1.) Design a class named Fan to represent a fan. The class contains:
- Three constants named SLOW, MEDIUM and FAST with values 1,2 and 3 to denote the fan speed.
- An Int data field named speed that specifies the speed of the fan (default SLOW).
- A Boolean data field named f_on that specifies whether the fan is on(default false).
- A double data field named radius that specifies the radius of the fan (default 4).
- A data field named color that specifies the color of the fan (default blue).
- A no-arg constructor that creates a default fan.
- A parameterized constructor initializes the fan objects to given values.
- A method named display() will display description for the fan. If the fan is on, the display() method displays speed, color and radius. If the fan is not on, the method returns fan color and radius along with the message “fan is off”.
Write a test program that creates two Fan objects. One with default values and the other with medium speed, radius 6, color brown, and turned on status true. Display the descriptions for two created Fan objects.

```
package FAN;

public class GTU_PPR_1 {
       public static void main(String args[])
       {
              Fan f1 = new Fan();
              f1.display();
              Fan f2 = new Fan(2,true,6,"brown");
              f2.display();
       }

   }
   class Fan
   {
       public static final int SLOW = 1;
       public static final int MEDIUM = 2;
       public static final int FAST = 3;
       int speed;
       boolean f_on;
       double radius;
       String color;
       Fan()
       {
              speed = SLOW;
              f_on = false;
              radius = 4;
              color = "blue";
       }
       Fan(int s,boolean f, double r, String c)
       {
              speed = s;
              f_on = f;
              radius = r;
              color = c;
       }
       public void display()
       {
              if(f_on)
              {
                     System.out.println("Speed of Fan : "+speed);
                     System.out.println("Radius of Fan : "+radius);
                     System.out.println("Color of Fan : "+color);
              }
              else
              {
                     System.out.println("Fan is Off.");
                     System.out.println("Radius of Fan : "+radius);
                     System.out.println("Color of Fan : "+color);
```

```
            }
        }
    }
```

**2.) Define the Rectangle class that contains:
Two double fields x and y that specify the center of the rectangle, the data field width and height ,
A no-arg constructor that creates the default rectangle with (0,0) for (x,y) and 1 for both width and height.
A parameterized constructor creates a rectangle with the specified x,y,height and width.
A method getArea() that returns the area of the rectangle.
A method getPerimeter() that returns the perimeter of the rectangle.
A method contains(double x, double y) that returns true if the specified point (x,y) is inside this rectangle.
Write a test program that creates two rectangle objects. One with default values and other with user specified values. Test all the methods of the class for both the objects.**

```
package Area;
class Rectangle
{
    double centerX,centerY,width,height;
    double x1,x2,x3,x4,y1,y2,y3,y4;
    Rectangle()
    {
            centerX=centerY=0;
            width=height=1;
    }
    Rectangle(double x,double y,double w,double h)
    {
            centerX=x;
            centerY=y;
            width=w;
            height=h;
    }
    double getArea()
    {
            return (width*height);
    }
    double getPerimeter()
    {

            return ( 2*(width+height) );
    }
    boolean contains(double x,double y)
    {
            x1=x3= ( centerX - (width/2) );
            x2=x4= ( centerX + (width/2) );
            y1=y2= (centerY + (height/2));
            y3=y4= (centerY - (height/2));
            if( (x > x1 && x < x4) && (y < y1 && y > y3))
                    return true;
            else
                    return false;
    }
}

public class GTU_PPR_2 {
    public static void main(String args[])
    {
            Rectangle r1Obj = new Rectangle();
            Rectangle r2Obj = new Rectangle(200,200,100,50);
            System.out.println("Rectangle 1 : Area "+r1Obj.getArea());
            System.out.println("Rectangle 1 : Perimeter "+r1Obj.getPerimeter());
            System.out.println("Rectangle 1 : Contains (3,2) "+r1Obj.contains(3,2));
            System.out.println();
            System.out.println("Rectangle 2 : Area "+r2Obj.getArea());
            System.out.println("Rectangle 2 : Perimeter "+r2Obj.getPerimeter());
            System.out.println("Rectangle        2        :        Contains        (160,190)
"+r2Obj.contains(160,190));
    }

}
```

3.) Describe abstract class called Shape which has three subclasses say Triangle,Rectangle,Circle. Define one method area() in the abstract class and override this area() in these three subclasses to calculate for specific object i.e. area() of Triangle subclass should calculate area of triangle etc. Same for Rectangle and Circle.

```
 package Shape;
abstract class Shape
{
    double dimension1,dimension2,radius;
    abstract double area();
}
class Triangle extends Shape
{
    Triangle(double d1,double d2)
    {
            dimension1=d1;
            dimension2=d2;
    }
    double area()
    {
            System.out.print("Area Of Triangle : ");
            return ((dimension1*dimension2)/2);
    }
}
class Rectangle extends Shape
{
    Rectangle(double d1,double d2)
    {
            dimension1=d1;
            dimension2=d2;
    }
    double area()
    {
            System.out.print("Area Of Rectangle : ");
            return (dimension1*dimension2);
    }
}
class Circle extends Shape
{
    Circle(double d1)
    {
            radius = d1;
    }
    double area()
    {
            System.out.print("Area Of Circle : ");
            return ((22*radius*radius)/7);
    }
}

public class GTU_PPR_3 {
    public static void main(String args[])
    {
            Shape t = new Triangle(25,20);
            Shape r = new Rectangle(15,20);
            Shape c = new Circle(49);
            System.out.println(t.area());
            System.out.println();
            System.out.println(r.area());
            System.out.println();
            System.out.println(c.area());
    }
}
```

4.) The abstract Vegetable class has three subclasses named Potato, Brinjal and Tomato. Write an application that demonstrates how to establish this class hierarchy. Declare one instance variable of type String that indicates the color of a vegetable. Create and display instances of these objects. Override the toString() method of Object to return a string with the name of the vegetable and its color.

```
package Vegetable;
import java.lang.*;

abstract class Vegetable
{
   String colorVeg;
   abstract public String toString();
}
class Potato extends Vegetable
{
   public String toString()
   {
         colorVeg = "Yellow";
         return colorVeg;
   }
}
class Brinjal extends Vegetable
{
   public String toString()
   {
         colorVeg = "Purpul";
         return colorVeg;
   }
}
class Tomato extends Vegetable
{
   public String toString()
   {
         colorVeg = "Red";
         return colorVeg;
   }
}

public class GTU_PPR_4 {
   public static void main(String args[])
   {
         Vegetable p = new Potato();
         Vegetable b = new Brinjal();
         Vegetable t = new Tomato();
         System.out.println("Color of Potato : "+p.toString());
         System.out.println("Color of Brinjal : "+b.toString());
         System.out.println("Color of Tomato : "+t.toString());
   }

}
```

5.) Write a program that illustrates interface inheritance. Interface P is extended by P1 and P2. Interface P12 inherits from both P1 and P2.Each interface declares one constant and one method. class Q implements P12.Instantiate Q and invoke each of its methods. Each method displays one of the constants.

```
package Interface;
interface P
{
   public static final int p = 5;
   void methodP();
}
interface P1 extends P
{
   public static final int p1 = 10;
   void methodP1();
}
interface P2 extends P
{
   public static final int p2 = 15;
   void methodP2();
}
interface P12 extends P1,P2
{
   public static final int p12 = 20;
   void methodP12();
}
class Q implements P12
{
   public void methodP12()
   {
          System.out.println("P12 class's method and Constant : "+p12);
   }
   public void methodP1()
   {
          System.out.println("P1 class's method and Constant : "+p1);
   }
   public void methodP2()
   {
          System.out.println("P2 class's method and Constant : "+p2);
   }
   public void methodP()
   {
          System.out.println("P class's method and Constant : "+p);
   }
}

public class GTU_PPR_5 {
   public static void main(String args[])
   {
          Q obj = new Q();
          obj.methodP12();
          obj.methodP1();
          obj.methodP2();
          obj.methodP();
   }

}
```

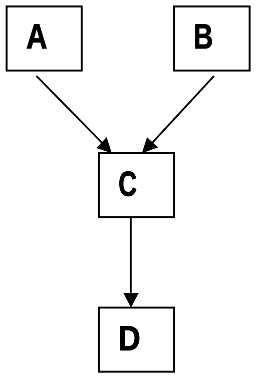
6.) The Transport interface declares a deliver() method. The abstract class Animal is the super class of the Tiger, Camel, Deer and Donkey classes. The Transport interface is implemented by the Camel and Donkey classes. Write a test program that initialize an array of four Animal objects. If the object implements the Transport interface, the deliver() method is invoked.

```
package Animale;
interface Transport
{
   void deliver();
}
abstract class Animal
{
   abstract void jungle();
}
class Tiger extends Animal
{
   public void jungle()
   {
          System.out.println("HI, I M TIGER..");
   }
}
class Camel extends Animal implements Transport
{
   public void jungle()
   {
          System.out.println("HI, I M CAMEL");
   }
```

```
    public void deliver()
    {
           System.out.println("HEY, I m interface's method");
    }
}
class Deer extends Animal
{
    public void jungle()
    {
           System.out.println("HI, I M DEER");
    }
}
class Donkey extends Animal implements Transport
{
    public void jungle()
    {
           System.out.println("HI, YOU R A DONKEY..!!!!!");
    }
    public void deliver()
    {
           System.out.println("HEY, I m interface's method");
    }
}

public class GTU_PPR_6 {
    public static void main(String args[])throws ArrayIndexOutOfBoundsException
    {
           Transport t1 = new Donkey();
           t1.deliver();
           Deer t2 = new Deer();
           t2.jungle();
           Transport t3 = new Camel();
           t3.deliver();
           Tiger t4 = new Tiger();
           t4.jungle();
    }
    }
```

7.) Explain single level and multiple inheritances in java. Write a program to demonstrate combination of both types of inheritance as shown in figure 1. i.e.hybrid inheritance
(A,B)->C->D

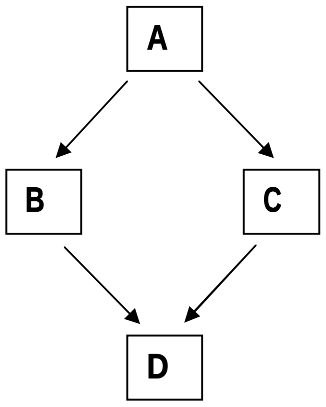


```
package Multiple_Inheritance;
import java.io.*;
interface A
    {
           void sum();
    }
interface B
    {
           void sub();
    }
class C implements A,B
    {
           int a=30;
```

```
          int b=20;
          public void sum()
          {
                System.out.println("addition="+(a+b));
          }
          public void sub()
          {
                System.out.println("subtraction="+(a-b));
          }
    }

public class D extends C
{
    public static void main(String args[])
    {
          C c= new C();
          c.sub();
          c.sum();
    }
}
```

8.) Write a program to demonstrate the mulipath inheritance for the classes having relation as show in figre 2. A->(B,C)->D



```
package multipath_inherintace;
import java.io.*;
interface  A
    {
          int a=10;
          int b=20;
    }
interface  B extends A
    {
          void sum();
    }
interface  C extends A
    {
           void mul();
    }
 class D implements B,C
{

          public void sum()
          {
                int c=a+b;
                System.out.println("Additionn="+c);
          }
          public void mul()
          {
                int d=a*b;
                System.out.println("Multiplication="+d);
          }
}


public class Multipath
{
   public static void main(String args[])
   {
          D d1 = new D();
          d1.mul();
          d1.sum();
   }
}
```

9.) **Write an applet that contains three buttons OK,CANCEL and HELP and one textfield. if OK is pressed shown on the status bar-“OK is pressed” and the text field should turn red. When CANCEL is pressed -shown on the status bar-“ CANCEL is pressed “and text field should turn green. When HELP is pressedshown on the status bar-“HELP is pressed” and the text field should turn yellow.**

```
package Applet;
import java.io.*;
import java.applet.*;
import java.awt.*;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;

public class Button_3 extends Applet implements ActionListener {

   Button ok,cancel,help;
   TextField tx;
   String msg="";
   public void init()
   {
          ok=new Button("OK");
          cancel=new Button("CANCEL");
          help=new Button("HELP");
          tx=new TextField("TYPE HERE");
          add(ok);
          add(cancel);
          add(help);
          add(tx);
          ok.addActionListener(this);
          cancel.addActionListener(this);
          help.addActionListener(this);
   }
   public void actionPerformed(ActionEvent e)
   {
          String str=e.getActionCommand();
          if(str.equals("OK"))
          {
                 msg="OK BUTTON IS PRESSED";

          }
          else if(str.equals("CANCEL"))
          {
                 msg="CANCEL BUTTON IS PRESSED";
          }
          else
          {
                 msg="HELP BUTTON IS PRESSED";
          }
          repaint();
   }
```

```
        public void paint(Graphics g)
        {
              if(msg.equals("OK BUTTON IS PRESSED"))
              {
                    tx.setBackground(Color.RED);
              }
              else if(msg.equals("CANCEL BUTTON IS PRESSED"))
              {
                    tx.setBackground(Color.GREEN);
              }
              else if(msg.equals("HELP BUTTON IS PRESSED"))
              {
                    tx.setBackground(Color.YELLOW);
              }

              else
              {
                    tx.setBackground(Color.WHITE);
              }
              showStatus(msg);
        }

    }
```

10.) Write an applet that draws four horizontal bars of equal size & of different colors such that they cover up the whole applet area. The applet should operate correctly even if it is resized.

```
    package Applet;
    import java.applet.*;
    import java.awt.*;
    import java.awt.event.ActionListener;
    import java.awt.event.ComponentEvent;
    import java.awt.event.ComponentListener;

    public class hvapplet extends Applet implements ComponentListener {

        int fullAppletHeight = 400, fullAppletWidth = 400;
         int oneRectHeight = (fullAppletHeight/4);
        public void init()
        {
              addComponentListener(this);
        }
        public void componentHidden(ComponentEvent e)
        {
              repaint();

        }
        public void componentMoved(ComponentEvent e)
        {
              repaint();
         }
        public void componentResized(ComponentEvent e)
        {
              fullAppletWidth = e.getComponent().getWidth();
              fullAppletHeight = e.getComponent().getHeight();
            oneRectHeight = (fullAppletHeight/4);
              repaint();
        }
        public void componentShown(ComponentEvent e)
        {
              repaint();
        }
        public void paint(Graphics g)
        {
            g.setColor(Color.ORANGE);
              g.fillRect(0,0,fullAppletWidth,oneRectHeight);
              g.setColor(Color.WHITE);
              g.fillRect(0,oneRectHeight,fullAppletWidth,oneRectHeight);
              g.setColor(Color.GREEN);
              g.fillRect(0,(oneRectHeight*2),fullAppletWidth,oneRectHeight);
              g.setColor(Color.BLACK);
              g.fillRect(0,(oneRectHeight*3),fullAppletWidth,oneRectHeight);
        }
    }
```

11.)Write an applet that tracks the position of the mouse when it is dragged or moved. At the current mouse position, it displays message (x, y) showing current position of the mouse. The message should disappear as soon as the user releases the mouse.

```
import java.io.*;
import java.applet.*;
import java.awt.*;
import java.awt.event.MouseListener;
import java.awt.event.MouseMotionListener;

public class MouseEvent extends Applet implements MouseListener,MouseMotionListener {

    String msg="";
    int mouseX=0,mouseY=0;
    public void init()
    {
            addMouseListener(this);
            addMouseMotionListener(this);
    }
    public void mouseClicked(java.awt.event.MouseEvent me) {
            repaint();
    }
    public void mouseEntered(java.awt.event.MouseEvent me) {
            repaint();
    }
    public void mouseExited(java.awt.event.MouseEvent me) {
            repaint();
    }
    public void mousePressed(java.awt.event.MouseEvent me) {
            repaint();
    }
    public void mouseReleased(java.awt.event.MouseEvent me) {
                        mouseX=me.getX();
                        mouseY=me.getY();
                        msg="";
                        repaint();
    }
    public void mouseDragged(java.awt.event.MouseEvent me) {
            mouseX=me.getX();
            mouseY=me.getY();
            msg="Mouse at("+mouseX+","+mouseY+")";
            repaint();
    }
    public void mouseMoved(java.awt.event.MouseEvent me) {
            mouseX=me.getX();
            mouseY=me.getY();
            msg="Mouse at("+mouseX+","+mouseY+")";
            repaint();
    }
    public void paint(Graphics g)
    {
            g.drawString(msg, mouseX, mouseY);
    }
}
```

12.)Write a program to create a frame with exit capabilities. Handle events for mouse pressed, mouse released, mouse clicked and mouse dragged by displaying appropriate message describing the event at the coordinates where the event has taken place.

```
package Frame;
import java.applet.*;
import java.awt.*;
import java.awt.event.MouseAdapter;
import java.awt.event.MouseListener;
import java.awt.event.MouseEvent;
import java.awt.event.MouseMotionAdapter;
import java.awt.event.WindowAdapter;
import java.awt.event.WindowEvent;

class MyMouseAdapter extends MouseAdapter
{
   Frame_Event_Handling fe1;
   public MyMouseAdapter(Frame_Event_Handling fe1)
   {
         this.fe1=fe1;
   }

   public void mousePressed(MouseEvent me)
   {
         fe1.mouseX=me.getX();
         fe1.mouseY=me.getY();
         fe1.str="Mouse Down at"+fe1.mouseX+","+fe1.mouseY;
         fe1.repaint();
   }

   public void mouseClicked(MouseEvent me)
   {
         fe1.mouseX=me.getX();
         fe1.mouseY=me.getY();
         fe1.str="Mouse Clicked at"+fe1.mouseX+","+fe1.mouseY;
         fe1.repaint();
   }

   public void mouseReleased(MouseEvent me)
   {
         fe1.mouseX=me.getX();
         fe1.mouseY=me.getY();
         fe1.str="Mouse Released at"+fe1.mouseX+","+fe1.mouseY;
         fe1.repaint();
   }

}

class MyMouseMotionAdapter extends MouseMotionAdapter
{
   Frame_Event_Handling fe1;
   public MyMouseMotionAdapter(Frame_Event_Handling fe1)
   {
         this.fe1=fe1;
   }

   public void mouseDragged(MouseEvent me)
   {
         fe1.mouseX=me.getX();
         fe1.mouseY=me.getY();
         fe1.str="Mouse Draggedd at"+fe1.mouseX+","+fe1.mouseY;
         fe1.repaint();
   }
}

class MyWindowAdapter extends WindowAdapter
{
   public void windowClosing(WindowEvent we)
   {
          System.exit(0);
   }
}
public class Frame_Event_Handling extends Frame {
   String str="";
   int mouseX=30;
   int mouseY=30;
   public  Frame_Event_Handling()
   {
          addMouseListener(new MyMouseAdapter(this));
          addMouseMotionListener(new MyMouseMotionAdapter(this) );
          addWindowListener(new MyWindowAdapter());
   }
   public void paint(Graphics g)
   {
          g.drawString(str, mouseX, mouseY);
   }
   public static void main(String args[])
   {
          Frame_Event_Handling fe = new Frame_Event_Handling();
          fe.setSize(new Dimension(300,300));
          fe.setTitle("HI THIS IS THE EXAMPLE OF FRAME EVENT HANDLING");
          fe.setVisible(true);
   }
}
```

13.)Write a complete program to create a frame for providing GUI to implement a stack for storing integer numbers. There are two buttons called PUSH & POP and a text field. Clicking of button PUSH pushes the number entered in the text field onto the stack. The click of button POP pops an element from the stack and displays that in the text field.

```
package Frame;
import java.applet.*;
import java.awt.*;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import javax.swing.*;

public class Frame_Stack {
   JLabel lbl;
   JButton pop,push;
   JTextField text;
   String stcarray[] = new String[10];
   int top=-1;
    Frame_Stack()
   {
          JFrame frm = new JFrame("Stack Implementation in Frame");
          frm.setLayout(new FlowLayout());
          frm.setSize(300,300);
          frm.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
          push=new JButton("PUSH");
          pop=new JButton("POP");
          text=new JTextField(12);
          push.addActionListener(new ActionListener()
          {

                 public void actionPerformed(ActionEvent ae) {
                                    ++top;
                                    if(top<100)
                                    {
                                           stcarray[top]=text.getText();
                                           lbl.setText("PUSH ELEMENT :"+text.getText());
                                    }
                                    else
                                    {
                                           lbl.setText("STACK IS FULL");
                                    }
                        }
                }
        );
        pop.addActionListener(new ActionListener()
        {

                public void actionPerformed(ActionEvent ae) {
                                    if(top>=0)
                                    {
                                            lbl.setText("POP ELEMENT :"+stcarray[top]);
                                            --top;
                                    }
                                    else
                                    {
                                            lbl.setText("STACK IS EMPTY");
                                    }

                }
        }
        );
        frm.add(text);
        frm.add(push);
        frm.add(pop);
        lbl=new JLabel("PRESS BUTTON");
        frm.add(lbl);
        frm.setVisible(true);

                }
        public static void main(String args[])
        {
                SwingUtilities.invokeLater(new Runnable()
                {
                        public void run()
                        {
                                new Frame_Stack();
                        }
                }
                );
        }
    }
```

14.) Write an event handling program to handle e-mail sending form using your creativity.

```
    package Applet;
    import java.awt.*;
    import java.awt.event.ActionEvent;
    import java.awt.event.ActionListener;
    import java.applet.*;

    public class Email_Event_Handling extends Applet implements ActionListener {

        Label l1,l2,l3;
        TextField t1,t2;
        TextArea ta;
        Button b1,b2;
         public void init()
                {
                l1=new Label("TO: ");
                t1= new TextField(20);
                l2 = new Label("Subject: ");
                t2 = new TextField(20);
                l3= new Label("Body: ");
                ta = new TextArea(20,20);
                b1 = new Button("SUBMIT");
                b2=new Button("CANCEL");
                add(l1);
                add(t1);
```

```
          add(t2);
          add(l2);
          add(l3);
          add(ta);
          add(b1);
          add(b2);
          b1.addActionListener(this);
          b2.addActionListener(this);
          }
          public void actionPerformed(ActionEvent e) {
                        String str = e.getActionCommand();
                        if(str.equals("SUBMIT"))
                        {
                               showStatus("Submit Successfully..!!");
                        }
                        else
                        {
                               showStatus("Cancel Button Pressed");
                        }

    }
}
```

15.) Write an event handling program to handle feedback form of GTU examination system using your creativity.

```
package Applet;
import java.applet.*;
import java.awt.*;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;

public class GTU_FeedBack_Form extends Applet implements ActionListener {

    Button Submit,Cancel;
    Label tx_En_No,tx_Name,tx_Branch,tx_College,tx_City;
    TextField tf_En_No,tf_Name,tf_Branch,tf_College,tf_City;
    TextArea ta_Suggestions;
    public void init()
    {
          Submit = new Button("SUBMIT");
          Cancel = new Button("CANCEL");
          tx_En_No = new Label("Enter your enrollement number:");
          tf_En_No = new TextField(20);
          tx_Name = new Label("Enter your name:");
          tf_Name = new TextField(20);
          tx_Branch = new Label("Enter your branch:");
          tf_Branch = new TextField(20);
          tx_College = new Label("Enter your college name:");
          tf_College = new TextField(30);
          tx_City = new Label("Enter your city name:");
          tf_City = new TextField (20);
          add(Submit);
          add(Cancel);
          add(tx_En_No);
          add(tf_En_No);
          add(tx_Name);
          add(tf_Name);
          add(tx_Branch);
          add(tf_Branch);
          add(tx_College);
          add(tf_College);
          add(tx_City);
          add(tf_City);
          Submit.addActionListener(this);
          Cancel.addActionListener(this);
          setName("GTU FEED BACK FORM");
```

```
      }

      public void actionPerformed(ActionEvent e) {
                    String str=e.getActionCommand();
                    if(str.equals("SUBMIT"))
                    {
                          showStatus("Successfully submited data");
                    }
                    else
                    {
                          showStatus("Error..!!!");
                    }
      }

   }
```

16.) Write a program using BufferedInputStream, FileInputStream, BufferedOutputStream, FileOutputStream to copy Content of one file File1.txt into another file File2.txt.

```
package FILEIO;
import java.io.*;

public class FileIO {
    public static void main(String args[]) throws IOException{
          int ch;
          FileInputStream fin = new FileInputStream("pqr.txt");
          BufferedInputStream bfin = new BufferedInputStream(fin);
          FileOutputStream fout = new FileOutputStream("xyz.txt");
          BufferedOutputStream bfout = new BufferedOutputStream(fout);
          while((ch=bfin.read()) != -1)
                {
                 bfout.write(ch);
                        System.out.println((char) ch);
                }
          bfout.close();
                fout.close();
                bfin.close();
                fin.close();
    }
}
```

17.) Write a program to display the bytes of a file in reverse sequence. Provide the name of the file as a command line argument. (Use RandomAccessFile)

```
package FILEIO;

import                                                            java.io.*;
public                              class                      File_Reverse{
     public     static     void     main(String[]     args)     throws     IOException{
             File            file            =            new            File(args[0]);
             if(!file.exists())
             {
                     System.out.println("File         does         not         exist.");
                     System.exit(0);
             }
             try{
                     RandomAccessFile     ra     =     new     RandomAccessFile(file,"r");
                     int                                                i=(int)ra.length();
                     System.out.println("Length:            "            +            i);

                     for(int      ct      =      (i-1);      ct      >      0;      ct--){
                             ra.seek(ct);
                             byte            b            =            ra.readByte();
                             System.out.print(b);                        }
                     ra.close();
             }
                     catch(IOException                                              e)
```

```
                {
                System.out.println(e.getMessage());
                }
        }
    }
```

18.) Write a program that takes input for filename and search word from command line arguments and checks whether that file exists or not. If exists, the program will display those lines from a file that contains given search word.

```
package FILEIO;

import java.io.*;
class File_Serach
{
   public static void main(String args[]) throws IOException
   {
          int fileExistFlag;
          boolean wordExists=false;
                File f = new File(args[0]);
          fileExistFlag = f.exists()?1:0;
          if(fileExistFlag == 1)
          {
                 BufferedReader fileReader = new BufferedReader(new FileReader(f));
                 String wholeLine;
                 String word = args[1];
                 while ((wholeLine = fileReader.readLine()) != null)
                 {
                        String wordArr[] = wholeLine.split(" ");
                        for(String wordElem:wordArr)
                        {
                               if(wordElem.equalsIgnoreCase(word))
                               {
                                      System.out.println("'"+word+"'"  +  "  exists  in  line
'" + wholeLine+"'");
                                      wordExists = true;
                               }
                        }
                 }
                 if(!wordExists)
                        System.out.println("Word Not Exists in File");
          }
          else
          {
                 System.out.println("File Not Exists");
          }

   }
}
```

19.) Write a program to replace all "word1" by "word2" from a file1, and output is written to file2 file and display the no. of replacement.

```
package File;
import java.io.*;
import java.lang.*;
public class Word_Change {
   public static void main(String args[])
   {
          try
          {
                 File f = new File("File1.txt");
                 BufferedReader reader = new BufferedReader(new FileReader(f));
                 String line="",oldtxt="";
                 while((line = reader.readLine()) != null)
```

```
                    {
                           oldtxt+=line +"\n";
                    }
                    reader.close();
                    String newtxt=oldtxt.replace("word1", "word2");
                    FileWriter writer = new FileWriter("File2.txt");
                    writer.write(newtxt);
             }
             catch(IOException io)
             {
                    io.getMessage();
             }
      }

   }
```

20.) Write a program to create directories (/home/abc/bcd/def/ghi/jkl) in the home directory /home/abc and list the files and directories showing file/directory, file size. Read-write-execute permissions. Write destructor to destroy the data of a class.

```
package Directory;
import java.io.*;

public class Directory_Prgm {
       public static void main(String args[])
       {
              String str = "/bcd/def/ghi/jkl";
              File f1 = new File("C:/home/abc");
              File f2 = new File(f1,str);
              try
              {
                     boolean b = f2.mkdirs();
                     if (b)
                     {
                            System.out.println("Directories:" + str + "created");
                     }
              }
              catch(Exception e)
              {
                     System.out.println("Eroor..!!"+e);
              }
              String file[] = f1.list();
              for(int i=0;i<file.length;i++)
              {
                     File f = new File("C:/home/abc"+file[i]);
                     if(f.isDirectory())
                     {
                            System.out.println(file[i]+"Directory");
                     }
                     else
                     {
                            String s = f.canRead()?"Readable":"not Readable";
                            String s1 = f.canRead()?"Editable":"not editable";
                            System.out.println(file[i] +"\tfile\t" + f.length() +"\t" + s1);
                     }
              }
       }
}
```

21.) Write a method for computing xy by doing repetitive multiplication. x and y are of type integer and are to be given as command line arguments. Raise and handle exception(s) for invalid values of x and y. Also define method main. Use finally in above program and explain its usage.

```
package Power;

public class Power_Example {
       public static void main(String args[])
       {
```

```
		int x,y,z;
		try
		{
			x=Integer.parseInt(args[0]);
			y=Integer.parseInt(args[1]);
			z=1;
			for(int i=0;i<y;i++)
			{
				z=z*x;
			}
			System.out.println(x+"^"+y+":" +z);
		}
		catch(NumberFormatException n)
		{
			System.out.println("Number Format Error..!!"+n);
		}
		finally
		{
			System.out.println("It executes every time..!!");
		}
	}

}
```

22.) It is required to have total two threads, both capable of acting as a produce as well as a consumer. If first thread acts as a producer then, the second thread becomes the consumer and vice-versa. They communicate with each other through a buffer, storing one integer number. One of the threads initiates the communication by sending 1 to the other thread. The second thread, on receiving 1 sends 2 to the first thread. On receiving 2, the first thread sends three integer numbers, one by one to the second thread. The second thread consumes the numbers by displaying them. Both threads terminate after that. Note that both threads must be capable of initiating the communication. Write complete multi-threaded program to meet above requirements.

```
package Multithread;

class CI
{
	int n;
	boolean valueSet = false;
	synchronized int recieve()
	{
		while(!valueSet)
			try
			{
				wait();
			}
			catch(InterruptedException e)
			{
				System.out.println("InterruptedException caught");
			}
		System.out.println("Recieve : " + n);
		valueSet = false;
		notify();
		return n;
	}
	synchronized void send(int n)
	{
		while(valueSet)
			try
			{
				wait();
			}
			catch(InterruptedException e)
			{
				System.out.println("InterruptedException caught");
			}
```

```
            this.n = n;
            valueSet = true;
            System.out.println("Send : "+n);
            notify();
        }
}
class Producer implements Runnable
{
        CI ci;
        Producer(CI ci)
        {
            this.ci = ci;
            new Thread(this,"Producer").start();
        }
        public void run()
        {
            int i=0;
            while(true)
            {
                ci.send(i++);
            }
        }
}
class Consumer implements Runnable
{
        CI ci;
        Consumer(CI ci)
        {
            this.ci = ci;
            new Thread(this,"Consumer").start();
        }
        public void run()
        {
            while(true)
            {
                ci.recieve();
            }
        }
}

public class Mul_Producer_Consumer {
        public static void main(String args[])
        {
            CI ci = new CI();
            new Producer(ci);
            new Consumer(ci);
            System.out.println("Press Control-C to Stop.");
        }

}
```

23) Write a program that generates custom exception if any integer value given from its command line arguments is negative.

```
class NegativeNumberException extends Exception {
        private int detail;
        NegativeNumberException(int a) {
            detail = a;
        }
        public String toString() {
            return "NegativeNumberException [" + detail + "]";
        }
}
class CustomExceptionDemo {
        public static void main(String args[]) {
            try {
                int n = Integer.parseInt(args[0]);
                if(n < 0)
```

```
                    {
                          throw new NegativeNumberException(n);
                    }
                    else
                    {
                          System.out.println("You have entered positive number...");
                    }
              }
              catch (NegativeNumberException e)
              {
                    System.out.println("Caught " + e);
              }
       }
}
```

24) Write a program to check that whether the name given from command line is file or not? If it is a file then print the size of file and if it is directory then it should display the name of all files in it.

```
import java.io.File;
class DirList {
       public static void main(String args[]) {
              String dirname = args[0];
              File f1 = new File(dirname);
              if(f1.isFile())
              {
                    System.out.println("File size: " + f1.length() + " Bytes");
              }
              else if(f1.isDirectory())
              {
                    System.out.println("Directory of " + dirname);
                    String s[] = f1.list();
                    for (int i=0; i < s.length; i++) {
                          File f = new File(dirname + "/" + s[i]);
                          if (f.isDirectory()) {
                                System.out.println(s[i] + " is a directory");
                          }
                          else {
                                System.out.println(s[i] + " is a file");
                          }
                    }
              }
       }
}
```

25) Write a program that counts the no. of words in a text file. The file name is passed as a command line argument. The program should check whether the file exists or not. The words in the file are separated by white space characters.

```
import java.io.*;

class WordCount
{
       public static void main(String[] args) throws Exception
       {
              FileInputStream f = new FileInputStream("FileInputStreamDemo.java");
              int l = f.available();
              int wordcount = 0;
              for(int i=0;i<l;i++)
              {
                    if((char)f.read() ==  ' ' || (char)f.read() ==  '\t' || (char)f.read()
==  '\n')
                          wordcount++;
                    //System.out.print((char)f.read());
              }
              System.out.println("No. of words = " + wordcount);
              f.close();
       }
}
```

26) Write an applet that draws four horizontal bars of equal size & of different colors such that they cover up the whole applet area. The applet should operate correctly even if it is resized.

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

public class Prg4bOr extends Applet
{
	public void paint(Graphics g)
	{
		int y = 0;
		g.setColor(Color.RED);
		g.fillRect(0,y,getWidth(),getHeight()/4);
		y+=getHeight()/4;
		g.setColor(Color.GREEN);
		g.fillRect(0,y,getWidth(),(int)getHeight()/4);
		y+=getHeight()/4;
		g.setColor(Color.BLUE);
		g.fillRect(0,y,getWidth(),(int)getHeight()/4);
		y+=getHeight()/4;
		g.setColor(Color.YELLOW);
		g.fillRect(0,y,getWidth(),(int)getHeight()/4);
	}
}
```

27) Write a program that creates three threads. Make sure that the main thread executes last.

```
class NewThread implements Runnable
{
	String name; // name of thread
	Thread t;
	NewThread(String threadname)
	{
		name = threadname;
		t = new Thread(this, name);
		System.out.println("New thread: " + t);
		t.start(); // Start the thread
	}
	// This is the entry point for thread.
	public void run()
	{
		try
		{
			for(int i = 5; i > 0; i--)
			{
				System.out.println(name + ": " + i);
				Thread.sleep(1000);
			}
		}
		catch (InterruptedException e)
		{
			System.out.println(name + "Interrupted");
		}
		System.out.println(name + " exiting.");
	}
}
class MultiThreadDemo
{
	public static void main(String args[])
	{
		NewThread n1 = new NewThread("One"); // start threads
		NewThread n2 = new NewThread("Two");
		NewThread n3 = new NewThread("Three");
		try
		{
			n1.t.join();
			n2.t.join();
			n3.t.join();
		}
		catch (InterruptedException e)
		{
			System.out.println("Main thread Interrupted");
		}
		System.out.println("Main thread exiting.");
	}
}
```

28) Write a program that uses thread synchronization to guarantee data integrity in a multithreaded application.

```
// This program uses a synchronized block.
class Callme
{
	void call(String msg)
	{
		System.out.print("[" + msg);
		try
		{
			Thread.sleep(1000);
		}
		catch (InterruptedException e)
		{
			System.out.println("Interrupted");
		}
		System.out.println("]");
	}
}
class Caller implements Runnable
{
	String msg;
	Callme target;
	Thread t;
	public Caller(Callme targ, String s)
	{
		target = targ;
		msg = s;
		t = new Thread(this);
		t.start();
	}
	// synchronize calls to call()
	public void run()
	{
		synchronized(target)
		{ // synchronized block
			target.call(msg);
		}
	}
}
class Prg5aOr
{
	public static void main(String args[]) \
	{
		Callme target = new Callme();
		Caller ob1 = new Caller(target, "Hello");
		Caller ob2 = new Caller(target, "Synchronized");
		Caller ob3 = new Caller(target, "World");
		// wait for threads to end
		try
		{
			ob1.t.join();
			ob2.t.join();
			ob3.t.join();
		}
```

```
                catch(InterruptedException e)
                {
                        System.out.println("Interrupted");
                }
        }
}
```