Java lab programs:

Exp1: To debug step by step with small program of about 10 to 15 lines which contains at least one if else condition and a for loop.

------------------------------------------------------

Exp2: AIM: To develop a program that works as simple calculator .Uses grid layout to arrange buttons for the digits and for the +,-,\*,% operations. Add text field to display the results, Handle any possible exceptions like divide by zero.

----------------------------------------------------------------

Exp3:a) To develop an Applet in java that displays a Simple Message

import java.applet.\*;

import java.awt.\*;

public class myapplet extends Applet

{

public void init()

{

setForeground(Color.green);

setBackground(Color.red);

}

public void paint(Graphics g)

{

g.drawString("hello applet",50,50);

}

}

/\*

<html>

<applet code="myapplet.class" height=500 width=500>

</applet>

</html>

\*/

Exp3.b) To develop an Applet in java that receives an integer in one Text Field, and computes its Factorial value and returns it in another text Field ,when button named “Compute” is clicked

import java.applet.\*; import java.awt.event.\*; import java.awt.\*;

public class FactorialApplet extends Applet implements ActionListener

{

Label l1,l2; TextField t1,t2; Button b1,b2; public void init()

{

l1=new Label("Enter a value: "); l2=new Label("Result:");

t1=new TextField(10);

t2=new TextField(10);

b1=new Button("Calculate"); b2=new Button("Clear");

add(l1);

add(t1);

add(b1);

add(b2);

add(l2);

add(t2); b1.addActionListener(this); b2.addActionListener(this);

}

public void actionPerformed(ActionEvent ae)

{

int n=Integer.parseInt(t1.getText()); int fact=1;

if(ae.getSource()==b1)

{

if(n==0||n==1)

{

fact=1; t2.setText(String.valueOf(fact));

}

else

{

for(int i=1;i<=n;i++) fact=fact\*i;

}

t2.setText(String.valueOf(fact));

}

else if(ae.getSource()==b2)

{

t1.setText("");

t2.setText("");

}

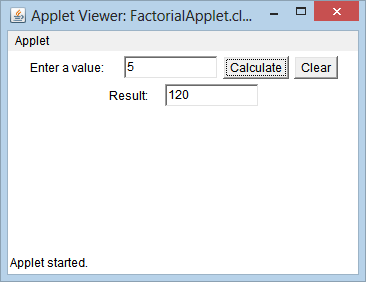
}

}

/\*<applet code="Factorial.class" width=500 height=500>

</applet> \*/

Output



Exp4: To creates User Interface to perform Integer Divisions. The user enters two numbers in text fields, Num1 and Num2.The division of Num1 and Num2 is displayed in the result field when the divide button clicked. If Num1 or Num2 were not integer, the program would throw a NumberFormatException, If Num2 is Zero, and the program would throw an Arithmetic Exception. Display the Exception in message box.

import java.applet.\*; import java.awt.\*; import java.awt.event.\*;

public class DivideDemo extends Applet implements ActionListener,TextListener

{

Label l1,l2,l3; TextField t1,t2,t3; Button b1;

int x,y; float res=0;

public void init()

{

l1=new Label("Num1:"); l2=new Label("Num2:"); l3=new Label("Num3:"); t1=new TextField(10); t2=new TextField(10); t3=new TextField(10); b1=new Button("Display"); add(l1);

add(t1);

add(l2);

add(t2);

add(b1);

add(l3);

add(t3); b1.addActionListener(this); t1.addTextListener(this); t2.addTextListener(this);

}

public void actionPerformed(ActionEvent ae)

{

try

{

int x=Integer.parseInt(t1.getText()); int y=Integer.parseInt(t2.getText()); if(ae.getSource()==b1)

res=x/y;

}

catch(NumberFormatException nfe)

{

javax.swing.JOptionPane.showMessageDialog(this, "Please enter Integer

value");

}

catch(ArithmeticException afe)

{

javax.swing.JOptionPane.showMessageDialog(this,"Please enter non-zero value");

}

t3.setText(String.valueOf(res));

}

public void textValueChanged(TextEvent te)

{

if(!t1.getText().equals("")&&(!t2.getText().equals("")))

{

b1.setEnabled(true);

}

else

b1.setEnabled(false);

}

}

/\*

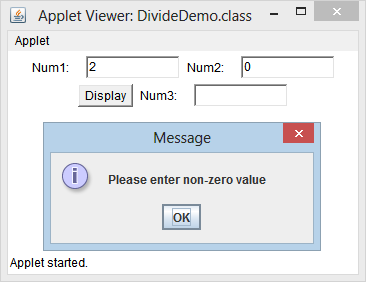
<html>

<applet code="DivideDemo" width=500 height=500>

</applet>

</html> \*/

Output



Exp5: To implements a multithread application that has three threads .First thread generates random integer for every second and if the value is even ,second thread computes the square of number and prints .If the value is odd ,the third thread will print the value of cube of number.

Exp 6:**Write a java program for the following :**

**Create a doubly linked list of elements and display the elements**

public class DoublyLinkedList {

    //Represent a node of the doubly linked list

    class Node{

        int data;

        Node previous;

        Node next;

        public Node(int data) {

            this.data = data;

        }

    }

    //Represent the head and tail of the doubly linked list

    Node head, tail = null;

    //addNode() will add a node to the list

    public void addNode(int data) {

        //Create a new node

        Node newNode = new Node(data);

        //If list is empty

        if(head == null) {

            //Both head and tail will point to newNode

            head = tail = newNode;

            //head's previous will point to null

            head.previous = null;

            //tail's next will point to null, as it is the last node of the list

            tail.next = null;

        }

        else {

            //newNode will be added after tail such that tail's next will point to newNode

            tail.next = newNode;

            //newNode's previous will point to tail

            newNode.previous = tail;

            //newNode will become new tail

            tail = newNode;

            //As it is last node, tail's next will point to null

            tail.next = null;

        }

    }

    //display() will print out the nodes of the list

    public void display() {

        //Node current will point to head

        Node current = head;

        if(head == null) {

            System.out.println("List is empty");

            return;

        }

        System.out.println("Nodes of doubly linked list: ");

        while(current != null) {

            //Prints each node by incrementing the pointer.

            System.out.print(current.data + " ");

            current = current.next;

        }

    }

    public static void main(String[] args) {

        DoublyLinkedList dList = new DoublyLinkedList();

        //Add nodes to the list

        dList.addNode(6);

        dList.addNode(2);

        dList.addNode(1);

        dList.addNode(4);

        dList.addNode(7);

        //Displays the nodes present in the list

        dList.display();

    }

}

**Output:**

Nodes of doubly linked list:

6 2 1 4 7

---------------------------------------------------------------------------------------

Exp7: To simulate a Traffic Light. The program lets the use select one of three lights :red, yellow or Green with radio buttons. On selecting radio button, an appropriate message with “stop” or “Ready” or “GO” should appear above the button in selected color. Intially ,there is no message Shown.

import java.applet.\*; import java.awt.\*; import java.awt.event.\*;

public class Signals extends Applet implements ItemListener

{

String msg="";

Checkbox stop,ready,go;

CheckboxGroup cbg;

public void init()

{

cbg = new CheckboxGroup();

stop = new Checkbox("Stop", cbg, false); ready = new Checkbox("Ready", cbg, false); go= new Checkbox("Go", cbg, false); add(stop);

add(read);

add(go);

stop.addItemListener(this); ready.addItemListener(this); go.addItemListener(this);

}

public void itemStateChanged(ItemEvent ie)

{repaint();

}

public void paint(Graphics g)

{

msg=cbg.getSelectedCheckbox().getLabel(); g.drawOval(165,40,50,50); g.drawOval(165,100,50,50); g.drawOval(165,160,50,50);

if(msg.equals("Stop"))

{

g.setColor(Color.red); g.fillOval(165,40,50,50);

}

else if(msg.equals("Ready"))

{

g.setColor(Color.yellow);

g.fillOval(165,100,50,50);

}

else

{

g.setColor(Color.green); g.fillOval(165,160,50,50);

} } }

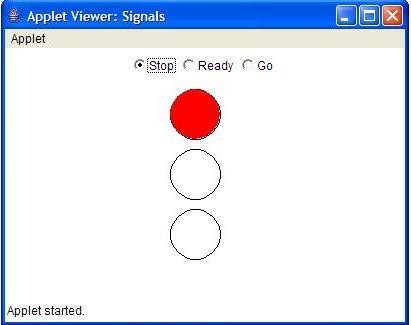
/\*<html>

<applet code="signals.class" width=400 height=250>

</applet>

</html> \*/

**Output:**



Exp8: To create an abstract class named shape that contains two integers and an empty method named printArea .Provide three classes named Rectangle ,Triangle and Circle subclass that each one of the classes extends the Class Shape .Each one of the classes contains only the method printArea() that prints the area of Shape.

Exp9: To display a table using Label in Grid Layout. Suppose that a table named Table.txt is stored in a text file. The First line in the file is the header, and the remaining lines correspond rows in table. The elements are separated by commas.

-----------------------------------------------------------------------------

Exp 10: To handle all mouse events and show event name at the center of the window when the mouse event is fired. (without Adapter Classes)

10.a) import java.awt.\*;

import java.awt.event.\*;

public class MouseListenerExample extends Frame implements MouseListener,MouseMotionListener{

Label l;

MouseListenerExample()

{

addMouseListener(this); addMouseMotionListener(this);

l=new Label(); l.setBounds(20,50,100,20); add(l);

setSize(300,300);

setLayout(null);

setVisible(true);

}

public void mouseClicked(MouseEvent e) { l.setText("Mouse Clicked");

}

public void mouseEntered(MouseEvent e) { l.setText("Mouse Entered");

}

public void mouseExited(MouseEvent e) { l.setText("Mouse Exited");

}

public void mousePressed(MouseEvent e) { l.setText("Mouse Pressed");

}

public void mouseReleased(MouseEvent e) { l.setText("Mouse Released");

}

public void mouseDragged(MouseEvent e) { l.setText("Mouse dragged");

}

public void mouseMoved(MouseEvent e) { l.setText("Mouse moved");

}

public static void main(String[] rgs) { new MouseListenerExample();

} }

-

Exp 10.b) **MOUSE EVENTS WITH ADAPTER CLASS program**

import java.awt.\*;

import java.awt.event.\*;

public class adapter1 extends MouseAdapter

{

Frame f;

adapter1()

{

f=new Frame();

f.addMouseListener(this);

f.setSize(400,400);

f.setVisible(true);

}

public void mouseClicked(MouseEvent e)

{

Graphics g = f.getGraphics();

g.setColor(Color.blue);

g.drawString("mouse clicked",50,50);

}

public static void main(String args[])

{

adapter1 obj=new adapter1();

}

}

-------------------------------------------------------------------------------

Exp 11: To load the names and phone numbers from the text file where data is organized as one line per record and each field in record are separated by a tab(\t).It takes a name or phone number as input and prints corresponding other value from hash table(hint: use Hash Table)

Exp12: Write a java program that correctly implements the producer –consumer problem using the concept of inter thread communication

Exp13: Write a java program to list all the files in a directory including the files present in all its subdirectories.ADDITIONAL PROGRAMS

1. WRITE A JAVA PROGRAM TO DEMONSTRATE MULTIPLE INHERITANCE
2. WRITE A JAVA PROGRAM TO DEMONSTRATE POLYMORPHISM
3. WRITE A JAVA PROGRAM TO DEMONSTRATE USER DEFINED EXCEPTION
4. WRITE A JAVA PROGRAM TO DEMONSTRATE PARAMETER PASSING TO AN APPLET
5. WRITE A JAVA PROGRAM TO DEMONSTRATE KEY EVENTS

import java.awt.\*;

import java.awt.event.\*;

public class KeyListenerExample extends Frame implements KeyListener

{

Label l;

TextArea area; KeyListenerExample()

{

l=new Label(); l.setBounds(20,50,100,20); area=new TextArea(); area.setBounds(20,80,300, 300); area.addKeyListener(this);

add(l);add(area); setSize(400,400); setLayout(null); setVisible(true);

}

public void keyPressed(KeyEvent e) {

l.setText("Key Pressed");

}

public void keyReleased(KeyEvent e) {

l.setText("Key Released");

public void keyTyped(KeyEvent e) {

l.setText("Key Typed");

}

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

} }

1. WRITE A JAVA PROGRAM TO DEMONSTRATE AL LAYOUT MANAGERS