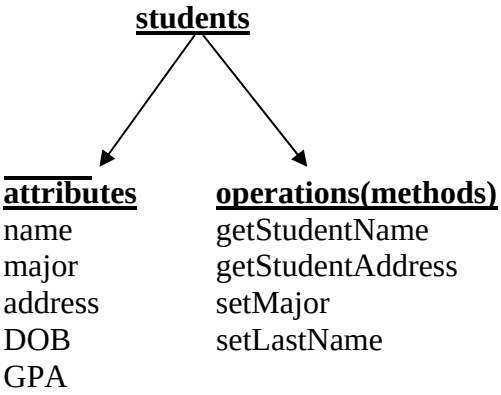
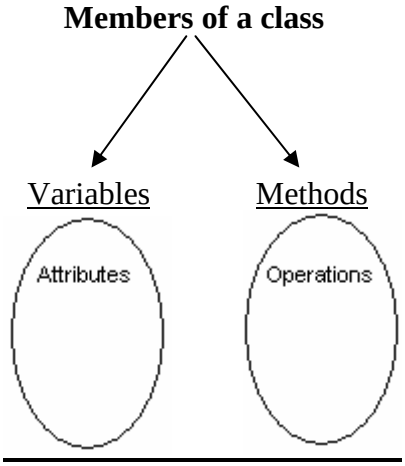
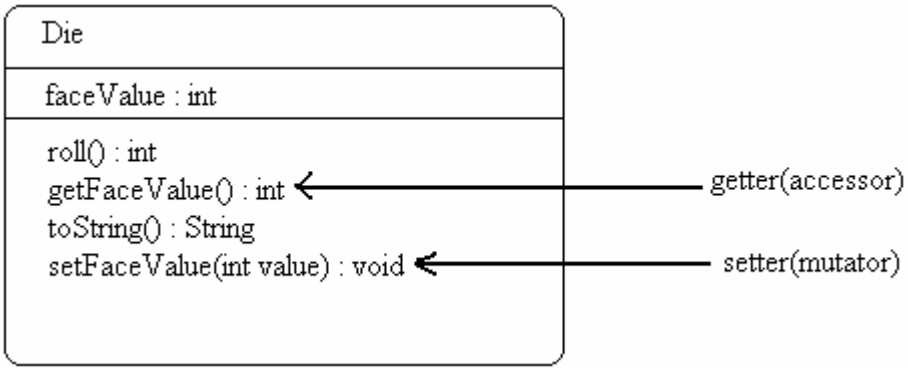
WRITING CLASSES (CHAPTER 4)

Three fundemental concepts of object-oriented programming :
- encapsulation
- inheritance
- polymorphism





Die class: How to use, how to construct the class




```
public class RollingDice
//creates two dice objects and rolls them several times
  public static void main (String[] args)
        {
          Die die1,die2;
          int sum;

          die1 = new Die();
          die2 = new Die();

          die1.roll();
          die2.roll();
          System.out.println("Die one: " + die1 + " Die two: " + die2);

          //Output:
          //Die one : 5    Die two : 2

          die1.roll();
          die2.setFaceValue(4);
          System.out.println("Die one: " + die1 + " Die two: " + die2);

          //Output:
          //Die one : 1    Die two : 4
```

```
            Sum = die1.getFaceValue() + die2.getFaceValue();
            System..out.println( “Sum: “ + sum);
            //Output:
            //Sum:5
  }
}


public class Die
{
        private final int MAX = 6;
        private int faceValue;

        //constructor
        public Die()
        {
                faceValue = 1;
        }

        public Die (int initialFaceValue)
        {
                faceValue = initialFaceValue;
        }

        //----------------------------------------
        //roll the die and return the result
        //----------------------------------------

        public int Roll( )
        {
                faceValue = (int) (Math.random( ) * MAX) +1;
                return ( faceValue );
        }

        //face value accessor
        public int getFaceValue( )
        {
                return faceValue;
        }

        //face value mutator
        public void setFaceValue( int value )
        {
                faceValue = value;
        }
```

```
public void setFaceValue (int value)
{
        if (value <= 6 && value >0)
                faceValue = value;
        else
                faceValue =6;
}

public String toString()//it converts an integer to a String
{
        String result = Integer.toString(feeValue);
        return result;
}

public class Die
{
        private final int MAX = 6;
        private int faceValue;

  //.....................................
  //constructor
  //.....................................

  public Die()
  {
        faceValue = 1;
  }

  //.....................................
  public Die (int k)
  {
        faceValue = k;
  }

  public Die ( Die otherDie)
  {
        faceValue = otherDie.faceValue;
  }

  public String toString()
  {
        String result = Integer.toString(faceValue);
        return result;
  }

  System.out.println( " die 1: " + die1 + "die2: " +die2);
```

Assume that we have no toString()

die1 = @ 14E4FA
die2 = @ EE44BA

int = 10;
System.out.println(" i : " +i);

Assume that we have to toString() method
die1 = 1
die2 = 4

```
public int rollDie()
{
        faceValue = (int)(Math.random()*MAX) + 1;
        return faceValue;
}

Die die1 = new Die();
System.out.println("die1: "+die1);
die1 =1

die1.rollDie();
System.out.println("die1: "+die1);
die1 =5

public int getFaceValue()
{
        return faceValue;
}

private String convertToString()
{
  String r;
  r = Integer.toString(faceValue);
  return r;
}

System.out.println(die1.convertToString());

//reimplement toString()

public String toString ()
{
  String result;
```

```
            result = convertToString();
            return result;
          }

         System.out.println("die1 : " + die1);
}
```

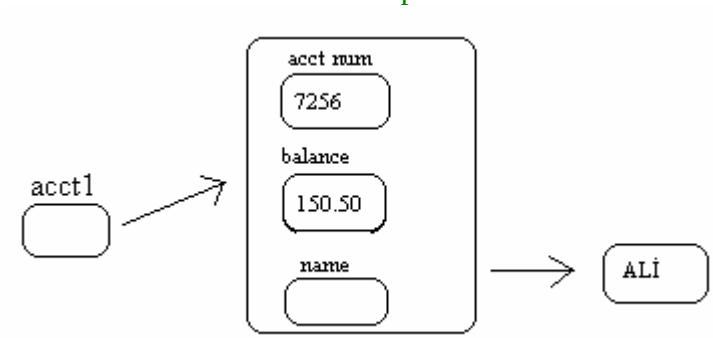
Bank Account Example

```
import java.text.NumberFormat;
public class Acount
{
  private final double RATE = 0.035;
  private long acctNumber;
  private double balance;
  private String name;

  //constructor (there is no return type for the constructor)
  public Account ( String owner, long account, double initial);
                    //formal parameters(arguments)
  {
            name = owner;
            acctNumber = account;
            balance = initial;
  }

  public double deposit (double amount)
  {
            balance = balance + amount,
            return balance;
  }

  public double withdraw (double amount, double fee)
  {
            balance = balance- amount-fee;
            return balance;
  }

  public double addInterest()
  {
            balance = balance + (balance*RATE);
            return balance;
  }

  public double getBalance()
  {
            return balance;
```

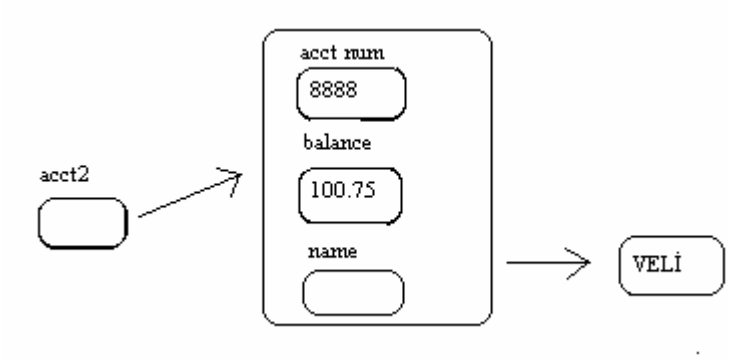
```
  }

  public String toString()
  {
        NumberFormat fmt = NumberFormat.getCurrencyInstance();
        return ( acctNumber + "/t" + name+ fmt.format(balance));
  }

}
```

MAIN:

```
  {     Account acct1 = new Account ("'ali', 7526, 150.50);
        acct1.deposit(10.50);
                    //actual parameter
```




```
        Account acct2 = new Account ('veli', 8888, 100.75);
        acct2.deposit(10.0);
```




```
        acct1.withdraw(5.0,1.0);
        acct2.withdraw(1.0);
        System.out.println("account2: "+acct2);
        //Output: account2 8888   Veli $ 104.75
```

Graphical Objects

```
import java.awt.*;
public class Circle
{
  private int diameter,x,y;
  private Color circleColor;
  public Circle (int size,Color shade, int upperX, int upperY);
  {
        diameter = size;
        circleColor = shade;
        x = upperX;
        y = upperY;

  }

  public void draw (Graphics page)
  {
        page.setColor(circleColor);
        page.fillOval(x,y,diameter,diameter);
  }
}
```

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

public class SplatPanel extends JPanel
{
  private Circle circle1, circle2, circle3, circle4, circle5;
  public SplatPanel()
  {
        circle1 = new Circle(30,Color.red,70,35);
        circle1 = new Circle(50,Color..green,30,20);
        circle1 = new Circle(100,Color..cyan,60,85);
        circle4 = new ....
        circle5 = new ....

        this.setPrefferedSize (new Dimension(300,200));
        this.setBackground(Color.black);
  }

  public void PaintComponent(Graphics page)
  {
```

```
            super.paintComponent(page);
            circle1.draw(page);
            circle2.draw(page);
            circle3.draw(page);
            circle4.draw(page);
            circle5.draw(page);
    }
}
```

```
 //*****************

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

public class Splat

{
  //create and display the BMI GUI

  public static void main ( String [] args)
  {
            JFrame frame = new JFrame ("splat");
            frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

            splatPanel panel = new splatPanel();
            frame.getContentPane().add(panel);
            frame.pack();

            frame.setVisible(true);
  }
}
```

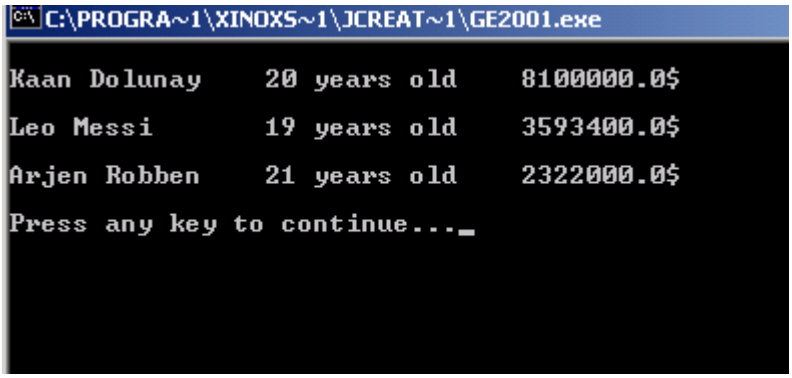
QUESTIONS

1- **Design and implement a class called PlayerValue contains instance data that represents the player's name, age, and value. Write a method that increases the player's value %1 for aech goal he scores and a method that increases the value %0,5 for each assist he makes. Include toString method that returns a one line description of player's information, Create a driver class called PlayerInformation whose main method instantiates and updates several PlayerValue objects.**

Answer1:

```
//driver class

public class PlayerInformation
{

   public static void main(String[] args)
   {
      PlayerValue p1=new PlayerValue("Kaan Dolunay",20,6000000);
      PlayerValue p2=new PlayerValue("Leo Messi",19,3000000);
      PlayerValue p3=new PlayerValue("Arjen Robben",21,2000000);

      p1.goal(25);
      p2.goal(13);
      p3.goal(8);

      p1.asist(16);
      p2.asist(12);
      p3.asist(15);

      System.out.println();
      System.out.println(p1);
      System.out.println();
      System.out.println(p2);
      System.out.println();
      System.out.println(p3);
      System.out.println();
   }
}


//class


public class PlayerValue
{

private String name;
private int agep;
private double fee;
private final double goalPers=0.01,asistPers=0.005;

public PlayerValue(String player,int age,double value)
{
         name=player;
         agep=age;
```

```
        fee=value;
    }

    public double goal(int score)
    {
        fee=fee+(fee*score*goalPers);
        return fee;
    }

    public double asist(int pas)
    {
        fee=fee+(fee*pas*asistPers);
        return fee;
    }

    public String toString()
    {
        return (name+"\t"+agep+" years old\t"+fee+"$");

    }


    }
```



2- Develop an application that implements a user interface for converting euro value to YTL value. And uses a convert button to calculate the result.

Answer2:

```
import javax.swing.JFrame;
/**
 * AWT Sample application
 *
```

```
 * @author
 * @version 1.00 04/08/31
 */
public class CurrencyConverter
{

   public static void main(String[] args)
   {

      JFrame frame=new JFrame("CurrencyConverter");

      CurrencyPanel panel=new CurrencyPanel();

      frame.getContentPane().add(panel);
      frame.pack();
      frame.setVisible(true);

   }
}



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

public class CurrencyPanel extends JPanel
{
     private JLabel label,label2,label3;
     private JTextField euro;
     private JButton convert;
     public CurrencyPanel()

     {
               convert=new JButton("convert");
               convert.addActionListener(new CurrencyListener());

               label=new JLabel("Enter the Euro value: ");
               label2=new JLabel("Value in YTL: ");
               label3=new JLabel("  ");

               euro = new JTextField(5);
               euro.addActionListener(new CurrencyListener());
               add (convert);
               add (label);
               add (euro);
               add (label2);
```

```
            add (label3);


            setPreferredSize(new Dimension(200,100));
            setBackground(Color.cyan);

    }

    private class CurrencyListener implements ActionListener

    {
            public void actionPerformed(ActionEvent event)

            {
                    double tl;
                    int eu;

                    String text=euro.getText();

                    eu=Integer.parseInt(text);
                    tl=eu*1.875;

                    label3.setText(Double.toString(tl));

            }



    }
}
```

3- Develop an application that implements user interface to change the value of the number which user enters. Use 3 buttons to increase the number one by one, to decrease the number one by one, and to take the square of the number.

Answer3:

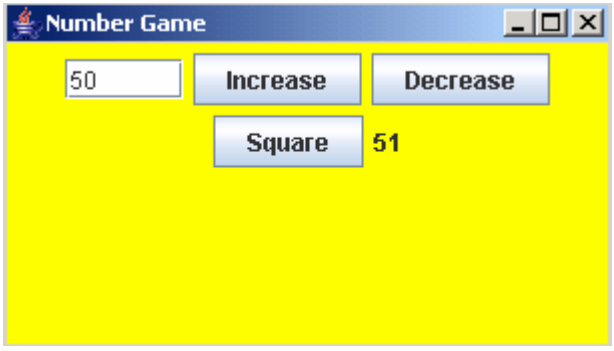
```
import java.awt.*;
```

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

public class CurrencyPanel extends JPanel
{
    private JLabel label,label2,label3;
    private JTextField euro;
    private JButton convert;
    public CurrencyPanel()

    {
            convert=new JButton("convert");
            convert.addActionListener(new CurrencyListener());

            label=new JLabel("Enter the Euro value: ");
            label2=new JLabel("Value in YTL: ");
            label3=new JLabel("  ");

            euro = new JTextField(5);
            euro.addActionListener(new CurrencyListener());
            add (convert);
            add (label);
            add (euro);
            add (label2);
            add (label3);


            setPreferredSize(new Dimension(200,100));
            setBackground(Color.cyan);

    }

    private class CurrencyListener implements ActionListener

    {
            public void actionPerformed(ActionEvent event)

            {
                    double tl;
                    int eu;

                    String text=euro.getText();

                    eu=Integer.parseInt(text);
                    tl=eu*1.875;

                    label3.setText(Double.toString(tl));
```

```
            }



    }
}



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

public class NumPanel extends JPanel

{
    private JButton increase,decrease,square;
    private JLabel label;
    private int count;
    private JTextField num;

    public NumPanel()

    {



            num=new JTextField(5);
            num.addActionListener(new ButtonListener());

            increase=new JButton("Increase");
            increase.addActionListener(new ButtonListener());

            decrease=new JButton("Decrease");
            decrease.addActionListener(new Button1Listener());

            square=new JButton("Square");
            square.addActionListener(new Button2Listener());


            label=new JLabel(""+count);

            add (num);
            add (increase);
```

```
        add (decrease);
        add (square);
        add (label);

        setPreferredSize(new Dimension(300,150));
        setBackground(Color.yellow);


}

private class ButtonListener implements ActionListener

{

        public void actionPerformed (ActionEvent event)
        {


           String text=num.getText();

                count=Integer.parseInt(text);


                count++;
                label.setText(Integer.toString(count));


        }


}



private class Button1Listener implements ActionListener

{

        public void actionPerformed (ActionEvent event)
        {


           String text=num.getText();

                count=Integer.parseInt(text);
                count--;
```

```
                label.setText(Integer.toString(count));
            }


    }


    private class Button2Listener implements ActionListener

    {

            public void actionPerformed (ActionEvent event)
            {


               String text=num.getText();

                    count=Integer.parseInt(text);
                    count=count*count;

                    label.setText(Integer.toString(count));
            }


    }

}
```



4- Explain what component, event and listener means.

Answer4:

GUI component is an object that defines a screen element to display information or allow the user to interact with a program in a certain way/

An event is an object that represents some occurrence, in which we may be interested,

A listener is an object that “waits” for an event to occur and responds in some way when it does.