LAB CYCLE 5

SUBMITTED BY,
SIJI JOSE
20MCA237

PROGRAM -1

AIM: Program to draw Circle, Rectangle, Line in Applet.

ALGORITHM

```
STEP 1: Start
STEP 2: Define a class ‘draw’ that extends Applet class.
STEP 3: Draw a line, rectangle and circle using drawLine, drawRect and drawOval
        methods of Graphics class respectively
STEP 4: Stop
```

PROGRAM CODE

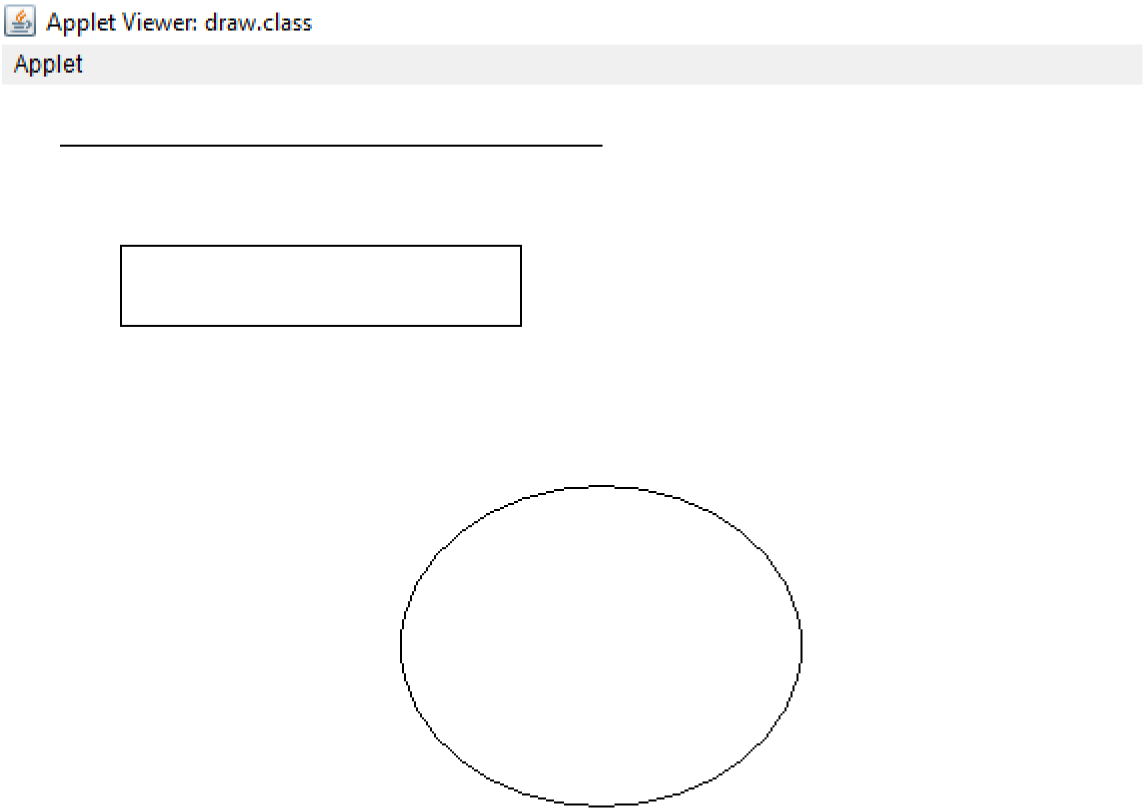
draw.java	import java.applet.Applet;
	import java.awt.Graphics;
	public class draw extends Applet{
	public void paint(Graphics g){
	g.drawLine(30,30, 300, 30);
	g.drawRect(60, 80, 200, 40);
	g.drawOval(200, 200, 200, 160);
	}
	}

draw.html	<applet code = "draw.class" width = "500" height = "700"></applet>

RESULT

The above program is executed and obtains the output.

OUTPUT



PROGRAM -2

AIM: Program to find a maximum of three numbers using AWT.

ALGORITHM

STEP 1: Start
STEP 2: Define a class ‘largest’ that extends Applet class and implements ActionListener interface.
STEP 3: Using TextField class object, construct the required no. of text fields wide enough to hold the values entered by the user.
STEP 4: Using TextField class object, construct the required no. of text fields wide enough to hold the values entered by the user.
STEP 5: Call addActionListener() method to send events from the button to the new listener.
STEP 6:Get the string values from text fields and then parse them as integers.
STEP 7:Compare each value using if-else statements to find the maximum value and set the result accordingly
STEP 8: Stop

PROGRAM CODE

largest.java

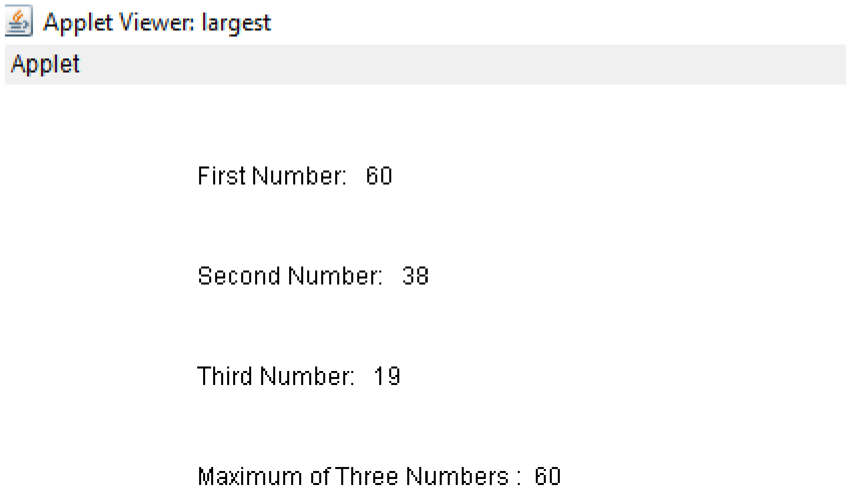
```
import java.awt.*;
import java.applet.*;
import java.io.*;

/*
<applet code="largest" width=500 height=500>
<param name="a" value="60">
<param name="b" value="38">
<param name="c" value="19">
</applet>
*/

public class largest extends Applet
{
        int a;
        int b;
        int c;
     int d;
        String str;

        public void start()
        {
                String s1;

                s1 = getParameter("a");
```

```
                    a = Integer.parseInt(s1);

                    s1 = getParameter("b");
                    b = Integer.parseInt(s1);

              s1 = getParameter("c");
                    c = Integer.parseInt(s1);
          }

          public void paint(Graphics g)
          {
                    if( a >= b && a >= c)
                            d = a;
                    else if (b >= a && b >= c)
                  d=b;
            else
                  d=c;
            g.drawString("First Number:   " + a, 100, 50);
                    g.drawString("Second Number:   " + b, 100, 100);
            g.drawString("Third Number:   " + c, 100, 150);
                    g.drawString("Maximum of Three Numbers :  " +
d, 100, 200);
          }
}
```

RESULT

The above program is executed and obtains the output.

OUTPUT



```
First Number:   60

Second Number:   38

Third Number:   19

Maximum of Three Numbers :  60
```

PROGRAM -3

AIM: Find the percentage of marks obtained by a student in 5 subjects. Display a happy face if he secures above 50% or a sad face if otherwise.

ALGORITHM

```
STEP 1: Start
STEP 2:Define a class ‘mark’ that extends Applet class and implements
        ActionListener interface.
STEP 3:Using TextField class object, construct text fields to receive marks of 5 subjects
        from the user.
STEP 4:Using Button class object, construct a labeled button that sends an instance
        of ActionEvent.
STEP 5: Call addActionListener() method to send events from the button to the new listener
STEP 6:Get the string values from text fields and then parse them as float values.
STEP 7:Calculate the percentage
STEP 8:Define a paint() method that contains functions from Graphics class to display
        a happy face if student secures above 50% or a sad face if otherwise
STEP 9: Stop
```

PROGRAM CODE

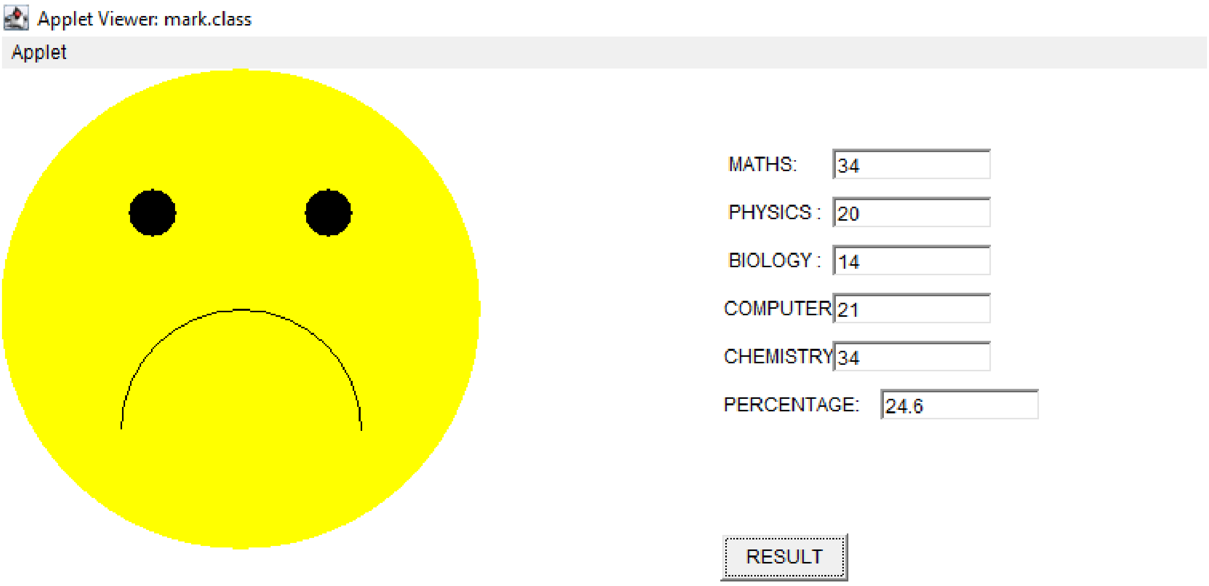
mark.java

```
import java.applet.*;
import java.awt.*;
import java.awt.Graphics;
import java.awt.event.*;


public class mark extends Applet implements ActionListener {
        Label l1,l2,l3,l4,l5,l6;
  TextField t1,t2,t3,t4,t5,t6;
  Button b;
  public void init(){
        l1 = new Label(" MATHS:");
        t1 = new TextField();
        l2 = new Label(" PHYSICS :");
        t2 = new TextField();
        l3 = new Label(" BIOLOGY :");
        t3 = new TextField();
        l4 = new Label("COMPUTER :");
        t4 = new TextField();
        l5 = new Label("CHEMISTRY:");
        t5 = new TextField();
        l6 = new Label("PERCENTAGE:");
        t6 = new TextField();


        b = new Button("RESULT");

        setLayout(null);

        l1.setBounds(450,50,70,20);
    t1.setBounds(520,50,100,20);
    l2.setBounds(450,80,70,20);
    t2.setBounds(520,80,100,20);
    l3.setBounds(450,110,70,20);
    t3.setBounds(520,110,100,20);
        l4.setBounds(450,140,70,20);
        t4.setBounds(520,140,100,20);
        l5.setBounds(450,170,70,20);
        t5.setBounds(520,170,100,20);
        l6.setBounds(450,200,100,20);
        t6.setBounds(550,200,100,20);

        b.setBounds(450,290,80,30);

    add(l1);
    add(l2);
    add(l3);
    add(l4);
        add(l5);
        add(l6);
    add(t1);
    add(t2);
    add(t3);
    add(t4);
    add(t5);
    add(t6);
    add(b);
    b.addActionListener(this);

}
public void actionPerformed(ActionEvent e){
        float m1, m2,m3, m4,m5,percent;

    m1= Float.parseFloat(t1.getText());
    m2= Float.parseFloat(t2.getText());
    m3= Float.parseFloat(t3.getText());
    m4= Float.parseFloat(t4.getText());
    m5= Float.parseFloat(t5.getText());

    percent=((m1+m2+m3+m4+m5)*100)/500;

    t6.setText(String.valueOf(percent));
    repaint();
}
public void paint(Graphics g){
        float p;
        p= Float.parseFloat(t6.getText());

        if(p> 50.0) {
             g.setColor(Color.YELLOW);
             g.fillOval(0,0,300,300);
             g.setColor(Color.BLACK);
             g.fillOval(80,75,30,30);
             g.fillOval(190,75,30,30);
             g.setColor(Color.black);
             g.fillArc (75,100,150,150,0,-180);
        }
        else {
             g.setColor(Color.YELLOW);
            g.fillOval(0,0,300,300);
            g.setColor(Color.BLACK );
            g.fillOval(80,75,30,30);
            g.fillOval(190,75,30,30);
            g.setColor(Color.black);
            g.drawArc(75,150,150,150,0,180);

        }
    }
}
/*
<applet code="mark.class" border="2" width="500"
height="500">
</applet>
*/
```

RESULT

The above program is executed and obtains the output.

OUTPUT



Applet Viewer: mark.class

Applet

MATHS: 34

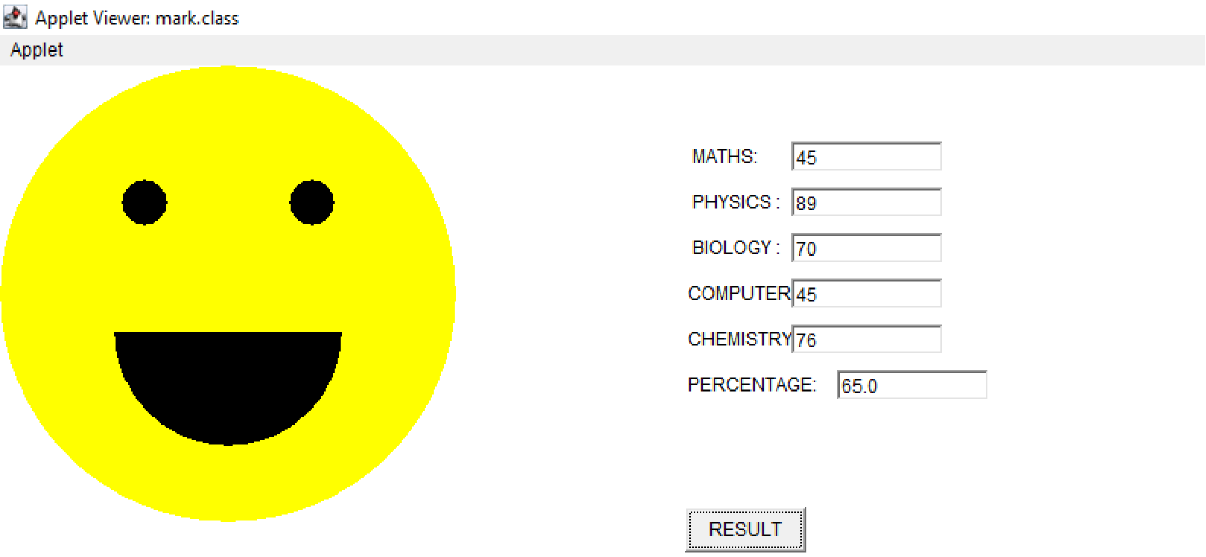
PHYSICS : 20

BIOLOGY : 14

COMPUTER 21

CHEMISTRY 34

PERCENTAGE: 24.6



MATHS: 45

PHYSICS : 89

BIOLOGY : 70

COMPUTER 45

CHEMISTRY 76

PERCENTAGE: 65.0

PROGRAM -4

AIM: Using 2D graphics commands in an Applet, construct a house. On mouse click event, change the color of the door from blue to red.

ALGORITHM

STEP 1: Start
STEP 2:Define a class ‘house’ that extends Applet and implements MouseListener.
STEP 3:Define methods to add MouseListener to the panel.
STEP 4: Using getX() and getY() methods, get the coordinates of the door to repaint
when the MousePressed event occurs
STEP 5: Stop

PROGRAM CODE

house.java

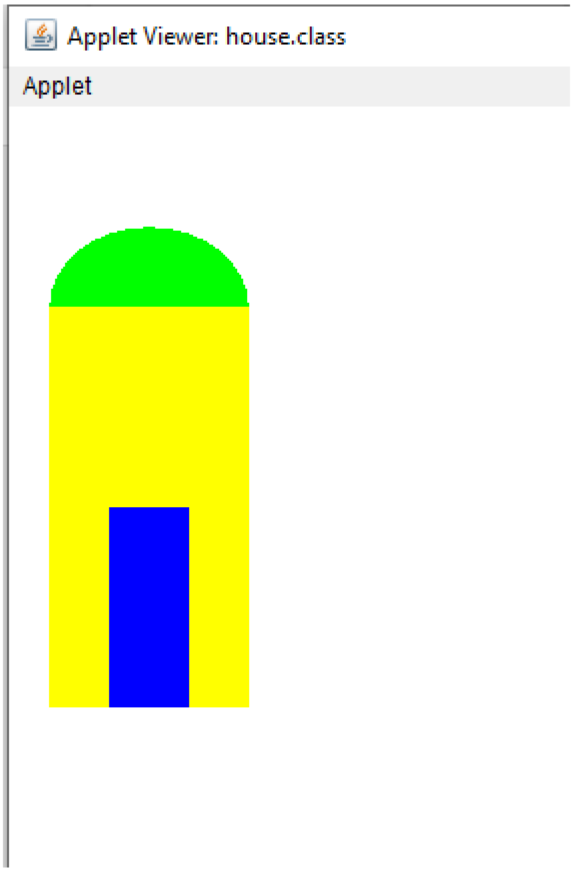
```
import java.applet.*;
import java.awt.*;
import java.awt.event.*;
/* <applet code="house.class" width="1200"
height="1200"></applet> */
public class house extends Applet implements MouseListener{
int x,y;
public void init(){
addMouseListener( this);
   }
public void paint(Graphics g){
g.setColor(Color.yellow);
g.fillRect(20,100,100,200);
g.setColor(Color.green);
g.fillArc(20,60,100,80,0,180);
g.setColor(Color.blue);
g.fillRect(50,200,40,100);
 if(x>200 && x<300 && y>200 && y<300)
      {
         g.setColor(Color.red);
         g.fillRect(50,200,40,100);
      }

}
   public void mouseClicked(MouseEvent e)
   {

   }
   public void mouseEntered(MouseEvent e)
   {

   }
```

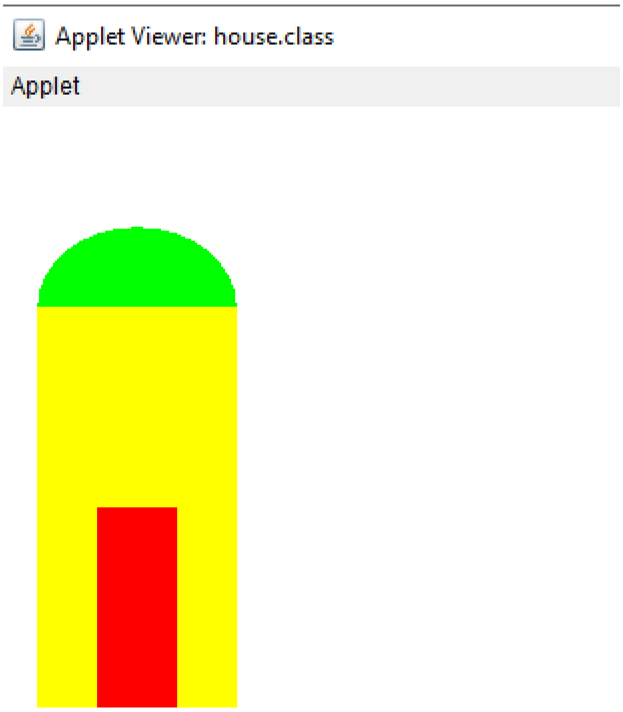
```
    public void mouseExited(MouseEvent e) {

    }

    public void mousePressed(MouseEvent e)
    {
        x=e.getX();
        y=e.getY();
        repaint();
    }
    public void mouseReleased(MouseEvent e)
    {

    }

}
```

RESULT

The above program is executed and obtains the output.

OUTPUT





Applet Viewer: house.class
Applet

PROGRAM -5

AIM: Implement a simple calculator using AWT components.

ALGORITHM

```
STEP 1: Start
STEP 2:Define a class ‘calculator’ that extends Frame and implements
        ActionListener interface.
STEP 3:Using TextField class object, construct the required no. of text fields wide enough
        to   hold the values entered by the user.
STEP 4: Using Label class objects, construct and provide the appropriate labels.
STEP 5: Using Button class object, construct labeled buttons that send the instances
        of ActionEvent.
STEP 6:Call addActionListener() method to send events from the button to the new listener.
STEP 7:Get the string values from text fields and then parse them as integers.
STEP 8:Perform various methods to add, subtract, multiply and divide those integers
STEP 9: Stop
```

PROGRAM CODE

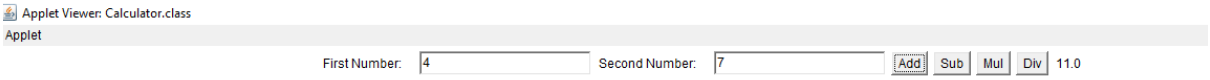
calculator.java

```
import java.applet.*;
import java.awt.*;
import java.awt.event.*;
/* <applet code = "Calculator.class" width = "200" height =
"400"></applet> */
public class Calculator extends Applet implements ActionListener
{
  Label l1,l2,l3;
  TextField t1,t2;
  Button b1,b2,b3,b4;

  public void init()
  {
   setLayout(new FlowLayout());

   l1=new Label("First Number:");
   t1=new TextField(20);

   l2=new Label("Second Number:");
   t2=new TextField(20);

   b1=new Button("Add");
   b2=new Button("Sub");
   b3=new Button("Mul");
   b4=new Button("Div");

   l3=new Label("Result");

   add(l1);
   add(t1);
   add(l2);
   add(t2);
   add(b1);
   add(b2);
   add(b3);
   add(b4);
   add(l3);

   b1.addActionListener(this);
   b2.addActionListener(this);
   b3.addActionListener(this);
   b4.addActionListener(this);
  }

 public void actionPerformed(ActionEvent ae)
  {
  Double num1=Double.parseDouble(t1.getText());
  Double num2=Double.parseDouble(t2.getText());
  if(ae.getSource()==b1)
   {
    Double value=num1+num2;
    l3.setText(""+value);
   }
  if(ae.getSource()==b2)
   {
    Double value=num1-num2;
    l3.setText(""+value);
   }
  if(ae.getSource()==b3)
   {
    Double value=num1*num2;
    l3.setText(""+value);
   }
  if(ae.getSource()==b4)
   {
    Double value=num1/num2;
    l3.setText(""+value);
   }
 }
}
```

RESULT

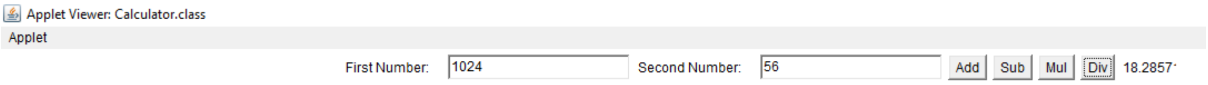
The above program is executed and obtains the output.

OUTPUT


Applet Viewer: Calculator.class

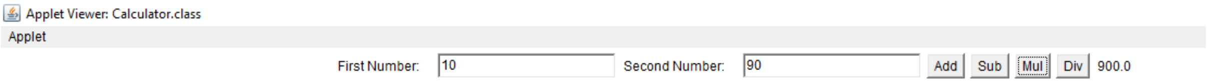
Applet

First Number: 4 Second Number: 7 11.0


Applet Viewer: Calculator.class

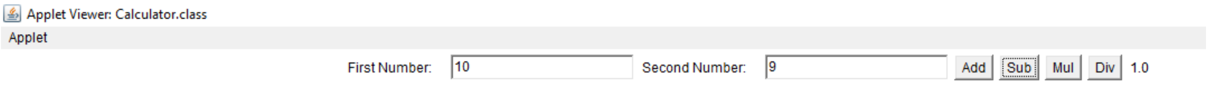
Applet

First Number: 1024 Second Number: 56 18.2857


Applet Viewer: Calculator.class

Applet

First Number: 10 Second Number: 90 900.0


Applet Viewer: Calculator.class

Applet

First Number: 10 Second Number: 9 1.0

PROGRAM -6

AIM: Develop a program that has a Choice component which contains the names of shapes such as rectangle, triangle, square and circle. Draw the corresponding shapes for given parameters as per user's choice.

ALGORITHM

```
STEP 1: Start
STEP 2: Define a class 'shape' that extends Applet class and implements
            ItemListener interface.:
STEP 3: Declare a new constructor of the Choice class to create an empty Choice menu.
STEP 4: Use add() method to include items in the menu.
STEP 5: Using getSelectedItem() method, get the item chosen by the user and
            repaint   accordingly
STEP 6: Stop
```

PROGRAM CODE

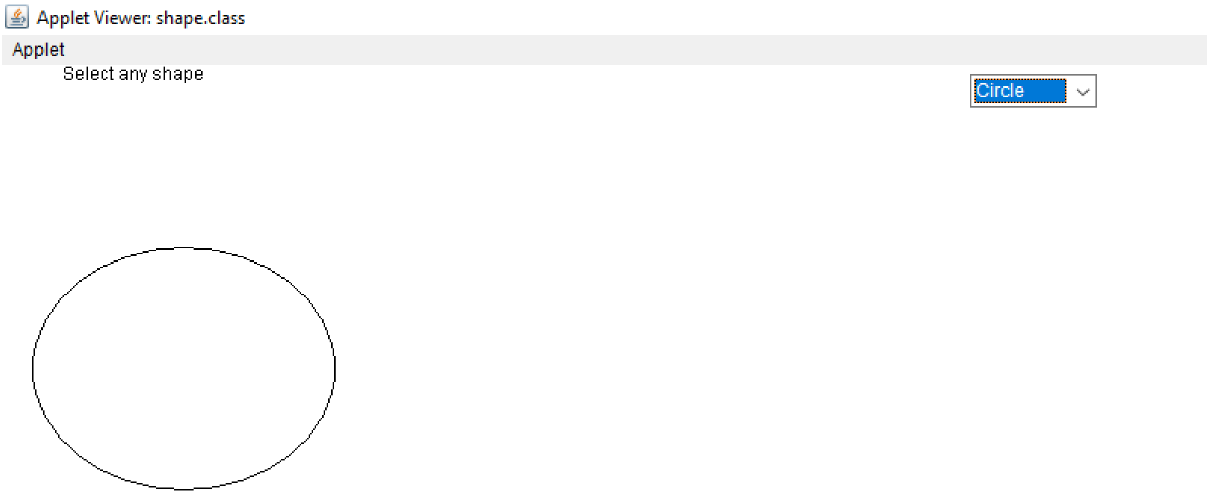
shape.java

```
import java.applet.*;
import java.awt.*;
import java.awt.event.*;
/* <applet code ="shape.class" width=700 height=900></applet> */
public class shape extends Applet implements ItemListener{
    String figure;
    Choice ch = new Choice();
        public void init(){
                ch.add("Rectangle");
                ch.add("Circle");
                ch.add("Triangle");
                ch.add("Square");
                add(ch);

        ch.addItemListener(this);
}
    public void itemStateChanged(ItemEvent e){
        figure = ch.getSelectedItem();
        repaint();
}
  public void paint(Graphics g){
        g.drawString("Select any shape",40,10);
        if (figure.equals("Rectangle")){
                g.drawRect(50,50,330,120);
        }
         if (figure.equals("Square")){
                g.drawRect(50,50,100,100);
        }
              if (figure.equals("Triangle")){
                int x[] = {200,300,100};
                int y[] = {200,300,300};
                int n = 3;
                g.drawPolygon(x, y, n);
        }

               if (figure.equals("Circle")){
                g.drawOval(20, 120, 200, 160);
        }
    }
}
```

RESULT

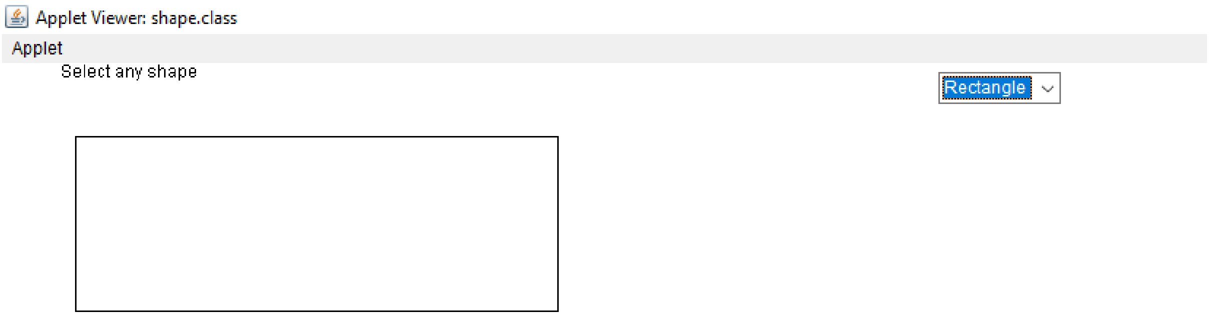
The above program is executed and obtains the output.

OUTPUT



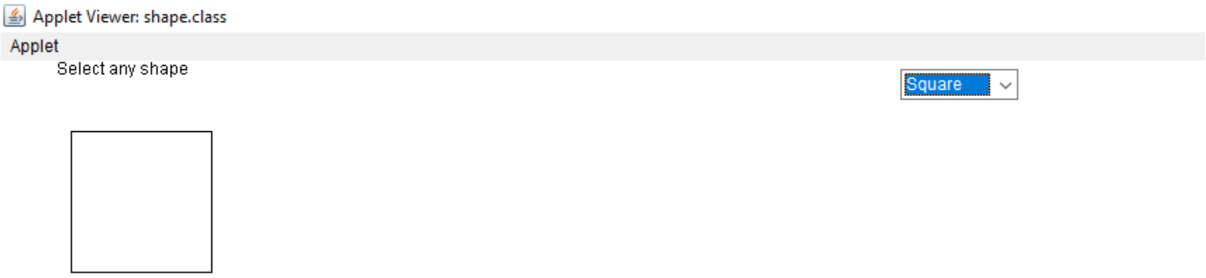
Select any shape

Circle



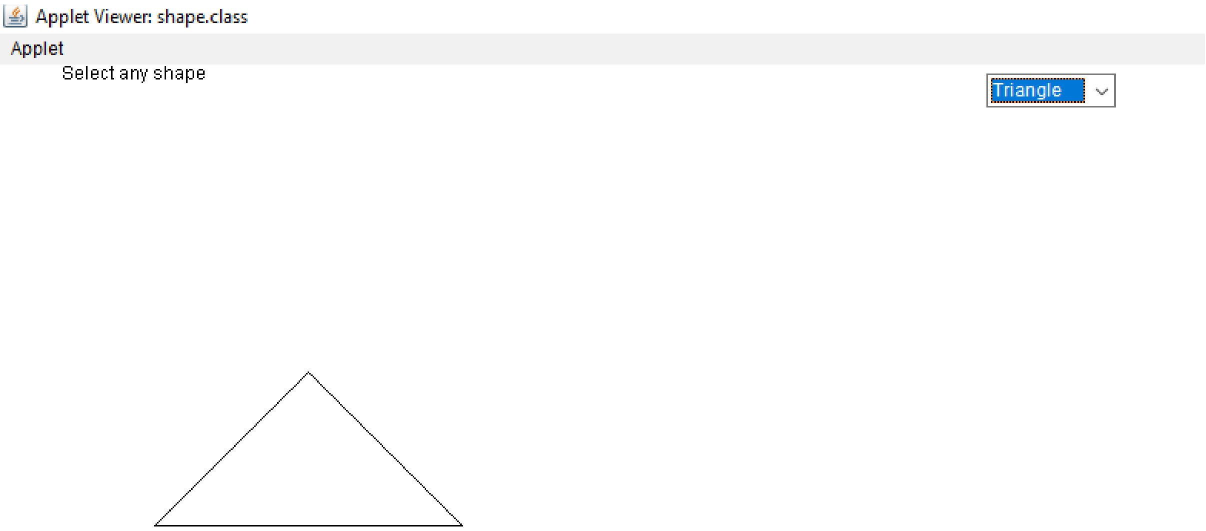
Select any shape

Rectangle



Select any shape

Square



Select any shape

Triangle

PROGRAM -7

AIM: Develop a program to handle all mouse events and window events

ALGORITHM

STEP 1: Start
STEP 2:Define a class Mouse that extends Applet class and implements
MouseListener interface
STEP 3:Define methods to add MouseListener to the panel
STEP 4:Using getX() and getY() methods, get the location (or movements) of the
mouse pointer on the panel. Use them to display the necessary message in the output.
STEP 5:Display the appropriate message in the output
STEP 6: Stop

PROGRAM CODE

mouse.java

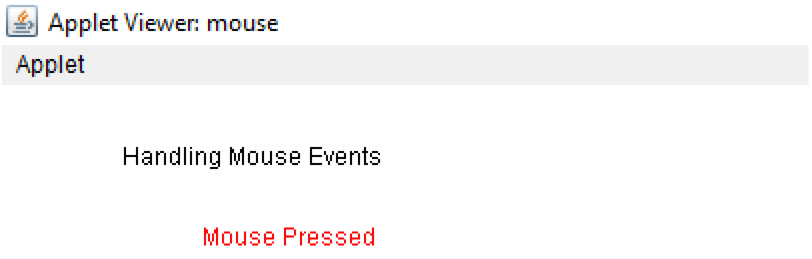
---window events not done---

```
import java.awt.*;
import java.applet.*;
import java.awt.event.*;
/*<applet code="mouse" width=300 height=300>
</applet>*/
public class mouse extends Applet implements MouseListener
{
int x=0;
int y=0;
String msg="";
public void init()
{
addMouseListener(this);
}
public void mouseClicked(MouseEvent me)
{
x=20;
y=40;
msg="Mouse Clicked";
repaint();
}
public void mousePressed(MouseEvent me)
{
x=30;
y=60;
msg="Mouse Pressed";
repaint();
}
public void mouseReleased(MouseEvent me)
{
x=30;
y=60;
msg="Mouse Released";
repaint();
}
public void mouseEntered(MouseEvent me)
{
x=40;
y=80;
msg="Mouse Entered";
repaint();
}
public void mouseExited(MouseEvent me)
{
x=40;
y=80;
msg="Mouse Exited";
repaint();
}
public void mouseDragged(MouseEvent me)
{
x=me.getX();
y=me.getY();
showStatus("Currently mouse dragged"+x+" "+y);

repaint(); }
public void mouseMoved(MouseEvent me)
{
x=me.getX();
y=me.getY();
showStatus("Currently mouse is at"+x+" "+y);
repaint();
}
public void paint(Graphics g)
{
g.drawString("Handling Mouse Events",60,40);
g.setColor(Color.red);
g.drawString(msg,100,80);
}
}
```

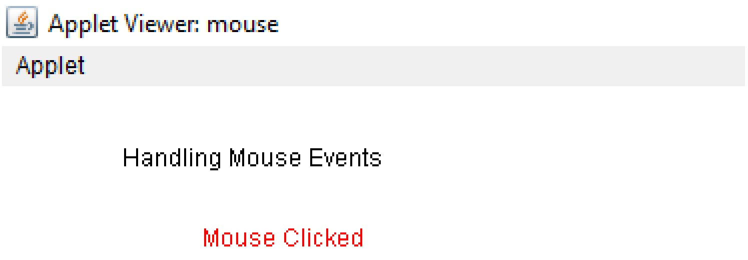
RESULT

The above program is executed and obtains the output.

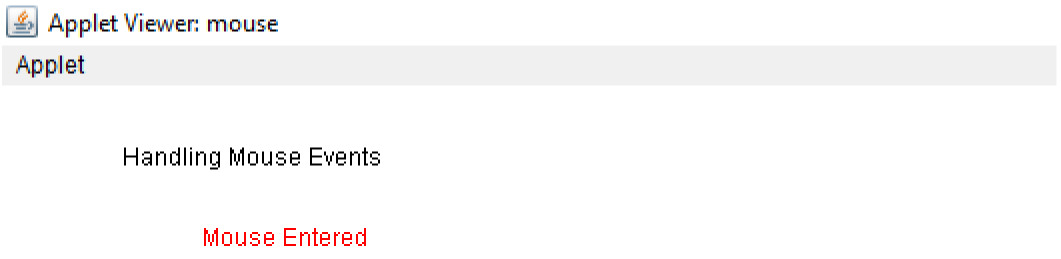
OUTPUT



```
Handling Mouse Events

      Mouse Pressed
```



```
Handling Mouse Events

      Mouse Clicked
```



```
Handling Mouse Events

      Mouse Entered
```

PROGRAM -8

AIM: Develop a program to handle Key events.

ALGORITHM

```
STEP 1: Start
STEP 2:Define a class key that extends Applet and implements KeyListener.
STEP 3: Define methods to add KeyListener to the panel
STEP 4: Using getKeyChar(), get the unicode and character representation of the
        key pressed. Use them to display the necessary message in the output.
STEP 5: Stop
```

PROGRAM CODE

events.java

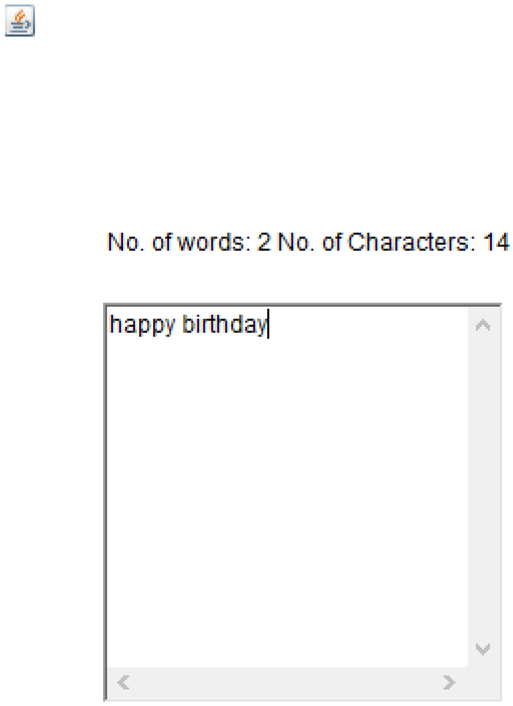
```java
import java.awt.*;
import java.awt.event.*;
import java.awt.event.KeyListener;
/* <applet code="events.class" width ="500"
height="700"></applet> */
public class events extends Frame implements KeyListener{
Label l;
TextArea a;
public events(){
l=new Label();
l.setBounds(60,100,420,60);
a=new TextArea();
a.setBounds(60 , 160 , 200 , 200);
a.addKeyListener(this);
add(l);
add(a);
setSize(800,800);
setLayout(null);
setVisible(true);
}
public void keyPressed( KeyEvent e ) {
}
public void keyReleased( KeyEvent e )
{
String t=a.getText();
String w[]=t.split("\\s");
l.setText("No. of words: "+ w.length +" No. of Characters: "+
t.length() );
}
public void keyTyped( KeyEvent e ) {
```

	}
	public static void main(String[] args) {
	new events();
	}
	}

RESULT

The above program is executed and obtains the output.

OUTPUT



```
No. of words: 2 No. of Characters: 14

happy birthday
```

LAB CYCLE 6

PROGRAM -1

AIM: Program to list the sub directories and files in a given directory and also search for a file name.

ALGORITHM

```
STEP 1: Start
STEP 2:: Create a class named ‘MyFilenameFilter’ that implements interface.
STEP 3: Create an object for the class File to to initialize its constructor with the file source.
STEP 4: Using list(), get the names of all the files present in the directory.
STEP 5: Create an object for the FileNameFilter interface that contains the method
        Boolean accept ( File dir, String name) to test if a specified file should be included
        in the file list or not.
STEP 6: Filter accordingly and store the file names to the list.
STEP 7:Display the list
STEP 8: Stop
```

PROGRAM CODE

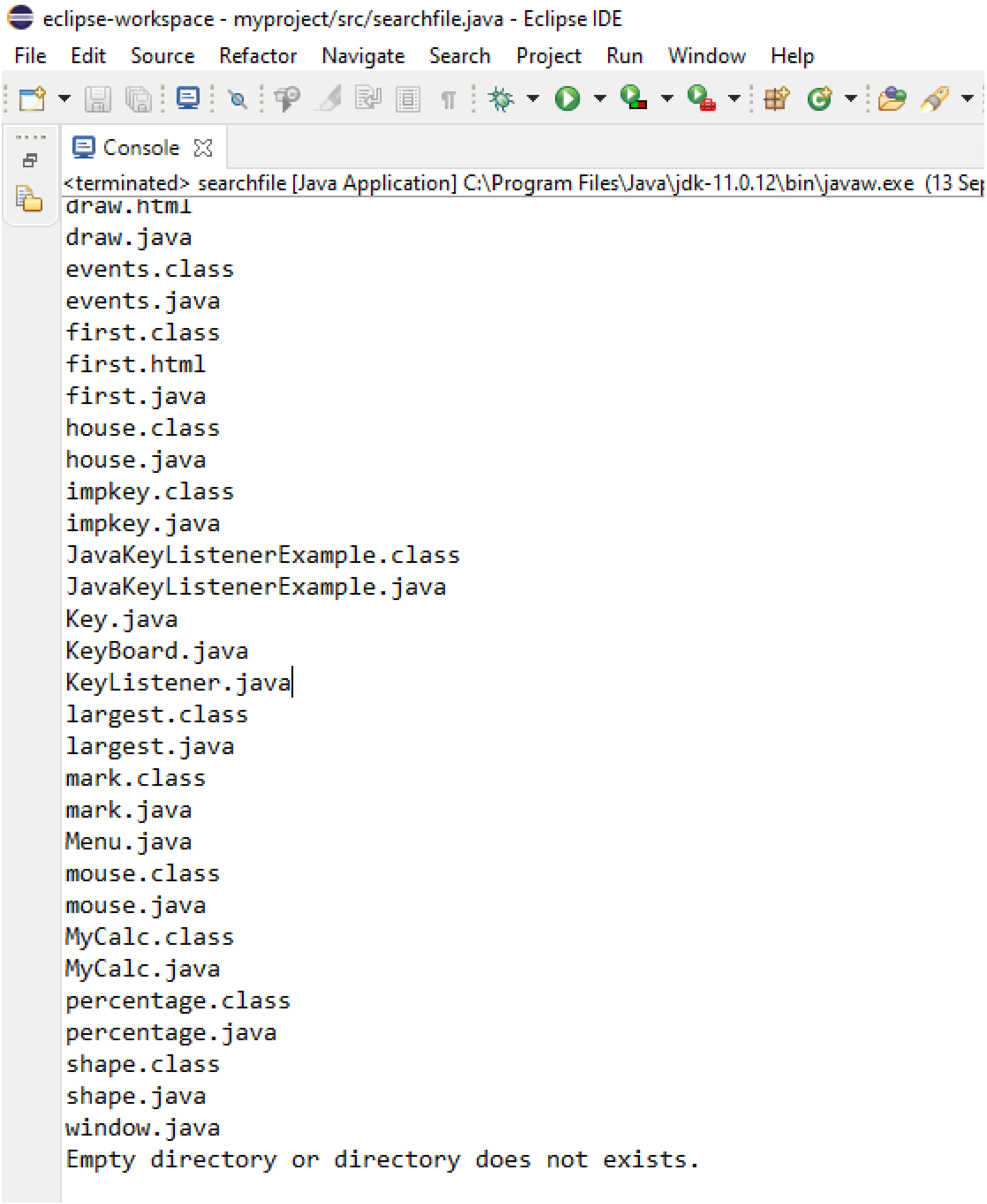
searchfile.java

```
import java.io.File;
import java.io.FilenameFilter;
class MyFilenameFilter implements FilenameFilter {

    String init;

    public MyFilenameFilter(String initials)
    {
        this.init = init;
    }

    public boolean accept(File dir, String name)
    {
        return name.startsWith(init);
    }
}

class searchfile {
  public static void main(String[] args) {
    File file = new File("C:\\Users\\Siji Jose\\Desktop\\java");
    String[] fileList = file.list();

    for(String str : fileList) {
      System.out.println(str);
    }
    File directory = new File("/home/user/");

    MyFilenameFilter filter= new MyFilenameFilter("mark.java");

    String[] fli = directory.list(filter);
    if (fli == null) {
      System.out.println(
        "Empty directory or directory does not exists.");
    }
    else {

      for (int i = 0; i < fli.length; i++) {
        System.out.println(fli[i]+" FOUND");
      }
    }
  }
}
```

RESULT

The above program is executed and obtains the output.

OUTPUT


eclipse-workspace - myproject/src/searchfile.java - Eclipse IDE
File
Edit
Source
Refactor
Navigate
Search
Project
Run
Window
Help
Console

```
<terminated> searchfile [Java Application] C:\Program Files\Java\jdk-11.0.12\bin\javaw.exe  (13 Sep
draw.html
draw.java
events.class
events.java
first.class
first.html
first.java
house.class
house.java
impkey.class
impkey.java
JavaKeyListenerExample.class
JavaKeyListenerExample.java
Key.java
KeyBoard.java
KeyListener.java
largest.class
largest.java
mark.class
mark.java
Menu.java
mouse.class
mouse.java
MyCalc.class
MyCalc.java
percentage.class
percentage.java
shape.class
shape.java
window.java
Empty directory or directory does not exists.
```

PROGRAM -2

AIM: Write a program to write to a file, then read from the file and display the contents on the console.

ALGORITHM

STEP 1: Start
STEP 2:Create a class named ‘writeread’.
STEP 3:Create an object of the class File to initialize its constructor with the file source.
STEP 4: Create and use an object for the FileWriter class to write the file.
STEP 5: Create and use an object for the BufferedReader class to read the stream
of characters the specified file.
STEP 6: Display the contents read from the file on the console.
STEP 7: Stop

PROGRAM CODE

writeread.java

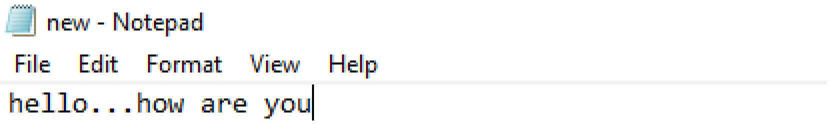
```
import java.io.FileReader;
import java.io.FileWriter;
import java.io.IOException;
class writeread{
        public static void main(String[] args)
        {
                try {
                        FileReader fr = new
FileReader("new.txt");
                        FileWriter fw = new
FileWriter("output.txt");
                        String str = "";
                        int i;
                        while ((i = fr.read()) != -1) {
                                str += (char)i;
                        }

                        System.out.println(str);
                        fw.write(str);
                        fr.close();
                        fw.close();
                        System.out.println("File reading and
writing both done");
                }
                catch (IOException e) {
                        System.out.println("There are some
IOException");
```

	} } }

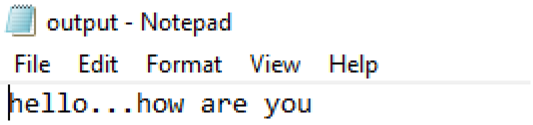
RESULT

The above program is executed and obtains the output.

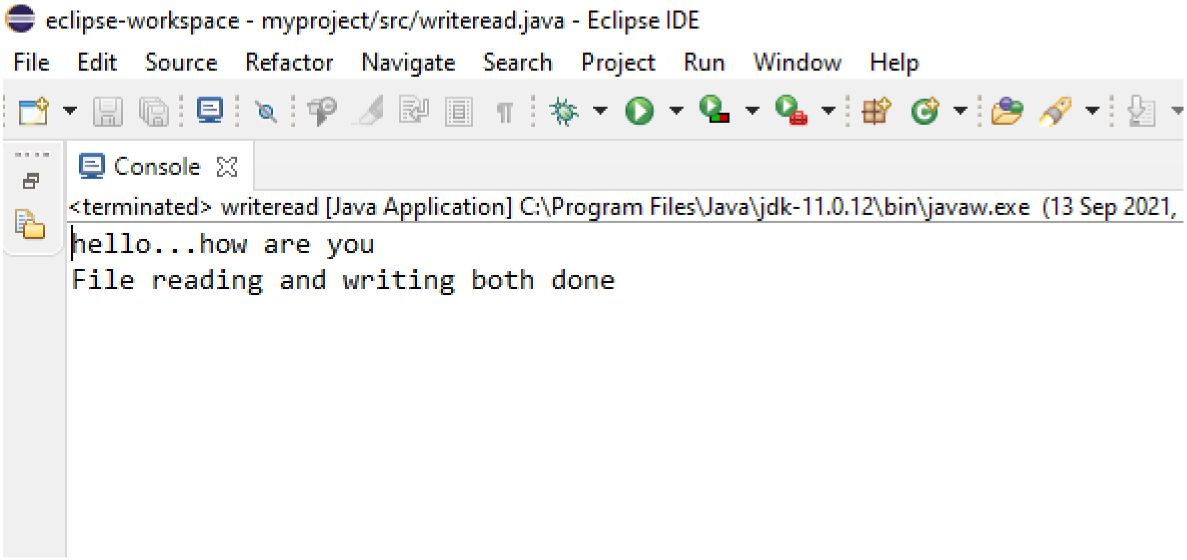
OUTPUT



```
hello...how are you
```



```
hello...how are you
```


eclipse-workspace - myproject/src/writeread.java - Eclipse IDE
File Edit Source Refactor Navigate Search Project Run Window Help
Console

```
<terminated> writeread [Java Application] C:\Program Files\Java\jdk-11.0.12\bin\javaw.exe  (13 Sep 2021,
hello...how are you
File reading and writing both done
```

PROGRAM -3

AIM: Write a program to copy one file to another.

ALGORITHM

STEP 1: Start
STEP 2:Create a class named ‘Copy’.
STEP 3:Create and use an object for the BufferedReader class to read the stream of characters from the specified file.
STEP 4:Create and use an object for the FileWriter class to write the stream of characters read by the BufferedReader, to the file.
STEP 5: Display the appropriate message on the console.
STEP 6: Stop

PROGRAM CODE

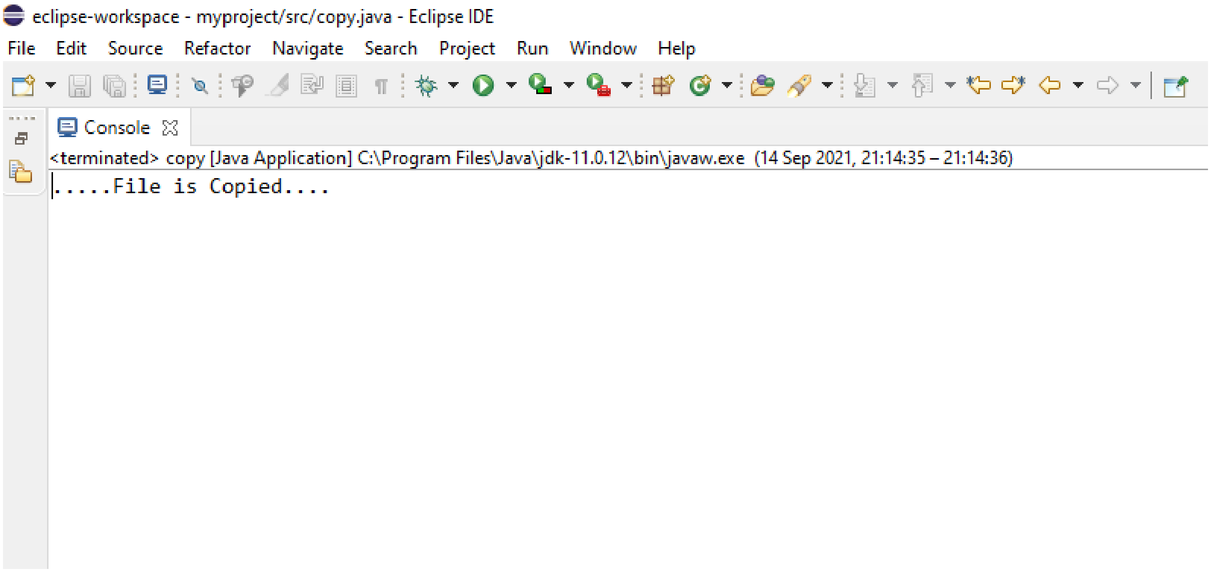
copy.java

```
import java.io.*;
class copy{
    public static void main(String args[])throws IOException {
        FileInputStream fr =new FileInputStream("new.txt");
        FileOutputStream fw=new FileOutputStream("copy.txt")
;
        System.out.println(".....File is Copied....");
        int c;
        while((c=fr.read())!=-1)
        fw.write((char)c);
        fr.close();
        fw.close();
      }
 }
```

RESULT

The above program is executed and obtains the output.

OUTPUT


eclipse-workspace - myproject/src/copy.java - Eclipse IDE
File Edit Source Refactor Navigate Search Project Run Window Help
Console

```
<terminated> copy [Java Application] C:\Program Files\Java\jdk-11.0.12\bin\javaw.exe  (14 Sep 2021, 21:14:35 – 21:14:36)
.....File is Copied....
```

PROGRAM -4

AIM: Write a program that reads from a file having integers. Copy even numbers and odd numbers to separate files.

ALGORITHM

STEP 1: Start
STEP 2:Create a class named ‘numbers’
STEP 3: Create an object for the class File to initialize its constructor with the given file.
STEP 4: Get user inputs via the console, for the integers to be inserted into the file.
STEP 5: Using an object for the FileWriter class, write those integers into the file.
STEP 6:Using objects for the FileOutputStream class, create two separate files to store even and odd integers respectively and copy the integers accordingly to separate files just created.
STEP 7: Stop

PROGRAM CODE

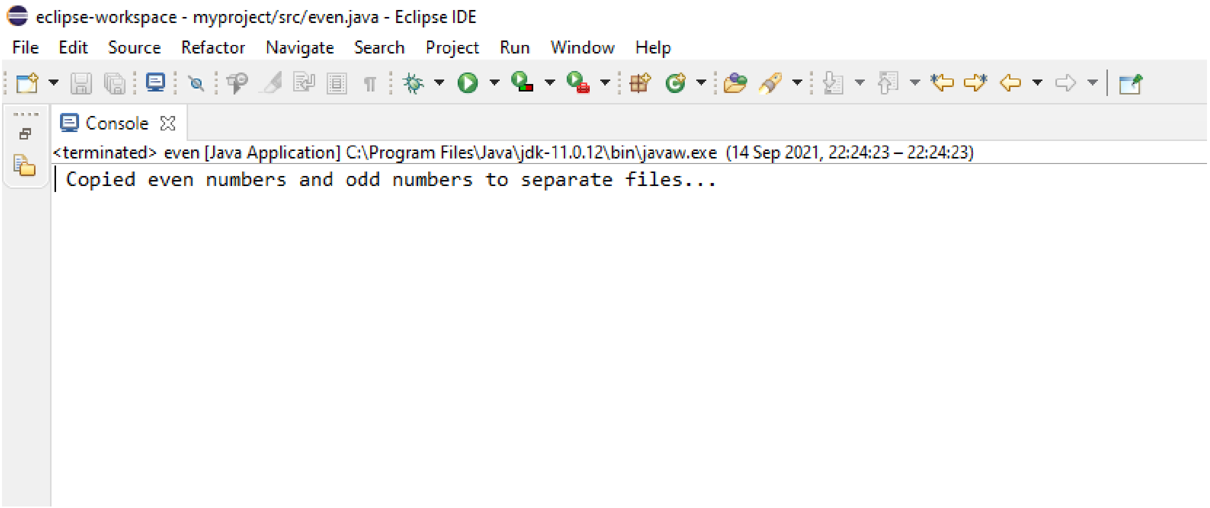
numbers.java

```
import java.io.*;

 class numbers
{
         public static void main(String args[]) throws IOException
{
   FileInputStream fr = new FileInputStream("num.txt");
   FileOutputStream fw1 = new FileOutputStream("even.txt");
   FileOutputStream fw2 = new FileOutputStream("odd.txt");
   System.out.println(" Copied even numbers and odd numbers to separate files...");
   int i;
   while((i=fr.read()) != -1)

   {
    if(i%2==0)
    fw1.write(i);
    else
    fw2.write(i);
   }

   fr.close();
   fw1.close();
   fw2.close();

 }}
```

RESULT

The above program is executed and obtains the output.

OUTPUT



eclipse-workspace - myproject/src/even.java - Eclipse IDE
File Edit Source Refactor Navigate Search Project Run Window Help
Console

```
<terminated> even [Java Application] C:\Program Files\Java\jdk-11.0.12\bin\javaw.exe (14 Sep 2021, 22:24:23 – 22:24:23)
Copied even numbers and odd numbers to separate files...
```

PROGRAM -5

AIM: Client server communication using Socket – TCP/IP

ALGORITHM

```
STEP 1: Start
STEP 2: To create the Client application, create an instance of ClientSocket class.
        2.1: Initiate connection to the server using hostname and a port number.
        2.2: Send data to the server using an OutputStream object.
        2.3: Read data from the server using an InputStream object.
        2.4: Close the connection.
STEP.3: To create the Server application, create an instance of ServerSocket class.
        3.1: Wait till a connection is established.
                    Socket s = ss.accept();
        3.2: Receive data from the client using an InputStream object.
        3.3: Send data to the client using an OutputStream object.
        3.4: Close the connection.
STEP 4: Stop
```

PROGRAM CODE

client.java

```
import java.net.*;
import java.io.*;
public class client {
public static void main(String args[])throws Exception{
        try {
          Socket sk=new Socket("localhost",2665);
          PrintWriter pw=new
PrintWriter(sk.getOutputStream(),true);
          pw.println("Client: Hello Server!!!");

InputStreamReader isr=new
InputStreamReader(sk.getInputStream());
BufferedReader br=new BufferedReader(isr);
String s1=br.readLine();
System.out.println(" Message is "+s1);
pw.close();
sk.close();
        }
        catch(Exception e) {}
}
}
```

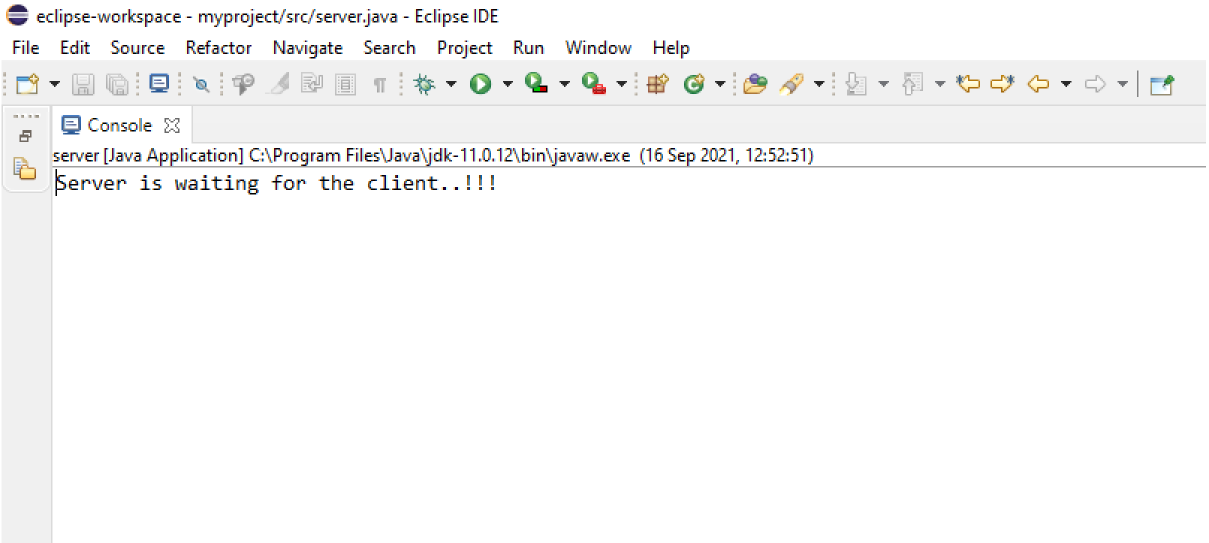
server.java

```
import java.net.*;
import java.io.*;
public class server {
public static void main(String args[])throws Exception{
          try {
ServerSocket ss=new ServerSocket(2665);
System.out.println("Server is waiting for the client..!!!");
Socket sk=ss.accept();
System.out.println("Connection established...!!");
InputStreamReader isr=new
InputStreamReader(sk.getInputStream());
BufferedReader br=new BufferedReader(isr);
String s=br.readLine();
System.out.println("Message is"+s);
PrintWriter pw=new PrintWriter(sk.getOutputStream(),true);
pw.println("Server: Hello Client!!!");
pw.close();
}
catch(Exception e) {}
}
}
```

RESULT

The above program is executed and obtains the output.

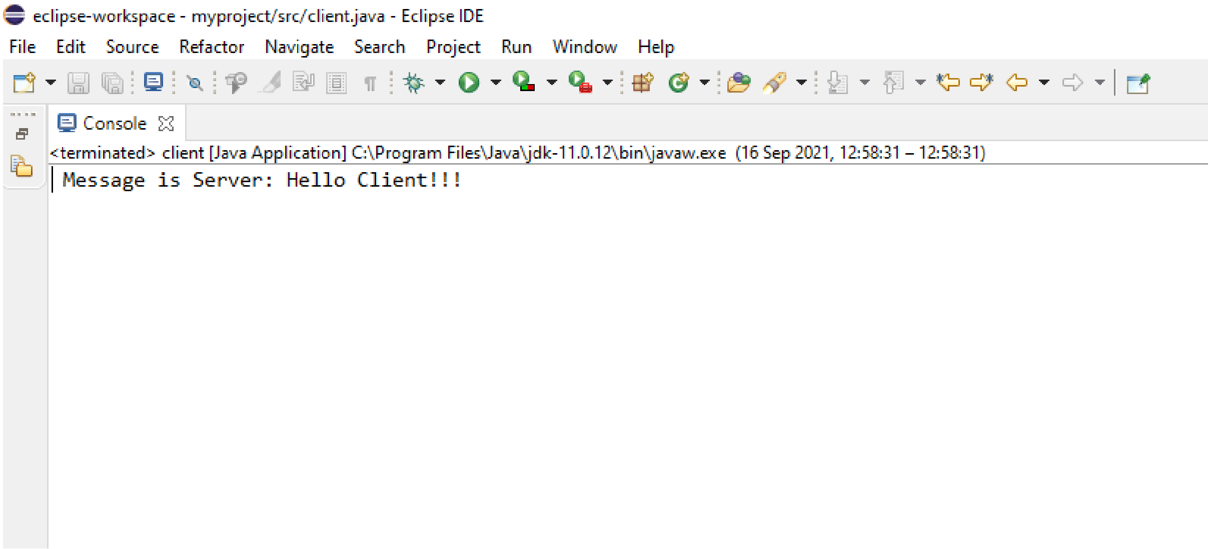
OUTPUT



eclipse-workspace - myproject/src/server.java - Eclipse IDE
File Edit Source Refactor Navigate Search Project Run Window Help
Console

server [Java Application] C:\Program Files\Java\jdk-11.0.12\bin\javaw.exe (16 Sep 2021, 12:52:51)

```
Server is waiting for the client..!!!
```



eclipse-workspace - myproject/src/client.java - Eclipse IDE
File Edit Source Refactor Navigate Search Project Run Window Help
Console

<terminated> client [Java Application] C:\Program Files\Java\jdk-11.0.12\bin\javaw.exe (16 Sep 2021, 12:58:31 – 12:58:31)

```
Message is Server: Hello Client!!!
```

PROGRAM -6

AIM: Client Server communication using DatagramSocket - UDP

ALGORITHM

```
STEP 1: Start
STEP 2:Create the Client application:
        2.1: Create a DatagramSocket object to carry the packet to the destination and
             to receive it whenever the server sends any data.
        2.2: Create the packet for sending/receiving data via a DatagramSocket.
             DatagramPacket(byte buf[], int length, InetAddress inetaddress, int port):-
        2.3: Invoke a send() or receive() call on a socket object.
        2.4: Close the connection.
STEP.3: Create the Server application:
        3.1: Create a DatagramSocket object to listen at the port specified.
        3.2: Create the packet for sending/receiving data via a DatagramSocket.
        3.3: Invoke a send() or receive() call on a socket object.
        3.4: Close the connection.
STEP 4: Stop
```

PROGRAM CODE

udpclient.java

```
import java.io.*;
import java.net.*;
public class udpclient {
        public static void main(String[] args) throws IOException {
                DatagramSocket client= new DatagramSocket();
                InetAddress
add=InetAddress.getByName("localhost");
                String str ="Hello client!!!";
                byte[] buf=str.getBytes();
                DatagramPacket p=new
DatagramPacket(buf,buf.length,add,4160);
                client.send(p);
                client.close();
        }

}
```

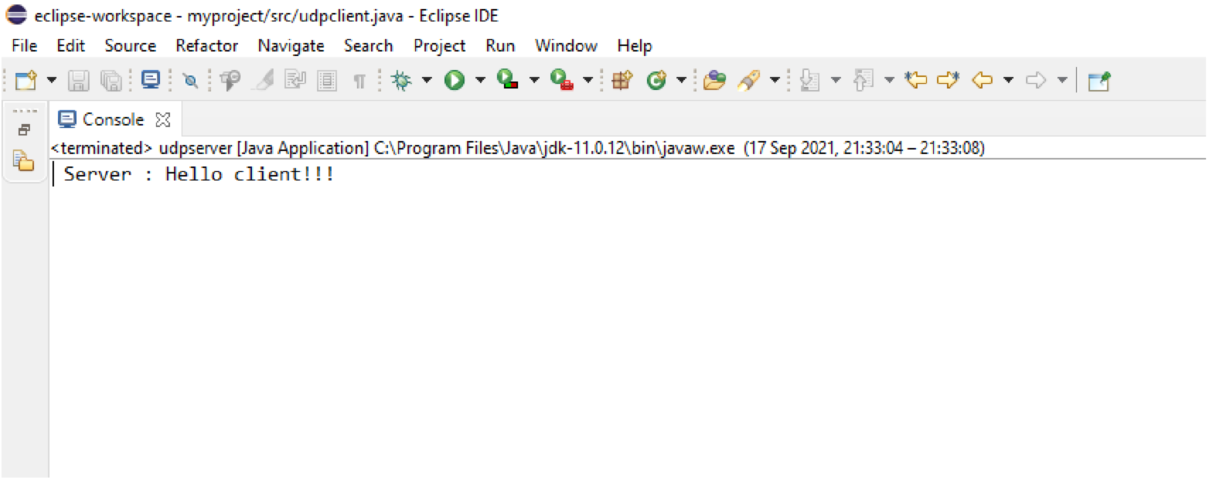
udpserver.java

```
import java.io.*;
import java.net.*;
public class udpserver {
        public static void main(String[] args) throws
IOException {
                DatagramSocket server=new
DatagramSocket(4160);
                byte[] buf=new byte[256];
                DatagramPacket packet=new
DatagramPacket(buf,buf.length);
                server.receive(packet);
                String response =new
String(packet.getData());
                System.out.println(" Server : "+response);
                server.close();
        }

}
```

RESULT

The above program is executed and obtains the output.

OUTPUT


eclipse-workspace - myproject/src/udpclient.java - Eclipse IDE
File Edit Source Refactor Navigate Search Project Run Window Help
Console

```
<terminated> udpserver [Java Application] C:\Program Files\Java\jdk-11.0.12\bin\javaw.exe (17 Sep 2021, 21:33:04 – 21:33:08)
Server : Hello client!!!
```