UNIVERSITY OF RAJASTHAN, JAIPUR

UNIVERSITY MAHARAJA COLLEGE, JAIPUR

NAME : KRISHNA KUMAR DADHICH

FATHER'S NAME : JATASHANKER DADHICH

ROLL NO :

ENROLL NO : 18/5984

CENTRE CODE : 424

EXAMINATION : B.C.A. FINAL YEAR 2020-2021

PROJECT NAME : CALCULATOR APPLICATION

SUBJECT : JAVA

EXAM DATE : ___/10/2021

COLLEGE NAME : UNIVERSITY MAHARAJA COLLEGE, JAIPUR(RAJ.)

The Calculator Application

Learning Objectives

The development process of the Calculator application will aid the students to:

- Create a simple Java console application
- Understand the object-oriented concepts of inheritance, polymorphism and data hiding
- Create application which request input from users, validate, process the input received and provide desired output.
- Use features of java like type conversion, interfaces, inheriting interfaces, looping and branching, packages and I/O classes.
-

Understanding the Calculator Application

The Calculator application performs both basic and scientific operations. The application provides user an option to choose between the basic mode and scientific mode. Based on the option selected by the user, the application calls the corresponding class and the user can perform various mathematical operations provided in the class. There is a base class in the application which contains all the methods for calculation, basic as well as scientific. The application validates the user input also and provides appropriate messages when wrong input is given by the user.

Creating the Calculator Application

To create the Calculator application, 5 java files were created. First, an interface iCalc, with the file name "iCalc.java" is created. Then, we create the base class Calculate, with the file name "Calculate.java" which contains all the methods for calculation. After the base class, two classes, Calculator and ScientificCalculator, with the file names as "Calculator.java" and "ScientificCalculator.java" are created. These classes call the methods defined in the base class Calculate. Class Calculator contains an instance of Class Calculate, whereas Class ScientificCalculator inherits Class Calculate and then uses its methods. After creation of all the above classes, a main class UseCalculate is created, with the file name “UseCalculate.java” which provides creates instances of Class Calculator or Class ScientificCalculator, based on the option selected by user.

Creating the Java Files

The iCalc Interface (iCalc.java)

Interface iCalc provides the structure of methods which can be used in any calculator application. It contains the following two methods:

- doCalculation(): Declares a method for providing methods for calculation.
- getResult(): Declares a method for extracting the result of the calculation.

The Calculate Class (Calculate.java)

Class Calculate contains the business logic of the Calculator application. It contains the methods for calculation of various mathematical operations like addition, divide and tangent. Class Calculate uses interfaces by implementing Interface iCalc. The class contains following methods:

Method	Description
Calculate()	Default constructor for the class without any arguments.
Calculate(Double dblNum, char cOperator)	Constructor containing two arguments. This constructor is used for scientific calculations.
Calculate(int iFirstNum, char cOperator, int iSecondNum)	Constructor containing three arguments. This constructor is used for basic calculations.

doCalculation()	Calculates the result based on the numbers and operator inputted by the user. Overriding the doCalculation function.of iCalc interface.
getResult()	Prints the result of calculation. Overriding the getResult function.of iCalc interface.
checkSecondNum()	In case of division of two numbers, it checks for value 0 in the second number entered.
checkInt()	Checks if basic calculation is performed.
checkDouble()	Checks if scientific calculation is performed.

The Calculator Class (Calculator.java)

Class Calculator calculates basic operations, namely, addition, subtraction, multiplication and division of two numbers. The class provides option to user to enter first number to be calculated, then the operation to be performed and then, the second number to be used for calculation. The input is received using java class BufferedReader. Class calculator creates an object of Class Calculate, by calling its constructor by passing three arguments, First Number, Operator and Second Number. After the creation of object of Class Calculate, doCalculation() method is called followed by getResult() method, which presents the result of calculation to the user.

Class Calculator also uses a do-while loop to provide an option to the user perform multiple calculations, till the user does not indicate the end of processing by typing ‘n’.

The ScientificCalculator Class (ScientificCalculator.java)

Class Calculator performs calculation of scientific operations, namely, sine, cosine, tangent and log of a number. The class provides option to user to enter the operation to be performed and the number to be calculated, The input is received using java class BufferedReader. Class ScientificCalculator inherits Class Calculate to use its methods. The class passes the user entered values to Class Calculate by calling its constructor having two arguments, Operator and the Number. The class calls doCalculation() method which is followed by getResult() method, which shows the result of calculation to the user.

Class ScientificCalculator also uses a do-while loop to provide an option to the user perform multiple calculations, till the user does not indicate the end of processing by typing 'n'.

The CalculatorLayout Class (CalculatorLayout.java)

Class Calculator provides two options to the user, Basic or Scientific. Based on the option entered by the user, the class creates an instance of Class Calculator for Basic operations or an instance of Class ScientificCalculator for scientific operations.

Class UseCalculator also uses a do-while loop to provide an option to the user perform multiple calculations, till the user does not indicate the end of processing by typing 'n'.

Generating the Class Files

Command for generating the class files:

javac <classname.java>

The steps for generating the class files for Calculator application are:

- Place all java files in a directory named as "Calculator".
- In the command prompt, go to the directory where the java files are stored for the Calculator application.
- Compile the following java files in the sequence given below using the command for compiling the java files:

    - iCalc.java
    - Calculate.java
    - Calculator.java
    - ScientificCalculator.java
    - UseCalculator.java

Working with the Calculator application

The steps for working with the Calculator application are:

- In the command prompt, go to the parent directory of “Calculator” directory which contains the class files for Calculator application.
- Enter the following command to run the Calculator application:

```
java Calculator.UseCalculator
```

- Enter ‘b’ (for Basic operations) or ‘s’ (for scientific operations) depending on the operations to be performed.
- If ‘b’ is entered, the following input is to be entered by the user:
    - First Number
    - Operator
    - Second Number
- The result is shown on the command prompt based on the above values.
- Enter ‘y’ to continue or ‘n’ to discontinue using the application.

- If ‘s’ is entered, the following input is to be entered by the user:

    - Operator
    - Number

- The result is shown on the command prompt based on the above values.
- Enter ‘y’ to continue or ‘n’ to discontinue using the application.

Code for the Calculator Application

iCalc.java

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


-- Interface iCalc represents the basic methods for the Calculate class

-- Creates Interface Structure

-- Can be used for creating any class which would do any sort of calculations


*************/


// Adds the Interface to the Package

package Calculator;


//Interface Definition

interface iCalc

{
```

```
            public void doCalculation();

            public void getResult();

}
```

MyCalculator.java

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


-- Class Calculator Performs basic operations like Add, Substract, Multiply, Division for two numbers

-- Uses class Calculate by creating its objects and then calling its methods


*************/
            import java.awt.*;
            import java.awt.event.*;

            public class MyCalculator extends Frame
            {

               public boolean setClear=true;
               double number, memValue;
               char op;

               String digitButtonText[] = {"7", "8", "9", "4", "5", "6", "1", "2", "3", "0", "+/-", "." };
               String operatorButtonText[] = {"/", "sqrt", "*", "%", "-", "1/X", "+", "=" };
               String memoryButtonText[] = {"MC", "MR", "MS", "M+" };
               String specialButtonText[] = {"Backspc", "C", "CE" };

               MyDigitButton digitButton[]=new MyDigitButton[digitButtonText.length];
               MyOperatorButton operatorButton[]=new
            MyOperatorButton[operatorButtonText.length];
               MyMemoryButton memoryButton[]=new MyMemoryButton[memoryButtonText.length];
               MySpecialButton specialButton[]=new MySpecialButton[specialButtonText.length];

               Label displayLabel=new Label("0",Label.RIGHT);
               Label memLabel=new Label(" ",Label.RIGHT);
```

```
    final int FRAME_WIDTH=325,FRAME_HEIGHT=325;
    final int HEIGHT=30, WIDTH=30, H_SPACE=10,V_SPACE=10;
    final int TOPX=30, TOPY=50;
    ///////////////////////////
    MyCalculator(String frameText)//constructor
    {
      super(frameText);

      int tempX=TOPX, y=TOPY;
      displayLabel.setBounds(tempX,y,240,HEIGHT);
      displayLabel.setBackground(Color.BLUE);
      displayLabel.setForeground(Color.WHITE);
      add(displayLabel);

      memLabel.setBounds(TOPX,  TOPY+HEIGHT+ V_SPACE,WIDTH, HEIGHT);
      add(memLabel);

// set Co-ordinates for Memory Buttons
      tempX=TOPX;
      y=TOPY+2*(HEIGHT+V_SPACE);
      for(int i=0; i<memoryButton.length; i++)
      {
        memoryButton[i]=new
MyMemoryButton(tempX,y,WIDTH,HEIGHT,memoryButtonText[i], this);
        memoryButton[i].setForeground(Color.RED);
        y+=HEIGHT+V_SPACE;
      }

//set Co-ordinates for Special Buttons
      tempX=TOPX+1*(WIDTH+H_SPACE); y=TOPY+1*(HEIGHT+V_SPACE);
      for(int i=0;i<specialButton.length;i++)
      {
        specialButton[i]=new
MySpecialButton(tempX,y,WIDTH*2,HEIGHT,specialButtonText[i], this);
        specialButton[i].setForeground(Color.RED);
        tempX=tempX+2*WIDTH+H_SPACE;
      }
```

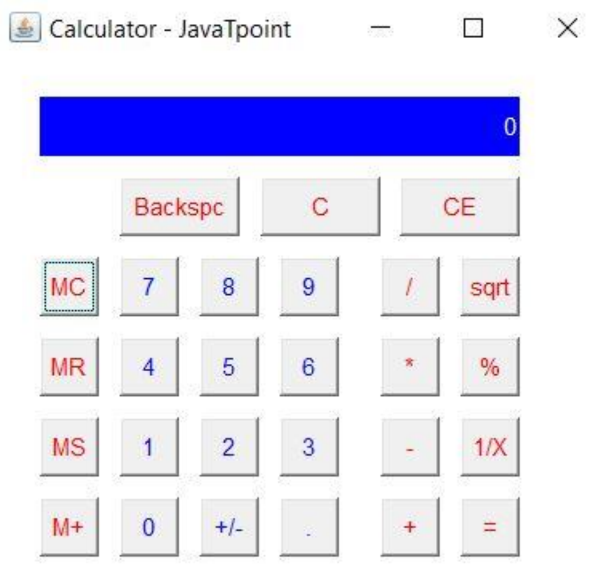
```
//set Co-ordinates for Digit Buttons
    int digitX=TOPX+WIDTH+H_SPACE;
    int digitY=TOPY+2*(HEIGHT+V_SPACE);
    tempX=digitX;  y=digitY;
    for(int i=0;i<digitButton.length;i++)
    {
      digitButton[i]=new MyDigitButton(tempX,y,WIDTH,HEIGHT,digitButtonText[i], this);
      digitButton[i].setForeground(Color.BLUE);
      tempX+=WIDTH+H_SPACE;
      if((i+1)%3==0){tempX=digitX; y+=HEIGHT+V_SPACE;}
    }

//set Co-ordinates for Operator Buttons
    int opsX=digitX+2*(WIDTH+H_SPACE)+H_SPACE;
    int opsY=digitY;
    tempX=opsX;  y=opsY;
    for(int i=0;i<operatorButton.length;i++)
    {
      tempX+=WIDTH+H_SPACE;
      operatorButton[i]=new
MyOperatorButton(tempX,y,WIDTH,HEIGHT,operatorButtonText[i], this);
      operatorButton[i].setForeground(Color.RED);
      if((i+1)%2==0){tempX=opsX; y+=HEIGHT+V_SPACE;}
    }

    addWindowListener(new WindowAdapter()
    {
      public void windowClosing(WindowEvent ev)
      {System.exit(0);}
    });

    setLayout(null);
    setSize(FRAME_WIDTH,FRAME_HEIGHT);
    setVisible(true);
  }
  //////////////////////////////////
  static String getFormattedText(double temp)
  {
    String resText=""+temp;
    if(resText.lastIndexOf(".0")>0)
      resText=resText.substring(0,resText.length()-2);
    return resText;
  }
  ////////////////////////////////////////
  public static void main(String []args)
  {
    new MyCalculator("Calculator - JavaTpoint");
  }
}


class MyDigitButton extends Button implements ActionListener
{
  MyCalculator cl;

  //////////////////////////////////////////
  MyDigitButton(int x,int y, int width,int height,String cap, MyCalculator clc)
  {
    super(cap);
    setBounds(x,y,width,height);
    this.cl=clc;
    this.cl.add(this);
    addActionListener(this);
  }
  ////////////////////////////////////////////////
  static boolean isInString(String s, char    ch)
  {
    for(int i=0; i<s.length();i++) if(s.charAt(i)==ch) return true;
    return false;
  }
  /////////////////////////////////////////////////
  public void actionPerformed(ActionEvent ev)
  {
    String tempText=((MyDigitButton)ev.getSource()).getLabel();

    if(tempText.equals("."))
    {
      if(cl.setClear)
```

```
          {cl.displayLabel.setText("0.");cl.setClear=false;}
          else if(!isInString(cl.displayLabel.getText(),'.'))
             cl.displayLabel.setText(cl.displayLabel.getText()+".");
          return;
       }

       int index=0;
       try{
          index=Integer.parseInt(tempText);
       }catch(NumberFormatException e){return;}

       if (index==0 && cl.displayLabel.getText().equals("0")) return;

       if(cl.setClear)
       {cl.displayLabel.setText(""+index);cl.setClear=false;}
       else
          cl.displayLabel.setText(cl.displayLabel.getText()+index);
    }//actionPerformed
}//class defination



class MyOperatorButton extends Button implements ActionListener
{
    MyCalculator cl;

    MyOperatorButton(int x,int y, int width,int height,String cap, MyCalculator clc)
    {
       super(cap);
       setBounds(x,y,width,height);
       this.cl=clc;
       this.cl.add(this);
       addActionListener(this);
    }
    ////////////////////////
    public void actionPerformed(ActionEvent ev)
    {
       String opText=((MyOperatorButton)ev.getSource()).getLabel();
```

```
        cl.setClear=true;
        double temp=Double.parseDouble(cl.displayLabel.getText());

        if(opText.equals("1/x"))
        {
          try
          {double tempd=1/(double)temp;
            cl.displayLabel.setText(MyCalculator.getFormattedText(tempd));}
          catch(ArithmeticException excp)
          {cl.displayLabel.setText("Divide by 0.");}
          return;
        }
        if(opText.equals("sqrt"))
        {
          try
          {double tempd=Math.sqrt(temp);
            cl.displayLabel.setText(MyCalculator.getFormattedText(tempd));}
          catch(ArithmeticException excp)
          {cl.displayLabel.setText("Divide by 0.");}
          return;
        }
        if(!opText.equals("="))
        {
          cl.number=temp;
          cl.op=opText.charAt(0);
          return;
        }
    // process = button pressed
        switch(cl.op)
        {
          case '+':
            temp+=cl.number;break;
          case '-':
            temp=cl.number-temp;break;
          case '*':
            temp*=cl.number;break;
          case '%':
            try{temp=cl.number%temp;}
            catch(ArithmeticException excp)
            {cl.displayLabel.setText("Divide by 0."); return;}
            break;
          case '/':
            try{temp=cl.number/temp;}
            catch(ArithmeticException excp)
            {cl.displayLabel.setText("Divide by 0."); return;}
            break;
        }//switch

        cl.displayLabel.setText(MyCalculator.getFormattedText(temp));
//cl.number=temp;
    }//actionPerformed
}//class


class MyMemoryButton extends Button implements ActionListener
{
    MyCalculator cl;

    ////////////////////////////////
    MyMemoryButton(int x,int y, int width,int height,String cap, MyCalculator clc)
    {
        super(cap);
        setBounds(x,y,width,height);
        this.cl=clc;
        this.cl.add(this);
        addActionListener(this);
    }
    ////////////////////////////////////////////////
    public void actionPerformed(ActionEvent ev)
    {
        char memop=((MyMemoryButton)ev.getSource()).getLabel().charAt(1);

        cl.setClear=true;
        double temp=Double.parseDouble(cl.displayLabel.getText());

        switch(memop)
        {
          case 'C':
                cl.memLabel.setText(" ");cl.memValue=0.0;break;
            case 'R':
                cl.displayLabel.setText(MyCalculator.getFormattedText(cl.memValue));break;
            case 'S':
                cl.memValue=0.0;
            case '+':
                cl.memValue+=Double.parseDouble(cl.displayLabel.getText());
                if(cl.displayLabel.getText().equals("0") || cl.displayLabel.getText().equals("0.0")  )
                    cl.memLabel.setText(" ");
                else
                    cl.memLabel.setText("M");
                break;
        }//switch
    }//actionPerformed
}//class


class MySpecialButton extends Button implements ActionListener
{
    MyCalculator cl;

    MySpecialButton(int x,int y, int width,int height,String cap, MyCalculator clc)
    {
        super(cap);
        setBounds(x,y,width,height);
        this.cl=clc;
        this.cl.add(this);
        addActionListener(this);
    }
    //////////////////////
    static String backSpace(String s)
    {
        String Res="";
        for(int i=0; i<s.length()-1; i++) Res+=s.charAt(i);
        return Res;
    }

    //////////////////////////////////////////////////////////
    public void actionPerformed(ActionEvent ev)
```

```
  {
    String opText=((MySpecialButton)ev.getSource()).getLabel();
//check for backspace button
    if(opText.equals("Backspc"))
    {
      String tempText=backSpace(cl.displayLabel.getText());
      if(tempText.equals(""))
        cl.displayLabel.setText("0");
      else
        cl.displayLabel.setText(tempText);
      return;
    }
//check for "C" button i.e. Reset
    if(opText.equals("C"))
    {
      cl.number=0.0; cl.op=' '; cl.memValue=0.0;
      cl.memLabel.setText(" ");
    }

//it must be CE button pressed
    cl.displayLabel.setText("0");cl.setClear=true;
  }//actionPerformed
}//class
```

Output :-



<u>ScientificCalculator.java</u>

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

-- Class ScientificCalculator Performs scientific calculations like Sine, Cosine, Tangent and Log of a number

-- Inherits class Calculate

-- Methods of Super class Calculate can be directly called by using the object of this Sub class ScientificCalculator

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

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

public class ScientificCalculator extends JFrame implements ActionListener {
  JTextField tfield;
  double temp, temp1, result, a;
  static double m1, m2;
  int k = 1, x = 0, y = 0, z = 0;
  char ch;
  JButton b1, b2, b3, b4, b5, b6, b7, b8, b9, zero, clr, pow2, pow3, exp,
      fac, plus, min, div, log, rec, mul, eq, addSub, dot, mr, mc, mp,
      mm, sqrt, sin, cos, tan;
  Container cont;
  JPanel textPanel, buttonpanel;

  ScientificCalculator() {
    cont = getContentPane();
    cont.setLayout(new BorderLayout());
    JPanel textpanel = new JPanel();
    tfield = new JTextField(25);
    tfield.setHorizontalAlignment(SwingConstants.RIGHT);
    tfield.addKeyListener(new KeyAdapter() {
      public void keyTyped(KeyEvent keyevent) {
        char c = keyevent.getKeyChar();
        if (c >= '0' && c <= '9') {
        } else {
          keyevent.consume();
        }
      }
    });
    textpanel.add(tfield);
    buttonpanel = new JPanel();
    buttonpanel.setLayout(new GridLayout(8, 4, 2, 2));
    boolean t = true;
    mr = new JButton("MR");
    buttonpanel.add(mr);
    mr.addActionListener(this);
    mc = new JButton("MC");
    buttonpanel.add(mc);
    mc.addActionListener(this);

    mp = new JButton("M+");
    buttonpanel.add(mp);
    mp.addActionListener(this);

    mm = new JButton("M-");
    buttonpanel.add(mm);
    mm.addActionListener(this);

    b1 = new JButton("1");
    buttonpanel.add(b1);
    b1.addActionListener(this);
    b2 = new JButton("2");
    buttonpanel.add(b2);
    b2.addActionListener(this);
    b3 = new JButton("3");
    buttonpanel.add(b3);
    b3.addActionListener(this);

    b4 = new JButton("4");
    buttonpanel.add(b4);
    b4.addActionListener(this);
    b5 = new JButton("5");
    buttonpanel.add(b5);
    b5.addActionListener(this);
    b6 = new JButton("6");
    buttonpanel.add(b6);
    b6.addActionListener(this);

    b7 = new JButton("7");
    buttonpanel.add(b7);
    b7.addActionListener(this);
    b8 = new JButton("8");
    buttonpanel.add(b8);
    b8.addActionListener(this);
    b9 = new JButton("9");
```

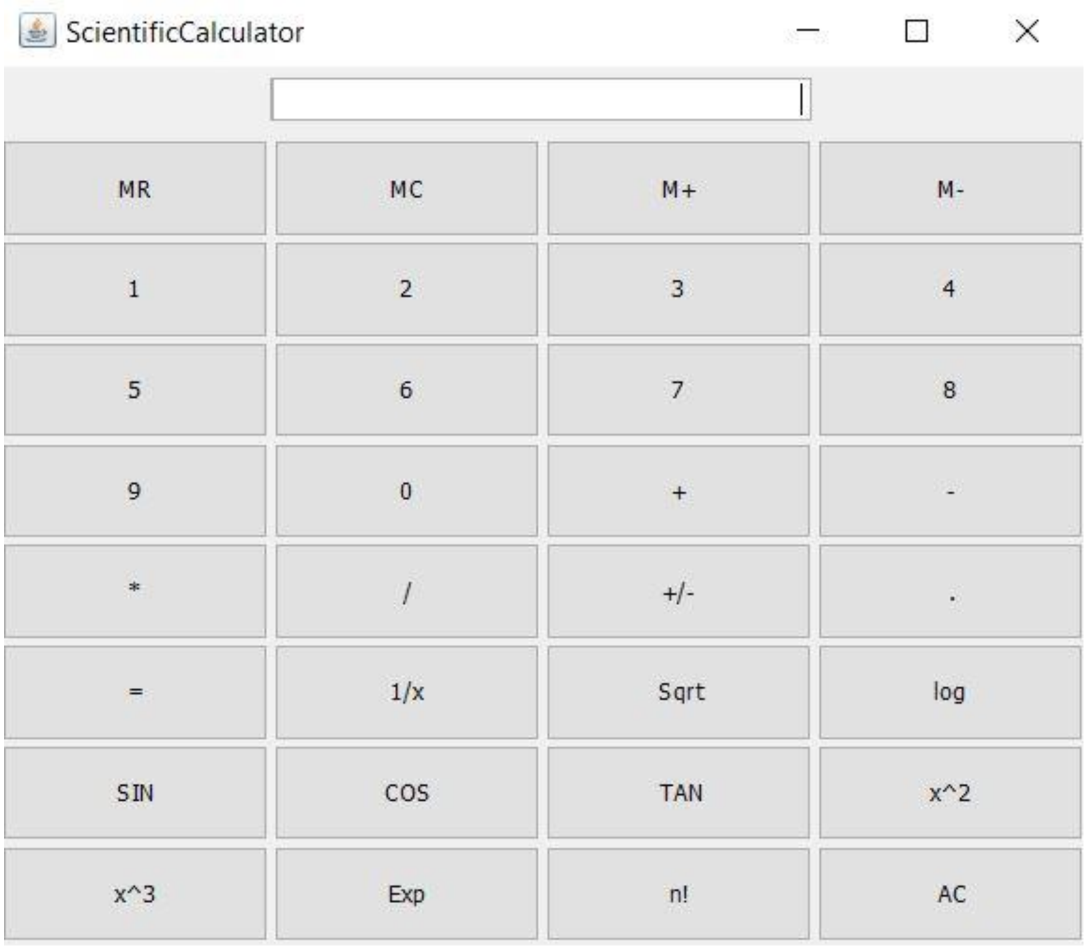
```
buttonpanel.add(b9);
b9.addActionListener(this);

zero = new JButton("0");
buttonpanel.add(zero);
zero.addActionListener(this);

plus = new JButton("+");
buttonpanel.add(plus);
plus.addActionListener(this);

min = new JButton("-");
buttonpanel.add(min);
min.addActionListener(this);

mul = new JButton("*");
buttonpanel.add(mul);
mul.addActionListener(this);

div = new JButton("/");
div.addActionListener(this);
buttonpanel.add(div);

addSub = new JButton("+/-");
buttonpanel.add(addSub);
addSub.addActionListener(this);

dot = new JButton(".");
buttonpanel.add(dot);
dot.addActionListener(this);

eq = new JButton("=");
buttonpanel.add(eq);
eq.addActionListener(this);

rec = new JButton("1/x");
buttonpanel.add(rec);
rec.addActionListener(this);
sqrt = new JButton("Sqrt");
```

```
    buttonpanel.add(sqrt);
    sqrt.addActionListener(this);
    log = new JButton("log");
    buttonpanel.add(log);
    log.addActionListener(this);

    sin = new JButton("SIN");
    buttonpanel.add(sin);
    sin.addActionListener(this);
    cos = new JButton("COS");
    buttonpanel.add(cos);
    cos.addActionListener(this);
    tan = new JButton("TAN");
    buttonpanel.add(tan);
    tan.addActionListener(this);
    pow2 = new JButton("x^2");
    buttonpanel.add(pow2);
    pow2.addActionListener(this);
    pow3 = new JButton("x^3");
    buttonpanel.add(pow3);
    pow3.addActionListener(this);
    exp = new JButton("Exp");
    exp.addActionListener(this);
    buttonpanel.add(exp);
    fac = new JButton("n!");
    fac.addActionListener(this);
    buttonpanel.add(fac);

    clr = new JButton("AC");
    buttonpanel.add(clr);
    clr.addActionListener(this);
    cont.add("Center", buttonpanel);
    cont.add("North", textpanel);
    setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
  }

  public void actionPerformed(ActionEvent e) {
    String s = e.getActionCommand();
    if (s.equals("1")) {
            if (z == 0) {
                tfield.setText(tfield.getText() + "1");
            } else {
                tfield.setText("");
                tfield.setText(tfield.getText() + "1");
                z = 0;
            }
        }
        if (s.equals("2")) {
            if (z == 0) {
                tfield.setText(tfield.getText() + "2");
            } else {
                tfield.setText("");
                tfield.setText(tfield.getText() + "2");
                z = 0;
            }
        }
        if (s.equals("3")) {
            if (z == 0) {
                tfield.setText(tfield.getText() + "3");
            } else {
                tfield.setText("");
                tfield.setText(tfield.getText() + "3");
                z = 0;
            }
        }
        if (s.equals("4")) {
            if (z == 0) {
                tfield.setText(tfield.getText() + "4");
            } else {
                tfield.setText("");
                tfield.setText(tfield.getText() + "4");
                z = 0;
            }
        }
        if (s.equals("5")) {
            if (z == 0) {
                tfield.setText(tfield.getText() + "5");
            } else {
```

```
            tfield.setText("");
            tfield.setText(tfield.getText() + "5");
            z = 0;
        }
    }
    if (s.equals("6")) {
        if (z == 0) {
            tfield.setText(tfield.getText() + "6");
        } else {
            tfield.setText("");
            tfield.setText(tfield.getText() + "6");
            z = 0;
        }
    }
    if (s.equals("7")) {
        if (z == 0) {
            tfield.setText(tfield.getText() + "7");
        } else {
            tfield.setText("");
            tfield.setText(tfield.getText() + "7");
            z = 0;
        }
    }
    if (s.equals("8")) {
        if (z == 0) {
            tfield.setText(tfield.getText() + "8");
        } else {
            tfield.setText("");
            tfield.setText(tfield.getText() + "8");
            z = 0;
        }
    }
    if (s.equals("9")) {
        if (z == 0) {
            tfield.setText(tfield.getText() + "9");
        } else {
            tfield.setText("");
            tfield.setText(tfield.getText() + "9");
            z = 0;
        }
    }
    if (s.equals("0")) {
        if (z == 0) {
            tfield.setText(tfield.getText() + "0");
        } else {
            tfield.setText("");
            tfield.setText(tfield.getText() + "0");
            z = 0;
        }
    }
    if (s.equals("AC")) {
        tfield.setText("");
        x = 0;
        y = 0;
        z = 0;
    }
    if (s.equals("log")) {
        if (tfield.getText().equals("")) {
            tfield.setText("");
        } else {
            a = Math.log(Double.parseDouble(tfield.getText()));
            tfield.setText("");
            tfield.setText(tfield.getText() + a);
        }
    }
    if (s.equals("1/x")) {
        if (tfield.getText().equals("")) {
            tfield.setText("");
        } else {
            a = 1 / Double.parseDouble(tfield.getText());
            tfield.setText("");
            tfield.setText(tfield.getText() + a);
        }
    }
    if (s.equals("Exp")) {
        if (tfield.getText().equals("")) {
            tfield.setText("");
        } else {
            a = Math.exp(Double.parseDouble(tfield.getText()));
            tfield.setText("");
            tfield.setText(tfield.getText() + a);
        }
    }
    if (s.equals("x^2")) {
        if (tfield.getText().equals("")) {
            tfield.setText("");
        } else {
            a = Math.pow(Double.parseDouble(tfield.getText()), 2);
            tfield.setText("");
            tfield.setText(tfield.getText() + a);
        }
    }
    if (s.equals("x^3")) {
        if (tfield.getText().equals("")) {
            tfield.setText("");
        } else {
            a = Math.pow(Double.parseDouble(tfield.getText()), 3);
            tfield.setText("");
            tfield.setText(tfield.getText() + a);
        }
    }
    if (s.equals("+/-")) {
        if (x == 0) {
            tfield.setText("-" + tfield.getText());
            x = 1;
        } else {
            tfield.setText(tfield.getText());
        }
    }
    if (s.equals(".")) {
        if (y == 0) {
            tfield.setText(tfield.getText() + ".");
            y = 1;
        } else {
            tfield.setText(tfield.getText());
        }
    }
```

```
if (s.equals("+")) {
    if (tfield.getText().equals("")) {
        tfield.setText("");
        temp = 0;
        ch = '+';
    } else {
        temp = Double.parseDouble(tfield.getText());
        tfield.setText("");
        ch = '+';
        y = 0;
        x = 0;
    }
    tfield.requestFocus();
}
if (s.equals("-")) {
    if (tfield.getText().equals("")) {
        tfield.setText("");
        temp = 0;
        ch = '-';
    } else {
        x = 0;
        y = 0;
        temp = Double.parseDouble(tfield.getText());
        tfield.setText("");
        ch = '-';
    }
    tfield.requestFocus();
}
if (s.equals("/")) {
    if (tfield.getText().equals("")) {
        tfield.setText("");
        temp = 1;
        ch = '/';
    } else {
        x = 0;
        y = 0;
        temp = Double.parseDouble(tfield.getText());
        ch = '/';
        tfield.setText("");
    }
    tfield.requestFocus();
}
if (s.equals("*")) {
    if (tfield.getText().equals("")) {
        tfield.setText("");
        temp = 1;
        ch = '*';
    } else {
        x = 0;
        y = 0;
        temp = Double.parseDouble(tfield.getText());
        ch = '*';
        tfield.setText("");
    }
    tfield.requestFocus();
}
if (s.equals("MC")) {
    m1 = 0;
    tfield.setText("");
}
if (s.equals("MR")) {
    tfield.setText("");
    tfield.setText(tfield.getText() + m1);
}
if (s.equals("M+")) {
    if (k == 1) {
        m1 = Double.parseDouble(tfield.getText());
        k++;
    } else {
        m1 += Double.parseDouble(tfield.getText());
        tfield.setText("" + m1);
    }
}
if (s.equals("M-")) {
    if (k == 1) {
        m1 = Double.parseDouble(tfield.getText());
        k++;
    } else {
        m1 -= Double.parseDouble(tfield.getText());
        tfield.setText("" + m1);
    }
}
if (s.equals("Sqrt")) {
    if (tfield.getText().equals("")) {
        tfield.setText("");
    } else {
        a = Math.sqrt(Double.parseDouble(tfield.getText()));
        tfield.setText("");
        tfield.setText(tfield.getText() + a);
    }
}
if (s.equals("SIN")) {
    if (tfield.getText().equals("")) {
        tfield.setText("");
    } else {
        a = Math.sin(Double.parseDouble(tfield.getText()));
        tfield.setText("");
        tfield.setText(tfield.getText() + a);
    }
}
if (s.equals("COS")) {
    if (tfield.getText().equals("")) {
        tfield.setText("");
    } else {
        a = Math.cos(Double.parseDouble(tfield.getText()));
        tfield.setText("");
        tfield.setText(tfield.getText() + a);
    }
}
if (s.equals("TAN")) {
    if (tfield.getText().equals("")) {
        tfield.setText("");
    } else {
        a = Math.tan(Double.parseDouble(tfield.getText()));
        tfield.setText("");
        tfield.setText(tfield.getText() + a);
    }
    }
    if (s.equals("=")) {
        if (tfield.getText().equals("")) {
            tfield.setText("");
        } else {
            temp1 = Double.parseDouble(tfield.getText());
            switch (ch) {
                case '+':
                    result = temp + temp1;
                    break;
                case '-':
                    result = temp - temp1;
                    break;
                case '/':
                    result = temp / temp1;
                    break;
                case '*':
                    result = temp * temp1;
                    break;
            }
            tfield.setText("");
            tfield.setText(tfield.getText() + result);
            z = 1;
        }
    }
    if (s.equals("n!")) {
        if (tfield.getText().equals("")) {
            tfield.setText("");
        } else {
            a = fact(Double.parseDouble(tfield.getText()));
            tfield.setText("");
            tfield.setText(tfield.getText() + a);
        }
    }
    tfield.requestFocus();
}

double fact(double x) {
    int er = 0;
    if (x < 0) {
        er = 20;
        return 0;
    }
    double i, s = 1;
    for (i = 2; i <= x; i += 1.0)
        s *= i;
    return s;
}

public static void main(String args[]) {
    try {
        UIManager
                .setLookAndFeel("com.sun.java.swing.plaf.windows.WindowsLookAndFeel");
    } catch (Exception e) {
    }
    ScientificCalculator f = new ScientificCalculator();
    f.setTitle("ScientificCalculator");
    f.pack();
    f.setVisible(true);
}
}
```

Output :-



<u>CalculatorLayout.java</u>(Complete Calculator)

/*************

Krishna Dadhich-Maharajas College

```
-- Class CalculatorLayout is the Main Class for the Calculator Application.

-- Provides options to the user: Basic or Scientific Calculator.

-- In short it is complete calculator.

*************/




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

@SuppressWarnings("serial")
public class CalculatorLayout extends JFrame implements ActionListener
{
   // =====================Declaration of variable==============
   //----------Common keys---------------
   private JButton bOne = new JButton("1");
   private JButton bTwo = new JButton("2");
   private JButton bThree = new JButton("3");
   private JButton bFour = new JButton("4");
   private JButton bFive = new JButton("5");
   private JButton bSix = new JButton("6");
   private JButton bSeven = new JButton("7");
   private JButton bEight = new JButton("8");
   private JButton bNine = new JButton("9");
   private JButton bZero = new JButton("0");
   private JButton bMul = new JButton("\u00D7");
   private JButton bDiv = new JButton("\u00F7");
   private JButton bAdd = new JButton("+");
   private JButton bSub = new JButton("\u02D7");
   private JButton bEqual = new JButton("=");
   private JButton bPoint = new JButton(".");
   private JButton bDel = new JButton("DE");
   private JButton bClear = new JButton("C");
   private JButton bSquare = new JButton("x\u00B2");
   private JButton bCube = new JButton("x\u00B3");
   private JButton bSqrt = new JButton("\u221A");
   private JButton bPercent = new JButton("%");
   private JButton bMod = new JButton("Mod");
   private JButton bOneByN = new JButton("1/n");
   private JButton bPlusMinus = new JButton("\u00B1");

   //----------Scientific keys---------------
   private JButton bSin = new JButton("sin");
   private JButton bCos = new JButton("cos");
   private JButton bTan = new JButton("tan");
   private JButton bAsin = new JButton("asin");
   private JButton bAcos = new JButton("acos");
   private JButton bAtan = new JButton("atan");
   private JButton bSinH = new JButton("sinh");
   private JButton bCosH = new JButton("cosh");
   private JButton bTanH = new JButton("tanh");
   private JButton bPowerOfTen = new JButton("10^n");
   private JButton bLog = new JButton("log");
   private JButton bLn = new JButton("ln");
   private JButton bAbs = new JButton("abs");
   private JButton bExit = new JButton("EXIT");

   //------------container variables---------------
   private JTextField tfDisplay = new JTextField();//result displaying screen
   private JTextField tfRawInput = new JTextField();
   private String sRawInput = "";
   private String sDisplay = "";//Input string
   private boolean isPlus = true;//Is the sign of the operand is plus
   private boolean isPoint = false;//is there is decimal point in the operands
   private boolean isOperation = false;
   private double number1 = 0;// 1st operand
   private double number2 = 0;// 2nd operand
   private double result = 0;// Result
   private char operation = ' ';// Operation
   static Color windowColor = new Color(110, 119, 129);//Color of container window


   //=================Default Constructor to Design the layout of the
```

```
calculator===========================
  public CalculatorLayout()
  {
    setBackground(windowColor);
    setLayout(null);
    JPanel pScreen1 = new JPanel(); // The screen that display the input
    JPanel pScreen2 = new JPanel(); // The screen that show the result
    JPanel pKeypad1 = new JPanel(); // The keypad that contain the common keys
    JPanel pKeypad2 = new JPanel(); // The keypad that contains the scientific keys.

    //---------------------Fonts & Colors------------------------
    Font fontResDisplay = new Font("Times New Roman", Font.BOLD, 35);//Font for displaying result
    Font fontKeypad = new Font("Times New Roman", Font.PLAIN, 20);//Font for key character
    Font fontKeypad1 = new Font("Times New Roman", Font.PLAIN, 15);//Font for key character

    Color screenColor = new Color(91, 178, 91);
    Color numberKeyColor = new Color(212, 212, 212);
    Color equalColor = new Color(63, 132, 243);
    Color exitColor = new Color(224, 67, 67);
    Color otherColor = new Color(247, 247, 247);
    Color copyRightColor = new Color(0, 0, 255);

    //===========================Designing the Result screen of the calculator
===========================
    add(pScreen1).setBounds(0, 0, 343, 30);
    pScreen1.add(tfRawInput);
    pScreen1.setLayout(null);
    tfRawInput.setBounds(0, 0, 343, 30);
    tfRawInput.setHorizontalAlignment(JTextField.LEFT);
    tfRawInput.setFont(new Font("Times New Roman", Font.PLAIN, 20));
    tfRawInput.setText("0");
    tfRawInput.setEditable(false);
    tfRawInput.setBackground(screenColor);
    tfRawInput.setForeground(Color.BLACK);
    //===========================Designing the Result screen of the calculator
===========================
    add(pScreen2).setBounds(0, 30, 343, 50);
    pScreen2.add(tfDisplay);
    pScreen2.setLayout(null);
```

```
        tfDisplay.setBounds(0, 0, 343, 50);
        tfDisplay.setHorizontalAlignment(JTextField.RIGHT);
        tfDisplay.setFont(fontResDisplay);
        tfDisplay.setText("0");
        tfDisplay.setEditable(false);
        tfDisplay.setBackground(screenColor);
        tfDisplay.setForeground(Color.BLACK);

        //==========================Designing the keypad1(Common Keys) of the calculator
====================
        add(pKeypad1).setBounds(0, 100, 343, 190);
        pKeypad1.setLayout(null);
        pKeypad1.setBackground(windowColor);

        //------------setting font, color and style of the common keys-------
        bOne.setFont(fontKeypad);  bOne.setBackground(numberKeyColor);    bOne.setFocusable(false);
        bTwo.setFont(fontKeypad);  bTwo.setBackground(numberKeyColor);
bTwo.setFocusable(false);
        bThree.setFont(fontKeypad);    bThree.setBackground(numberKeyColor);
bThree.setFocusable(false);
        bFour.setFont(fontKeypad); bFour.setBackground(numberKeyColor);   bFour.setFocusable(false);
        bFive.setFont(fontKeypad); bFive.setBackground(numberKeyColor);   bFive.setFocusable(false);
        bSix.setFont(fontKeypad);  bSix.setBackground(numberKeyColor);    bSix.setFocusable(false);
        bSeven.setFont(fontKeypad);    bSeven.setBackground(numberKeyColor);
bSeven.setFocusable(false);
        bEight.setFont(fontKeypad);    bEight.setBackground(numberKeyColor);
bEight.setFocusable(false);
        bNine.setFont(fontKeypad); bNine.setBackground(numberKeyColor);
bNine.setFocusable(false);
        bZero.setFont(fontKeypad); bZero.setBackground(numberKeyColor);   bZero.setFocusable(false);
        bAdd.setFont(fontKeypad);  bAdd.setBackground(otherColor);    bAdd.setFocusable(false);
        bSub.setFont(fontKeypad);  bSub.setBackground(otherColor);    bSub.setFocusable(false);
        bMul.setFont(fontKeypad);  bMul.setBackground(otherColor);    bMul.setFocusable(false);
        bDiv.setFont(fontKeypad);  bDiv.setBackground(otherColor);    bDiv.setFocusable(false);
        bPoint.setFont(fontKeypad);    bPoint.setBackground(numberKeyColor);
bPoint.setFocusable(false);
        bEqual.setFont(new Font("Times New Roman", Font.PLAIN, 40));
bEqual.setBackground(equalColor);  bEqual.setFocusable(false);
        bDel.setFont(fontKeypad1); bDel.setBackground(Color.ORANGE);  bDel.setFocusable(false);
        bClear.setFont(fontKeypad);    bClear.setBackground(exitColor);   bClear.setFocusable(false);
        bSquare.setFont(fontKeypad);   bSquare.setBackground(otherColor);
bSquare.setFocusable(false);
        bSqrt.setFont(fontKeypad); bSqrt.setBackground(otherColor);   bSqrt.setFocusable(false);
        bCube.setFont(fontKeypad); bCube.setBackground(otherColor);   bCube.setFocusable(false);
        bPercent.setFont(fontKeypad);  bPercent.setBackground(otherColor);
bPercent.setFocusable(false);
        bMod.setFont(new Font("Times New Roman", Font.PLAIN, 10));
bMod.setBackground(otherColor);    bMod.setFocusable(false);
        bOneByN.setFont(fontKeypad1);  bOneByN.setBackground(otherColor);
bOneByN.setFocusable(false);
        bPlusMinus.setFont(fontKeypad);    bPlusMinus.setBackground(numberKeyColor);
bPlusMinus.setFocusable(false);

        //-----------------------------placing the common keys-------------------------------------
        // 1st row
        pKeypad1.add(bDel).setBounds(226, 0, 54, 38);     pKeypad1.add(bClear).setBounds(280, 0, 54,
38);
        // 2nd row
        pKeypad1.add(bSeven).setBounds(10, 38, 54, 38);pKeypad1.add(bEight).setBounds(64, 38, 54,
38);pKeypad1.add(bNine).setBounds(118, 38, 54, 38);
        pKeypad1.add(bMul).setBounds(172, 38, 54, 38);pKeypad1.add(bDiv).setBounds(226, 38, 54,
38);pKeypad1.add(bSquare).setBounds(280, 38, 54, 38);
        // 3rd row
        pKeypad1.add(bFour).setBounds(10, 76, 54, 38);pKeypad1.add(bFive).setBounds(64, 76, 54,
38);pKeypad1.add(bSix).setBounds(118, 76, 54, 38);
        pKeypad1.add(bAdd).setBounds(172, 76, 54, 38);pKeypad1.add(bSub).setBounds(226, 76, 54,
38);pKeypad1.add(bCube).setBounds(280, 76, 54, 38);
        // 4th row
        pKeypad1.add(bOne).setBounds(10, 114, 54, 38);pKeypad1.add(bTwo).setBounds(64, 114, 54,
38);pKeypad1.add(bThree).setBounds(118, 114, 54, 38);
        pKeypad1.add(bEqual).setBounds(172, 114, 108, 38);pKeypad1.add(bMod).setBounds(280, 114,
54, 38);
        // 5th row
        pKeypad1.add(bZero).setBounds(10, 152, 54, 38);pKeypad1.add(bPoint).setBounds(64, 152, 54,
38);pKeypad1.add(bPlusMinus).setBounds(118, 152, 54, 38);
        pKeypad1.add(bOneByN).setBounds(172, 152, 54, 38);pKeypad1.add(bPercent).setBounds(226,
152, 54, 38);pKeypad1.add(bSqrt).setBounds(280, 152, 54, 38);
```

```
	//=========================Designing the keypad2(scientific Keys) of the calculator
====================

	add(pKeypad2).setBounds(0, 310, 343, 145);
	pKeypad2.setLayout(null);
	pKeypad2.setBackground(windowColor);

	//---------------------copyright tag----------------------------------------------------
	JLabel copyRight = new JLabel("\u00A9 2021  KD- JAVA Tech.");
	copyRight.setForeground(copyRightColor);
	pKeypad2.add(copyRight).setBounds(205, 130, 150, 15);

	//----------------------------placing the Scientific keys------------------------------------
	// 1st row
	pKeypad2.add(bSin).setBounds(10, 0, 65, 38);pKeypad2.add(bCos).setBounds(75, 0, 65,
38);pKeypad2.add(bTan).setBounds(140, 0, 65, 38);
	pKeypad2.add(bLog).setBounds(205, 0, 65, 38);pKeypad2.add(bLn).setBounds(270, 0, 64, 38);
	// 2nd row
	pKeypad2.add(bAsin).setBounds(10, 38, 65, 38);pKeypad2.add(bAcos).setBounds(75, 38, 65,
38);pKeypad2.add(bAtan).setBounds(140, 38, 65, 38);
	pKeypad2.add(bPowerOfTen).setBounds(205, 38, 65, 38);pKeypad2.add(bAbs).setBounds(270,
38, 64, 38);
	// 3rd row
	pKeypad2.add(bSinH).setBounds(10, 76, 65, 38);pKeypad2.add(bCosH).setBounds(75, 76, 65,
38);pKeypad2.add(bTanH).setBounds(140, 76, 65, 38);
	pKeypad2.add(bExit).setBounds(205, 76, 130, 38);

	//------------setting font, color and style of the common keys-------
	bSin.setFont(fontKeypad);  bSin.setBackground(otherColor);    bSin.setFocusable(false);
	bCos.setFont(fontKeypad);  bCos.setBackground(otherColor);    bCos.setFocusable(false);
	bTan.setFont(fontKeypad);  bTan.setBackground(otherColor);    bTan.setFocusable(false);
	bAsin.setFont(fontKeypad1);    bAsin.setBackground(otherColor);   bAsin.setFocusable(false);
	bAcos.setFont(fontKeypad1);    bAcos.setBackground(otherColor);   bAcos.setFocusable(false);
	bAtan.setFont(fontKeypad1);    bAtan.setBackground(otherColor);   bAtan.setFocusable(false);
	bSinH.setFont(fontKeypad1);    bSinH.setBackground(otherColor);   bSinH.setFocusable(false);
	bCosH.setFont(fontKeypad1);    bCosH.setBackground(otherColor);   bCosH.setFocusable(false);
	bTanH.setFont(fontKeypad1);    bTanH.setBackground(otherColor);   bTanH.setFocusable(false);
	bLog.setFont(fontKeypad);  bLog.setBackground(otherColor);    bLog.setFocusable(false);
```

```
        bLn.setFont(fontKeypad);   bLn.setBackground(otherColor);    bLn.setFocusable(false);
        bAbs.setFont(fontKeypad);  bAbs.setBackground(otherColor);    bAbs.setFocusable(false);
        bExit.setFont(fontKeypad); bExit.setBackground(exitColor);       bExit.setFocusable(false);
        bPowerOfTen.setFont(fontKeypad1);  bPowerOfTen.setBackground(otherColor);
bPowerOfTen.setFocusable(false);

        // =====================================Adding actionListener
=======================================
        //common keys
        bOne.addActionListener(this);
        bTwo.addActionListener(this);
        bThree.addActionListener(this);
        bFour.addActionListener(this);
        bFive.addActionListener(this);
        bSix.addActionListener(this);
        bSeven.addActionListener(this);
        bEight.addActionListener(this);
        bNine.addActionListener(this);
        bZero.addActionListener(this);
        bAdd.addActionListener(this);
        bSub.addActionListener(this);
        bMul.addActionListener(this);
        bDiv.addActionListener(this);
        bPoint.addActionListener(this);
        bEqual.addActionListener(this);
        bDel.addActionListener(this);
        bClear.addActionListener(this);
        bSquare.addActionListener(this);
        bSqrt.addActionListener(this);
        bCube.addActionListener(this);
        bPercent.addActionListener(this);
        bMod.addActionListener(this);
        bOneByN.addActionListener(this);
        bPlusMinus.addActionListener(this);
        //scientific keys
        bSin.addActionListener(this);
        bCos.addActionListener(this);
        bTan.addActionListener(this);
        bAsin.addActionListener(this);
```

```
    bAcos.addActionListener(this);
    bAtan.addActionListener(this);
    bSinH.addActionListener(this);
    bCosH.addActionListener(this);
    bTanH.addActionListener(this);
    bPowerOfTen.addActionListener(this);
    bLog.addActionListener(this);
    bLn.addActionListener(this);
    bAbs.addActionListener(this);
    bExit.addActionListener(this);

  }

  //================================Main method============================
  public static void main(String[] CHAND)
  {
    CalculatorLayout frame = new CalculatorLayout();
    frame.setTitle("CI- Calculator");
    frame.setSize(350, 500);
    frame.getContentPane().setBackground(windowColor);
    frame.setLocationRelativeTo(null);
    //frame.setMaximizedBounds(new Rectangle(300, 200));
    frame.setResizable(false);
    frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    frame.setVisible(true);
  }

  //=============================ActionListener of All keys ===========================
  @Override
  public void actionPerformed(ActionEvent e)
  {
    if (e.getSource() == bOne)
    {
      if(operation == '=')
      {
        sDisplay = "1";
        sRawInput = "1";
        tfRawInput.setText(sRawInput);
        operation = ' ';
        }
        else
        {
            sDisplay = sDisplay + "1";
            sRawInput += "1";
            tfRawInput.setText(sRawInput);
        }
    }
    else if (e.getSource() == bTwo)
    {
        if(operation == '=')
        {
            sDisplay = "2";
            sRawInput = "2";
            tfRawInput.setText(sRawInput);
            operation = ' ';
        }
        else
        {
            sDisplay = sDisplay + "2";
            sRawInput += "2";
            tfRawInput.setText(sRawInput);
        }
    }
    else if (e.getSource() == bThree)
    {
        if(operation == '=')
        {
            sDisplay = "3";
            sRawInput = "3";
            tfRawInput.setText(sRawInput);
            operation = ' ';
        }
        else
        {
            sDisplay = sDisplay + "3";
            sRawInput += "3";
            tfRawInput.setText(sRawInput);
        }
```

```
}
else if (e.getSource() == bFour)
{
  if(operation == '=')
  {
    sDisplay = "4";
    sRawInput = "4";
    tfRawInput.setText(sRawInput);
    operation = ' ';
  }
  else
  {
    sDisplay = sDisplay + "4";
    sRawInput += "4";
    tfRawInput.setText(sRawInput);
  }
}
else if (e.getSource() == bFive)
{
  if(operation == '=')
  {
    sDisplay = "5";
    sRawInput = "5";
    tfRawInput.setText(sRawInput);
    operation = ' ';
  }
  else
  {
    sDisplay = sDisplay + "5";
    sRawInput += "5";
    tfRawInput.setText(sRawInput);
  }
}
else if (e.getSource() == bSix)
{
  if(operation == '=')
  {
    sDisplay = "6";
    sRawInput = "6";
```

```
            tfRawInput.setText(sRawInput);
            operation = ' ';
        }
        else
        {
            sDisplay = sDisplay + "6";
            sRawInput += "6";
            tfRawInput.setText(sRawInput);
        }
    }
    else if (e.getSource() == bSeven)
    {
        if(operation == '=')
        {
            sDisplay = "7";
            sRawInput = "7";
            tfRawInput.setText(sRawInput);
            operation = ' ';
        }
        else
        {
            sDisplay = sDisplay + "7";
            sRawInput += "7";
            tfRawInput.setText(sRawInput);
        }
    }
    else if (e.getSource() == bEight)
    {
        if(operation == '=')
        {
            sDisplay = "8";
            sRawInput = "8";
            tfRawInput.setText(sRawInput);
            operation = ' ';
        }
        else
        {
            sDisplay = sDisplay + "8";
            sRawInput += "8";
            tfRawInput.setText(sRawInput);
        }
    }
    else if (e.getSource() == bNine)
    {
        if(operation == '=')
        {
            sDisplay = "9";
            sRawInput = "9";
            tfRawInput.setText(sRawInput);
            operation = ' ';
        }
        else
        {
            sDisplay = sDisplay + "9";
            sRawInput += "9";
            tfRawInput.setText(sRawInput);
        }
    }
    else if (e.getSource() == bZero)
    {
        if(sDisplay.equals(""))//zero at first
        {
            sDisplay = "0";
            sRawInput += "0";
            tfRawInput.setText(sRawInput);
        }
        else
        {
            sDisplay = sDisplay + "0";
            sRawInput += "0";
            tfRawInput.setText(sRawInput);
        }
    }
    else if (e.getSource() == bPoint) ///when bPoint is clicked
    {
        if(sDisplay.equals(""))//bPoint at starting of a number
        {
            sDisplay = "0.";
            sRawInput += "0.";
            tfRawInput.setText(sRawInput);
        }
        else if(!isPoint)//when there is no bPoint till now then just add the bPoint
        {
            sDisplay = sDisplay + ".";
            sRawInput += ".";
            tfRawInput.setText(sRawInput);
        }
        isPoint = true;//when the bPoint button is clicked once set the bPoint flag to true
    }
    else if(e.getSource() == bPlusMinus && !sDisplay.equals("") && !isOperation)//plus minus(sign)
button
    {
        if(isPlus)
        {
            sDisplay = "-" + sDisplay;
            sRawInput = sDisplay;
            tfRawInput.setText(sRawInput);
            isPlus = false;
        }
        else
        {
            sDisplay = sDisplay.substring(1, sDisplay.length());
            sRawInput = sDisplay.substring(1, sDisplay.length());
            tfRawInput.setText(sRawInput);
            isPlus = true;
        }
    }
    else if (e.getSource() == bAdd && (!sDisplay.equals("") || operation == '=')) // + button is clicked
and input is not empty
    {
        if(!isOperation)//to check if plus without number1
        {
            number1 = Double.parseDouble(sDisplay);
            sDisplay = "";
            operation = '+';
            isPlus = true;
            sRawInput += " + ";
            tfRawInput.setText(sRawInput);
            if(isPoint)//when there is any bPoint number or any bDivision operation, there may be a
bPoint in the result
            {
              tfDisplay.setText(""+number1);
            }
            else if(!isPoint)//when there is no bPoint in the result
            {
              tfDisplay.setText(""+(long)number1);
            }
        }
        else if(isOperation && operation != '=')
        {
            number2  = Double.parseDouble(sDisplay);

            if(operation == '+')
            {
              result = number1 + number2;
            }
            else if(operation == '-')
            {
              result = number1 - number2;
            }
            else if(operation == '*')
            {
              result = number1 * number2;
            }
            else if(operation == '/')
            {
              result = number1 / number2;
            }
            else if(operation == '%')
            {
              result = number1 % number2;
            }
            else
            {
              result = number2;
            }
```

```
                String temp = "";
                if(isPoint || operation == '/')//when there is any bPoint number or any bDivision operation,
there may be a bPoint in the result
                {
                    tfDisplay.setText(""+result);
                    temp = ""+result;
                }
                else if(!isPoint)//when there is no bPoint in the result
                {
                    tfDisplay.setText(""+(long)result);
                    temp = ""+(long)result;
                }
                operation = '+';
                sDisplay = "";
                number1 = result;
                isPlus = true;
                isPoint = false;
                isOperation = true;
                sRawInput += " + ";
                tfRawInput.setText(sRawInput);
                sRawInput = temp + " + ";
            }
            else if(operation == '=')
            {
                sDisplay = "";
                operation = '+';
                isPlus = true;
                isOperation = true;
                sRawInput += " + ";
                tfRawInput.setText(sRawInput);
            }
            isOperation = true;
        }
        else if (e.getSource() == bSub && (!sDisplay.equals("") || operation == '='))  // - button is clicked
and input is not empty
        {
            if(!isOperation)//to check if plus without number1
            {
                number1 = Double.parseDouble(sDisplay);
                sDisplay = "";
                operation = '-';
                isPlus = true;
                sRawInput += " - ";
                tfRawInput.setText(sRawInput);
                if(isPoint)//when there is any bPoint number or any bDivision operation, there may be a
bPoint in the result
                {
                    tfDisplay.setText(""+number1);
                }
                else if(!isPoint)//when there is no bPoint in the result
                {
                    tfDisplay.setText(""+(long)number1);
                }
            }
            else if(isOperation && operation != '=')
            {
                number2  = Double.parseDouble(sDisplay);

                if(operation == '+')
                {
                    result = number1 + number2;
                }
                else if(operation == '-')
                {
                    result = number1 - number2;
                }
                else if(operation == '*')
                {
                    result = number1 * number2;
                }
                else if(operation == '/')
                {
                    result = number1 / number2;
                }
                else if(operation == '%')
                {
                    result = number1 % number2;
                }
                else
                {
                    result = number2;
                }
                String temp = "";
                if(isPoint || operation == '/')//when there is any bPoint number or any bDivision operation,
there may be a bPoint in the result
                {
                    tfDisplay.setText(""+result);
                    temp = ""+result;
                }
                else if(!isPoint)//when there is no bPoint in the result
                {
                    tfDisplay.setText(""+(long)result);
                    temp = ""+(long)result;
                }
                operation = '-';
                sDisplay = "";
                number1 = result;
                isPlus = true;
                isPoint = false;
                isOperation = true;
                sRawInput += " - ";
                tfRawInput.setText(sRawInput);
                sRawInput = temp + " + ";
            }
            else if(operation == '=')
            {
                sDisplay = "";
                operation = '-';
                isPlus = true;
                isOperation = true;
                sRawInput += " - ";
                tfRawInput.setText(sRawInput);
            }
            isOperation = true;
         }
         else if (e.getSource() == bMul && (!sDisplay.equals("") || operation == '='))  // * button is clicked
and input is not empty
```

```
    {
        if(!isOperation)//to check if plus without number1
        {
            number1 = Double.parseDouble(sDisplay);
            sDisplay = "";
            operation = '*';
            isPlus = true;
            sRawInput += " \u00D7 ";
            tfRawInput.setText(sRawInput);
            if(isPoint)//when there is any bPoint number or any bDivision operation, there may be a
bPoint in the result
            {
                tfDisplay.setText(""+number1);
            }
            else if(!isPoint)//when there is no bPoint in the result
            {
                tfDisplay.setText(""+(long)number1);
            }
        }
        else if(isOperation && operation != '=')
        {
            number2  = Double.parseDouble(sDisplay);

            if(operation == '+')
            {
                result = number1 + number2;
            }
            else if(operation == '-')
            {
                result = number1 - number2;
            }
            else if(operation == '*')
            {
                result = number1 * number2;
            }
            else if(operation == '/')
            {
                result = number1 / number2;
            }
            else if(operation == '%')
            {
               result = number1 % number2;
            }
            else
            {
               result = number2;
            }
            String temp = "";
            if(isPoint || operation == '/')//when there is any bPoint number or any bDivision operation,
there may be a bPoint in the result
            {
               tfDisplay.setText(""+result);
               temp = ""+result;
            }
            else if(!isPoint)//when there is no bPoint in the result
            {
               tfDisplay.setText(""+(long)result);
               temp = ""+(long)result;
            }
            operation = '*';
            sDisplay = "";
            number1 = result;
            isPlus = true;
            isPoint = false;
            isOperation = true;
            sRawInput += " \u00D7 ";
            tfRawInput.setText(sRawInput);
            sRawInput = temp + " \u00D7 ";
         }
         else if(operation == '=')
         {
            sDisplay = "";
            operation = '*';
            isPlus = true;
            isOperation = true;
            sRawInput += " \u00D7 ";
            tfRawInput.setText(sRawInput);
         }
```

```
            isOperation = true;
        }
        else if (e.getSource() == bDiv && (!sDisplay.equals("") || operation == '='))  // bDivision button is
clicked and input is not empty
        {
            if(!isOperation)//to check if plus without number1
            {
                number1 = Double.parseDouble(sDisplay);
                sDisplay = "";
                operation = '/';
                isPlus = true;
                sRawInput += " / ";
                tfRawInput.setText(sRawInput);
                if(isPoint)//when there is any bPoint number or any bDivision operation, there may be a
bPoint in the result
                {
                    tfDisplay.setText(""+number1);
                }
                else if(!isPoint)//when there is no bPoint in the result
                {
                    tfDisplay.setText(""+(long)number1);
                }
            }
            else if(isOperation && operation != '=')
            {
                number2  = Double.parseDouble(sDisplay);

                if(operation == '+')
                {
                    result = number1 + number2;
                }
                else if(operation == '-')
                {
                    result = number1 - number2;
                }
                else if(operation == '*')
                {
                    result = number1 * number2;
                }
                else if(operation == '/')
                {
                    result = number1 / number2;
                }
                else if(operation == '%')
                {
                    result = number1 % number2;
                }
                else
                {
                    result = number2;
                }
                String temp = "";
                if(isPoint || operation == '/')//when there is any bPoint number or any bDivision operation,
there may be a bPoint in the result
                {
                    tfDisplay.setText(""+result);
                    temp = ""+result;
                }
                else if(!isPoint)//when there is no bPoint in the result
                {
                    tfDisplay.setText(""+(long)result);
                    temp = ""+(long)result;
                }
                operation = '*';
                sDisplay = "";
                number1 = result;
                isPlus = true;
                isPoint = false;
                isOperation = true;
                sRawInput += " / ";
                tfRawInput.setText(sRawInput);
                sRawInput = temp + " / ";
            }
            else if(operation == '=')
            {
                sDisplay = "";
                operation = '/';
                isPlus = true;
```

```
            isOperation = true;
            sRawInput += " / ";
            tfRawInput.setText(sRawInput);
         }
         isOperation = true;
      }
      else if (e.getSource() == bMod && (!sDisplay.equals("") || operation == '='))//Modules button
      {
         if(!isOperation)//to check if plus without number1
         {
            number1 = Double.parseDouble(sDisplay);
            sDisplay = "";
            operation = '%';
            isPlus = true;
            sRawInput += " mod ";
            tfRawInput.setText(sRawInput);
            if(isPoint)//when there is any bPoint number or any bDivision operation, there may be a
bPoint in the result
            {
               tfDisplay.setText(""+number1);
            }
            else if(!isPoint)//when there is no bPoint in the result
            {
               tfDisplay.setText(""+(long)number1);
            }
         }
         else if(isOperation && operation != '=')
         {
            number2  = Double.parseDouble(sDisplay);

            if(operation == '+')
            {
               result = number1 + number2;
            }
            else if(operation == '-')
            {
               result = number1 - number2;
            }
            else if(operation == '*')
            {
               result = number1 * number2;
            }
            else if(operation == '/')
            {
               result = number1 / number2;
            }
            else if(operation == '%')
            {
               result = number1 % number2;
            }
            else
            {
               result = number2;
            }
            String temp = "";
            if(isPoint || operation == '/')//when there is any bPoint number or any bDivision operation,
there may be a bPoint in the result
            {
               tfDisplay.setText(""+result);
               temp = ""+result;
            }
            else if(!isPoint)//when there is no bPoint in the result
            {
               tfDisplay.setText(""+(long)result);
               temp = ""+(long)result;
            }
            operation = '%';
            sDisplay = "";
            number1 = result;
            isPlus = true;
            isPoint = false;
            isOperation = true;
            sRawInput += " mod ";
            tfRawInput.setText(sRawInput);
            sRawInput = temp + " * ";
         }
         else if(operation == '=')
         {
            sDisplay = "";
            operation = '%';
            isPlus = true;
            isOperation = true;
            sRawInput += " mod ";
            tfRawInput.setText(sRawInput);
        }
        isOperation = true;
    }
    else if (e.getSource() == bEqual && !sDisplay.equals(""))//when bEqual button is clicked and the
input is not empty
    {
        number2  = Double.parseDouble(sDisplay);

        if(operation == '+')
        {
            result = number1 + number2;
        }
        else if(operation == '-')
        {
            result = number1 - number2;
        }
        else if(operation == '*')
        {
            result = number1 * number2;
        }
        else if(operation == '/')
        {
            result = number1 / number2;
        }
        else if(operation == '%')
        {
            result = number1 % number2;
        }
        else
        {
            result = number2;
        }
        String temp = "";
```

```
            if(isPoint || operation == '/')//when there is any bPoint number or any bDivision operation,
there may be a bPoint in the result
            {
               tfDisplay.setText(""+result);
               temp = ""+result;
            }
            else if(!isPoint)//when there is no bPoint in the result
            {
               tfDisplay.setText(""+(long)result);
               temp = ""+(long)result;
            }
            sDisplay = "";
            number1 = result;
            isPlus = true;
            isPoint = false;
            isOperation = true;
            sRawInput += " = ";
            tfRawInput.setText(sRawInput);
            sRawInput = temp;
            operation = '=';
         }
         else if (e.getSource() == bDel && !sDisplay.equals(""))//DE button
         {
            sDisplay = sDisplay.substring(0, sDisplay.length()-1);
            sRawInput = sRawInput.substring(0, sRawInput.length()-1);
            if(sRawInput.equals(""))//after deleting the last digit
            {
               //tfDisplay.setText("0");
               tfRawInput.setText("0");
            }
            else
            {
               //tfDisplay.setText(sDisplay);
               tfRawInput.setText(sRawInput);
            }
         }
         else if (e.getSource() == bClear)//Clear button
         {
            number1 = number2 = result = 0;
```

```
        sDisplay = "";
        operation = ' ';
        tfDisplay.setText("0");
        isPoint = false;
        isPlus = true;
        isOperation = false;
        sRawInput = "";
        tfRawInput.setText("0");
    }
    else if (e.getSource() == bSquare && !sDisplay.equals(""))//Square button
    {
        number1 = Double.parseDouble(sDisplay);
        result = Math.pow(number1, 2);
        String temp = "";
        if(!isPoint)
        {
            tfDisplay.setText(""+(long)result);
            temp = ""+(long)result;
        }
        else
        {
            tfDisplay.setText(""+result);
            temp = ""+result;
        }
        sRawInput += "^2 = ";
        tfRawInput.setText(sRawInput);
        sRawInput = temp;
        sDisplay = "";
        number1 = result;
        operation = '=';
        isPoint = false;
        isOperation = true;
        isPlus = true;
    }
    else if (e.getSource() == bSqrt && !sDisplay.equals(""))//root button
    {
        number1 = Double.parseDouble(sDisplay);
        result = Math.sqrt(number1);
        sRawInput =  "\u221A" + sRawInput;
        tfRawInput.setText(sRawInput);
        tfDisplay.setText(""+result);
        sRawInput = ""+result;
        sDisplay = "";
        number1 = result;
        operation = '=';
        isPoint = true;
        isOperation = true;
        isPlus = true;
    }
    else if (e.getSource() == bCube && !sDisplay.equals("")) //Cube button
    {
        number1 = Double.parseDouble(sDisplay);
        result = Math.pow(number1, 3);
        String temp = "";
        if(!isPoint)
        {
            tfDisplay.setText(""+(long)result);
            temp = ""+(long)result;
        }
        else
        {
            tfDisplay.setText(""+result);
            temp = ""+result;
        }
        sRawInput +="^3 = ";
        tfRawInput.setText(sRawInput);
        sRawInput = temp;
        sDisplay = "";
        number1 = result;
        operation = '=';
        isPoint = false;
        isOperation = true;
        isPlus = true;

    }
    else if (e.getSource() == bOneByN && !sDisplay.equals(""))// 1/n button
    {
        number1 = Double.parseDouble(sDisplay);
```

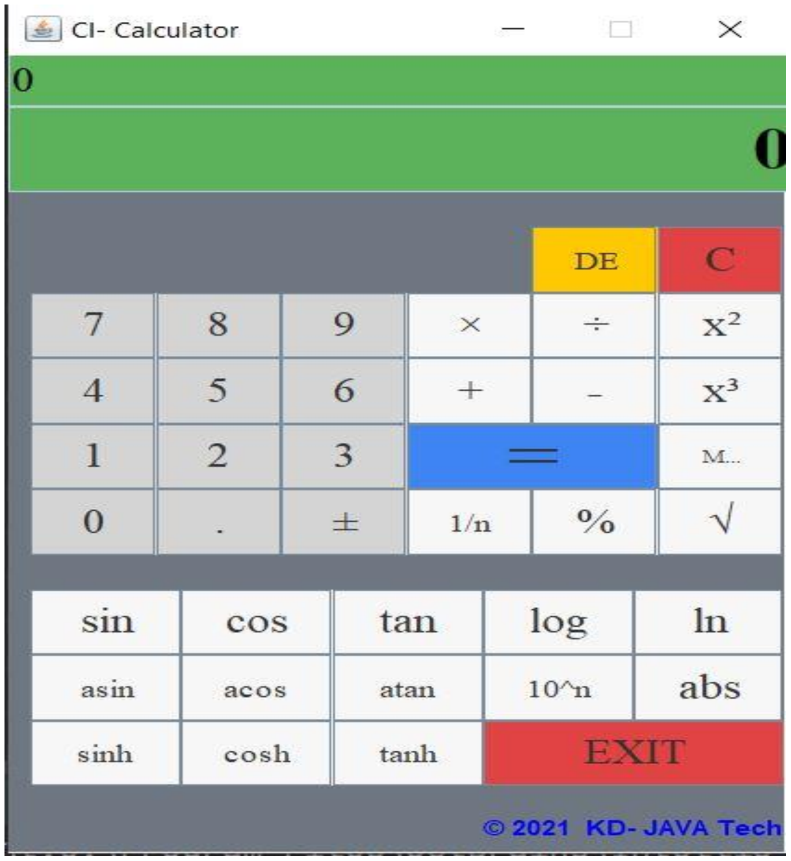
```
            result = 1 / number1;
            if(isPoint)
            {
               sRawInput = "1 / " + number1;
            }
            else
            {

               sRawInput = "1 / " + (long)number1;
            }
            tfRawInput.setText(sRawInput);
            tfDisplay.setText(""+result);
            sDisplay = "";
            sRawInput = ""+result;
            number1 = result;
            operation = '=';
            isPoint = true;
            isOperation = true;
            isPlus = true;
         }
         else if(e.getSource() == bPercent && !sDisplay.equals("") && operation=='*')// % button is
pressed
         {
            number2  = Double.parseDouble(sDisplay);
            result = number1 * (number2 / 100);
            sRawInput = number1+" \u00D7 "+number2+"%";
            tfRawInput.setText(sRawInput);
            tfDisplay.setText(""+result);
            sDisplay = "";
            number1 = result;
            operation = '=';
            isPlus = true;
            isOperation = true;
            isPoint = true;
            sRawInput = ""+result;
         }
         else if(e.getSource() == bSin && !sDisplay.equals(""))//sin function
         {
            number1 = Double.parseDouble(sDisplay);
            if(number1 == 30)
            {
               result = Math.sin(Math.toRadians(number1)) + 0.0000000000000001;
            }
            else
            {
               result = Math.sin(Math.toRadians(number1));
            }
            sRawInput =  "sin("+sRawInput+")";
            tfRawInput.setText(sRawInput);
            tfDisplay.setText(""+result);
            sRawInput = ""+result;
            sDisplay = "";
            number1 = result;
            operation = '=';
            isPoint = true;
            isOperation = true;
            isPlus = true;
         }
         else if(e.getSource() == bCos && !sDisplay.equals(""))//cosine function
         {
            number1 = Double.parseDouble(sDisplay);
            if(number1 == 60)
            {
               result = Math.cos(Math.toRadians(number1)) - 0.0000000000000001;
            }
            else if (number1 == 90)
            {
               result = 0;
            }
            else
            {
               result = Math.cos(Math.toRadians(number1));
            }
            sRawInput =  "cos("+sRawInput+")";
            tfRawInput.setText(sRawInput);
            tfDisplay.setText(""+result);
            sRawInput = ""+result;
            sDisplay = "";
            number1 = result;
```

```
    operation = '=';
    isPoint = true;
    isOperation = true;
    isPlus = true;
  }
  else if(e.getSource() == bTan && !sDisplay.equals(""))//tan function
  {
    number1 = Double.parseDouble(sDisplay);
    if(number1 == 45)
    {
      result = Math.tan(Math.toRadians(number1)) + 0.0000000000000001;
    }
    else if(number1 == 90)
    {
      result = 0;
      tfDisplay.setText("Invalid");
    }
    else
    {
      result = Math.tan(Math.toRadians(number1));
    }
    sRawInput =  "tan("+sRawInput+")";
    tfRawInput.setText(sRawInput);
    if(number1 != 90)
    {
      tfDisplay.setText(""+result);
    }
    sRawInput = ""+result;
    sDisplay = "";
    number1 = result;
    operation = '=';
    isPoint = true;
    isOperation = true;
    isPlus = true;
  }
  else if(e.getSource() == bSinH && !sDisplay.equals(""))//sinh function
  {
    number1 = Double.parseDouble(sDisplay);
    result = Math.sinh(Math.toRadians(number1));
```

```
        sRawInput =  "sinh("+sRawInput+")";
        tfRawInput.setText(sRawInput);
        tfDisplay.setText(""+result);
        sRawInput = ""+result;
        sDisplay = "";
        number1 = result;
        operation = '=';
        isPoint = true;
        isOperation = true;
        isPlus = true;
    }
    else if(e.getSource() == bCosH && !sDisplay.equals(""))//cosh function
    {
        number1 = Double.parseDouble(sDisplay);
        result = Math.cosh(Math.toRadians(number1));
        sRawInput =  "cosh("+sRawInput+")";
        tfRawInput.setText(sRawInput);
        tfDisplay.setText(""+result);
        sRawInput = ""+result;
        sDisplay = "";
        number1 = result;
        operation = '=';
        isPoint = true;
        isOperation = true;
        isPlus = true;
    }
    else if(e.getSource() == bTanH && !sDisplay.equals(""))//tanh function
    {
        number1 = Double.parseDouble(sDisplay);
        result = Math.tanh(Math.toRadians(number1));
        sRawInput =  "tanH("+sRawInput+")";
        tfRawInput.setText(sRawInput);
        tfDisplay.setText(""+result);
        sRawInput = ""+result;
        sDisplay = "";
        number1 = result;
        operation = '=';
        isPoint = true;
        isOperation = true;
        isPlus = true;
    }
    else if(e.getSource() == bAsin && !sDisplay.equals(""))//asin function
    {
        number1 = Double.parseDouble(sDisplay);
        result = Math.asin(Math.toRadians(number1));
        sRawInput =  "asin("+sRawInput+")";
        tfRawInput.setText(sRawInput);
        tfDisplay.setText(""+result);
        sRawInput = ""+result;
        sDisplay = "";
        number1 = result;
        operation = '=';
        isPoint = true;
        isOperation = true;
        isPlus = true;
    }
    else if(e.getSource() == bAcos && !sDisplay.equals(""))//acos function
    {
        number1 = Double.parseDouble(sDisplay);
        result = Math.acos(Math.toRadians(number1));
        sRawInput =  "acos("+sRawInput+")";
        tfRawInput.setText(sRawInput);
        tfDisplay.setText(""+result);
        sRawInput = ""+result;
        sDisplay = "";
        number1 = result;
        operation = '=';
        isPoint = true;
        isOperation = true;
        isPlus = true;
    }
    else if(e.getSource() == bAtan && !sDisplay.equals(""))//atan function
    {
        number1 = Double.parseDouble(sDisplay);
        result = Math.atan(Math.toRadians(number1));
        sRawInput =  "atan("+sRawInput+")";
        tfRawInput.setText(sRawInput);
        tfDisplay.setText(""+result);
        sRawInput = ""+result;
        sDisplay = "";
        number1 = result;
        operation = '=';
        isPoint = true;
        isOperation = true;
        isPlus = true;
    }
    else if(e.getSource() == bLog && !sDisplay.equals(""))//log function
    {
        number1 = Double.parseDouble(sDisplay);
        result = Math.log10(number1);
        sRawInput =  "log"+sRawInput;
        tfRawInput.setText(sRawInput);
        tfDisplay.setText(""+result);
        sRawInput = ""+result;
        sDisplay = "";
        number1 = result;
        operation = '=';
        isPoint = true;
        isOperation = true;
        isPlus = true;
    }
    else if(e.getSource() == bLn && !sDisplay.equals(""))//ln function
    {
        number1 = Double.parseDouble(sDisplay);
        result = Math.log(number1);
        sRawInput =  "ln"+sRawInput;
        tfRawInput.setText(sRawInput);
        tfDisplay.setText(""+result);
        sRawInput = ""+result;
        sDisplay = "";
        number1 = result;
        operation = '=';
        isPoint = true;
        isOperation = true;
        isPlus = true;
    }
    else if(e.getSource() == bAbs && !sDisplay.equals(""))//abs function
```

```
{
   number1 = Double.parseDouble(sDisplay);
   result = Math.abs(number1);
   sRawInput =  "abs("+sRawInput+")";
   tfRawInput.setText(sRawInput);
   tfDisplay.setText(""+result);
   sRawInput = ""+result;
   sDisplay = "";
   number1 = result;
   operation = '=';
   isPoint = true;
   isOperation = true;
   isPlus = true;
}
else if(e.getSource() == bPowerOfTen && !sDisplay.equals(""))//power of ten function
{
   number1 = Double.parseDouble(sDisplay);
   result = Math.pow(10, number1);
   sRawInput =  "10^"+sRawInput;
   tfRawInput.setText(sRawInput);
   tfDisplay.setText(""+result);
   sRawInput = ""+result;
   sDisplay = "";
   number1 = result;
   operation = '=';
   isPoint = true;
   isOperation = true;
   isPlus = true;
}
else if(e.getSource() == bExit)//exit button
{
   System.exit(0);
}   }  }
```

Output :-


CI- Calculator
0
0
DE
C
7
8
9
×
÷
x²
4
5
6
+
-
x³
1
2
3
=
M...
0
.
±
1/n
%
√
sin
cos
tan
log
ln
asin
acos
atan
10^n
abs
sinh
cosh
tanh
EXIT
© 2021 KD- JAVA Tech