# Tentamen för TDA143 PROGRAMMERADE SYSTEM

DAG: 17-06-09 TID: 8:30 – 13:30

---

| | | |
|---|---|---|
| **Ansvarig**: | | Christer Carlsson, ankn 1038 |
| **Förfrågningar**: | | Christer Carlsson |
| **Resultat**: | | erhålls via Ladok |
| **Allmän info**: | | tentamen är uppdelad i två delar: del 1 omfattar översikt av datateknik och del 2 omfattar programmering |
| **Betygsgränser**: | 3:a | 10 poäng på del 1 och 26 poäng på del 2 |
| | 4:a | 13 poäng på del 1 och 36 poäng på del 2 |
| | 5:a | 16 poäng på del 1 och 48 poäng på del 2 |
| | maxpoäng | 20 poäng på del 1 och 60 poäng på del 2 |
| Siffror inom parentes: | | anger maximal poäng på uppgiften. |
| Granskning: | | Efter att resultatet meddelas i Ladok kommer tentorna att finnas på instutitionens studieexpedition. |
| Hjälpmedel: | | En valfri lärobok i Java eller de på kursen utdelade föreläsningsanteckningarna (OH-bilderna) som behandlar **programmeringsdelen** av kursen. |
| | | Förtydligande noteringar får finnas i boken resp. föreläsningsanteckningarna. |
| Var vänlig och: | | Skriv tydligt och disponera papperet på lämpligt sätt. |
| | | Börja varje uppgift på nytt blad. Skriv ej på baksidan av papperet. |
| Observera: | | Uppgifterna är ej ordnade efter svårighetsgrad. Titta därför igenom hela tentamen innan du börjar skriva. |
| | | Alla program skall vara väl strukturerade, lätta att överskåda samt enkla att förstå. |
| | | Vid rättning av uppgifter där programkod ingår bedöms principiella fel allvarligare än smärre språkfel. |
| | | På de programmeringsuppgifter som ingår i tentamenstesen kan vissa poäng erhållas även om ett fullständigt program inte redovisas, detta kräver dock att en algoritm som löser problemet presenteras. |

# LYCKA TILL!!!!

**Uppgift 1**

Denna uppgift består av 10 påståenden. Frågorna skall besvaras med ”sant” eller ”falskt”.

Ett korrekt svar ger 1 poäng, ett felaktigt svar ger -0.5 poäng och ett utelämnat svar ger 0 poäng. Totalt på uppgiften kan aldrig minuspoäng erhållas.

a) Om heltalet $-105_{10}$ lagras med 8 bitar på två-komplementsform fås bitmönstret 10010101.

b ) *Memory Manager* är den del av operativsystemet som handhar läsning och skrivning på sekundärminne.

c ) En *trojansk häst* är skadlig kod (*malware*) som infekterar andra program.

d) I ett binärt sökträd finns det största värdet i roten.

e) Valet av underliggande datastruktur är vanligtvis kritiskt för att kunna konstruera effektiva algoritmer.

f) En *E/R-modell* används för att på ett överskådligt sätt skissa designen av en databas.

g) Idag är objektorienterade databaser den vanligat typen av databaser.

h) Ett exempel på så kallad *black-box testing* kan vara att ge ut en *beta version*.

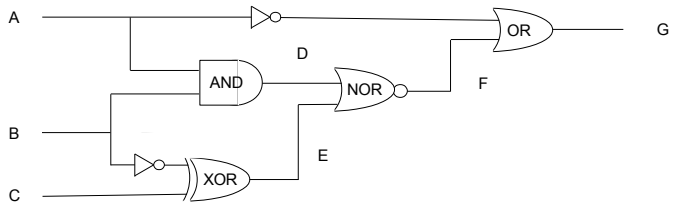
i) När man utvecklar programvara skall man, för att åstadkomma en så god systemstruktur som möjligt, eftersträva att de ingående komponenterna i systemet har *hög coupling* och *låg cohesion*.

j) 3D-rendering innebär att en bild projiceras på på ett 3D-objekt.

(10 poäng)

**Uppgift 2**

Ange sanningstabellen för den logiska kretsen nedan:




(2 poäng)

**Uppgift 3**

Tidskomplexiteten för algoritmer uttrycks ofta med Θ-notationen.

a) Förklara vad Θ-notationen innebär.

b) Ordna nedanstående tidskomplexiteter från den snabbaste till den långsammaste:

$\Theta(n)$, $\Theta(n^2)$, $\Theta(1)$, $\Theta(^2\log n)$, $\Theta(n^3)$, $\Theta(n\ ^2\log n)$

(2 poäng)

**Uppgift 4**

Beskriv hur binärsökning fungerar. Illustrera med ett exempel. Vilken tidskomplexitet har algoritmen?

(2 poäng)

**Uppgift 5**

Ange hur protokoll hierarkin i Internet är uppbyggd samt beskriv kortfattat vilken uppgift de olika lagren har.

(2 poäng)

**Uppgift 6**

Antag att du har följande databastabeller:

**NewCars**:

| RegNr | Company | Model | Year | To100 |
|---|---|---|---|---|
| GHX681 | Audi | TT RS | 2017 | 5.3 |
| CDC573 | Porsche | Cayman S | 2016 | 4.6 |
| HTY980 | Audi | S3 | 2016 | 5.5 |
| XYZ123 | BMW | Z4 sDrive35i | 2016 | 5.2 |

**OldCars**:

| RegNr | Company | Model | Year | To100 |
|---|---|---|---|---|
| THB123 | Audi | R8 5.2 V8 | 2007 | 4.2 |
| FGT051 | Porsche | Carrera 911 | 2004 | 4.7 |
| SDH931 | Porsche | Boxster S | 2012 | 5.0 |

**PriceList**

| Item | Price |
|---|---|
| THB123 | 500000 |
| CDC573 | 530000 |
| HTY980 | 350000 |
| GHX681 | 480000 |
| FGT051 | 360000 |
| SDH931 | 280000 |
| XYZ123 | 420000 |

Hur ser tabellen ut som erhålls genom nedstående SQL-query?

```
SELECT Model, Price, Year, To100
FROM NewCars, OldCars, PriceList
WHERE (Company = 'BMW' OR Company = 'Porsche') AND Price < 450000 AND RegNr = Item
```

(2 poäng)

# DEL 2: Programmeringsteknik

**Uppgift 7**

a) Vad skrivs ut när main-metoden i klassen Uppgift7a exekveras?

```
public class Uppgift7a {
    public static void main(String[] args) {
        int[] a1 = {1, 2,3,4, 5};
        mystery(a1);
        System.out.println(java.util.Arrays.toString(a1));
    }//main

    public static void mystery(int[] a) {
        for (int i = 0; i < a.length; i = i + 1) {
            a[i] = a[i] + a[i / 2];
        }
    }//mystery
}//Uppgift7a
```

(2 poäng)

b) Vad skrivs ut när main-metoden i klassen Uppgift7b exekveras?

```
public class Mystery {
    public String text;
    public static int answer;

    public Mystery(String text, int answer) {
        this.text = text;
        this. answer = answer;
    }//constructor

    public String getText() {
        return text;
    }//getText

    public int getAnswer () {
        return answer;
    }//getAnswer
}//Mystery

public class Uppgift7b {
    public static void main(String[] arg) {
        Mystery m1 = new Mystery("Vad blir 4*4?", 16);
        Mystery m2 = new Mystery("Vad blir 5*6?", 30);
        System.out.println(m1.getText() + " " + m1.getAnswer());
        System.out.println(m2.getText() + " " + m2.getAnswer());
    }//main
}//Uppgift7b
```

(2 poäng)

c) Vad skrivs ut när main-metoden i klassen Uppgift7c exekveras?

```
public class  Uppgift7c {
    public static void  main(String arg[]) {
        String s = "Det finns många problem";
        for  (int  i = 0; i < s.length(); i = i + 1) {
            char  ch = s.charAt(s.length() - 1 - i);
            ch = Character.toUpperCase(ch);
            System.out.print(ch);
        }
        System.out.println();
    }//main
}//Uppgift7c
```

(2 poäng)

d) Betrakta nedanstående klass

```
public class Wrong {
   public void main(String [] args) {
   int number =  -10 ;
        while (number <= 10) {
             if ( number %  2 == 0)
                  System.out.println(number);
             number =  number + 1 ;
        }
   }//main
}//Wrong
```

Klassen går att kompilera, men vid exekvering erhålls följande felutskrift:

```
Exception in thread "main" java.lang.NoSuchMethodError: main
```

Förklara vad som är galet!

(2 poäng)

**Uppgift 8**

På skatteverkets hemsida kan man under rubriken ”Statlig inkomstskatt, fysiska personer” läsa:

*Det är ingen statlig inkomstskatt på beskattningsbar förvärvsinkomst upp till 438 900 kr. Mellan 438 900 kr och 638 500 kr är den statliga skatten 20 procent på den beskattningsbara förvärvsinkomsten. På den beskattningsbara förvärvsinkomsten som överstiger 638 500 kr är den statliga inkomstskatten 25 procent.*

*Statlig inkomstskatt på kapitalinkomster är 30 %.*

Skriv ett program som upprepade gånger läser in beskattningsbar förvärvsinkomst samt kapitalinkomst och skriver ut den statliga inkomstskatten.

Du får själv välja om du vill göra in- och utmatning via dialogrutor eller använda System.in respektive System.out (se exemplen nedan).

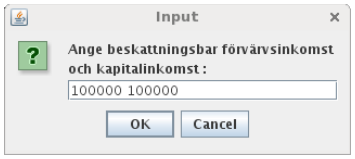
För att kunna erhålla full poäng på uppgiften:

- skall programmet utformas på så sätt att inläsningen upprepas tills användaren avbryter exekveringen (vid användning av dialogrutor genom att användaren trycker på Cancel-knappen och vid användning av System.in genom att användaren lämpligen ger ctrl z)
- skall förvärvsinkomst och kapitalinkomst läsas med en inläsningssats (dvs ett Scanner-objekt skall användas)
- skall programmet innehålla en metod

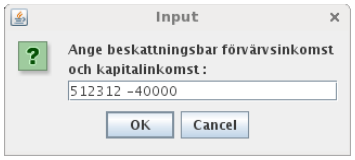
  **public static int** computeTax(**int** incomeOfSalary, **int** incomeOfCapital)

  som tar förvärvsinkomst samt kapitalinkomst och returnerar den statliga skatten
- skall felutskrift göras om användaren ger negativa indatavärden

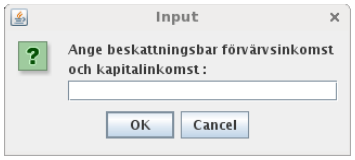
**Med användning av dialogrutor**












**Med användning av System.in resp System.out**

Ange beskattningsbar förvärvsinkomst och kapitalinkomst: 100000 100000

Inkomstskatt är: 30000 kr

Ange beskattningsbar förvärvsinkomst och kapitalinkomst: 512312 -40000

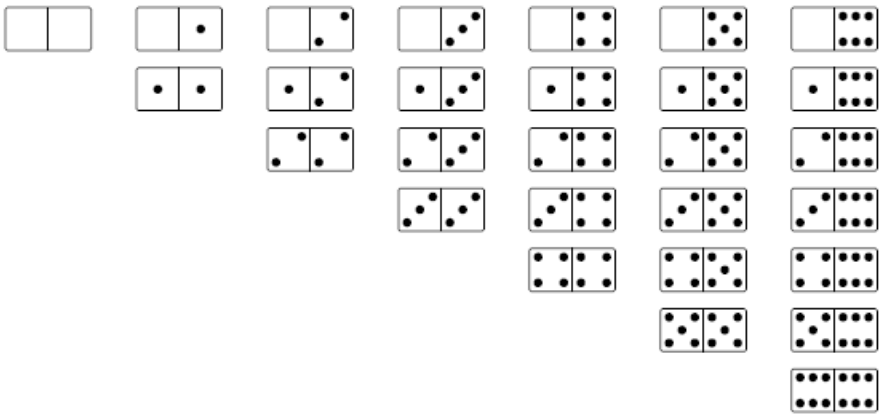
Ogiltig indata! Kan inte vara negativ!

Ange beskattningsbar förvärvsinkomst och kapitalinkomst:

(11 poäng)

**Uppgift 9**

En spelbricka i Domino är uppdelad i en vänster sida och en höger sida. Sidorna är försedda med prickar i likhet med en tärning. Antalet prickar på respektive sida kan variera mellan 0 och 6. I spelet används 28 unika brickor enligt:



För att avbilda en enskild dominobricka finns klassen Tile. Klassen innehåller följande:

| | |
|---|---|
| **public** Tile(**int** left, **int** right) | konstruktor som skapar en bricka med left prickar på brickans vänstra sida och right prickar på brickans högra sida |
| **public int** getLeft() | returnerar antalet prickar på brickans vänstra sida |
| **public int** getRight() | returnerar antal prickar på brickans högra sida |

Din uppgift är att implementera en klass DominoModel. Klassen skall innehålla följande:

| | |
|---|---|
| **private** ArrayList<Tile> tiles | instansvariabel för att lagra dominobrickor. |
| **public** DominoModel() | konstruktor som initierar listan tiles att innehålla samtliga 28 brickorna som ingår i spelet (se ovan). Obs! Endast en förekomst av varje bricka skall förekomma. Till exempel är brickan  samma bricka som brickan  (brickan är enbart roterad). |
| **public int** nrOfTiles() | returnerar antalet kvarvarande brickor i listan tiles. |
| **public** Tile pickTile() | som väljer ut och returnerar en slumpmässig bricka från listan tiles. Om ingen bricka finns kvar i tiles returneras **null**. |
| **public static boolean** match(Tile tile1, Tile tile2) | returnerar **true** om någon sida på bricka tile1 har samma antal prickar som någon sida på bricka tile2, annars returneras **false**. |

(10 poäng)

**Uppgift 10**

a) I ett visst datorsystem kräver man lösenord som uppfylla följande krav:

- lösenordet skall vara minst 8 tecken långt
- lösenordet skall innehålla minst en stor bokstav och minst en liten bokstav
- lösenordet skall innehålla minst två tecken som inte är en bokstäver

Skriv en metod

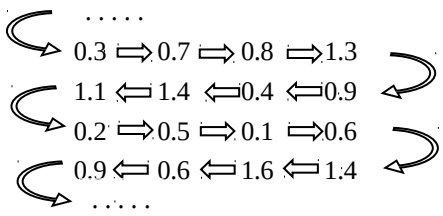
**public static boolean** isValid(String password)

som tar en sträng password och returnerar **true** om password uppfyller ovanstående villkor för att vara ett giltigt lösenord, annars returnerar metoden **false**.

Tips: Klassen Character tillhandahåller en del användbara metoder.

(6 poäng)

b) Ett teleskop avsöker en rektangulär yta av natthimmelen och samlar in data. Varje datavärde utgörs av ett reellt tal och anger den ljusmängd som teleskopet upptäcker. Teleskopet sveper fram och tillbaks över himmelen enligt den ordning som anges av figuren nedan:




Teleskopet lagrar datavärdena i ett endimensionellt fält av typen **double**. Resultatet av svepningen som visas i figuren ovan blir alltså ett fält med följande utseende:

| 0.3 | 0.7 | 0.8 | 1.3 | 0.9 | 0.4 | 1.4 | 1.1 | 0.2 | 0.5 | 0.1 | 0.6 | 1.4 | 1.6 | 0.6 | 0.9 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|

Skriv en metod

**public static double**[][] toMatrix(**double**[] scanned, **int** nrRows)

som tar ett endimensionellt fält scanned och ett heltal nrOfRows. Parametern scanned innehåller de datavärden som teleskopet lagrat under en datainsamlingssession och nrOfRows anger hur många svep som teleskopet gjort under sessionen. Metoden skall returnera ett tvådimensionellt fält som representera den ursprungliga rektangulära ytan av himmelen.

Antag att det endimensionella fältet som visas ovan har lagrats i variabeln data då skall anropet toMatrix(data, 4) returnera matrisen:

| 0.3 | 0.7 | 0.8 | 1.3 |
|---|---|---|---|
| 1.1 | 1.4 | 0.4 | 0.9 |
| 0.2 | 0.5 | 0.1 | 0.6 |
| 0.9 | 0.6 | 1.6 | 1.4 |

(7 poäng)

c) Det finns olika format för att representera färger.

RGB-formatet specificerar nivån av färgerna röd (R), grön (G) och blå (B). Nivån för varje färg anges som ett heltal i intervallet [0, 255]. RGB är standardformatet för bl.a LCD-displayer, digitala kameror och webbsidor.

CMYK-formatet specificerar nivån för färgerna cyan (C), magenta (M), gul (Y) och svart (K). Nivån för varje färg anges som ett reellt tal i intervallet[0.0, 1.0]. CMYK är det dominerande formatet vid publicering av böcker och tidskrifter.

Din uppgift är att skriva en metod

**public static double**[][][] fromRGBtoCMYK(**int**[][][] picture)

som tar en färgbild picture representerad med RGB-format och returnerar samma färgbild men med CMYK-formatet.

De matematiska formlerna för att konvertera från RGB-format till CMYK-format anges nedan:

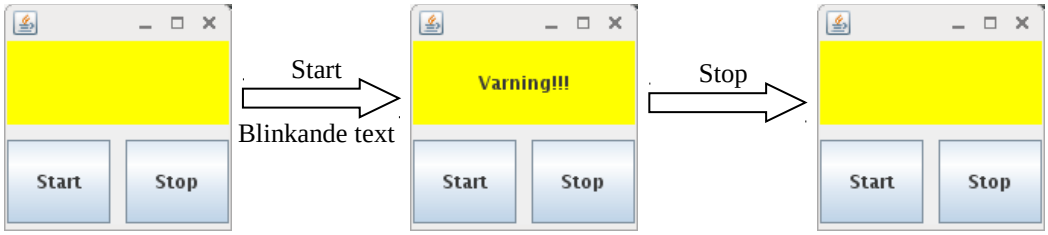
white = max{red/255, green/255, blue/255}
cyan = (white -red/255) / white
magenta = (white – green/255) / white
yellow = (white – blue/255) / white
black = (1 - white)

Tips: Math.max(x, y) returnerar det största värdet av x och y.

(7 poäng)

**Uppgift 11**

Skriv ett program som skapar ett fönster enligt figurerna nedan. Fönstret består av två knappar och en etikett. När Start-knappen trycks, börjar en text att blinka i etiketten och när Stop-knappen trycks ”släcks” texten i etiketten.




a) Börja med att skriva en klass TwinkleLable som realiserar den blinkande etiketten. Denna klass skall vara en subklass till standardklassen JLable. Ett instans av klassen TwinkleLable kan befinna sig i två olika tillstånd, blinkande eller inte blinkande. För att byta mellan dessa tillstånd skall klassen tillhandahålla instansmetoderna:

**public void** setTwinkleOn()
**public void** setTwinkleOff()

För att handha med vilken takt texten på etiketten skall blinka nyttjas en instans av klassen javax.swing.Timer. Lämpligen är texten ”tänd” under 0.5 sekunder och ”släckt” under 0.5 sekunder. Texten som blinkar på etiketten anges via konstruktorn, dvs. konstruktorn har utseendet:

**public** TwinkleLable(String text)

b) Skriv klassen ShowTwinkleLable som definiera fönstret med knapparna och den blinkande etiketten. Den blinkande texten skall vara ”Varning!!!”. Observera att du kan lösa denna deluppgift utan att du gjort förra deluppgiften genom att anta att klassen TwinkleLable redan finns.

c) Givetvis behövs också en main-metod implementeras, som skapar och visar upp en instans av klassen ShowTwinkleLable. Detta skall görs i en separat klass namet MainTwinkle.

(11 poäng)

**Uppgift 1.MÖJLIG**

Skriv en klass Book som beskriver en bok med en titel och författare. Det kan finnas flera författare till boken. Lagra författarna i en instansvariabel av typen ArrayList<String>.

Klassen skall ha två konstruktorer:

```
Book(String title, String author)
Book(String title, ArrayList<String> authors)
```

Instansvariablerna skall vara oåtkomliga utifrån klassen, istället skall det finnas instansmetoder som ger önskad information:

```
int getNrOfAuthors()
String getTitel()
ArrayList<String> getAuthors()
```

I klassen behöver inte finnas några instansmetoder som sätter tillstånd. Eftersom det kan finnas en eller två författare behövs två olika konstruktorer.

Klassen skall innehålla en metod String toString() som ger en strängrepresentation av objektet på formen:

```
Röda rummet: August Strindberg
Polis, polis, potatismos: Maj Sjövall & Per Wahlö
An Introduction to Computer Science Using Java: Kamin, Mickunas & Reingold
```

beroende på om det finns en eller flera författare.

I klassen skall även finnas en metod

Book getCopy() som ger en kopia av objektet.

```
import java.util.ArrayList;
public class Book {
   private String title;
   private ArrayList<String> authors = new ArrayList<String>();

   public Book (String title, String author) {
        this.title = title;
        authors.add(author);
   }

   public Book (String title, ArrayList<String> authors) {
        this.title = title;
        for (String author : authors)
             this.authors.add(author);
   }

   public int getNrOfAuthors() {
        return authors.size();
   }

   public String getTitle() {
        return title;
   }

   public ArrayList<String> getAuthors() {
        ArrayList<String> theAuthors = new ArrayList<String>();
        for (String author : this.authors)
             theAuthors.add(author);
        return theAuthors;
   }

   public Book getCopy() {
        return new Book(this.title, this.authors);
   }

   public String toString() {
        String output = title + ": ";
        for (int i = 0; i < authors.size(); i = i + 1) {
             output = output + authors.get(i);
             if (i < authors.size() - 2)
                  output = output + ", ";
             else if (i < authors.size() - 1)
                  output = output + " & ";
        }
        return output;
   }//toString
}//Book
```

**MÖJLIG**

Din uppgift är att skriva en metod

**public static int**[][][] frosty(**int**[][][] samples)

som tar en bild samples och returnerar en ny bild som ger effekten att se bilden samples genom en frostad glasskiva. Detta görs genom att låta färgerna i punkten (x,y) i den nya bilden tas från en annan, närliggande punkt i samples; denna punkt väljs slumpmässigt så att dess x- och y-koordinater ligger högst fem bildpunkter från x resp y.

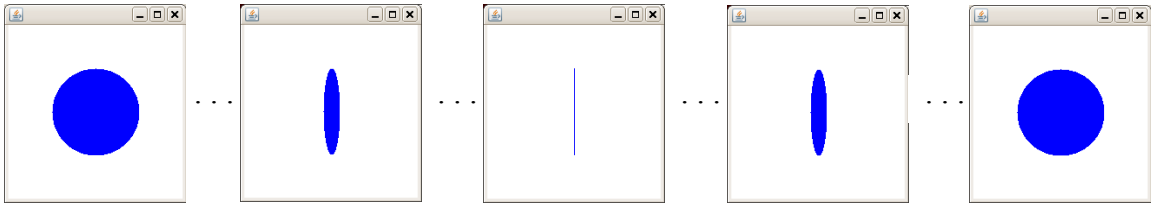
Original bild

Filtrerad bild

(8 poäng)

**MÖJLIG**

Din uppgift är att skriva ett program som visar ett fönster med en roterande cirkel, enligt bildsekvensen nedan.



a) Skriv en klass RotatingCircel som utökar JPanel och som ritar ut den roterande cirkeln. Animeringen av cirkeln görs genom att rita ut en fylld oval, vars diameter ändras allt eftersom rotationen fortskrider. Rotationshastigneten skall styras med hjälp av ett Timer-objekt.

Klassen skall ha en konstruktor

```
RotatingCircle(int size, Color color, int rotationTime)
```

där parametern size anger diametern på cirkeln (i pixels), parametern color vilken färg cirkeln skall ha och parametern rotationTime anger tiden för att rotera cirkeln ett helt varv (i millisekunder). Intervallet med vilket Timer-objekt skall generera händelser bestäms således av parametrarna size och rotationTime (låt varje händelse som Timer-objekt genereras ändrar diametern hos ovalen som ritas ut med 1 pixel).

När ett objekt av klassen RotationCircle skapas skall en hela cirkeln visas (som i den första bilden ovan). Det skall också gälla att cirkeln alltid skall ritas ut centrerad i mitten av ritytan.

b) Skriv en klass ShowRotatingCircle som skapar ett fönster som innehåller ett objekt av klassen RotatingCircle med en blå cirkel som har en diameter av 100 pixels och vars rotationstid är 2000 milisekunder. Skriv också en main-metod som visar upp fönstret (antingen i klassen ShowRotatingCircle eller i en separat klass).

c) I bildsekvensen ovan är bredden och höjden av RotatingCircel-objekt dubbelt så stor som diametern av cirkeln som ritas ut. Hur åstadkommer man detta?

Tips: Först måste man förhindra att fönstrets storlek gör att förändra.

(12 poäng)

(12 poäng)

```
import java.awt.*;
import javax.swing.*;
import java.awt.event.*;
public class RotatingCircle extends JPanel implements ActionListener {
   private int maxSize;
   private int currentSize;
   private Color color;
   private javax.swing.Timer t;
   private final int INCREASE = 0;
   private final int DECREASE = 1;
   private int status;

   public RotatingCircle(int size, Color color, int rotationTime) {
        this.color = color;
        setPreferredSize(new Dimension(2*size,2*size));
        maxSize = size;
        currentSize= size;
         t = new javax.swing.Timer(rotationTime/size, this);
        status = DECREASE;
        setBackground(Color.WHITE);
        t.start();
   }//constructor

   public void paintComponent(Graphics g) {
        super.paintComponent(g);
        int width = getSize().width;
        int height = getSize().height;
        g.setColor(color);
        g.fillOval(width/2 - currentSize/2, height/2 - maxSize/2, currentSize, maxSize);
   }//paintComponent

   public void actionPerformed(ActionEvent e){
        if (currentSize == 1)
             status = INCREASE;
        if (currentSize == maxSize)
             status = DECREASE;
        if (status == INCREASE)
             currentSize = currentSize + 1;
        else
             currentSize = currentSize - 1;
        repaint();
   }//actionPerformed
}//RotatingCircle

import javax.swing.*;
import java.awt.*;
public class ShowRotatingCircle extends JFrame {
   public ShowRotatingDisc() {
        add(new RotatingCircle(200, Color.BLUE, 2000));
        pack();
        setResizable(false);
        setVisible(true);
        setDefaultCloseOperation(EXIT_ON_CLOSE);
   }
   public static void main(String[] args){
        ShowRotatingCircle r = new ShowRotatingCircle();
   }//main
}//ShowRotatingCircle
```

c) Vad skrivs ut när nedanstående program exekveras?

```
public class Uppgift1c {
   public static void main(String[] args) {
      int[] vekt = {1, 2, 3, 4, 5};
      mystery(vekt);
      System.out.println(java.util.Arrays.toString(vekt));
   }//main

   public static void mystery(int[] arr) {
      for (int i = 0; i < arr.length; i++) {
         arr[i] = arr[i] + arr[arr.length - 1 - i];
      }
   }//mystery
}//Uppgift1c
```

(2 poäng)

d) Vad skrivs ut när nedanstående programexekveras?

```
public class Robot {
   private int energy;
   private static int nrOfGames  = 0;
   public Robot(int energy) {
      this.energy = energy;
   }
   public boolean defeat(Robot opponent) {
      nrOfGames = nrOfGames +1;
      return (energy > opponent.energy());
   }
   public int energy() {
      return energy;
   }
   public void pickEnergy(Robot opponent) {
      energy = energy + opponent.energy() / 2;
   }
   public void visa() {
      System.out.println("En robot med energi: " + energy);
   }
   public static int antalMatcher() {
      return nrOfGames;
   }
}//Robot

public class Main {
   public static void main(String[] args) {
      Robot theBeast = new Robot(40);
      Robot theKiller = null;
      for (int i = 0; i < 3; i = i + 1) {
         theKiller = new Robot(i * 20);
         if (theKiller.defeat(theBeast))
            theBeast = theKiller;
         else
            theBeast.pickEnergy(theKiller);
      }
      theBeast.visa();
      theKiller.visa();
      System.out.println("Antal matcher: " + Robot.antalMatcher());
   }//main
}//Main
```

(4 poäng)

c) Vad blir utskriften när main-metoden i klassen TestSqurae nedan exekveras?

```
public class Square {
   private int side;
   public Square(int side) {
        this.side = side;
   }
   public void setSide(int side) {
        this.side = side;
   }
   public int getArea(){
        return side * side;
   }
}//Square

public class TestSquare {
   public static void main(String args[]) {
        Square  s1 = new Square(2);
        Square  s2 = new Square(4);
        Square  s3 = s2;
        System.out.println(s1.getArea()+" "+s2.getArea() +" "+s3.getArea());
        s1 = s3;
        s3.setSide (5);
        System.out.println(s1.getArea()+" "+s2.getArea() +" "+s3.getArea());
   }
}
```

/SLÅ IHOP!!!!!

a) Vad blir utskriften när nedanstående program exekveras?

```
public class Uppgift7a {
    public static void main(String[] arg) {
        int a = 5, b = 10;
        System.out.println("a = " + a + " och b = " + b);
        swap(a, b);
        System.out.println("a = " + a + " och b = " + b);
    }//main

    public static void swap(int x, int y) {
        int temp = x;
        x  = y;
        y = temp;
    }//swap
}//Uppgift7a
```

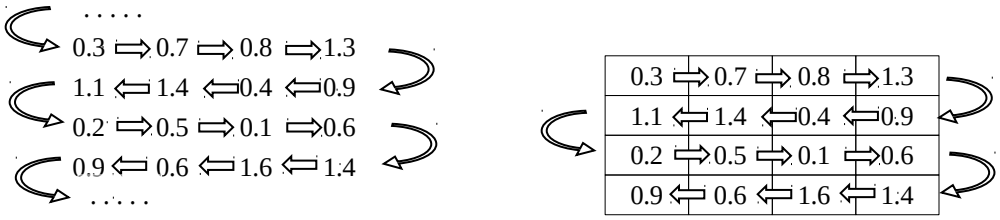
b) Vad blir utskriften när nedanstående program exekveras?

```
public class Uppgift7b {
    public static void main(String[] arg) {
        int[] v = {1, 2, 3, 4, 5};
        print(v);
        swap(0, 4, v);
        print(v);
    }//main
    public static void swap(int i, int j,  int[] f) {
        int temp =  f[i];
        f[i]  = f[j];
    f[j] = temp;
    }//swap
    public static void print(int[] f) {
        for (int i = 0; i < f.length; i = i + 1)
            System.out.print(f[i] + " ");
        System.out.println();
    } //print
}//Uppgift7b
```

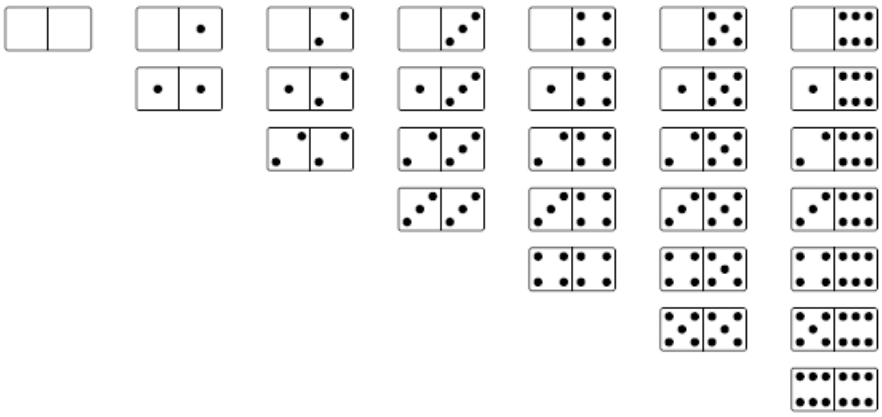
Vad skrivs ut när följande kodsegment  exekveras?

```
int number = 123;
int  newNumber = 0;
while (number != 0) {
    newNumber = newNumber*10 + number%10;
    number = number / 10;
}
System.out.println(newNumber);
```

(2 poäng)




En spelbricka i Domino är uppdelad i en vänster sida och en höger sida. Sidorna är försedda med prickar i likhet med en tärning. Antalet prickar på respektive sida kan variera mellan 0 och 6. I spelet används 28 unika brickor enligt:



För att avbilda en enskild dominobricka finns klassen Tile. Klassen innehåller följande:

| | |
|---|---|
| **public** Tile(**int** left, **int** right) | konstruktor som skapar en bricka med left prickar på brickans vänstra sida och right prickar på brickans högra sida |
| **public int** getLeft() | returnerar antalet prickar på brickans vänstra sida |
| **public int** getRight() | returnerar antal prickar på brickans högra sida |

Din uppgift är att implementera en klass DominoModel. Klassen skall innehålla följande:

| | |
|---|---|
| **private** ArrayList<DominoTile> tiles | instansvariabel för att lagra dominobrickor. |
| **public** DominoModel() | konstruktor som initierar listan tiles att innehålla samtliga 28 brickorna som ingår i spelet (se ovan). Obs! Endast en förekomst av varje bricka skall förekomma. Till exempel är brickan  samma bricka som brickan  (brickan är enbart roterad). |
| **public int** nrOfTiles() | returnerar antalet kvarvarande brickor i listan tiles. |
| **public** DominoTile pickTile() | som väljer ut och returnerar en slumpmässig bricka från listan tiles. Om ingen bricka finns kvar i tiles returneras **null**. |
| **public static boolean** match(Tile tile1, Tile tile2) | returnerar **true** om någon sida på bricka tile1 har samma antal prickar som någon sida på bricka tile2, annars returneras **false**. |

(10 poäng)

| |
|---|
| Applikationslagret |
| Transportlagret |
| Nätverkslagret |
| Länklagret |
| Fysiaka lagret |

| | | | |
|---|---|---|---|
| 0.3 | 0.7 | 0.8 | 1.3 |
| | 1.4 | 0.4 | 0.9 |
| 0.2 | 0.5 | 0.1 | 0.6 |
| 0.9 ⇦ | 0.6 | 1.6 | 1.4 |

application: supporting network applications

- FTP, SMTP, HTTP

transport: process-process data transfer

- TCP, UDP

network: routing of datagrams from source to destination

- IP, routing protocols

link: data transfer between neighboring network elements

- Ethernet, 802.11 (WiFi)

physical: bits “on the wire”