**PROBLEM DEFINITION**

As a group of Computer Science Engineering students we are interested to build a calculator that enables people to perform mathematical operations easily. The user can easily perform various mathematical operations such as addition, subtraction, multiplication and division. This program shall reduce the complexity of performing various calculations to a very simple task. The program is designed in such a way that the user doesn’t have to consider the BODMAS/PEMDAS rule while performing calculations, the program will automatically take into consideration the same. The user will be able to enter an expression, for example:

Eg:- 12+3\*2-4

This would be done simply by pressing various buttons denoting the numbers as well as operators and then calculate the result simply by pressing a button denoted by ”=”. For example the result of the above expression would be displayed as follows:

14

**Aim:**

The main purpose of the Calculator is to provide a simple solution for the user when performing mathematical calculations. The program is a simple swing based GUI program which can be used on any computer supporting JAVA. It is a program built with the consideration of the BODMAS/PEMDAS rule, which can be quite tasking for someone to keep in mind while using a calculator that does not support the same. After inserting the expression into the program, the user does not need to worry about performing the calculations, and hence it reduces work required to perform calculations.

**Objectives:**

Performing various calculations is a very common task done by people of all ages, from children to adults, often times slowing down important processes to be done because of slow calculation. Calculating is a task involved in basically all places, we as people have been performing various mathematical calculations since decades. Through this project we present a simple solution for performing various calculations, which is easy to understand, easy to use, and makes day to day life easier for the users.

The goals of the project are:-

* To perform addition of multiple numbers
* To perform subtraction of multiple numbers
* To perform multiplication of multiple numbers
* To perform division of multiple numbers
* To use the BODMAS/PEMDAS rule in the calculation
* To simplify the process of calculations

**Code:-**

import java.awt.\*;

import java.awt.event.\*;

import javax.swing.\*;

import java.util.\*;

class Calc

{

boolean opstate,ispoint=false;

JPanel p;

JButton invert, zero, point, solve, one, two, three, four, five, six, seven, eight, nine, plus, minus, multiply, divide, back, cancel, cancele;

JTextField ip;

JTextField exp;

Calculate cobj=new Calculate();

Calc()

{

JFrame f=new JFrame("Calculator");

f.setLayout(null);

f.addWindowListener(new WindowAdapter() {

public void windowClosing(WindowEvent e) {

System.exit(0);

}

});

p=new JPanel();

p.setLayout(null);

p.setBounds(0,0,400,400);

invert=new JButton("\u00B1");

invert.setBounds(50,280,80,30);

zero=new JButton("0");

zero.setBounds(130,280,80,30);

point=new JButton(".");

point.setBounds(210,280,80,30);

solve=new JButton("=");

solve.setBounds(290,280,80,30);

one=new JButton("1");

one.setBounds(50,250,80,30);

two=new JButton("2");

two.setBounds(130,250,80,30);

three=new JButton("3");

three.setBounds(210,250,80,30);

four=new JButton("4");

four.setBounds(50,220,80,30);

five=new JButton("5");

five.setBounds(130,220,80,30);

six=new JButton("6");

six.setBounds(210,220,80,30);

seven=new JButton("7");

seven.setBounds(50,190,80,30);

eight=new JButton("8");

eight.setBounds(130,190,80,30);

nine=new JButton("9");

nine.setBounds(210,190,80,30);

plus=new JButton("\u002B");

plus.setBounds(290,250,80,30);

minus=new JButton("\u2212");

minus.setBounds(290,220,80,30);

multiply=new JButton("\u00D7");

multiply.setBounds(290,190,80,30);

divide=new JButton("\u00F7");

divide.setBounds(290,160,80,30);

back=new JButton("\u232B");

back.setBounds(210,160,80,30);

cancel=new JButton("C");

cancel.setBounds(130,160,80,30);

cancele=new JButton("CE");

cancele.setBounds(50,160,80,30);

ip=new JTextField("0");

ip.setBounds(50,130,320,30);

ip.setHorizontalAlignment(JTextField.RIGHT);

ip.setEditable(false);

exp=new JTextField("");

exp.setBounds(50,95,320,30);

exp.setHorizontalAlignment(JTextField.RIGHT);

exp.setEditable(false);

ActionListener al=new ActionListener(){

public void actionPerformed(ActionEvent ae){

String inp=ip.getText();

if(inp.equals("0")||opstate)

{

inp="";

}

if(ae.getSource()==zero)

ip.setText(inp+"0");

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

ip.setText(inp+"1");

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

ip.setText(inp+"2");

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

ip.setText(inp+"3");

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

ip.setText(inp+"4");

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

ip.setText(inp+"5");

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

ip.setText(inp+"6");

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

ip.setText(inp+"7");

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

ip.setText(inp+"8");

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

ip.setText(inp+"9");

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

{

if(inp.equals(""))

ip.setText("0");

else if(inp.substring(0,1).equals("~"))

ip.setText(inp.substring(1,inp.length()));

else

ip.setText("~"+inp);

}

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

{

if(ispoint&&!opstate)

return;

if(inp.equals("")||opstate)

ip.setText("0.");

else

ip.setText(inp+".");

ispoint=true;

}

opstate=false;

}

};

ActionListener al1=new ActionListener(){

public void actionPerformed(ActionEvent ae){

String inp=ip.getText();

if(inp.equals("Invalid"))

return;

if(opstate)

opstate=false;

if(opercheck())

inp="";

ispoint=false;

if(ae.getSource()==plus)

{

exp.setText(exp.getText()+inp+"+");

ip.setText("0");

}

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

{

exp.setText(exp.getText()+inp+"-");

ip.setText("0");

}

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

{

exp.setText(exp.getText()+inp+"\*");

ip.setText("0");

}

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

{

exp.setText(exp.getText()+inp+"/");

ip.setText("0");

}

}

};

ActionListener al2=new ActionListener(){

public void actionPerformed(ActionEvent ae){

String inp=ip.getText();

if(inp.equals("Invalid"))

return;

if(ae.getSource()==back)

{

if(opstate)

return;

ispoint=false;

if(!inp.equals("0"))

{

if(inp.substring(inp.length()-1,inp.length()).equals("."))

ispoint=false;

ip.setText(inp.substring(0,inp.length()-1));

}

if(ip.getText().equals(""))

ip.setText("0");

}

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

{

ispoint=false;

exp.setText("");

ip.setText("0");

}

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

{

ispoint=false;

ip.setText("0");

}

}

};

ActionListener al3=new ActionListener(){

public void actionPerformed(ActionEvent ae){

if(ip.getText().equals("Invalid"))

return;

String send=exp.getText()+ip.getText();

cobj.setExpression(send);

exp.setText("");

ip.setText(cobj.getResult());

opstate=true;

}

};

zero.addActionListener(al);

one.addActionListener(al);

two.addActionListener(al);

three.addActionListener(al);

four.addActionListener(al);

five.addActionListener(al);

six.addActionListener(al);

seven.addActionListener(al);

eight.addActionListener(al);

nine.addActionListener(al);

invert.addActionListener(al);

point.addActionListener(al);

plus.addActionListener(al1);

minus.addActionListener(al1);

multiply.addActionListener(al1);

divide.addActionListener(al1);

back.addActionListener(al2);

cancel.addActionListener(al2);

cancele.addActionListener(al2);

solve.addActionListener(al3);

p.add(invert);

p.add(point);

p.add(zero);

p.add(solve);

p.add(one);

p.add(two);

p.add(three);

p.add(four);

p.add(five);

p.add(six);

p.add(seven);

p.add(eight);

p.add(nine);

p.add(plus);

p.add(minus);

p.add(multiply);

p.add(divide);

p.add(back);

p.add(cancel);

p.add(cancele);

p.add(ip);

p.add(exp);

f.add(p);

f.setSize(400,400);

f.setVisible(true);

}

boolean opercheck()

{

String chk=exp.getText();

String inex=ip.getText();

if(chk.equals(""))

return false;

if(inex.equals("0"))

{

char last=chk.charAt(chk.length()-1);

if(last=='+'||last=='-'||last=='\*'||last=='/')

{

exp.setText(chk.substring(0,chk.length()-1));

return true;

}

}

return false;

}

public static void main(String args[])

{

new Calc();

}

}

class Calculate

{

String exp[];

int length;

void setExpression(String ex)

{

String delimit="((?<=\\+)|(?=\\+)|(?<=-)|(?=-)|(?<=\\\*)|(?=\\\*)|(?<=/)|(?=/)|(?<=~)|(?=~))";

exp=ex.split(String.format(delimit, ";"));

length=exp.length;

}

String getResult()

{

int i=0,j=0;

Vector<String> stk=new Vector<String>();

String postfix[]=new String[length];

while(i<length)

{

if(exp[i].equals("+")||exp[i].equals("-")||exp[i].equals("\*")||exp[i].equals("/"))

{

while((stk.size()>=1)&&(getPriority(stk.elementAt(stk.size()-1))>getPriority(exp[i])))

{

postfix[j]=stk.elementAt(stk.size()-1);

stk.removeElementAt(stk.size()-1);

j++;

}

stk.add(exp[i]);

i++;

}

else

{

postfix[j]=exp[i];

i++;

j++;

}

}

while(stk.size()>=1)

{

postfix[j]=stk.elementAt(stk.size()-1);

stk.removeElementAt(stk.size()-1);

j++;

}

return evaluate(postfix);

}

int getPriority(String op)

{

if(op.equals("+")||op.equals("-"))

return 0;

else

return 1;

}

String evaluate(String postfix[])

{

int i=0;

length=postfix.length;

String now=null;

Vector<Double> stk=new Vector<Double>();

double opnd1,opnd2,res=0;

while(i<length)

{

now=postfix[i];

if(isNumeric(now))

{

stk.add(Double.parseDouble(now));

}

else if(now.equals("~"))

{

i++;

now=postfix[i];

Double d=Double.parseDouble(now);

d=d\*-1;

stk.add(d);

}

else

{

opnd2=stk.elementAt(stk.size()-1);

stk.removeElementAt(stk.size()-1);

opnd1=stk.elementAt(stk.size()-1);

stk.removeElementAt(stk.size()-1);

switch(now)

{

case "+": res=opnd1+opnd2;

break;

case "-": res=opnd1-opnd2;

break;

case "\*": res=opnd1\*opnd2;

break;

case "/":

res=opnd1/opnd2;

if(Double.isNaN(res)||Double.isInfinite(res))

{

return "Invalid";

}

break;

}

stk.add(res);

}

i++;

}

res=stk.elementAt(stk.size()-1);

stk.removeElementAt(stk.size()-1);

now=Double.toString(res);

if(res<0)

now="~"+now.substring(1,now.length());

return now;

}

public static boolean isNumeric(String str) {

try

{

double d = Double.parseDouble(str);

}

catch(NumberFormatException nfe)

{

return false;

}

return true;

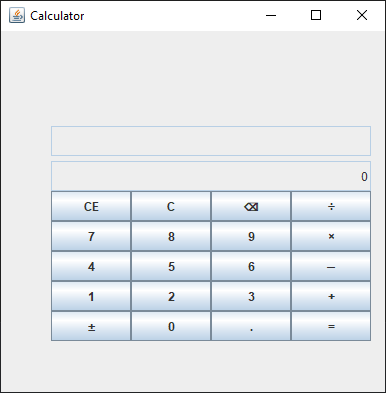
}

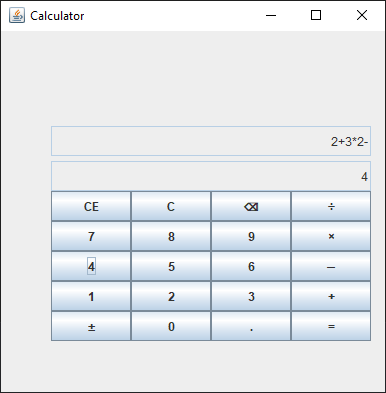
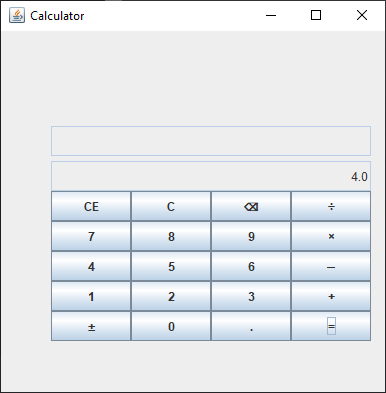
}

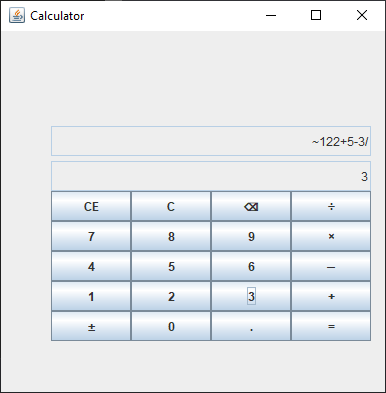
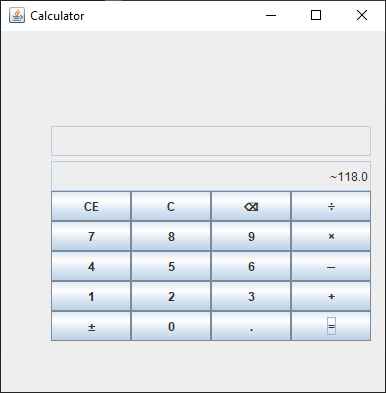
**Understanding the Calculator Application:-**

|  |  |  |
| --- | --- | --- |
| Class | Methods | Use |
| Calc | boolean opercheck() | A method used to check if the last input was an operator and if yes remove it from the expression and return true |
| WindowAdapter  (an anonymous class) | void windowClosing(WindowEvent e) | A method used to close the JFrame when user presses the close button. |
| ActionListener() (al)  (an anonymous class) | void actionPerformed(ActionEvent ae) | Action event for buttons: 0 to 9, invert and point to input them into the JTextfield |
| ActionListener() (al1)  (an anonymous class) | void actionPerformed(ActionEvent ae) | Action event for buttons: plus, minus, multiply and divide to input them into the JTextfield |
| ActionListener() (al2)  (an anonymous class) | void actionPerformed(ActionEvent ae) | Action event for buttons: back, cancel and cancele to do their respective operations on the JTextfield |
| ActionListener() (al3)  (an anonymous class) | void actionPerformed(ActionEvent ae) | Action event for button: equals to perform the calculation and to display the result in the JTextfield |
| Calc | public static void main(String args) | To start the program and create object of the class Calc |
| Calc | Calc() | A constructor to add the various swing components |
| Calculate | void setExpression(String ex) | To receive the expression string from the object of Calc and split it into an array of strings |
| Calculate | String getResult() | To convert the infix expression into postfix expression, call the method evaluate and return the result string |
| Calculate | int getPriority(String op) | To return the priority of the operator contained in op |
| Calculate | String evaluate(String postfix[]) | To evaluate the postfix expression and return the result string |
| Calculate | static boolean isNumeric(String str) | To check if string str is a number or not and return true if yes |

**Screenshots:-**



**Conclusion**

The project has been developed successfully and the performance of the program has been found satisfactory. Use of computer helps the user in reducing time wasted in performing repeated tasks. It further helps the user in performing complex calculations easily.

User friendly Graphical User Interface has been provided to the user to interact with the program. Users can perform various operations at the simple click of a button.

**Future Scope**

The scope of this project is providing the user with more possible operations such as factorial, square, square root, powers, permutations, combinations, trigonometric operations, brackets, complex numbers, and also providing various utilities like solving simultaneous equations, etc.

**Limitations**

* Brackets are not implemented

Brackets have not been implemented in this program to keep it simple and remove various possible errors that may occur due to faulty input they may cause.

* No history provided

No history has been provided for past calculations, this feature can be found in most present calculators

* Lacking various advanced operations

The various advanced operations such as factorial, square, square root, powers, permutations, combinations, trigonometric operations have not been provided in this project.

**References:**

1. [www.google.co.in](http://www.google.co.in)
2. [www.stackoverflow.com](http://www.stackoverflow.com)
3. [www.javatpoint.com](http://www.javatpoint.com)