika måste metoden `equals` användas. Likhetsoperatoren `==`

```
public static boolean proceed() {  
    Scanner scannerObject = new Scanner(System.in);  
    while (true) {  
        System.out.println("Answer yes to continue, and no to stop.");  
        String answer = scannerObject.next();  
        if (answer.equals("yes") || answer.equals("y"))  
            return true;  
        if (answer.equals("no") || answer.equals("n"))  
            return false;  
        else  
            System.out.println("Your answer should be yes or no!");  
    }  
} //proceed
```

c) Vad kompilatorn klagar på är att metoden `Math.sqrt()` inte finns! Detta kan ju verka något mystiskt eftersom vi alla vet att det i klassen `Math` finns en klassmetod `sqrt()`. Alla vet också att klassen `Math` finns i paketet `java.lang` och att detta paket inte behöver importeras. Så varför blir det då fel?

Jo, problemet är att också den egna klassen heter `Math`!! Vid namngivning i Java gäller närhetsregeln, därför avser `Math` i anropet `Math.sqrt(tal)` den egna klassen `Math` och inte `java.lang.Math`. Kompilatorn har givetvis rätt, i den egna klassen finns ingen metod `sqrt` definierad.

För att rätta till felet kan vi således ändra namnet på den egna klassen eller specificera att det är klassen `java.lang.Math` vi avser.

Specificera namnet på metoden:

```
import javax.swing.*;  
public class Math {  
    public static void main (String[] arg) {  
        String indata = JOptionPane.showInputDialog("Ange ett tal: ");  
        double tal = Double.parseDouble(indata);  
        double root = java.lang.Math.sqrt(tal);  
        JOptionPane.showMessageDialog(null, "Roten ur talet " + tal + " är " + root);  
    } //main  
} //Math
```

Ändra namnet på klassen:

```
import javax.swing.*;  
public class AnotherMath {  
    public static void main (String[] arg) {  
        String indata = JOptionPane.showInputDialog("Ange ett tal: ");  
        double tal = Double.parseDouble(indata);  
        double root = Math.sqrt(tal);  
        JOptionPane.showMessageDialog(null, "Roten ur talet " + tal + " är " + root);  
    } //main  
} //AnotherMath
```

Uppgift 8

```
import javax.swing.JOptionPane;
import java.util.Scanner;
public class ATG {

    private static int calculateNrOf6(int[] nrOfHorses) {
        int sum = 0;
        for (int i = 0; i < nrOfHorses.length; i = i + 1) {
            sum = sum + nrOfHorses[i] - 1;
        }
        return sum;
    } //calculateNrOf6

    private static int calculateNrOf5(int[] nrOfHorses) {
        int sum = 0;
        for (int i = 0; i < nrOfHorses.length - 1; i = i + 1) {
            for (int j = i + 1; j < nrOfHorses.length; j = j + 1) {
                sum = sum + (nrOfHorses[i]-1) * (nrOfHorses[j] - 1);
            }
        }
        return sum;
    } //calculateNrOf5

    public static void main(String[] args) {
        boolean done = false;
        while (!done) {
            String indata = JOptionPane.showInputDialog("Ange antal hästar per lopp: ");
            if (indata == null)
                done = true;
            else {
                Scanner sc = new Scanner(indata);
                int[] nrOfHorses = new int[7];
                for (int i = 0; i < nrOfHorses.length; i++) {
                    nrOfHorses[i] = sc.nextInt();
                }
                String output = "Antal rader med 6 rätt: " + calculateNrOf6(nrOfHorses) + "\n";
                output = output + "Antal rader med 5 rätt: " + calculateNrOf5(nrOfHorses);
                JOptionPane.showMessageDialog(null, output);
            }
        }
    } //main
} //ATG
```

Med användning av System.in och System.out:

```
public static void main(String[] args) {
    Scanner sc = new Scanner(System.in);
    boolean done = false;
    while (!done) {
        System.out.print("Ange antal hästar per lopp: ");
        if (!sc.hasNext())
            done = true;
        else {
            int[] nrOfHorses = new int[7];
            for (int i = 0; i < nrOfHorses.length; i++) {
                nrOfHorses[i] = sc.nextInt();
            }
            System.out.println("Antal rader med 6 rätt: " + calculateNrOf6(nrOfHorses) + "\n"
                               + "Antal rader med 5 rätt: " + calculateNrOf5(nrOfHorses));
        }
    }
} //main
```

Uppgift 9

```
import java.util.ArrayList;
public class Ship {
    private String name;
    private double capacity;
    private ArrayList<Goods> cargo;

    public Ship(String name, double capacity) {
        this.name = name;
        this.capacity = capacity;
        cargo = new ArrayList<Goods>();
    } //konstruktor

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

    public double getCapacity(){
        return capacity;
    } //getCapacity

    public double getTotalWeight() {
        int total = 0;
        for (Goods g : cargo) {
            total = total + g.getWeight();
        }
        return total;
    } //getTotalWeight

    public boolean loadGoods(Goods item) {
        if (getTotalWeight() + item.getWeight() > capacity) {
            return false;
        }
        else {
            cargo.add(item);
            return true;
        }
    } //loadGoods

    public Goods unloadGoods(int id){
        for (Goods g : cargo) {
            if (id == g.getID()) {
                cargo.remove(g);
                return g;
            }
        }
        return null;
    } //unloadGoods

    public ArrayList<Goods> willBeUnloadAt(String destination){
        ArrayList<Goods> toDestination = new ArrayList<Goods>();
        for (Goods g : cargo) {
            if (g.getDestination().equals(destination)) {
                toDestination.add(g);
            }
        }
        return toDestination;
    } //willBeUnloadAt
} //Ship
```

Alternativ lösning:

```
import java.util.ArrayList;
public class Ship {
    private String name;
    private double capacity;
    private ArrayList<Goods> cargo = new ArrayList<Goods>();
    private int totalWeight = 0;

    public Ship(String name, double capacity) {
        this.name = name;
        this.capacity = capacity;
    } //konstruktor

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

    public double getCapacity(){
        return capacity;
    } //getCapacity

    public double getTotalWeight(){
        return totalWeight;
    } //getTotalWeight

    public boolean loadGoods(Goods item) {
        if (totalWeight + item.getWeight() > capacity) {
            return false;
        }
        else {
            cargo.add(item);
            totalWeight = totalWeight + item.getWeight();
            return true;
        }
    } //loadGoods

    public Goods unloadGoods(int id){
        for (int i = 0; i < cargo.size(); i = i + 1) {
            Goods item = cargo.get(i);
            if (id == item.getID()) {
                cargo.remove(item);
                totalWeight = totalWeight - item.getWeight();
                return item;
            }
        }
        return null;
    } //unloadGoods

    public ArrayList<Goods> willBeUnloadAt(String destination){
        ArrayList<Goods> toDestination = new ArrayList<Goods>();
        for (int i = 0; i < cargo.size(); i = i + 1) {
            Goods item = cargo.get(i);
            if (item.getDestination().equals(destination)) {
                toDestination.add(item);
            }
        }
        return toDestination;
    } //willBeUnloadAt
} //Ship
```

Uppgift 10

a)

```
public static int numberOfTimesOccurring(char ch, String str) {
    int count = 0;
    char lowerCh = Character.toLowerCase(ch);
    String lowerStr = str.toLowerCase();
    for (int i = 0; i < lowerStr.length(); i++) {
        if (lowerStr.charAt(i) == lowerCh) {
            count++;
        }
    }
    return count;
} //numberOfTimesOccurring
```

b)

```
public static boolean lettersOccuringSameNumberOfTimes(String str1, String str2) {
    for (int i = 0; i < str1.length(); i++) {
        char curChar = str1.charAt(i);
        if (Character.isLetter(curChar)) {
            int timesInStr1 = numberOfTimesOccurring(curChar, str1);
            int timesInStr2 = numberOfTimesOccurring(curChar, str2);
            if (timesInStr1 != timesInStr2) {
                return false;
            }
        }
    }
    return true;
} // lettersOccuringSameNumberOfTimes
```

c)

```
public static boolean isAnagram(String str1, String str2) {
    return lettersOccuringSameNumberOfTimes(str1, str2) && lettersOccuringSameNumberOfTimes(str2, str1);
} //isAnagram
```

Uppgift 11

```
public static int[][][] scale(int[][][] samples, double factor) {
    int[][][] newSamples = new int[(int) (factor*samples.length)][(int) (factor*samples[0].length)][3];
    for (int row = 0; row < newSamples.length; row = row + 1)
        for (int col = 0; col < newSamples[row].length; col = col + 1) {
            for (int rgb = 0; rgb < 3; rgb = rgb + 1) {
                newSamples[row][col][rgb] = samples[(int) (row/factor)][(int) (col/factor)][rgb];
            }
        }
    return newSamples;
} //scale
```

Uppgift 12

```
import javax.swing.*;
import java.awt.*;
import java.awt.event.*;
public class TimeTrackerPanel extends JPanel implements ActionListener {
    private JButton track = new JButton("Track");
    private JButton pause = new JButton("Pause");
    private JLabel showTime = new JLabel("", JLabel.CENTER);
    private Timer timer = new Timer(1000, this);
    private Time nrOfSeconds = new Time();
    public TimeTrackerPanel() {
        Font buttonFont = new Font("Times", Font.PLAIN, 36);
        Font labelFont = new Font("Times", Font.PLAIN, 12);
        track.setFont(buttonFont);
        pause.setFont(buttonFont);
        showTime.setFont(labelFont);
        track.setForeground(Color.BLUE);
        track.setBackground(Color.YELLOW);
        pause.setForeground(Color.BLUE);
        pause.setBackground(Color.YELLOW);
        showTime.setForeground(Color.BLUE);
        showTime.setBackground(Color.WHITE);
        showTime.setText(nrOfSeconds.toString());
        showTime.setOpaque(true);
        setLayout(new GridLayout(3, 1));
        add(showTime);
        add(track);
        add(pause);
        track.addActionListener(this);
        pause.addActionListener(this);
        pause.setEnabled(false);
    } //constructor

    public void actionPerformed(ActionEvent e) {
        if (e.getSource() == track) {
            timer.start();
            track.setEnabled(false);
            pause.setEnabled(true);
        }
        else if (e.getSource() == pause) {
            timer.stop();
            track.setEnabled(true);
            pause.setEnabled(false);
        }
        else if (e.getSource() == timer) {
            nrOfSeconds.increase();
            showTime.setText(nrOfSeconds.toString());
        }
    } //actionPerformed
} //TimeTrackerPanel
```

```
import javax.swing.JFrame;
public class TimeTracker extends JFrame {
    public TimeTracker() {
        add(new TimeTrackerPanel());
        setSize(200, 150);
        setVisible(true);
        setDefaultCloseOperation(EXIT_ON_CLOSE);
    }//constructor

    public static void main(String[] args) {
        new TimeTracker();
    }//main
}//TimeTracker
```