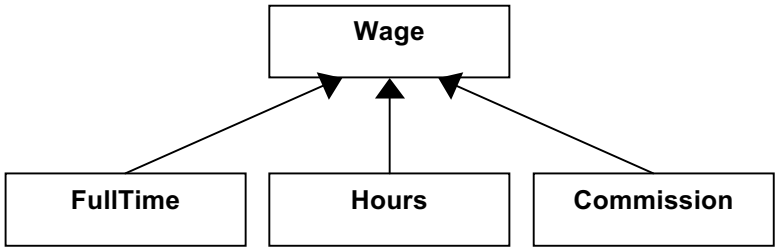
Laboration 14 - Abstrakt klass, arv grafisk komponent

Avsikten med laboration 14 är att du ska träna på att ärva abstrakt klass och ärva grafisk komponent. Sist i laborationen är det facit över uppgifterna. Konsultera detta när du är färdig med uppgiften.
Placera java-filerna i paketet ***laboration14***. Placera bildfilerna (jpg) i katalogen *images* i projektmappen.

Grundläggande uppgifter

Uppgift 1

I ett företag avlönas de anställda enligt tre olika modeller – fast månadslön, timanställning och provision på försäljning. Företaget använder en klass för varje lönemodell och dessa klasser ärver den abstrakta klassen ***Wage***:




```
package laboration14;

public abstract class Wage {
    private long id;

    public Wage( long id ) {
        this.id = id;
    }

    public long getId() {
        return this.id;
    }

    public String toString() {
        return "Id: " + this.id + ", lön denna månad: " + wage() + " kr";
    }

    public abstract double wage();
}
```

I Uppgift 1 ska du skriva klasserna ***FullTime***, ***Hours*** och ***Commission***.

Uppgift 1a

En fast månadslön representeras av klassen ***FullTime***. Klassen ska ärva ***Wage*** och ska innehålla:

Instansvariabler

wage : double

Konstruktorer

FullTime(long id, double wage)

Metoder

public void setWage(double)
public double getWage ()
public double wage(); // returnerar månadslönen

FullTime extends Wage
–wage : double
+FullTime(long, double) +setWage(double) +getWage() : double +wage() : double

Testa klassen *FullTime* med följande program:

```
FullTime employee = new FullTime( 19938278, 21500 );
System.out.println( employee.toString() );
System.out.println( "Anställd med id " + employee.getId() +
                    " har månadslönen " + employee.getWage() + " kr");
employee.setWage( 22250 );
System.out.println( "Lön denna månad: " + employee.wage() + " kr");
```

Körresultatet blir följande:

```
Id: 19938278, lön denna månad: 21500.0 kr
Anställd med id 19938278 har månadslönen 21500.0 kr
Lön denna månad: 22250.0 kr
```

Uppgift 1b

Timavlöning representeras av klassen ***Hours***. Klassen ska ärva ***Wage*** och ska innehålla:

Instansvariabler

hourlyWage : double
hours : double

Konstruktorer

Hours(long id, double hourlyWage)

Metoder

public void setHourlyWage(double)
public double getHourlyWage()
public void setHours(double)
public double getHours() : double
public double wage(); // returnerar (hourlyWage * hours)

Hours extends Wage
–hourlyWage : double –hours : double
+Hours(long, double) +setHourlyWage(double) +getHourlyWage() : double +setHours(double) +getHours() : double +wage() : double

Testa klassen *Hours* med följande program:

```
Hours employee = new Hours( 17233534, 95.0 );
employee.setHours( 128 );
System.out.println( employee.toString() );
System.out.println( "Anställd med id " + employee.getId() +
                " har arbetat " + employee.getHours() +
                " timmar till lönen " + employee.getHourlyWage() + " kr" );
employee.setHourlyWage( 98.50 );
System.out.println( "Lön denna månad: " + employee.wage() + " kr");
```

Körresultatet blir följande:

```
Id: 17233534, lön denna månad: 12160.0 kr
Anställd med id 17233534 har arbetat 128.0 timmar till lönen 95.0 kr
Lön denna månad: 12608.0 kr
```

Uppgift 1c

Lön som utgår som del av försäljningsresultat representeras av klassen ***Commission***. Klassen ska ärva ***Wage*** och ska innehålla:

Instansvariabler

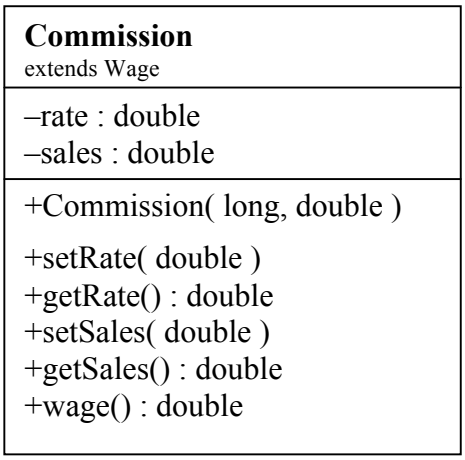
rate : double
sales : double

Konstruktorer

Commission(long id, double rate)

Metoder

public void setRate(double)
public double getRate()
public void setSales(double)
public double getSales() : double
public double wage(); // returnerar (rate * sales)



Testa klassen *Commission* med följande program:

```
Commission employee = new Commission( 19278865, 0.10 );
employee.setSales( 208000 );
System.out.println( employee.toString() );
System.out.println( "Anställd med id " + employee.getId() +
                    " har sålt för " + employee.getSales() +
                    " kr till provisionen " + employee.getRate()*100 +
                    " %" );
employee.setRate( 0.12 );
System.out.println( "Lön denna månad: " + employee.wage() + " kr");
```

Körresultatet blir följande:

```
Id: 19278865, lön denna månad: 20800.0 kr
Anställd med id 19278865 har sålt för 208000.0 kr till provisionen 10.0 %
Lön denna månad: 24960.0 kr
```

Uppgift 2 - Ärva JPanel

I denna uppgift ska du bygga en komponent som består av två JLabel-komponenter och två JTextField-komponenter. Den ska se ut som figuren till höger när den är färdig.

Klassen ska ärva JPanel. Strukturen för klassen ska vara så här:

```
package laboration14;
import java.awt.*;
import javax.swing.*;

public class NameProfession extends JPanel {
    // Instansvariabler

    public NameProfession() {
        :
    }

    // metoder som anropas från konstruktorn – vid behov

    // get-metoder
    public String getName() {
        :
    }

    public String getProfession() {
        :
    }
}
```

Layout

JLabel-komponenterna ska placeras i en panel (JPanel) och JTextField-komponenterna i en annan panel (JPanel). Dessa två paneler ska sedan placeras i den ärvda panelen. Lämpliga layout-managers är: BorderLayout i den ärvda panelen (placera panelerna i WEST och CENTER) och GridLayout i komponentpanelerna.

Instansvariabler

De grafiska komponenterna ska vara instansvariabler i klassen, dvs två JPanel-komponenter, två JLabel-komponenter och två JTextField-komponenter. JPanel-komponenterna kan ges Layout-manager redan vid konstruktionen, dvs. … = new JPanel(new GridLayout(2,1));

Konstruktorn

Sätt PreferredSize och LayoutManager för den ärvda panelen. Lägg till komponenter i panelerna och placera slutligen panelerna i den ärvda panelen. Om du vill kan du även ge panelerna en PreferredSize.

Det är tänkbart att du vill ändra Font-objektet som används av JLabel-komponenterna respektive JTextField-komponenterna. Det avgör du själv.

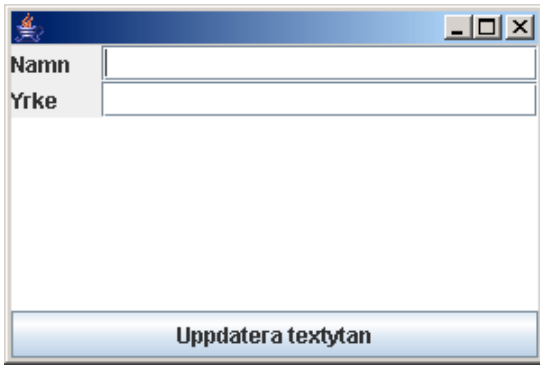
Metoder för att avläsa inmatade värden - getters

Metoderna ska returnera innehållet i vardera JTextField-komponenterna. Använd getText-metoden i klassen JTextField.

Ett fönster att placera NameProfession-panelen i

För att testa din *NameProfession*-panel kan du använda *TestNameProfession*.

Men du måste komplettera *actionPerformed*-metoden i *TestNameProfession* så att en text visas då användaren klickar på knappen. Texten ska vara på formen:
Namn: AAAA
Yrke: BBBB




```
package laboration14;
import javax.swing.*;
import java.awt.*;
import java.awt.event.*;

public class TestNameProfession extends JPanel implements ActionListener {
    private JButton update = new JButton("Uppdatera textytan");
    private JTextArea textarea = new JTextArea();
    private NameProfession nameProfession = new NameProfession();

    public TestNameProfession() {
           setLayout(new BorderLayout());
           textarea.setPreferredSize(new Dimension(300,100));
        textarea.setEditable(false);
        update.addActionListener(this);

        add(nameProfession, BorderLayout.NORTH);
        add(textarea, BorderLayout.CENTER);
        add(update, BorderLayout.SOUTH);
    }

    public void actionPerformed(ActionEvent e) {
        // komplettera
    }

    public static void main(String[] args) {
        SwingUtilities.invokeLater(new Runnable() {
            public void run() {
                JFrame frame = new JFrame();
                frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
                frame.add(new TestNameProfession());
                frame.pack();
                frame.setVisible(true);
            }
        });
    }
}
```

Uppgift 3 – Ärva JLabel

Det går utmärkt att ärva av andra komponenter än JFrame och JPanel. I denna uppgift ska du skapa en något ”specialiserad” JLabel-komponent. Följande gäller för komponenten

- Den skapas med ”Dialog” som typsnitt
- Texten är centrerad från början
- Bakgrundsfärgen syns
- När komponenten skapas måste man ange:
  - text som ska visas
  - stil
  - storlek
  - textfärg
  - bakgrundsfärg

För övrigt fungerar komponenten som vilken JLabel-komponent som helst. Så här ska klassen vara:

```
package laboration14;
import javax.swing.*;
import java.awt.*;

public class Title extends JLabel {
    public Title(String content, int style, int fontsize, Color text,
                 Color background) {
        Font font = new Font("Dialog", style, fontsize);
        setText(content);
        setHorizontalAlignment(JLabel.CENTER);
        setFont(font);
        setBackground(background);
        setOpaque(true);
        setForeground(text);
    }
}
```

Kommentarer

Klassen innehåller endast en konstruktor i vilka inställningarna görs:

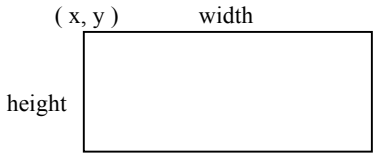
- `setText(content);` texten anges
- `setHorizontalAlignment(JLabel.CENTER);` texten ska vara centrerad
- `setFont(font);` typsnittet anges
- `setBackground(background);` `setOpaque(true);` bakgrundsfärgen anges och komponenten görs icke-transparent
- `setForeground(text);` textfärgen anges

Testprogram

För att test utseendet på **Title**-komponenten ska du skriva en klass, vilken ärver JPanel, och som innehåller tre Title-komponenter. LayoutManager i panelen ska vara BorderLayout och komponenterna ska placeras i NORTH, CENTER och SOUTH.

Sedan ska du placera panelen i en JFrame. När fönstert görs synligt ska det likna fönstret till höger.

Uppgift 4a – Klassen Rectangle



Du ska skriva klassen ***Rectangle*** vilken ska ärva klassen ***Shape***. Klassen *Shape* är *abstrakt* och kan endast ärvas. *Shape* innehåller en abstrakt metoder, ***paint***.

- Klassen *Rectangle* ska innehålla instansvariablerna *width* och *height*.
- Konstruktorn ska ta emot 5 argument – *x*, *y*, *width*, *height* och *color*
- *paint*-metoden måste implementeras i klassen *Rectangle*.

Ytterligare en klass ingår i uppgiften, nämligen ***PaintWindow***. Denna klass finns i laborationsmaterialet.

Som du ser i klassen *Shape* och i testprogrammet nedan så ska *paint*-metoden ta emot ett *PaintWindow*-objekt som parameter och sedan rita sig i detta fönster.

Metoden *paint*

I klassen *Rectangle* ska *paint*-metoden anropa metoden *line* i *PaintWindow*-objektet. Detta ska göras fyra gånger:

```
window.line(super.x, super.y, super.x + this.bredd, super.y, this.färg, 1);
// osv
```

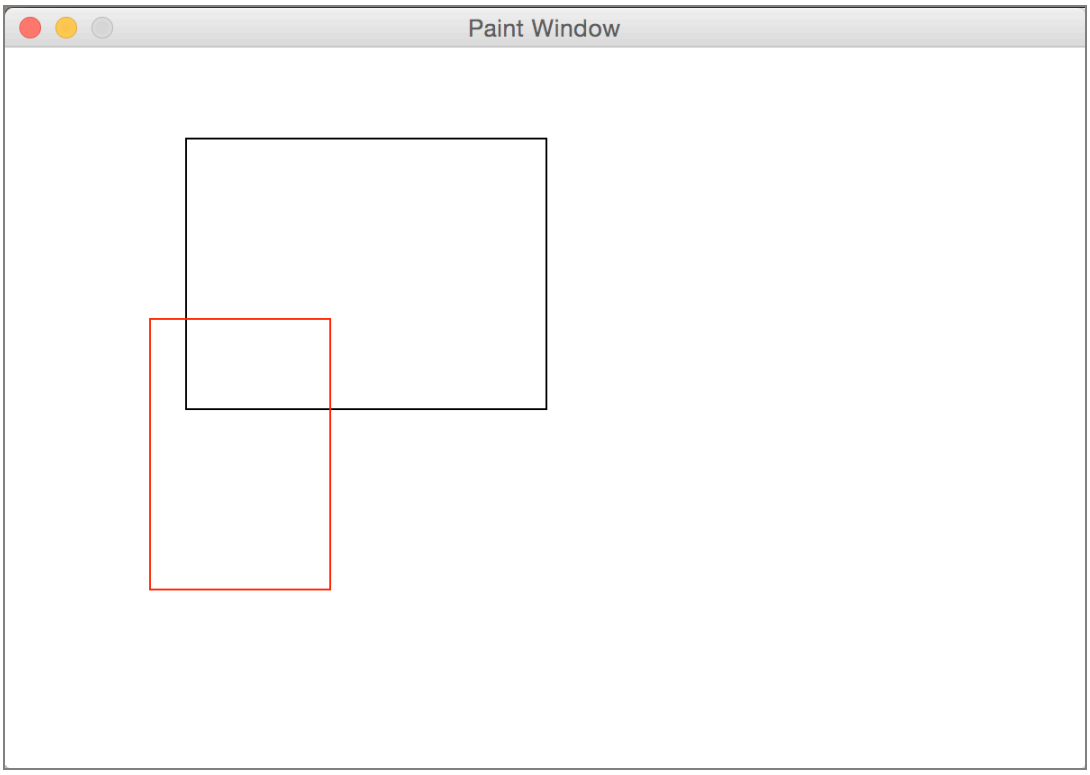
Konstruktor

```
public Rectangle( int x, int y, int width, int height, Color color )
```

När du kör nedanstående testprogram erhåller du körresultatet längst ner på sidan.

Testprogram

```
Rectangle rect1 = new Rectangle( 100, 50, 200, 150, Color.BLACK );
Rectangle rect2 = new Rectangle( 80, 150, 100, 150, Color.RED);
PaintWindow window = new PaintWindow();
rect1.paint( window );
rect2.paint( window );
```



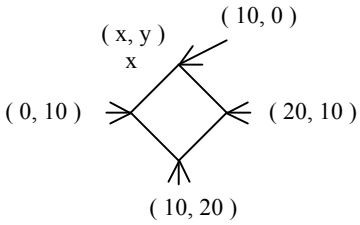
Uppgift 4b – Klassen NSides

Du ska skriva klassen ***NSides*** vilken ska ärva klassen ***Shape***. Klassen *Shape* är abstrakt och du måste implementera *paint*-metoden i klassen *NSides*.

- Klassen *NSides* ska innehålla instansvariabeln *points* av typen *Point[]*. Klassen ***Point*** är given och finns på kurssidan. Arrayen uttrycker punkter relativt figurens (x, y)-värde (se exempel nedan till höger)
- Konstruktorn ska ta emot fyra argument – x, y, punkter och färg
- *paint*-metoden ska implementeras så att det ritas linjer mellan punkterna i *Point*-arrayen.

Metoden *paint*

paint-metoden ska implementeras så att det ritas linjer mellan punkterna i *Point*-arrayen. Från punkt 1 ska det ritas linje till punkt2, Från punkt 2 till punkt 3 osv. Dessutom ska det vara en linje mellan den första punkten och den sista punkten.
I figuren till höger innehåller *Point*-arrayen objekten:
(10, 0), (20, 10), (10, 20) och (0, 10)



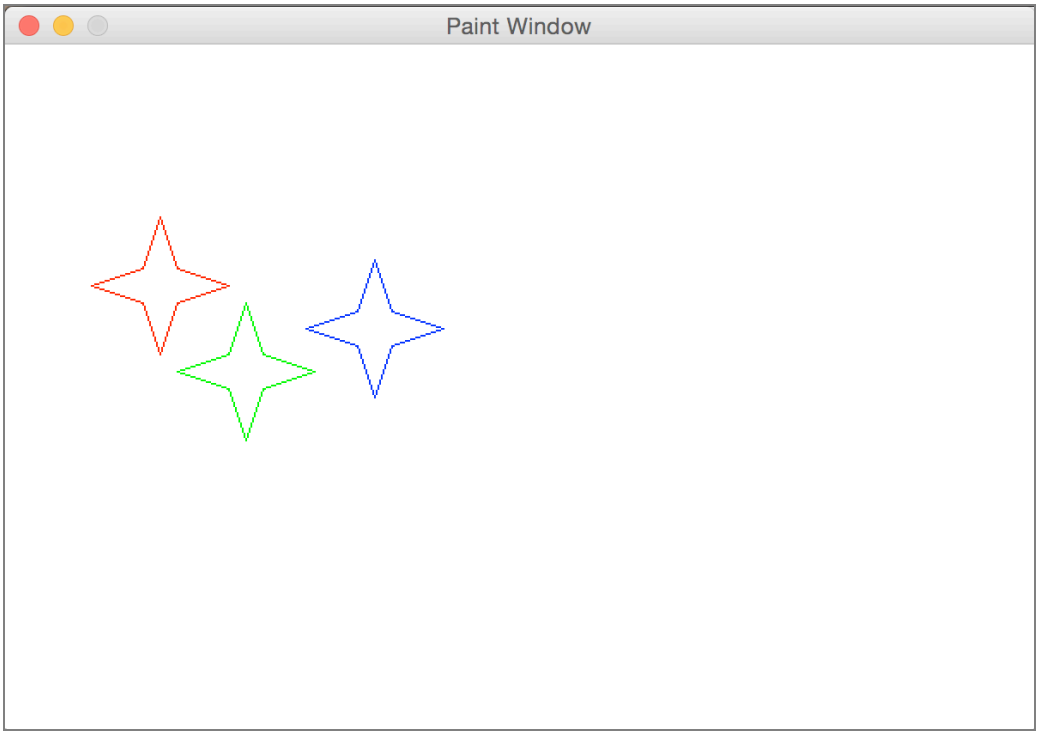
Konstruktor

```
public NSides( int x, int y, Point[] points, Color color )
```

När du kör nedanstående testprogram erhåller du körresultatet längst ner på sidan.

Testprogram

```
Point[] p = { new Point(0, 40), new Point(30, 30), new Point(40, 0),
              new Point(50, 30), new Point(80, 40), new Point(50, 50),
              new Point(40, 80), new Point(30, 50) };
NSides star1 = new NSides(50, 100, p, Color.RED);
NSides star2 = new NSides(100, 150, p, Color.GREEN);
NSides star3 = new NSides(175, 125, p, Color.BLUE);
PaintWindow window = new PaintWindow();
star1.paint(window);
star2.paint(window);
star3.paint(window);
```



Extrauppgifter

Uppgift 5

Du ska skriva klassen ***Animation*** vilken ska ärva *Shape*. Följande gäller:

- Klassen ska innehålla två instansvariabler,
  * *images* (ImageIcon[]) en array med ImageIcon-objekt
  * *index* (int) håller reda på vilken bild som ska visas. Ska ges lämpligt startvärde, t.ex. -1.
- Klassen ska ha en konstruktor som tar emot tre argument, en ImageIcon-array, x och y
- *paint*-metoden ska implementeras så att vid varje anrop ska en ny bild visas. Först visas bilden i position 0 (*images[0]*), sedan bilden i position 1 (*images[1]*) osv. När det inte finns fler bilder att visa ska åter bilden i position 0 visas osv.
  Exempel på vad *paint*-metoden ska göra:
  * Sudda bilden i position *index* (anropa metoden *hideImage*). Detta ska inte göras vid det första anropet till *paint*-metoden (använd lämplig if-sats).
  * Öka *index* med 1. Om *index* blir för stort (*index >= images.length*) så sätt *index* till 0. Det fungerar bra med en if-sats. I lösningarna styrs värdet på *index* med hjälp av %-operatorn.
  * Anropa metoden *showImage* så att bilden i position *index* visas. Anropet kan börja så här:
    ```
    window.showImage( images[index].getImage(), …);
    ```

När du kör nedanstående testprogram ska ordet **New** animeras i PaintWindow-fönstret.

```
PaintWindow window = new PaintWindow();
ImageIcon[] images = new ImageIcon[10];
for(int i=0; i<images.length; i++) {
    images[i] = new ImageIcon("images/new" + (i+1) + ".jpg");
}
Animation anim = new Animation(images, 100, 100);
for(int i=0; i<100; i++) {
    anim.paint(window);
    PaintWindow.pause(50);
}
```

Utökning

Lägg till metoderna ***setX***, ***setY***, ***getX*** och ***getY*** i klassen ***Animation***. Ändra sedan figurens x- respektive y-läge lite vid varje iteration i programmet ovan (t.ex. efter PaintWindow.pause(50)). Du kan t.ex. öka x med 4 och y med 2 varje iteration.

Uppgift 1a

```
package laboration14;

public class FullTime extends Wage {
    private double wage;

    public FullTime( long id, double lön ) {
        super( id ); // Anropa konstruktorn AnsalldLon( long )
        this.wage = lön;
    }

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

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

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

Uppgift 1b

```
package laboration14;

public class Hours extends Wage {
    private double hourlyWage;
    private double hours;

    public Hours( long id, double hourlyWage ) {
        super( id );
        this.hourlyWage = hourlyWage;
    }

    public double getHours() {
        return this.hours;
    }

    public void setHours(double hours) {
        this.hours = hours;
    }

    public double getHourlyWage() {
        return this.hourlyWage;
    }

    public void setHourlyWage(double hourlyWage) {
        this.hourlyWage = hourlyWage;
    }

    public double wage() {
        return this.hours * this.hourlyWage;
    }
}
```

Uppgift 1c

```
package laboration14;

public class Commission extends Wage {
    private double rate;
    private double sales;

    public Commission( long id, double rate ) {
        super( id );
        this.rate = rate;
    }

    public double getSales() {
        return this.sales;
    }

    public void setSales(double sales) {
        this.sales = sales;
    }

    public double getRate() {
        return this.rate;
    }

    public void setRate(double rate) {
        this.rate = rate;
    }

    public double wage() {
        return this.sales * this.rate;
    }
}
```

Uppgift 2

```
package laboration14;
import java.awt.*;
import javax.swing.*;

public class NameProfession extends JPanel {
    private JPanel labelPanel = new JPanel(new GridLayout(2,1));
    private JPanel inputPanel = new JPanel(new GridLayout(2,1));
    private JTextField tfName = new JTextField();
    private JTextField tfProfession = new JTextField();
    private JLabel lblName = new JLabel("Namn");
    private JLabel lblProfession = new JLabel("Yrke");

    public NameProfession() {
        Font dialogB12 = new Font("Dialog", Font.BOLD, 12);
        setLayout(new BorderLayout());
        setPreferredSize(new Dimension(300,60));

        labelPanel.setPreferredSize(new Dimension(50,0));
        lblName.setFont(dialogB12);
        lblProfession.setFont(dialogB12);
        labelPanel.add(lblName);
        labelPanel.add(lblProfession);
        inputPanel.add(tfName);
        inputPanel.add(tfProfession);

        add(labelPanel, BorderLayout.WEST);
        add(inputPanel, BorderLayout.CENTER);
```

```
    }

    /********** get-metoder ***********/
    public String getName() {
        return tfName.getText();
    }

    public String getProfession() {
        return tfProfession.getText();
    }
}
```

```
//Komplettering i TestNameProfession
public void actionPerformed(ActionEvent e) {
    textarea.setText("Namn: " + nameProfession.getName() +"\nYrke: " +
                     nameProfession.getProfession());
}
```

Uppgift 3

```
package laboration14;
import javax.swing.*;
import java.awt.*;

public class TestTitle extends JPanel {
    private Title title1 = new Title("Rött och svart", Font.BOLD, 24,
                                     Color.red, Color.black);
    private Title title2 = new Title("Blå text - gul bakgrund", Font.PLAIN,
                                     16, Color.blue, Color.yellow);
    private Title title3 = new Title("Denna rubrik är i SOUTH",
                                     Font.ITALIC + Font.BOLD, 12
                                     Color.white, Color.black);

    public TestTitle() {
        setLayout(new BorderLayout());
        setPreferredSize(new Dimension(400,150));
        add(title1, BorderLayout.NORTH);
        add(title2, BorderLayout.CENTER);
        add(title3, BorderLayout.SOUTH);
    }

    public static void main(String[] args) {
        SwingUtilities.invokeLater(new Runnable() {
            public void run() {
                JFrame frame = new JFrame();
                frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
                frame.add(new TestTitle());
                frame.pack();
                frame.setVisible(true);
            }
        });
    }
}
```

Uppgift 4a

```
package laboration14;
import java.awt.*;

public class Rectangle extends Shape {
    private int width;
    private int height;

    public Rectangle(int x, int y, int width, int height, Color color) {
        super( x, y, color );
        this.width = width;
        this.height = height;
    }

    public void paint( PaintWindow window ) {
        window.line( super.x, super.y, super.x + this.width, super.y,
                     super.color, 1 );
        window.line( super.x + this.width, super.y, super.x + this.width,
                     super.y + this.height, super.color, 1 );
        window.line( super.x, super.y + this.height, super.x + this.width,
                     super.y + this.height, super.color, 1 );
        window.line( super.x, super.y, super.x, super.y + this.height,
                     super.color, 1 );
    }
}
```

Uppgift 4b

```
package laboration14;
import java.awt.*;

public class NSides extends Shape {
    private Point[] points;

    public NSides(int x, int y, Point[] points, Color color) {
        super(x, y, color);
        this.points = points;
    }

    public void paint(PaintWindow window) {
        int x1, y1, x2, y2;
        if (points.length >= 2) {
            for (int i = 0; i < points.length - 1; i++) {
                x1 = super.x + points[i].getX();
                y1 = super.y + points[i].getY();
                x2 = super.x + points[i + 1].getX();
                y2 = super.y + points[i + 1].getY();
                window.line(x1, y1, x2, y2, super.color, 1);
            }
            x1 = super.x + points[0].getX();
            y1 = super.y + points[0].getY();
            x2 = super.x + points[points.length - 1].getX();
            y2 = super.y + points[points.length - 1].getY();
            window.line(x1, y1, x2, y2, super.color, 1);
        }
    }
}
```

Uppgift 5

```
package laboration14;
import javax.swing.ImageIcon;

public class Animation extends Shape {
    private ImageIcon[] images;
    private int index = -1;

    public Animation(ImageIcon[] images, int x, int y) {
        this.images = images;
        super.x = x;
        super.y = y;
    }

    // B
    public int getX() {
        return super.x;
    }
    // B
    public void setX(int x) {
        super.x = x;
    }
    // B
    public int getY() {
        return super.y;
    }
    // B
    public void setY(int y) {
        super.y = y;
    }

    public void paint(PaintWindow window) {
        if(index>=0) {
            window.hideImage(images[index]);
        }
        index = (index+1) % images.length;
        window.showImage(images[index], super.x, super.y);
    }
}
```