Laboration 7

Avsikten med laborationen är att ska skriva klasser som innehåller instansvariabler, instansmetoder och konstruktorer. Sedan ska du använda objekt av de klasser du skrivit i små program.

Skapa paketet **laboration7** innan du fortsätter med laborationen.

Uppgift 7a

Denna uppgift besvarar du på papper.

Gör ett **klassdiagram** som beskriver klassen **Circle** nedan.

```
package laboration7;

public class Circle {
    private double radius;

    public Circle() {
        this.radius = 1.0;
    }

    public Circle( double inRadius ) {
        this.radius = inRadius;
    }

    public void setRadius( double inRadius ) {
        this.radius = inRadius;
    }

    public double getRadius() {
        return this.radius;
    }

    public double area() {
        double area = Math.PI * this.radius * this.radius;
        return area;
    }

    public String toString() {
        return "CIRCLE, radius = " + this.radius;
    }
}
```

Uppgift 7b

Nedanstående program använder klassen **Circle** i Uppgift 7a. Tyvärr innehåller programmet en del kod som ger kompileringsfel (understrukna). Kompileringsfelen beror på att instansvariabeln *radius* är private-deklarerad medan programmet försöker komma åt *radius* direkt.
Börja med att skapa klassen **Circle** och klassen **CircleTest** i paketet *laboration7*. Sedan ska du lösa problemet på två sätt, först ett dåligt sätt (1 nedan) och sedan ett bra sätt (2 nedan).

1. Lösning 1 – dålig lösning
   public-deklarera *radius* i klassen Circle. Nu går det bra att kompilera och exekvera CircleTest. Du har nu fått ett exempel på hur man kommer åt *public*-deklarerade instansvariabler. Sådana förekommer då och då.
2. Lösning 2 – bra lösning
   Deklarera åter *radius* som *private*. Ersätt samtliga understrukna delar i CircleTest med metodanrop. Det är alltid med metodanrop som du alltid ska kommunicera med objekt. Teskör programmet när du åtgärdat samtliga fel i programmet och kontrollera så att körresultatet är det tänkta.

```
package laboration7;
/**
 * Programmet använder ett objekt av typen Circle. Men på felaktigt sätt.
 * Rätta till felen i programmet.
 * @author Rolf Axelsson
 */
public class CircleTest {
    public static void main(String[] args) {
        Circle c = new Circle();

        c.radius = 23.2; // setRadius eller initiera via konstruktor
        System.out.println( "CIRKEL med radie = " + c.radius ); // getRadius
        System.out.println( "CIRKEL med area = " + Math.PI * c.radius * c.radius );
// båda understrykningarna och lite till kan ersättas med area
        c.radius = c.radius + 5.7; // setRadius och getRadius, 1 el 2 rader med kod
        System.out.println( "CIRCLE, radius = " + c.radius ); // toString
    }
}
```

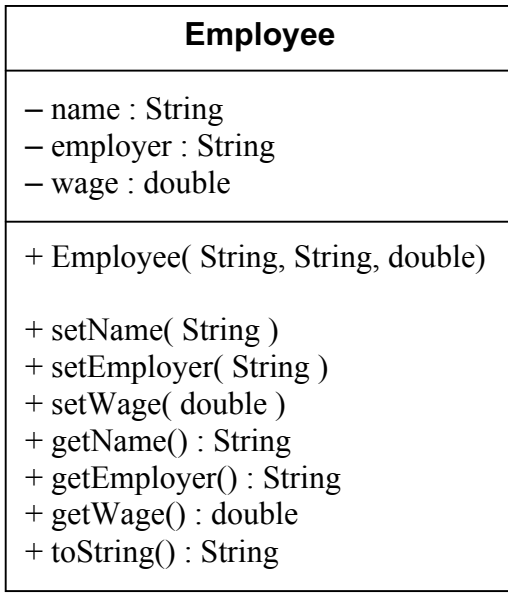
Körresultat

```
CIRKEL med radie = 23.2
CIRKEL med area = 1690.9308298681701
CIRKEL med radie = 28.9
```

Uppgift 7c

Du ska skapa en klass **Employee**. Klassdiagrammet till höger beskriver vad klassen ska innehålla. Diagrammet visar:

- Klassens namn
- Klassens attribut (instansvariabler). Minus-tecknet före namnet anger att variabeln ska vara private.
- Klassens konstruktorer och metoder (instansmetoder). Plus-tecknet före namnet anger att metoden ska vara public.



Hämta **EmployeeTest.java** från kurssidan och exekvera programmet.

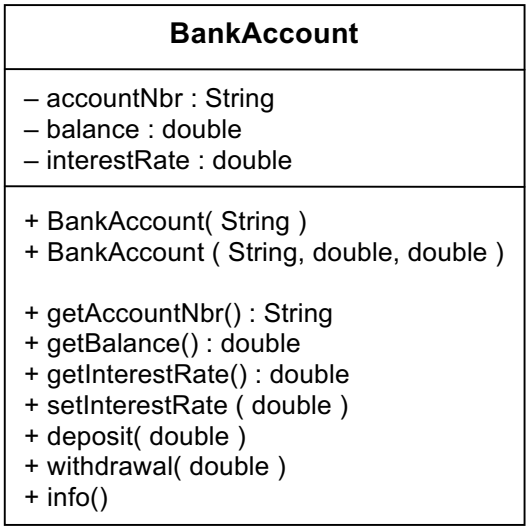
Var god vänd!

Körresultatet ska bli:

```
Namn: Ergil Tuveberg
Arbetar hos: Renlighet AB
Lön: 27000.0
EMPLOYEE: name = Ergil Tuveberg, employer = Renlighet AB, wage = 27000.0
```

Uppgift 7d

Klassdiagrammet till höger beskriver klassen **BankAccount**. Skriv klassen BankAccount i paketet *laboration7*.

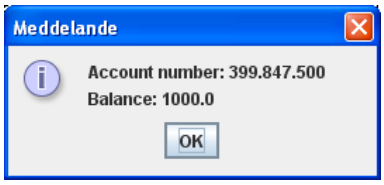


- **Konstruktorn** *med en parameter* (String accountNbr) initierar accountNbr till angivet värde. *balance* ska initieras till 0.0. *interestRate* ska initieras till 0.005.
  Konstruktorn *med tre parametrar* för över värdena till instansvariablerna.
- **get**-metoderna och **set**-metoden är som de brukar vara.
- Metoden **deposit** ska öka *balance* med parameterns värde. t.ex.

```
Konto konto = new Konto();
konto.setSaldo( 1000 );
konto.insättning( 1250 );
System.out.println( konto.getSaldo() + " kr" );
```

  ger utskriften: `2250.0 kr`

- Metoden **withdrawal** ska på liknande sätt minska *balance* med parameterns värde.
- Metoden **info** ska visa ett fönster liknande det till höger.



Hämta **BankAccountTest.java** från hemsidan. Om du kör programmet ska du få följande resultat:

Fråga 1: Hur ändrar man innehållet i *accountNbr* i ett BankAccount-objekt? Ändringen ska ske från klassen BankAccountTest.

Fråga 2: Hur ändrar man innehållet i *balance* i ett BankAccount-objekt? Ändringen ska ske från klassen BankAccountTest.

Uppgift 7e

Gör ett klassdiagram och skriv en klass som representerar ett bord (table).

Viktiga egenskaper för ett bord är *material (material)*, *bredd (width), djup (depth)* och *höjd (height)*. Välj själv lämpliga variabeltyper. Samtliga instansvariabler ska deklareras som private

Viktiga metoder för ett bord är:

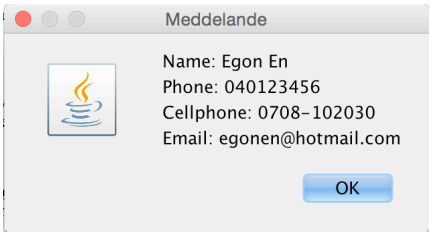
- *set-metoder* och *get-metoder* för samtliga egenskaper
- *size* vilken returnerar resultatet av (bredd · djup)
- *toString* som returnerar en sträng på formen.
  ”Table [material = trä, width = 120, depth = 70, height = 72]”

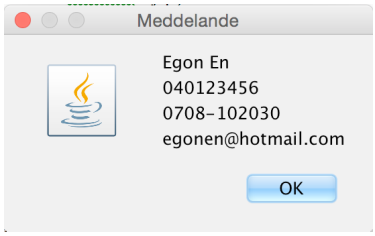
Uppgift 7f

Gör ett program som använder **Table**-klassen du skrivit i Uppgift 7e. Programmet ska anropa samtliga metoder i klassen Table.

Uppgift 7g

I programmet **ContactTest** (se nedan) används ett objekt av typen **Contact**. Skriv klassen **Contact** så att programmet fungerar. Testkör programmet **ContactTest**. Fönsterna nedan visar sig om användaren matar in ”Egon En”, ”040-123456”, ”0708-102030” och ”egonen@hotmail.com”:






```
package laboration7;
import javax.swing.*;

public class ContactTest {
    public void action() {
        Contact contact = new Contact();
        String name, phone;
        name = JOptionPane.showInputDialog("Ange namn");
        contact.setName(name);
        phone = JOptionPane.showInputDialog("Ange hemtelefon");
        contact.setPhone(phone);
        contact.setCellphone(JOptionPane.showInputDialog("Ange mobil"));
        contact.setEmail(JOptionPane.showInputDialog("Ange mail-adress"));
        JOptionPane.showMessageDialog(null, contact.toString());
        JOptionPane.showMessageDialog(null, contact.getName() + "\n" +
                                            contact.getPhone() + "\n" +
                                            contact.getCellphone() + "\n" +
                                            contact.getEmail());
    }

    public static void main(String[] args) {
        ContactTest prog = new ContactTest();
        prog.action();
    }
}
```

Uppgift 7h

Gör ett klassdiagram och skriv en klass som representerar ett hus (House).

Viktiga egenskaper för ett hus är *byggnadsår (year)*, *bostadsyta (size)* och *tomtyta (garden)*. Välj själv lämpliga variabeltyper. Samtliga instansvariabler ska deklareras som private.

Klassen ska innehålla en konstruktor där användaren kan ange värden på samtliga egenskaper.

Viktiga metoder för en villa är:

- *set-metod* till bostadyta och *get-metoder* för samtliga egenskaper
- *toString* som returnerar en sträng på formen.
  ”Hus byggt 1972 med 142 kvm bostadsyta. Tomten är 620 kvm stor.”

Gör ett program som använder **House**-klassen du skrivit. Programmet ska anropa samtliga metoder i klassen House.

Uppgift 7i

Inmatning från användaren via dialogfönster behövs i många program. T.ex. så kan inmatning av ett heltal ske via instruktionerna:

```
int res = Integer.MIN_VALUE;
String str = JOptionPane.showInputDialog( "Ett meddelande" );
if(str!=null && str.length()>0) {  // användaren klickade inte på Avbryt
    res = Integer.parseInt( str ); // och det finns inmatade tecken
}
```

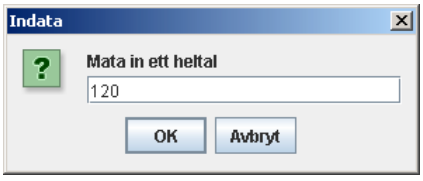
Instruktionerna ovan klarar av att användaren klickar på *Avbryt* och att användaren inte matar in några tecken innan klick på *OK*. Om något av detta inträffar så kommer *res* ha det minsta värdet som kan lagras i en int, nämligen konstanten *Integer.MIN_VALUE*.

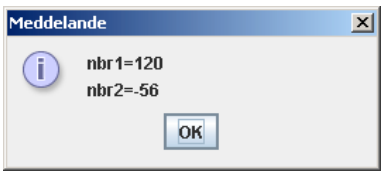
Instruktionerna ovan klarar däremot inte av om användaren matar in tecken som inte kan översättas till en *int*. Det ska vi åtgärda senare på kursen.

Placera klasserna ***Input*** respektive ***InputTest*** i paketet ***laboration7***. I klassen *Input* ser du klassmetoden ***getInt***, vilken låter användaren mata in ett heltal.

Studera *main*-metoden i klassen *InputTest* och exekvera sedan main-metoden.

Två dialoger låter användaren mata in totalt två heltal. Sedan skrivs de inmatade talen ut i en tredje dialog.





Testa programmet med olika inmatningar för att se vad som händer.

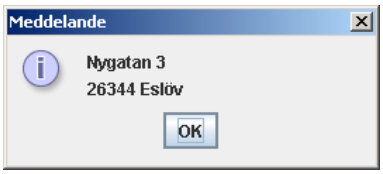
Lägg till klassmetoderna ***getLong*** respektive ***getDouble*** i klassen Input. Metoderna ska fungera på samma sätt som *getInt* men låta användaren mata in en *long* respektive en *double*. Testa metoderna så de fungerar bra.

När du skriver *getLong* så kan du intitera *res* med konstanten *Long.MIN_VALUE*.
När du skriver *getDouble* så kan du initiera *res* med konstanten *Double.NaN*.

Uppgift 7j

Skriv klassen **Address** så den överensstämmer med klassdiagrammet till höger.

Kör programmet **AddressTest** och kontrollera att du får körresultatet nedan. Av programmet framgår bl.a. hur *toString*-metoden ska skrivas.



och därefter en dialog med den adress du matar in under körningen, t.ex.

Address
– street : String – postalCode : int – town : String
+ Address(String,int,String) + getStreet() : String + getPostalCode() : int + getTown() : String + toString() : String

Uppgift 7k

PP 5.6 Design and implement a class called `Box` that contains instance data that represents the height, width, and depth of the box. Also include a `boolean` variable called `full` as instance data that represents if the box is full or not. Define the `Box` constructor to accept and initialize the height, width, and depth of the box. Each newly created `Box` is empty (the constructor should initialize `full` to false). Include getter and setter methods for all instance data. Include a `toString` method that returns a one-line description of the box. Create a driver class called `BoxTest`, whose `main` method instantiates and updates several `Box` objects.

driver class – testklass, program som anropar olika metoder i en eller flera klasser.

Uppgift 7m

PaintWindow innehåller konstruktorn

```
public PaintWindow(ImageIcon background)
```

Om du använder konstruktorn så kommer den bifogade bilden bli bakgrundsbild på ritytan och bildens storlek bestämmer ritytans storlek (dock max 800x800 pixlar).

PaintWindow innehåller metoderna *getBackgroundWidth()* och *getBackgroundHeight()*. Metoderna returnerar bredd respektive höjd på bakgrundsbilden.

Programmet ***Exercise7m*** använder klassen ***Picture*** för att lagra information om en bild. Klassdiagrammet till höger beskriver klassen ***Picture***.

Picture
- icon : ImageIcon - x : int - y : int - dx : int - dy : int
+Picture(ImageIcon i,int x,int y) +Picture(ImageIcon i, int x, int y, int dx, int dy)
+ get-metoder och set-metoder

Skriv klassen ***Picture*** och kör sedan programmet ***Exercise7m***. Ett fönster liknande fönstret till höger ska visa sig. Det förutsätter att du:

- Har bilderna gubbe.jpg, liggandegubbe.jpg och sommar.jpg i katalogen *images* i projektkatalogen
- *PaintWindow* finns tillgänglig i projektet

Studera följande frågor:

- Vad innehåller en instans av klassen Picture? Vilka är instansvariablerna?
- Hur instansieras ett Picture-objekt i Uppgift 7m? Vilka parametrar har konstruktorn?
- Hur läggs en bild till i fönstret?

Uppgift 7n

Om du kör ***Exercise7n*** så kommer den liggande gubben att röra sig åt höger över fönstret, ”studsa” vid högerkanten och sedan röra sig till vänster och försvinna förbi vänsterkanten.

Din uppgift är att se till att

1. Gubben även studsar vid vänsterkanten
2. Initiera dy till -1 och se till att gubben studsar mot övre kanten respektive undre kanten i fönstret.

Om du vill kan du lägga till kod för att även flytta den stående gubben.

Uppgift 7o

Exercise 7o är en tänkbar lösning på ***Exercise 7n***. Vissa förändringar i systemet ska göras:

1. Komplettera klassen ***Picture*** med metoden ***move()***, vilken ska ändra värdet i instansvariablerna x och y med hjälp av dx (t.ex. x = x + dx;) och dy.
   Ändra sedan nedanstående rader i Exercice 7o med anropet: ***man1.move();***

```
x = man1.getX() + man1.getDx();
man1.setX(x);
y = man2.getY() + man2.getDy(); // bort
man2.setY(y);
```

2. Komplettera metoden ***changeDiredtion(Picture picture, …)*** så att dx i bifogat picture-objekt ändras om picture-objektets x-värde är mindre än minX eller större än maxX. Gör på samma sätt med dy.
   Ersätt sedan nedanstående rader i Exercise 7o med anropet:
   changeDirection(man1,0,maxX,0.maxY);

```
if(man1.getX()<0 || man1.getX()>=maxX) {
    dx = man1.getDx();
    man1.setDx(-dx);
}
if(man1.getY()<0 || man1.getY()>=maxY) {
    dy = man1.getDy();
    man1.setDy(-dy);
}
```

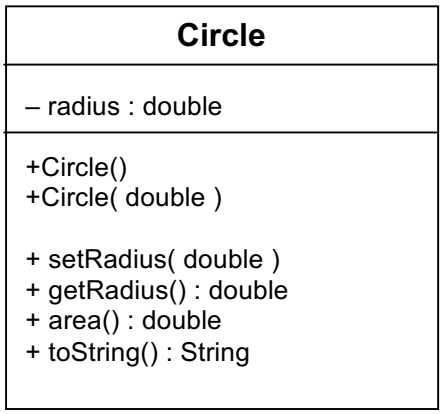
Nu ska while-loopen i programmet se ut så här:

```
while(true) {
    man1.move();
    showPicture(man1);
    changeDirection(man1, 0, maxX, 0, maxY);
    PaintWindow.pause(20);
}
```

Nu återstår det en ändring i programmet. Lägg till kod så att även den andra bilden studsar runt. Sätt t.ex. dx till 2 och dy till -3.

Lösningsförslag

Uppgift 7a



Uppgift 7b

```
package laboration7;
import javax.swing.*;

public class CircleTest {
    public static void main(String[] args) {
        Circle c = new Circle( 23.2 );

        System.out.println( "CIRKEL med radie = " + c.getRadius() );
        System.out.println( "CIRKEL med area = " + c.area() );
        c.setRadius( c.getRadius() + 5.7 );
        System.out.println( c.toString() );
    }
}
```

Uppgift 7c

```
package laboration7;

public class Employee {
    private String name;
    private String employer;
    private double wage;

    public Employee(String name, String employer, double wage) {
        this.name = name;
        this.employer = employer;
        this.wage = wage;
    }

    public void setName( String name ) {
        this.name = name;
    }

    public void setEmployer( String employer ) {
        this.employer = employer;
    }

    public void setWage( double wage ) {
        this.wage = wage;
    }

    public String getName() {
```

```
        return this.name;
    }

    public String getEmployer() {
        return this.employer;
    }

    public double getWage() {
        return this.wage;
    }

    public String toString() {
        return "EMPLOYEE: name = " + this.name + ", employer = " + this.employer +
", wage = " + this.wage;
    }
}
```

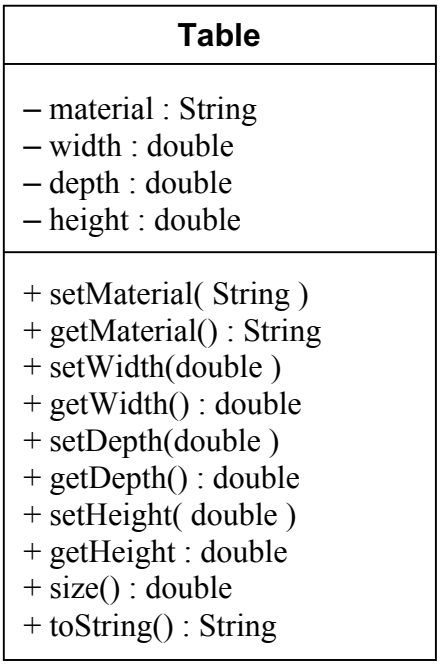
Uppgift 7d

```
package laboration7;
import javax.swing.JOptionPane;

public class BankAccount {
    private String accountNbr;
    private double balance;
    private double interestRate;

    public BankAccount(String accountNbr) {
        this.accountNbr = accountNbr;
        this.interestRate = 0.005;
    }

    public BankAccount(String accountNbr, double balance, double interestRate) {
        this.accountNbr = accountNbr;
        this.balance = balance;
        this.interestRate = interestRate;
    }

    public String getAccountNbr() {
        return accountNbr;
    }

    public double getBalance() {
        return balance;
    }

    public double getInterestRate() {
        return interestRate;
    }

    public void setInterestRate(double interestRate) {
        this.interestRate = interestRate;
    }

    public void deposit(double amount) {
        this.balance = this.balance + amount;
    }

    public void withdrawal(double amount) {
        this.balance = this.balance - amount;
    }

    public void info() {
        String message = "Account number: " + this.accountNbr + "\n" +
                         "Balance: " + this.balance;
        JOptionPane.showMessageDialog(null, message);
    }
}
```

Fråga 1:
Det går inte att ändra *accountNbr* efter att objektet av typen BankAccount har skapats.

Fråga 2:
Man ändrar *balance* genom anrop till någon av metoderna *deposit* eller *withdrawal*.

Uppgift 7e




```
package laboration7;

public class Table {
    private String material;
    private double width;
    private double depth;
    private double height;

    public String getMaterial() {
        return this.material;
    }

    public void setMaterial(String inMaterial) {
        this.material = inMaterial;
    }

    public double getWidth() {
        return this.width;
    }

    public void setWidth(double inWidth) {
        this.width = inWidth;
    }

    public double getDepth() {
        return this.depth;
    }

    public void setDepth(double inDepth) {
        this.depth = inDepth;
    }

    public double getHeight() {
        return this.height;
    }

    public void setHeight(double inHeight) {
        this.height = inHeight;
    }

    public double size() {
        return this.width * this.depth;
    }

    public String toString() {
        return "Table [ material = " + this.material + ", width = " +
               this.width + ", depth = " + this.depth + ", height = " +
               this.height + " ]";
    }
}
```

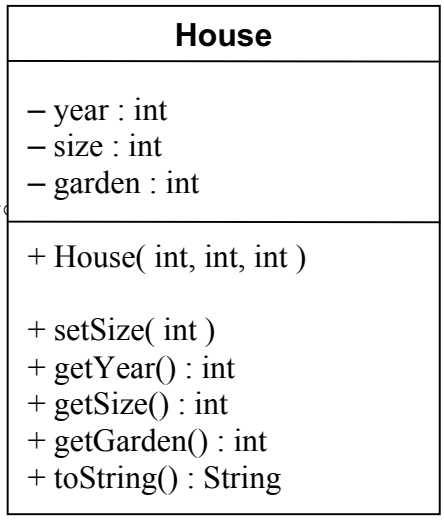
Uppgift 7g

```
package laboration7;

/**
 * Klassen innehåller kontaktinformation till en person
 * @author Rolf
 */
public class Contact {
```

```
    private String name;
    private String phone;
    private String cellphone;
    private String email;

    public void setName( String inName ) {
        this.name = inName;
    }

    public void setPhone( String inPhone ) {
        this.phone = inPhone;
    }

    public void setCellphone( String inCellphone ) {
        this.cellphone = inCellphone;
    }

    public void setEmail( String inEmail ) {
        this.email = inEmail;
    }

    public String getName() {
        return this.name;
    }

    public String getPhone() {
        return this.phone;
    }

    public String getCellphone() {
        return this.cellphone;
    }

    public String getEmail() {
        return this.email;
    }

    public String toString() {
        String res = "Name: " + this.name +
                     "\nPhone: " + this.phone +
                     "\nCellphone: " + this.cellphone +
                     "\nEmail: " + this.email;
        return res;
    }
}
```

Uppgift 7h

```
package laboration7;

public class House {
    private int year;
    private int size;
    private int garden;

    public House( int year, int sizeOfHouse, int sizeOfGar
        this.setYear(year);
        this.setSize(sizeOfHouse);
        this.setGarden(sizeOfGarden);
    }

    public void setSize(int sizeOfHouse) {
        this.size = sizeOfHouse;
    }

    public int getYear() {
        return this.year;
    }
```



```
    public int getSize() {
        return this.size;
    }

    public int getGarden() {
        return this.garden;
    }

    public String toString() {
        return "Hus byggt " + this.year + " med " + this.size + " kvm bostadsyta.
Tomten är " + this.garden + " kvm stor.";
    }
}
```

Uppgift 7i

```
package laboration7;
import javax.swing.JOptionPane;

public class Input {
    public static int getInt( String message ) {
        int res = Integer.MIN_VALUE;
        String str = JOptionPane.showInputDialog( message );
        if(str!=null && str.length()>0) {
            res = Integer.parseInt( str );
        }
        return res;
    }

    public static long getlong( String message ) {
        long res = Long.MIN_VALUE;
        String str = JOptionPane.showInputDialog( message );
        if(str!=null && str.length()>0) {
            res = Long.parseLong( str );
        }
        return res;
    }

    public static double getDouble( String message ) {
        double res = Double.NaN;
        String str = JOptionPane.showInputDialog( message );
        if(str!=null && str.length()>0) {
            res = Double.parseDouble( str );
        }
        return res;
    }
}
```

Uppgift 7j

```
package laboration7;

public class Address {
    private String street;
    private int postalCode;
    private String town;

    public Address(String street, int postalCode, String town) {
        super();
        this.street = street;
        this.postalCode = postalCode;
        this.town = town;
    }
    public String getStreet() {
        return street;
    }
    public int getPostalCode() {
        return postalCode;
    }
```

```
    public String getTown() {
        return town;
    }
    public String toString() {
        return this.street + "\n" + this.postalCode + " " + this.town;
    }
}
```

Uppgift 7k

```
package laboration7;

public class Box {
    private double height;
    private double width;
    private double depth;
    private boolean full;

    public Box(double height, double width, double depth) {
        this.height = height;
        this.width = width;
        this.depth = depth;
        this.full = false;
    }

    public double getHeight() {
        return height;
    }

    public void setHeight(double height) {
        this.height = height;
    }

    public double getWidth() {
        return width;
    }

    public void setWidth(double width) {
        this.width = width;
    }

    public double getDepth() {
        return depth;
    }

    public void setDepth(double depth) {
        this.depth = depth;
    }
    public String toString() {
        return "Box [width="+this.width+", depth="+this.depth+",
height="+this.height+", full="+this.full+"]";
    }
}
```

Uppgift 7m

```
package laboration7;
import javax.swing.ImageIcon;

public class Picture {
    private ImageIcon icon;
    private int x;
    private int y;
    private int dx;
    private int dy;

    public Picture(ImageIcon icon, int x, int y) {
```

```
        this.icon = icon;
        this.x = x;
        this.y = y;
    }

    public Picture(ImageIcon icon, int x, int y, int dx, int dy) {
        this.icon = icon;
        this.x = x;
        this.y = y;
        this.dx = dx;
        this.dy = dy;
    }

    public ImageIcon getIcon() {
        return icon;
    }

    public void setImage(ImageIcon icon) {
        this.icon = icon;
    }

    public int getX() {
        return x;
    }

    public void setX(int x) {
        this.x = x;
    }

    public int getY() {
        return y;
    }

    public void setY(int y) {
        this.y = y;
    }

    public int getDx() {
        return dx;
    }

    public void setDx(int dx) {
        this.dx = dx;
    }

    public int getDy() {
        return dy;
    }

    public void setDy(int dy) {
        this.dy = dy;
    }
}
```

Uppgift 7n

```
public class Exercise7n {
    :
    public void movePictures() {
        int x,dx;
        int y,dy;
        int maxX = window.getBackgroundWidth()-man1.getIcon().getIconWidth();
        int maxY = window.getBackgroundHeight()-man2.getIcon().getIconHeight();
        while(true) {
            x = man1.getX() + man1.getDx();
            man1.setX(x);
```

```
            y = man1.getY() + man1.getDy(); // bort
            man1.setY(y); // bort

            showPicture(man1);
            showPicture(man2);

            if(man1.getX()<0 || man1.getX()>=maxX) {  // bort efter ||
                dx = man1.getDx();
                man1.setDx(-dx);
            }
             // bort
            if(man1.getY()<0 || man1.getY()>=maxY) {
                dy = man1.getDy();
                man1.setDy(-dy);
            }

            PaintWindow.pause(20);
        }
    }
    :
}
```

Uppgift 7o

```
public class Picture {
    :
    public void move() {
        this.x += this.dx;
        this.y += this.dy;
    }
    :
}


public class Exercise7o {
    :
    public void movePictures() {
        int x,dx;
        int y,dy;
        int maxX = window.getBackgroundWidth()-man1.getIcon().getIconWidth();
        int maxY = window.getBackgroundHeight()-man2.getIcon().getIconHeight();
        while(true) {
            man1.move();
            man2.move();

            showPicture(man1);
            showPicture(man2);

            changeDirection(man1, 0, maxX, 0, maxY);
            changeDirection(man2, 0, maxX, 0, maxY);

            PaintWindow.pause(20);
        }
    }

    public void changeDirection(Picture picture, int minX, int maxX, int minY, int
maxY) {
        if(picture.getX()<minX || picture.getX()>maxX) {
            picture.setDx(picture.getDx()*(-1));
        }
        if(picture.getY()<minY || picture.getY()>maxY) {
            picture.setDy(picture.getDy()*(-1));
        }
    }
    :
}
```