TENTAMEN I

Programmeringsteknik F1

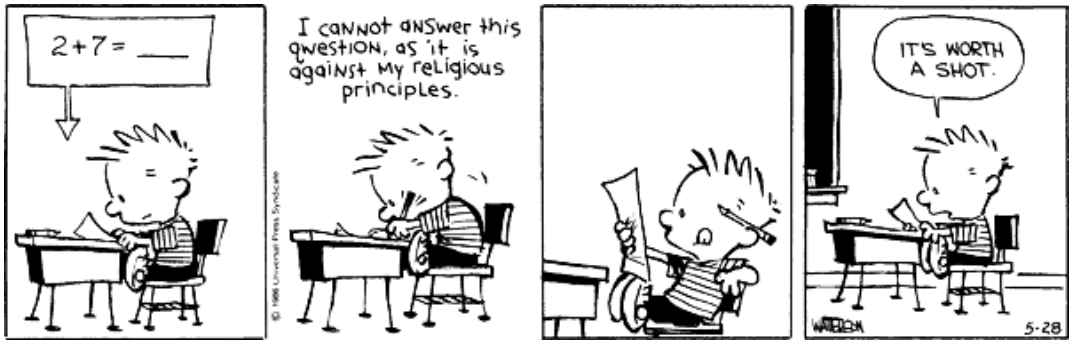
DAG: **TISDAG** TID: **14.00-18.00** SAL: **M**

Ansvarig:	Erland Holmström tel. 1007, hem 270358.
Resultat:	Anslås ej, meddelas via ladok.
Lösningar:	Anslås (eventuellt) på hemsidan samtidigt som resultatet är klart.
Granskning: kan	Tentan finns (efter anslagning av resultatet) på expeditionen där rättningen granskas. Tid för klagomål mot rättningen meddelas samtidigt som resultatet.
Betygsgränser:	**Godkänd - 26p, 4 - 36p, 5 - 46p, max=60.** (Siffror inom parentes anger maximal poäng på varje uppgift. I vissa fall ges även ungefärlig poängfördelning inom uppgiften.)
Hjälpmedel:	**Horstmann Java Concepts** ***eller*** Skansholm Java direkt om man läst kursen för Bondesson

Var vänlig och läs detta:

- Börja med att läsa igenom HELA tesen så du kan ställa frågor när jag kommer.
- Observera att svar skall motiveras där så är lämpligt.
- **Skriv läsligt!** Rita gärna figurer. Svårlästa lösningar bedöms ej!!
- Börja varje uppgift på nytt blad. (Dock ej deluppgifter). Skriv endast på en sida av pappret.
- Skriv ditt (person)nummer på **alla** blad.
- Programmen skall vara skrivna i Java och vara indenterade och kommenterade.
  I de uppgifter där det spelar någon roll, antas att man kör under UNIX-operativsystem.
- De råd och anvisningar som givits under kursen **skall** följas vid programkonstruktionerna. Det innebär bla att onödigt komplicerade, långa och/eller ostrukturerade lösningar i värsta fall ej bedöms. Programmen skall skrivas som generella enheter som är lätta att förstå (för andra än skrivaren) och lätta att ändra i när förutsättningarna ändras. De skall dessutom uppfylla normala krav på objektorienterade komponenter.

LYCKA TILL!!!



Såväl svar som motivering måste vara rätt för poäng.

Uppgift 1. a) Vad skrivs ut av följande programsnutt?

```
public class Test {
     public static void main(String[] args) {
// a1)
          boolean grabbed = false;
          if(grabbed=false) {
               System.out.println("Yes1");
          } else {
               System.out.println("No1");
          }
// a2)
          boolean sunny = false;
          boolean rich = true;
          if (sunny)
               if (rich)
                    System.out.println("Yes2");
          else
               System.out.println("No2");
     } // end main
}
```

b) Om du inte skriver en equals metod för en klass så tillhandahåller Java en. Var finns den och hur fungerar den?

Sant eller falsk? Motivera. Det är normalfall som gäller, obskyra specialfall kan du bortse från.

c) Uttrycket nedan kompilerar ok. (Antag att o inte deklarerats tidigare)

```
Object o = new Integer(5);
```

d) Uttrycket nedan evalueras till sant

```
(new Integer(11) == new Integer(11))
```

e) En statisk medlemsmetod kan referera icke statiska variabler i samma klass men endast efter att en instans av klassen skapats.

f) När ett objekt skickas som argumet till en metod så kan metoden ändra objektet.

g) Nästan alla variabler initieras till nåt men det finns variabler som inte initieras. (Var deklareras dom i så fall?)

(10p)

Uppgift 2. Förklara detaljerat vad som sker varje varv i loopen nedan och vad som skrivs ut av programmet om första anropet till slump.nextInt(20) ger talet 10?

```
import java.util.*;
public class Untitled {
     static Random slump = new Random();
     public static void newNumbers() {
          for (int i = 0; i < (slump.nextInt(20)); i++) {
               System.out.println("i= " + i);
          }
     }
     public static void main(String[] args) {
          newNumbers();
     }
}
```

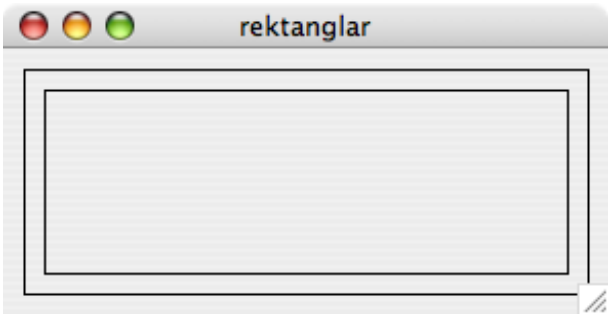
(4p)

Uppgift 3. Vad skrivs ut av följande program?

```
public class AnimalFarm {
     public static void main(String[] args) {
          final String pig = "length: 10";
          System.out.println("All Animals are equal: " + pig==pig);
     }
}
```

(3p)

Uppgift 4. Skriv ett program som ritar figuren nedan. När man drar i förstora-rutan så skall de två rectanglarna ändra storlek.



Rita i en panel (dvs en klass för detta) och skriv ett huvudprogram som *inte* får ärva en Jframe (men du skall använda en såklart). Föredragen startstorlek skall vara 340x340.

(8p)

Uppgift 5. Ett positivt heltal `N` kallas lyckligt (happy) om följande förfarande leder till talet 1:

Upprepa: `N` = summan av kvadraterna på siffrorna i `N`

T ex är 440 ett lyckligt tal, eftersom

440 -> 16+16+0 = 32 -> 9 + 4 = 13 -> 1 + 9 = 10 -> 1

Uppenbarligen är 1 ett lyckligt tal. Positiva heltal som inte leder till 1 kallas olyckliga. De leder i stället till 4 som vid fortsatt iteration ger en cykel 4 -> 16 -> 37 -> 58 -> 89 -> 145 -> 42 -> 20 -> 4.

a) Skriv en metod `int sumOfSqr(int n)` som ger efterföljaren till `n` enligt processen ovan. Dvs resultatet skall vara summan av kvadraterna på siffrorna i `n`. Det får här och i fortsättningen förutsättas att resultatet är mindre än det största tillåtna heltalet. Metoden *far inte* vara rekursiv.

b) Skriv en metod `boolean isHappy(int n)` som avgör om n är lyckligt eller ej. Det får förutsättas att antingen 1 eller 4 nås. Metoden *far inte* vara rekursiv. Utnyttja metoden ovan.

c) Gör nu a) igen men nu *skall* metoden vara rekursiv. (`sumOfSqrRek`)

d) Gör nu b) igen men nu *skall* metoden vara rekursiv. (`isHappyRek`)

e) Skriv ett fullständigt program som bestämmer och skriver ut antalet lyckliga tal mindre än 1000. Utnyttja metoderna ovan (a+b eller c+d), som naturligtvis inte behöver upprepas.

(11p)

Uppgift 6. Skriv ett litet minnesträningsprogram som när det startas visar upp ett fönster av formen:

Fönstret skall ha titeln Memory Training. Överst skall finnas en text, som från början är “Memory Training”. Därunder finns en rad med fem knappar, som innehåller talen 0-4 i en slumpmässig ordning och utan dubbletter. Underst finns tre knappar. Den högra avslutar programmet. Den högra knappens har röd bakgrund, de övriga gul.

Den mittersta gör att fönstret ser ut som vid programstart men med en ny slumpmässig blandning av siffrorna. Den vänstra döljer siffrorna. Programmet förväntar sig att användaren beter sig som följer (fusk kontrolleras inte):

1. Memorerar siffrornas placering.

2. Trycker på Hide, varvid siffrorna blir "osynliga".

3. Klickar successivt på sifferknapparna 0, 1, 2, 3 och 4 i rätt ordning. Om fel knapp trycks, skall texten ändras till “Memory Fault” (och användaren gå till nästa punkt). Om samtliga siffertangenter trycks ned korrekt skall texten i stället bli “You Made It”.

4. Trycker på New.

Till din hjälp har du en färdig klass `MTModel` se slutet.

Du skall använda M-V-C med klasserna MTControl, MTView och Main.

Fonten skall vara Times, plain, 24 punkter.

Antalet tal-knappar skall kunna varieras (se parametern till modellen). Dvs man skall kunna skapa spel med tex 10 tal att minnas så du kan inte skriva kod för en knapp i taget utan måste använda en loop till detta.

(24p)

```
import java.util.*;
public class MTModel extends Observable {
     private int[] numbers;
     private int antal = 5;  // default number of nbrs
     boolean gameOver = false;
     private int actualPosition = -1;
     private Random slump = new Random();

     public MTModel(int antal) {
          this.antal = antal;
          numbers = new int[antal];
     }

     public void newNumbers() {
          gameOver = false;
          actualPosition = antal-1;
          for (int i = 0; i < antal; i++) {
               numbers[i] = i;
          }
```

```
        // blanda
        int j, k;
        int temp;
        for (int i = 1; i <= antal; i++) {
            j = slump.nextInt(antal);
            k = slump.nextInt(antal);
            temp = numbers[j];
            numbers[j] = numbers[k];
            numbers[k] = temp;
        }
        setChanged();
        notifyObservers(new Integer(1));
    }

    public void tryNumber(int number) {
        if ( !gameOver) {
            actualPosition++;
            if ( actualPosition==number ) {
                // correct choise
                if ( actualPosition==antal-1 ) {
                    gameOver=true;
                    setChanged();
                    notifyObservers(new Integer(2));
                } else { // ok spelet fortsätter
                    setChanged();
                    notifyObservers(new Integer(3));
                }
            } else { // wrong choise
                gameOver=true;
                actualPosition--;
                setChanged();
                notifyObservers(new Integer(4));
            }
        }
    }

    public void hide(boolean hider) {
        gameOver = false; // nu kan man börja om
        actualPosition = -1;
        setChanged();
        notifyObservers(new Integer(5));
    } // end hide

    public int[] getNumbers() {
        int[] tmp = new int[antal];
        for( int i = 0; i < antal; ++i ) {
            if (numbers[i]<=actualPosition) {
                tmp[i] = numbers[i];
            } else {
                tmp[i] = -1;
            }
        }
        return tmp;
    }
    public int getAntal() {
        return antal;
    }
}
```

U1 ==========

Korrekt Motivering och rätt svar krävs.
(Inom parentes står ibland vad frågan testar)

a1) (En luring faktiskt)
(grabbed=false) är ett uttryck vars värde är false
(grabbed tilldelas false och värdet av uttrycket är false)
så det står alltså alltid if (false) vilket innebär "No1"

a2) (Semantiken för if satsen)
En else associeras alltid med sista if så indenteringen luras här.
Det borde sett ut så här:

```
if (sunny)
   if (rich)
      System.out.println("Yes2");
   else
      System.out.println("No2");
```

eller ännu hellre med block så blir det otvetydigt

```
if (sunny) {
   if (rich) {
      System.out.println("Yes2");
   } else {
      System.out.println("No2");
   }
}
```

och eftersom sunny är false så blir det ingen utskrift alls.

b) (2p) Den finns i klassen Object och den jämför om refrenserna är lika.
(likhet)

c) Ja det går bra. En Integer ärver ju Object dvs den är ett Object så en referens till ett Object kan också vara referens till en Integer. Operationerna på o är dock begränsade till de som finns på object om man inte typkonverterar. (Typer, polymorfism)

d) Nej det är två olika heltal och referenslikhet kommer att ge false.
new integer(11) är ett uttryck som returnerar en referens till en Integer vars värde är 11. 2 anrop ger olika referenser. (referens likhet, instansiering av objekt)

e) Nej, statiska metoder kan aldrig* komma åt nåt i ett objekt.
(*Här finns obskyra fall tex om en statisk metod instansierar ett objekt av sin egen klass så kommer den åt instansvariabler i det objektet)
(semantik för klasser, objekt)

f) Ja. Referensen till objektet kan inte ändras men det referensen pekar på
kan ändras (och det är objektet det). (parameteröverföring)

g) (2p) Ja. Variabler inuti metoder initieras inte. (semantik för variabler)

U2 ==========

(Semantiken för for-loopen)
Om man lägger in lite variabler så ser man tydligt vad som sker

```
    public static void newNumbers2() {
        int p;
        for (int i = 0; i < (p=slump.nextInt(20)); i++) {
            System.out.println("i= " + i + " nextInt= " + p);
        }
        System.out.println("sista p= "+ p);
    }
```

Uttrycket i for loopen evalueras varje varv och då får man ett nytt slumptal varje gång.
Det är alltså ganska osäkert vad som kommer att hända efter det första varvet,
det beror på vad slumptalsgeneratorn ger för slumptal. Några anrop:

```
    % java MystiskLoop
    i= 0 nextInt= 13
    i= 1 nextInt= 12
    i= 2 nextInt= 10
    sista p= 0
    % java MystiskLoop
    i= 0 nextInt= 9
    i= 1 nextInt= 12
    sista p= 0
    % java MystiskLoop
```

```
i= 0 nextInt= 7
i= 1 nextInt= 17
i= 2 nextInt= 19
i= 3 nextInt= 13
i= 4 nextInt= 19
i= 5 nextInt= 11
i= 6 nextInt= 10
i= 7 nextInt= 12
sista p= 7
```

U3 ==========

(Prioritetsordning på operatorer)
Eftersom + operatorn har högre prioritet än == operatorn så beräknas först
"All Animals are equal: " + pig
till strängen "All Animals are equal: length 10"
som naturligtvis inte är lika med strängen "length 10" så false skrivs ut.

U4 ==========

Den här uppgiften är en blandning av test om ni läst/hittar i boken och av om ni kan mer än skriva av dvs kan göra enkla förändringar i det som står i boken.
En stor skillnad mot bokens lösning är att rektanglarna skulle följa fönstrets storlek, en annan att man skulle använda en JPanel.

Om man då trasslar in sig i användandet av både en JComponenet (som boken gör) och en JPanel så är man illa ute såklart för då har man inte förstått att dessa två är ungefär samma sak (i det här sammanhanget märks nog ingen skillnad alls).
Sen svek mig mitt minne (jag borde ju kollat) när jag skrev att man inte fick ärva en JPanel, det skulle ju varit att man *måste* ärva en JPanel för att få lite skillnad även där men ingen blev väl lessen för det antar jag. Men poängen förskjuts naturligtvis från den del man kan skriva rakt av från boken.

```
import java.awt.*;
import javax.swing.*;

public class Rect extends JPanel {
    private static int size = 300;

    public Rect() {
        setPreferredSize(new Dimension(size+40, size+40));
    }

    public void paintComponent(Graphics g) {
    super.paintComponent(g);
        int width  = getWidth();  //avsaknad av dessa ger -4p
        int height = getHeight();
        g.drawRect(10, 10, width-20, height-20);
        g.drawRect(20, 20, width-40, height-40);
  }
}

import java.awt.*;
import javax.swing.*;
public class Main {

    public static void main(String [] args) { //(4)
        JFrame frame = new JFrame();
        frame.setTitle("rektanglar");
        frame.setLocation(50, 50);
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        frame.getContentPane().setLayout(new BorderLayout() );
        Rect bx = new Rect();
        frame.getContentPane().add(bx, BorderLayout.CENTER);
        frame.setVisible(true);
        frame.pack();
    }
}
```

U5 ==========

```
// Många alternativa lösningar är möjliga (och "rätt") men med strängar är det
// svårt att få elegans här, speciellt i de rekursiva lösningarna.

public class LyckligaTal {

   public static int sumOfSqr(int n) {
        // n antas vara >=0
        int sum = 0;
```

```
        while (n>0) {
                int ental = n%10;
                        n = n/10;
                        sum = sum + ental*ental;
        }
                return sum;
    }

        public static int sumOfSqrRek(int n) {
                if ( n<=0 ) {
                        return 0;
                } else {
                        int ental = n%10;
                        return ental*ental + sumOfSqrRek(n/10);
                }
        }

        public static boolean isHappy(int n) {
                int tmp = n;
                while ( tmp!=1 && tmp!=4 ) {
                        tmp = sumOfSqr(tmp);
                }
                return tmp==1;
        }

        public static boolean isHappyRek(int n) {
                if (n==1) {
                        return true;
                } else if (n==4){
                        return false;
                } else {
                        return isHappyRek(sumOfSqrRek(n));
                }
        }

        public static void main(String[] args) {
                int antal = 0;
                int start = 1;
                int slut = 1000;
                for(int i = start; i <= slut; ++i) {
                        //if ( isHappy(i) != isHappyRek(i) )  System.out.println("OOOps");
                        if ( isHappy(i)) {
                                System.out.println(i + " är ett lyckligt tal"); // ej nödvändig
                                antal = antal +1;
                        }
                }
                System.out.println("Det finns " + antal + " stycken lyckliga tal mellan " + start + " och " + slut);
        }
}

U6 ============================================================
public class MemoryTrainingMain {
        public static void main(String[] s) {
                MTModel m = new MTModel(5);
                MTView v = new MTView(m);
        }
}
============================================================

import javax.swing.*;
import java.awt.event.*;

public class MTControl implements ActionListener {
        MTModel m;

        public MTControl (MTModel m) {
                this.m = m;
        }

        public void actionPerformed(ActionEvent e) {

                if ( e.getActionCommand().equals("quit") ) {
                        System.exit(0);
                } else if (     e.getActionCommand().equals("newbe") ) {
                        m.newNumbers();
```

```
            } else if (     e.getActionCommand().equals("hide") ) {
                  m.hide(true);
            } else if (     e.getActionCommand().equals("avsluta") ) {
                  System.exit(0);
            } else { // assume number buttons now
                  int knapp = Integer.parseInt(butt.getActionCommand());
                  m.tryNumber(knapp);
            }
      } // end actionPerformed
}
=========================================================
import java.awt.*;
import javax.swing.*;
import java.util.*;

public class MTView extends JFrame implements Observer {

      private JButton[] siff;
      private JLabel  heading = new JLabel("MemoryTraining", JLabel.CENTER);

      public MTView(MTModel m) {
            JButton show = new JButton("Hide");
            JButton newbe = new JButton("New");
            JButton quit = new JButton("Quit");
            setTitle("Memory training");
            setLocation(50, 50);
            setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
            setLayout(new GridLayout(3,1,3,3)); // 3x1 med 3 pixlars avstånd
            Font font = new Font("Times", Font.PLAIN, 24);
            show.setFont(font); heading.setFont(font); quit.setFont(font); newbe.setFont(font);
            MTControl mtc = new MTControl(m);
            m.addObserver(this);
            int antal = m.getAntal();
            siff = new JButton[antal];
            m.newNumbers();
            int[] numbers = m.getNumbers();

            JPanel buttNumbers = new JPanel();
            buttNumbers.setLayout(new GridLayout(1, antal, 2, 2));
            for (int i = 0; i < antal; i++) {
                  siff[i] = new JButton("" + numbers[i]);
                  siff[i].setActionCommand("" + numbers[i]);
                  siff[i].addActionListener(mtc);
                  siff[i].setFont(font);
                  siff[i].setForeground(Color.BLACK);
                  siff[i].setBackground(Color.YELLOW);
                  buttNumbers.add(siff[i]);
            }
            JPanel buttPanel = new JPanel();
            buttPanel.setLayout(new GridLayout(1, 3, 2, 2));

            show.setActionCommand("hide");
            show.addActionListener(mtc);
            show.setBackground(Color.YELLOW);
            buttPanel.add(show);

            newbe.setActionCommand("newbe");
            newbe.addActionListener(mtc);
            newbe.setBackground(Color.YELLOW);
            buttPanel.add(newbe);

            quit.setActionCommand("quit");
            quit.addActionListener(mtc);
            quit.setBackground(Color.RED);
            buttPanel.add(quit);

            add(heading);
            add(buttNumbers);
            add(buttPanel);

            pack();
            setVisible(true);
      }

      public void update(Observable o, Object obj) {
```

```
        if (o instanceof MTModel) {
            MTModel m = (MTModel) o;
            int[] numbers = m.getNumbers();
            int antal = numbers.length;
            int action = ((Integer) obj).intValue(); // It better be

            switch ( action ) {
            case 1: // newNumbers
                heading.setText("Memory Training");
                for (int i = 0; i < antal; i++) {
                    if (siff[i]!=null) {
                        siff[i].setText("" + numbers[i]);
                        siff[i].setActionCommand("" + numbers[i]);
                    }
                }
            break;
            case 2: // tryNumber, game over You won
                heading.setText("You Made It");
            break;
            case 3: // game goes on
                for (int i = 0; i < antal; i++) {
                    if (numbers[i] == -1) {
                        siff[i].setText("");
                    } else {
                        siff[i].setText("" + numbers[i]);
                    }
                }
            break;
            case 4: // wrong choise
                heading.setText("Memory Fault, Start new game");
            break;
            case 5: // hide
                heading.setText("Ok lets try again");
                for (int i = 0; i < antal; i++) {
                    siff[i].setText("");
                }
            break;
            } // end switch
        } // end if
    } // end update
}
```