Proyecto 1: Estructuras de Datos

Calculadora

Otoño 2021

Profesora Silvia Guardati Buemo

Instituto Nacional Autónomo de México

Miembros del equipo

Acosta López Liliana

Díaz Barriga Gómez del Campo Daniel

Hellberg Yanci Alexander

Hernández Salas José Alejandro

Nieto Merodio Javier

Tabla de contenido

Descripción del problema

1.1. Objetivo

Diseñar un programa que simule una calculadora capaz de: resolver expresiones usando operaciones básicas, hacer un proceso de revisión de los valores entregados por el usuario y garantizar una salida exitosa del resultado. Esto, con un uso eficiente en las líneas de código implementadas.

1.2. Requisitos

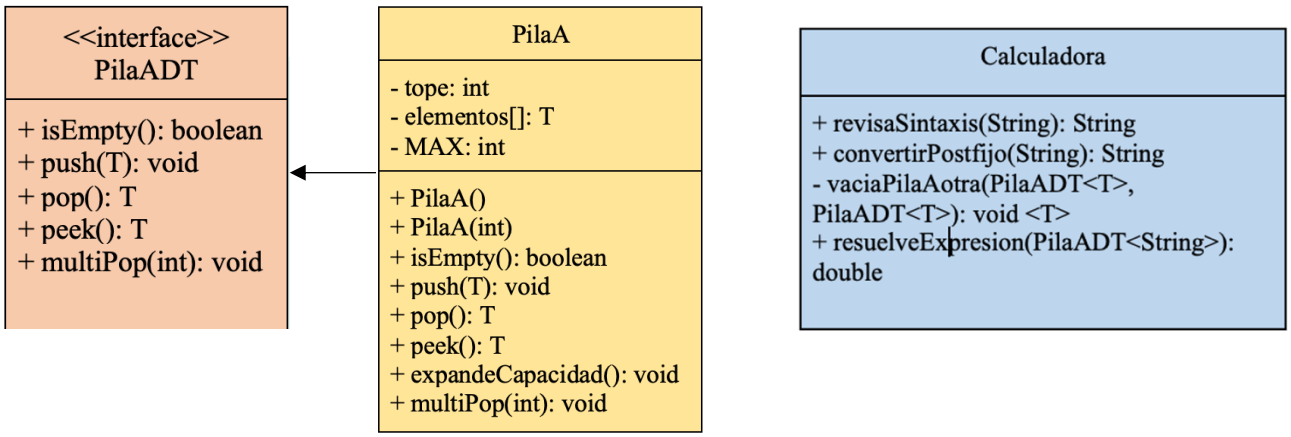
El programa diseñado debe tener una interfaz gráfica (GUI) en la que se ingrese la expresión a evaluar. Debe poder trabajar con números negativos, decimales y potencias sobre los números reales. Tiene que ser capaz de incluir en los cálculos las siguientes operaciones: suma, resta, división, multiplicación y potencias (raíces incluidas como potencias).

1.3. Restricciones

Para la solución del problema se debe utilizar Netbeans como IDE. Para poder crear el programa de la calculadora se deben implementar métodos que hagan una revisión de la sintaxis y/o errores de escritura. Además, es importante denotar que, para el proceso interno de evaluación, se debe convertir la expresión que ingrese el usuario de tipo infija a postfija.

2. Solución diseñada

2.1. UML de clases



2.2. Pseudocódigo

Método para revisar la cadena

```
Entrada: String operacion
Función revisaSintaxis
i := 0;
Mientras que i<longitud(cadena) y res≠ “” hacer
    Según cadena(i) hacer
        caso +:
        caso -:
          Si i=0 o i=longitud(cadena)-1 o operadores contains(cadena(i+1)) o operadores
          contains(cadena(i-1)) o cadena(i+1)≠ ( o cadena(i+1)≠ ) entonces
              error="ERROR EN SUMA O SUBTRACCION ["+i+"]";
```

Método para pasar la expresión a postfijo

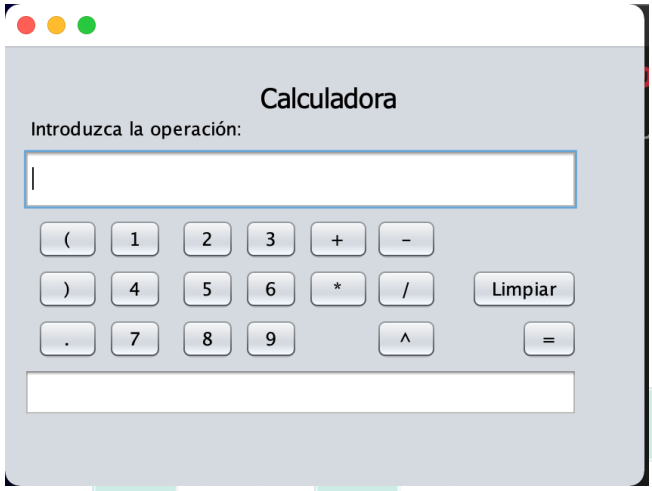
```
Entrada: PilaADT<String> pila
Función resolver expresión dada

PilaA<String> aux1:=new PilaA();
PilaA<String> aux2:=new PilaA();
PilaA<String> aux3:=new PilaA();
VaciaPilaAotra(pila, aux1);

Mientras aux1≠isEmpty() hacer
    simbolo:=aux1.pop();
    Según simbolo hacer
        caso + ;
          res:=Double.parseDouble(aux2.pop()) + Double.parseDouble(aux2.pop());
          aux2.push(String.valor(res));
```

Método para resolver la expresión

```
Entrada: String infijo
Función convertir infijo a postfijo
PilaADT<Caracter> aux := new PilaA<>();
Para i:=0 Hasta longitud(infijo) Con Paso i+1 Hacer
    neg:=0;
    ch:= caracter(infijo(i));
    Si ch= ( entonces
        aux.push(ch);
        neg:=0;
    Si no si ch=) entonces
        Mientras aux.peek() ≠( hacer
          postfijo:= postfijo+aux.pop();
        Fin Mientras
        aux.pop();
    Si no si ch.isDigit entonces
        postfijo:=postfijo+ch;
        neg:=1;
    Si no si +-*/^~.indexOf(ch)>=0 entonces
 Mientras aux≠isEmpty() y comparaPrioridad(aux.peek(), ch) hacer
            postfijo:=postfijo+aux.pop();
          Fin Mientras
          aux.push(ch);
    Fin Si
Fin Para
Mientras aux≠isEmpty() hacer
    postfijo:=postfijo+aux.pop();
Fin Mientras
return postfijo
```

Fin Función

2.3. Pantalla



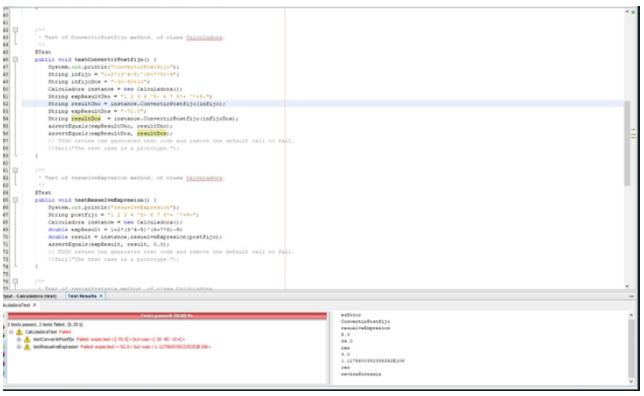
3. Pruebas

Cada método fue probado con diferentes datos, comprobando que cada uno cumpla su función adecuadamente, aunque también se incluyó una operación que se pasó por todos los métodos tal y como lo hace en el programa completo.

En revisarSintaxis se probó que el algoritmo encontrará errores cuando los haya, que mantendrá igual una operación bien escrita y, finalmente, que al encontrar un número negativo lo modificará de manera correcta. De manera similar, esError se probó usando textos de error como los que podrían surgir en revisarSintaxis y operaciones correctas.

Por otro lado, en convertirPostfijo, las pruebas se realizaron con operaciones largas que usen gran variedad de operadores y paréntesis; también se probó que actuara de manera correcta cuando encuentre puntos decimales y números negativos.

Finalmente, resuelveExpresion se probó usando operaciones en postfijo que usen distintos tipos de operadores, incluyendo también números con punto decimal y números negativos.



En un punto dado, 2 de 4 pruebas fueron fallidas. Se pudo analizar que estaban relacionadas a la precisión de las herramientas dadas por las librerías para poder calcular

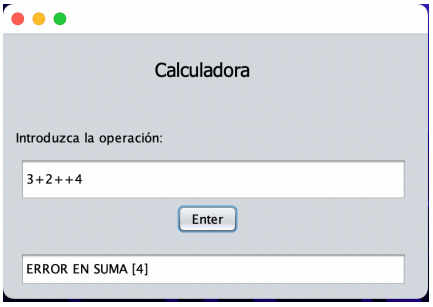
Los errores que llegaron a aparecer en el proceso de pruebas fueron menores y se solucionaron sin mayor problema, hasta que el programa pasó las pruebas.

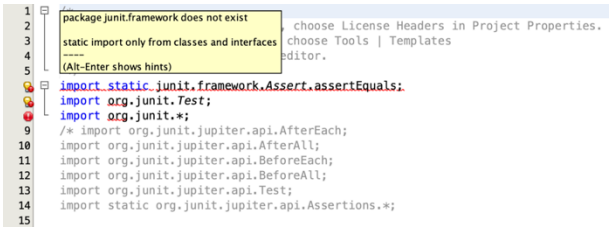
4. Limitaciones de la solución

Para diferenciar entre un número negativo y una resta en el código que se programó, se utilizaron los símbolos: "~" y "-" respectivamente. Cuando el usuario indique el uso de un número negativo, éste deberá señalar el botón (-). Este símbolo no podrá ser utilizado para una resta, ya que aparecerá "ERROR", sin embargo, si la operación comienza con el símbolo " ~ ", será válida. Entonces, el código interno maneja el negativo de un número y una resta de distinta manera. Cabe aclarar que el usuario ingresa un número negativo con el símbolo "-" como en cualquier calculadora. En primeras versiones del código, el usuario sí debía ingresar el símbolo "~". No obstante, esto pudo corregirse.

El usuario únicamente podrá realizar las operaciones básicas de suma, resta, multiplicación, división y potencia, ya que las raíces también pueden expresarse como potencias.

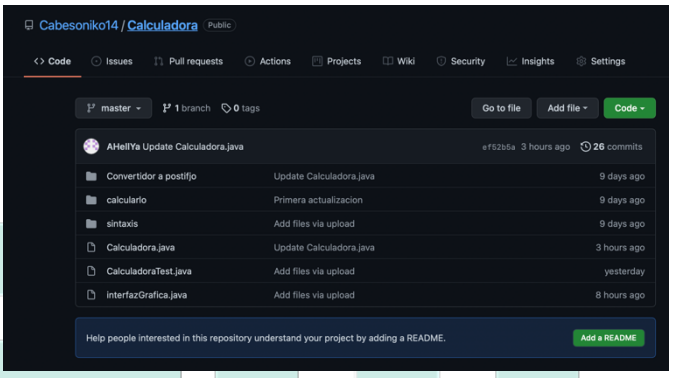
Si el usuario comete un error en la escritura de la operación, en el resultado aparecerá "ERROR", señalando también dónde se encuentra el error dentro de la operación, que se da por el mal uso de los paréntesis, el doble uso de operadores, entre otros; por lo que deberá volver a escribirla o corregirla.





Finalmente, para poder ejecutar la clase CalculadoraTest.java donde se hicieron las pruebas de JUnit, se deben tener las librerías de prueba de JUnit4. Si se tiene JUnit5, saltarán errores como el mostrado a continuación. Esto supone una gran limitación.

5. Posibles mejoras y conclusiones



Tras haber terminado el producto pedido, se llegaron a varias posibles mejoras para futuros proyectos de una calculadora postfija y, en general, para proyectos de programación en equipo. En primer lugar, en temas de organización y eficiencia en el área de trabajo, se llegó a una conclusión importante: es de vital importancia tener un buen espacio o programa donde todos puedan ver el código trabajado y hacer cambios sin ningún problema. Al principio, el equipo acudió a Google Drive para subir el código. Esta aplicación es útil para que todos vean el código, pero no para hacer modificaciones. Por ello, se cambió a la ampliamente recomendada plataforma GitHub tras haber hecho el proyecto. GitHub nos permitió tener un ambiente más orientado a la escritura de código, donde podíamos ver el programa y además hacer modificaciones sin mayor problema. Haber usado esta plataforma desde el principio habría ayudado en el proceso de hacer distintos métodos con la seguridad de que se estaban manejando los mismos tipos de datos.

Por el lado de la escritura de código, es ampliamente recomendable encontrar una forma eficiente para manejar los números de más de un dígito. Al recibir números en tipo String, a la hora de hacer el cálculo de la expresión en postfijo, como los operandos van juntos, puede confundirse cuáles son las cifras manejadas. También, luego de haber tenido un serio problema a la hora de diferenciar números negativos de una resta, concluimos que usar un símbolo diferente a “-” para denotar

números negativos (conversión interna del programa sin involucrar al usuario) es una gran solución.

A modo de conclusión, este proyecto nos sirvió a todos los integrantes para aprender a manejar proyectos más grandes de Java y entender aún mejor la aplicación de los conocimientos aprendidos. Pero, lo más importante: nos ayudó a entender mejor cómo es trabajar un programa en equipo. Entender los retos y beneficios de la comunicación, conocer espacios cómodos para compartir y comentar el código y generar habilidades sociales a la hora de incluir a varias personas en un mismo programa. Creemos que no solo es importante aprender a programar, sino también a colaborar en el mundo de la programación.

Bibliografía

1. Buemo, S. G. (2016). *Estructuras de datos básicas programación orientada a objetos con java* (1.ª ed.). Alfaomega GPO.

Apéndice con el código

Calculadora.java

```
package com.mycompany.calculadora;

import java.util.ArrayList;
import java.util.Arrays;

/*
 *<pre>
 * Clase Calculadora
 *
 * Recibe operaciones matematicas y las resuelve.
 *</pre>
 *@author Daniel, Javi, Liliana, Alexander, Jose
 */
public class Calculadora {

    /**
    * Metodo que recibe una operacion en infijo y la convierte a postfijo
    * @param infijo Cadena con la operacion en infijo
    * @return
    * <ul>
    * <li>La cadena convertida a postfijo</li>
```

```
 * </ul>
 */
public String ConvertirPostfijo(String infijo){
    String postfijo = "";
    PilaADT<Character> aux = new PilaA<>();
    int negativo = 0;
    for (int i = 0; i < infijo.length(); i++){

        char ch = infijo.charAt(i);
        if (ch == '(') {
            aux.push(ch);
        }
        else if (ch == ')'){
            while (aux.peek() != '(' )
                postfijo += aux.pop();
            aux.pop(); // Quitar paréntesis derecho
        }
        else if ( ch== '~'){
            negativo = 1;
        }
        else if (esUnDigito(ch)&&negativo == 0 ){
            postfijo += ch;
        }
        else if (esUnDigito(ch)&& negativo == 1){
            postfijo +='~';
            postfijo += ch;
            negativo = 0;
        }
```

```
        else if ("+-*/^".indexOf(ch)>=0){
            postfijo += ' ';
            while ((!aux.isEmpty()) && comparaPrioridad(aux.peek(), ch))
                postfijo += aux.pop();
            aux.push(ch);
        }

    }

    while (!aux.isEmpty())
        postfijo += aux.pop();
    return postfijo;
}

/**
 * Recibe un operador y le asigna una prioridad
 * @param ch Operador
 * @return <ul>
 * <li> 1: si el operador es una suma o una resta </li>
 * <li> 2: si el operador es una multiplicacion o una division </li>
 * <li> 3: si es un caret </li>
 *
 * </ul>
 */
private int prioridad(char ch){
    switch (ch) {
        case '+':
        case '-':
            return 1;
```

```
        case '*':
        case '/':
            return 2;
        case '^':
            return 3;
        default:
            return 0;
    }
}
/**
 * Compara la prioridad de dos operadores
 * @param enPila Operador que se encuentra en la pila
 * @param nuevo Operdor que se acaba recibir
 * @return <ul>
 * <li> true: cuando la prioridad de enPila es mayor al nuevo </li>
 * <li> false:  cuando la prioridad de en Pila es menor al nuevo </li>
 *
 * </ul>
 */
private boolean comparaPrioridad(char enPila, char nuevo){
    int prioPila = prioridad(enPila);
    int prioNuevo = prioridad(nuevo);
    boolean res = prioPila >= prioNuevo;
    return res;
}
/**
 *  Comprueba si un caracter es un numero o un punto.
 * @param ch Caracter a comprobar
 * @return<ul>
```

```
 * <li> true: cuando es un numero o un punto </li>
 * <li> false:  cuando no es un numero o un punto </li>
 *
 * </ul>
 *
 */
private boolean esUnDigito(char ch){
        return (ch>='0' && ch<='9') || ch=='.';
        }

 /**
 * Metodo que recibe una operacion en postfijo y la resuelve
 * @param postfijo Cadena con la operacion en postfijo
 * @return
 * <ul>
 * <li>El resultado de la operación</li>
 * </ul>
 */
public double resuelveExpresion(String postfijo){
    postfijo += " ";
    double res;
    PilaADT<Double> PilitaAux = new PilaA();
    String num = "";
    for (int i = 0; i < postfijo.length()-1; i++){
        char ch = postfijo.charAt(i);
       if (esUnDigito(ch)||ch == '~'){
          if (ch == '~')
              num += '-';
          else
```

```
            num += ch;
        System.out.println(ch);
    }
    if (!esUnDigito(postfijo.charAt(i+1)) && !num.equals("")){
        PilitaAux.push(Double.parseDouble(num));
        System.out.println(num+"Agregado");
        num = "";
    }
     switch (ch) {
         case '+' :{
             double x = (double) PilitaAux.pop();
             double y = (double) PilitaAux.pop();
             System.out.println(x);
             System.out.println(y);
             res = x+y;
             PilitaAux.push(res);
             break;
         }
         case '-' : {
             double x = (double) PilitaAux.pop();
             System.out.println(x);
             double y = (double) PilitaAux.pop();
             System.out.println(y);
             res = y-x;
             System.out.println("res"+res);
             PilitaAux.push(res);
             break;
         }
         case '*' : {
```

```
        double x = (double) PilitaAux.pop();
        double y = (double) PilitaAux.pop();
        System.out.println(x);
        System.out.println(y);
        res = x*y;
        PilitaAux.push(res);
        break;
    }
    case '/' : {
        double x = (double) PilitaAux.pop();
        double y = (double) PilitaAux.pop();
        res = y/x;
        PilitaAux.push(res);
        break;
    }
    case '^' : {
        double x = (double) PilitaAux.pop();
        double y = (double) PilitaAux.pop();
        System.out.println(x);
        System.out.println(y);
            if(y<0){
                if(x < 1 && x%1 != x){
                    int denominador = 1;

                    while(x%1 != x){
                        y*=10;
                        denominador*=10;
                    }
                    int a = (int) x;
```

```
                int b =denominador;

                while(a != 1){

                        while (b != 0) {

                         if (a > b)
                           a = a - b;
                         else
                        b = b - a;
                                        }

                        if(a !=1){
                                        x= x/a;
                                        denominador= denominador/a;
                                        }
                                }
                if(denominador%2==1){
                        res = -1*Math.pow(y,x);
                        res = Math.pow(res,1/denominador);

                }else{
                        System.out.println("Numeros negativos no tienen
raices pares.");
                                        }
                }

        }
```

```
                    else{
                  res = Math.pow(y,x);
                   PilitaAux.push(res);
              break;}
              }
          }
       }

    return (double) PilitaAux.pop();
}


 /**
  * Metodo que evalua una operacion en infijo y determina su vlaidez
  * @param cadena Cadena con la operacion en infijo
  * @return
  * <ul>
  * <li>La misma cadena si es correcta</li>
  * <li>Si tiene un elemento x negatico convierte el simbolo a ~</li>
  * <li>Error si la operación no es válida</li>
  * </ul>
  */
 public String revisaSintaxis(String cadena){
    String res="", error="";
    StringBuilder nuevaCadena = new StringBuilder(cadena);
    PilaADT<Character> aux=new PilaA<>();
    int contIz=0, contDer=0;
    ArrayList<Character> op=new ArrayList<>(Arrays.asList('+', '-', '*', '/','^','~'));
```

```
        for (int i = 0; i < cadena.length(); i++){

            switch (cadena.charAt(i)){
                case '+':
                    if(i==0 || i==cadena.length()-1 || op.contains(cadena.charAt(i+1))||
op.contains(cadena.charAt(i-1))|| cadena.charAt(i+1)==')')
                        error="ERROR EN SUMA ["+i+"]";
                    break;
                case '-':
                    if(i==cadena.length()-1 || (i!=0 && op.contains(cadena.charAt(i+1)))|| (i!=0
&& op.contains(cadena.charAt(i-1)))|| cadena.charAt(i+1)==')')
                        error="ERROR EN SUBTRACCION ["+i+"]";
                    else if (i == 0|| cadena.charAt(i-1)=='('){
                        nuevaCadena.setCharAt(i, '~');
                        cadena = nuevaCadena.toString();
                    }
                    break;
                case '*':
                case '/':
                case '^':
                    if(i==0 || i==cadena.length()-1 || op.contains(cadena.charAt(i+1)) ||
op.contains(cadena.charAt(i-1)) || cadena.charAt(i-1)=='(' || cadena.charAt(i+1)==')')
                        error="ERROR EN MULTIPLICACION DIVISION O
EXPONENCIAL ["+i+"]";
                    break;

                case '(':
                    if(i==cadena.length()-1  || op.contains(cadena.charAt(i+1)) ||
cadena.charAt(i+1)==')' )
```

```
                error="ERROR EN ( ["+i+"]";
            else{

                res+=cadena.charAt(i);
                aux.push(cadena.charAt(i));
                contIz++;}

            break;

        case ')':
            if(i==0 || op.contains(cadena.charAt(i-1)) || cadena.charAt(i-1)=='(' ||
aux.isEmpty())//|| cadena.charAt(i+1)!='+'|| cadena.charAt(i+1)!='-'||
cadena.charAt(i+1)=='('
                error= "ERROR EN ) ["+i+"]";
            else{
                aux.pop();
                contDer++;
            }
            break;

        default:
            if(esUnDigito(cadena.charAt(i)) || cadena.charAt(i)==' ' ){
                int cont=0;
                while(i<cadena.length() && cont<2 && cadena.charAt(i)!='(' &&
cadena.charAt(i)!=')' && !op.contains(cadena.charAt(i))){
                    if(cadena.charAt(i)=='.'){
                        cont++;
                    }
                    i++;
```

```
                }
                if(cont==2 || cadena.charAt(i-1)=='.')
                    error= "ERROR CON PUNTOS ["+i+"]";
                else{
                    i--;
                }
            }else{
                error = "ERROR: Caracter no reconocido ["+i+"]";
            }
            break;
        }
    }

    if(contIz!=contDer)
        error= "ERROR: No hay misma cantidad de parentesis ";
    if(error.equals("")){
        return cadena;
    }else{
        return error;
    }
}
/**
 * Verifica si fue hubo error o no al revisar la sintaxis
 * @param cad cadena revisada o anuncio de error
 * @return
 * <ul>
 * <li>true: si es un error</li>
 * <li>false: si no es un error</li>
 * </ul>
```

```
     */
    public boolean esError(String cad){
        String prueba = "";
        if ( cad.length()<5){
            return false;
        }
        for ( int i = 0; i < 5; i++){
          prueba += cad.charAt(i);
        }
        return prueba.equals("ERROR");
    }
}
```

Interfaz gráfica

```
package com.mycompany.calculadora;

/**
 *
 * @author Daniel, Javi, Liliana, Alexander, Jose
 */
public class interfazGrafica extends javax.swing.JFrame {
    private Calculadora calc;

    /**
     * Creates new form interfazGrafica
     */
```

```
    public interfazGrafica() {
        initComponents();
        calc = new Calculadora();
    }

    /**
     * This method is called from within the constructor to initialize the form.
     * WARNING: Do NOT modify this code. The content of this method is always
     * regenerated by the Form Editor.
     */
    @SuppressWarnings("unchecked")
    // <editor-fold defaultstate="collapsed" desc="Generated Code">//GEN-
BEGIN:initComponents
    private void initComponents() {

        jInternalFrame1 = new javax.swing.JInternalFrame();
        jButton1 = new javax.swing.JButton();
        jButton13 = new javax.swing.JButton();
        jButton14 = new javax.swing.JButton();
        jButton17 = new javax.swing.JButton();
        jPanel1 = new javax.swing.JPanel();
        jLabel1 = new javax.swing.JLabel();
        input = new javax.swing.JTextField();
        igual = new javax.swing.JButton();
        jLabel2 = new javax.swing.JLabel();
        Output = new javax.swing.JTextField();
        uno = new javax.swing.JButton();
        tres = new javax.swing.JButton();
        dos = new javax.swing.JButton();
```

```
        cuatro = new javax.swing.JButton();
        cinco = new javax.swing.JButton();
        seis = new javax.swing.JButton();
        siete = new javax.swing.JButton();
        ocho = new javax.swing.JButton();
        nueve = new javax.swing.JButton();
        suma = new javax.swing.JButton();
        resta = new javax.swing.JButton();
        multiplicacion = new javax.swing.JButton();
        division = new javax.swing.JButton();
        exponente = new javax.swing.JButton();
        izParentesis = new javax.swing.JButton();
        deParentesis = new javax.swing.JButton();
        punto = new javax.swing.JButton();
        limpiar = new javax.swing.JButton();

        jInternalFrame1.setVisible(true);

        javax.swing.GroupLayout jInternalFrame1Layout = new
javax.swing.GroupLayout(jInternalFrame1.getContentPane());
        jInternalFrame1.getContentPane().setLayout(jInternalFrame1Layout);
        jInternalFrame1Layout.setHorizontalGroup(

jInternalFrame1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADI
NG)
            .addGap(0, 0, Short.MAX_VALUE)
        );
        jInternalFrame1Layout.setVerticalGroup(
```

```
jInternalFrame1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADI
NG)
            .addGap(0, 0, Short.MAX_VALUE)
        );

        jButton1.setText("jButton1");

        jButton13.setText("jButton3");

        jButton14.setText("jButton3");

        jButton17.setText("jButton3");

        setDefaultCloseOperation(javax.swing.WindowConstants.EXIT_ON_CLOSE);
        setSize(new java.awt.Dimension(450, 300));

        jLabel1.setFont(new java.awt.Font("Tahoma", 0, 18)); // NOI18N
        jLabel1.setText("Calculadora");

        input.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                inputActionPerformed(evt);
            }
        });

        igual.setText("=");
        igual.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
```

```
        igualActionPerformed(evt);
    }
});

jLabel2.setText("Introduzca la operación:");

Output.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt) {
        OutputActionPerformed(evt);
    }
});

uno.setText("1");
uno.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt) {
        unoActionPerformed(evt);
    }
});

tres.setText("3");
tres.setActionCommand("3");
tres.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt) {
        tresActionPerformed(evt);
    }
});

dos.setText("2");
dos.addActionListener(new java.awt.event.ActionListener() {
```

```
    public void actionPerformed(java.awt.event.ActionEvent evt) {
        dosActionPerformed(evt);
    }
});

cuatro.setText("4");
cuatro.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt) {
        cuatroActionPerformed(evt);
    }
});

cinco.setText("5");
cinco.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt) {
        cincoActionPerformed(evt);
    }
});

seis.setText("6");
seis.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt) {
        seisActionPerformed(evt);
    }
});

siete.setText("7");
siete.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt) {
```

```
        sieteActionPerformed(evt);
    }
});

ocho.setText("8");
ocho.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt) {
        ochoActionPerformed(evt);
    }
});

nueve.setText("9");
nueve.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt) {
        nueveActionPerformed(evt);
    }
});

suma.setText("+");
suma.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt) {
        sumaActionPerformed(evt);
    }
});

resta.setText("-");
resta.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt) {
        restaActionPerformed(evt);
```

```
    }
});

multiplicacion.setText("*");
multiplicacion.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt) {
        multiplicacionActionPerformed(evt);
    }
});

division.setText("/");
division.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt) {
        divisionActionPerformed(evt);
    }
});

exponente.setText("^");
exponente.setActionCommand("^");
exponente.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt) {
        exponenteActionPerformed(evt);
    }
});

izParentesis.setText("(");
izParentesis.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt) {
        izParentesisActionPerformed(evt);
```

```
        }
    });

    deParentesis.setText(")");
    deParentesis.addActionListener(new java.awt.event.ActionListener() {
        public void actionPerformed(java.awt.event.ActionEvent evt) {
            deParentesisActionPerformed(evt);
        }
    });

    punto.setText(".");
    punto.addActionListener(new java.awt.event.ActionListener() {
        public void actionPerformed(java.awt.event.ActionEvent evt) {
            puntoActionPerformed(evt);
        }
    });

    limpiar.setText("Limpiar");
    limpiar.addActionListener(new java.awt.event.ActionListener() {
        public void actionPerformed(java.awt.event.ActionEvent evt) {
            limpiarActionPerformed(evt);
        }
    });

    javax.swing.GroupLayout jPanel1Layout = new javax.swing.GroupLayout(jPanel1);
    jPanel1.setLayout(jPanel1Layout);
    jPanel1Layout.setHorizontalGroup(

jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.TRAILING)
```

```
        .addGroup(jPanel1Layout.createSequentialGroup()

.addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEA
DING)
                .addGroup(jPanel1Layout.createSequentialGroup()
                    .addGap(11, 11, 11)
                    .addComponent(jLabel2)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)
                    .addComponent(jLabel1, javax.swing.GroupLayout.PREFERRED_SIZE,
103, javax.swing.GroupLayout.PREFERRED_SIZE))
                .addGroup(javax.swing.GroupLayout.Alignment.TRAILING,
jPanel1Layout.createSequentialGroup()
                    .addContainerGap(javax.swing.GroupLayout.DEFAULT_SIZE,
Short.MAX_VALUE)

.addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEA
DING)
                        .addComponent(Output,
javax.swing.GroupLayout.Alignment.TRAILING,
javax.swing.GroupLayout.PREFERRED_SIZE, 367,
javax.swing.GroupLayout.PREFERRED_SIZE)
                        .addComponent(input, javax.swing.GroupLayout.Alignment.TRAILING,
javax.swing.GroupLayout.PREFERRED_SIZE, 367,
javax.swing.GroupLayout.PREFERRED_SIZE)))
                .addGroup(jPanel1Layout.createSequentialGroup()
                    .addGap(15, 15, 15)
```

```
.addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.TRA
ILING)
                    .addComponent(izParentesis,
javax.swing.GroupLayout.PREFERRED_SIZE, 41,
javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addComponent(deParentesis,
javax.swing.GroupLayout.PREFERRED_SIZE, 41,
javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addComponent(punto, javax.swing.GroupLayout.PREFERRED_SIZE,
41, javax.swing.GroupLayout.PREFERRED_SIZE))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEA
DING)
                    .addGroup(jPanel1Layout.createSequentialGroup()
                        .addComponent(siete)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)
                        .addComponent(ocho)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                        .addComponent(nueve))
                    .addGroup(jPanel1Layout.createSequentialGroup()
                        .addComponent(cuatro)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)
                        .addComponent(cinco)
```

```
.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                        .addComponent(seis)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                        .addComponent(multiplicacion,
javax.swing.GroupLayout.PREFERRED_SIZE, 41,
javax.swing.GroupLayout.PREFERRED_SIZE))
                    .addGroup(jPanel1Layout.createSequentialGroup()
                        .addComponent(uno)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)
                        .addComponent(dos)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                        .addComponent(tres)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                        .addComponent(suma,
javax.swing.GroupLayout.PREFERRED_SIZE, 41,
javax.swing.GroupLayout.PREFERRED_SIZE)))
                .addGap(4, 4, 4)

.addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEA
DING)
                    .addGroup(jPanel1Layout.createSequentialGroup()
                        .addComponent(exponente,
javax.swing.GroupLayout.PREFERRED_SIZE, 41,
javax.swing.GroupLayout.PREFERRED_SIZE)
```

```
.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
                        .addComponent(igual))
                    .addGroup(jPanel1Layout.createSequentialGroup()
                        .addComponent(resta, javax.swing.GroupLayout.PREFERRED_SIZE,
41, javax.swing.GroupLayout.PREFERRED_SIZE)
                        .addGap(0, 0, Short.MAX_VALUE))
                    .addGroup(jPanel1Layout.createSequentialGroup()
                        .addComponent(division,
javax.swing.GroupLayout.PREFERRED_SIZE, 41,
javax.swing.GroupLayout.PREFERRED_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
                        .addComponent(limpiar)))))
            .addContainerGap())
     );
      jPanel1Layout.setVerticalGroup(

jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
          .addGroup(jPanel1Layout.createSequentialGroup()

.addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.TRA
ILING)
                .addComponent(jLabel1, javax.swing.GroupLayout.PREFERRED_SIZE, 56,
javax.swing.GroupLayout.PREFERRED_SIZE)
                .addComponent(jLabel2))
            .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
```

```
            .addComponent(input, javax.swing.GroupLayout.PREFERRED_SIZE, 40,
javax.swing.GroupLayout.PREFERRED_SIZE)
            .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BAS
ELINE)
                .addComponent(uno)
                .addComponent(tres)
                .addComponent(dos)
                .addComponent(suma)
                .addComponent(resta)
                .addComponent(izParentesis))
            .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BAS
ELINE)
                .addComponent(cuatro)
                .addComponent(cinco)
                .addComponent(seis)
                .addComponent(multiplicacion)
                .addComponent(division)
                .addComponent(deParentesis)
                .addComponent(limpiar))
            .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BAS
ELINE)
                .addComponent(igual)
                .addComponent(siete)
```

```
                .addComponent(ocho)
                .addComponent(nueve)
                .addComponent(exponente)
                .addComponent(punto))
            .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
            .addComponent(Output, javax.swing.GroupLayout.PREFERRED_SIZE, 32,
javax.swing.GroupLayout.PREFERRED_SIZE)
            .addContainerGap(javax.swing.GroupLayout.DEFAULT_SIZE,
Short.MAX_VALUE))
        );

        javax.swing.GroupLayout layout = new
javax.swing.GroupLayout(getContentPane());
        getContentPane().setLayout(layout);
        layout.setHorizontalGroup(
            layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(layout.createSequentialGroup()
                .addContainerGap()
                .addComponent(jPanel1, javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                .addContainerGap(38, Short.MAX_VALUE))
        );
        layout.setVerticalGroup(
            layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(layout.createSequentialGroup()
                .addContainerGap()
```

```
            .addComponent(jPanel1, javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
            .addContainerGap(40, Short.MAX_VALUE))
        );

        pack();
    }// </editor-fold>//GEN-END:initComponents

    private void igualActionPerformed(java.awt.event.ActionEvent evt) {//GEN-
FIRST:event_igualActionPerformed
        String ver = calc.revisaSintaxis(input.getText());
        if (!calc.esError(ver)){
        String res = calc.ConvertirPostfijo(ver);
        System.out.println(res);
        double bob = calc.resuelveExpresion(res);
        Output.setText(""+ bob);
        } else {
         Output.setText(""+ ver);
        }
    }//GEN-LAST:event_igualActionPerformed

    private void inputActionPerformed(java.awt.event.ActionEvent evt) {//GEN-
FIRST:event_inputActionPerformed
        // TODO add your handling code here:
    }//GEN-LAST:event_inputActionPerformed

    private void OutputActionPerformed(java.awt.event.ActionEvent evt) {//GEN-
FIRST:event_OutputActionPerformed
```

```
        // TODO add your handling code here:
    }//GEN-LAST:event_OutputActionPerformed

    private void izParentesisActionPerformed(java.awt.event.ActionEvent evt) {//GEN-
FIRST:event_izParentesisActionPerformed
        String a = input.getText();
        a += '(';
        input.setText(a);
    }//GEN-LAST:event_izParentesisActionPerformed

    private void unoActionPerformed(java.awt.event.ActionEvent evt) {//GEN-
FIRST:event_unoActionPerformed
        String a = input.getText();
        a += '1';
        input.setText(a);
    }//GEN-LAST:event_unoActionPerformed

    private void dosActionPerformed(java.awt.event.ActionEvent evt) {//GEN-
FIRST:event_dosActionPerformed
        String a = input.getText();
        a += '2';
        input.setText(a);
    }//GEN-LAST:event_dosActionPerformed

    private void tresActionPerformed(java.awt.event.ActionEvent evt) {//GEN-
FIRST:event_tresActionPerformed
        String a = input.getText();
        a += '3';
        input.setText(a);
```

```
    }//GEN-LAST:event_tresActionPerformed

    private void sumaActionPerformed(java.awt.event.ActionEvent evt) {//GEN-
FIRST:event_sumaActionPerformed
        String a = input.getText();
        a += '+';
        input.setText(a);
    }//GEN-LAST:event_sumaActionPerformed

    private void restaActionPerformed(java.awt.event.ActionEvent evt) {//GEN-
FIRST:event_restaActionPerformed
        String a = input.getText();
        a += '-';
        input.setText(a);
    }//GEN-LAST:event_restaActionPerformed

    private void deParentesisActionPerformed(java.awt.event.ActionEvent evt) {//GEN-
FIRST:event_deParentesisActionPerformed
        String a = input.getText();
        a += ')';
        input.setText(a);
    }//GEN-LAST:event_deParentesisActionPerformed

    private void cuatroActionPerformed(java.awt.event.ActionEvent evt) {//GEN-
FIRST:event_cuatroActionPerformed
        String a = input.getText();
        a += '4';
        input.setText(a);
    }//GEN-LAST:event_cuatroActionPerformed
```

```
    private void cincoActionPerformed(java.awt.event.ActionEvent evt) {//GEN-
FIRST:event_cincoActionPerformed
        String a = input.getText();
        a += '5';
        input.setText(a);
    }//GEN-LAST:event_cincoActionPerformed

    private void seisActionPerformed(java.awt.event.ActionEvent evt) {//GEN-
FIRST:event_seisActionPerformed
        String a = input.getText();
        a += '6';
        input.setText(a);
    }//GEN-LAST:event_seisActionPerformed

    private void multiplicacionActionPerformed(java.awt.event.ActionEvent evt) {//GEN-
FIRST:event_multiplicacionActionPerformed
        String a = input.getText();
        a += '*';
        input.setText(a);
    }//GEN-LAST:event_multiplicacionActionPerformed

    private void divisionActionPerformed(java.awt.event.ActionEvent evt) {//GEN-
FIRST:event_divisionActionPerformed
        String a = input.getText();
        a += '/';
        input.setText(a);
    }//GEN-LAST:event_divisionActionPerformed
```

```
    private void limpiarActionPerformed(java.awt.event.ActionEvent evt) {//GEN-
FIRST:event_limpiarActionPerformed
        input.setText("");
    }//GEN-LAST:event_limpiarActionPerformed

    private void puntoActionPerformed(java.awt.event.ActionEvent evt) {//GEN-
FIRST:event_puntoActionPerformed
        String a = input.getText();
        a += '.';
        input.setText(a);
    }//GEN-LAST:event_puntoActionPerformed

    private void sieteActionPerformed(java.awt.event.ActionEvent evt) {//GEN-
FIRST:event_sieteActionPerformed
        String a = input.getText();
        a += '7';
        input.setText(a);
    }//GEN-LAST:event_sieteActionPerformed

    private void ochoActionPerformed(java.awt.event.ActionEvent evt) {//GEN-
FIRST:event_ochoActionPerformed
        String a = input.getText();
        a += '8';
        input.setText(a);
    }//GEN-LAST:event_ochoActionPerformed

    private void nueveActionPerformed(java.awt.event.ActionEvent evt) {//GEN-
FIRST:event_nueveActionPerformed
        String a = input.getText();
```

```
        a += '9';
        input.setText(a);
    }//GEN-LAST:event_nueveActionPerformed

    private void exponenteActionPerformed(java.awt.event.ActionEvent evt) {//GEN-
FIRST:event_exponenteActionPerformed
        String a = input.getText();
        a += '^';
        input.setText(a);
    }//GEN-LAST:event_exponenteActionPerformed

    /**
     * @param args the command line arguments
     */
    public static void main(String args[]) {
        /* Set the Nimbus look and feel */
        //<editor-fold defaultstate="collapsed" desc=" Look and feel setting code (optional)
">
        /* If Nimbus (introduced in Java SE 6) is not available, stay with the default look and
feel.
         * For details see
http://download.oracle.com/javase/tutorial/uiswing/lookandfeel/plaf.html
         */
        try {
            for (javax.swing.UIManager.LookAndFeelInfo info :
javax.swing.UIManager.getInstalledLookAndFeels()) {
                if ("Nimbus".equals(info.getName())) {
                    javax.swing.UIManager.setLookAndFeel(info.getClassName());
                    break;
```

```
            }
        }
    } catch (ClassNotFoundException ex) {

java.util.logging.Logger.getLogger(interfazGrafica.class.getName()).log(java.util.logging.
Level.SEVERE, null, ex);
    } catch (InstantiationException ex) {

java.util.logging.Logger.getLogger(interfazGrafica.class.getName()).log(java.util.logging.
Level.SEVERE, null, ex);
    } catch (IllegalAccessException ex) {

java.util.logging.Logger.getLogger(interfazGrafica.class.getName()).log(java.util.logging.
Level.SEVERE, null, ex);
    } catch (javax.swing.UnsupportedLookAndFeelException ex) {

java.util.logging.Logger.getLogger(interfazGrafica.class.getName()).log(java.util.logging.
Level.SEVERE, null, ex);
    }
    //</editor-fold>

    /* Create and display the form */
    java.awt.EventQueue.invokeLater(new Runnable() {
        public void run() {
            new interfazGrafica().setVisible(true);
        }
    });
}
```

```
// Variables declaration - do not modify//GEN-BEGIN:variables
private javax.swing.JTextField Output;
private javax.swing.JButton cinco;
private javax.swing.JButton cuatro;
private javax.swing.JButton deParentesis;
private javax.swing.JButton division;
private javax.swing.JButton dos;
private javax.swing.JButton exponente;
private javax.swing.JButton igual;
private javax.swing.JTextField input;
private javax.swing.JButton izParentesis;
private javax.swing.JButton jButton1;
private javax.swing.JButton jButton13;
private javax.swing.JButton jButton14;
private javax.swing.JButton jButton17;
private javax.swing.JInternalFrame jInternalFrame1;
private javax.swing.JLabel jLabel1;
private javax.swing.JLabel jLabel2;
private javax.swing.JPanel jPanel1;
private javax.swing.JButton limpiar;
private javax.swing.JButton multiplicacion;
private javax.swing.JButton nueve;
private javax.swing.JButton ocho;
private javax.swing.JButton punto;
private javax.swing.JButton resta;
private javax.swing.JButton seis;
private javax.swing.JButton siete;
private javax.swing.JButton suma;
private javax.swing.JButton tres;
```

```
    private javax.swing.JButton uno;
    // End of variables declaration//GEN-END:variables
}
```