


Profesora Beltrán Martínez Beatriz

PROGRAMACION II

TERCER EXAMEN PARCIAL




Corona Jiménez Martín
Huertero Rivera Oswaldo
Castelán Carpinteyro Dante



ESPECIFICACIÓN DEL PROBLEMA

Realizar el diagrama de clases y programa que resuelva una calculadora de fracciones:

- Generar los constructores necesarios, para la creación de una fracción.
 - Las variables de instancia serán denominador (d) y numerador (n).
 - Se deben tener las operaciones de suma, resta, multiplicación y división de fracciones.
 - Para la suma y la resta se debe calcular el MCM previo a realizar la suma, teniendo un método recursivo que lo obtenga (el MCM).
 - Se debe tener la opción de leer los datos de entrada desde un archivo y guardar el resultado en un archivo, así como su recuperación.
 - Se debe considerar el manejo de excepciones propias de Java y al menos una del programador para los posibles errores.
 - Utilice ambiente gráfico.
- 



EXPLICACIÓN DE LA SOLUCIÓN

COLORS.JAVA:

public class Colors

- Es una clase de utilidad que define la paleta de colores del programa.
- Usa constantes estáticas (static final) para mantener consistencia en los colores.
- Define colores personalizados usando valores RGB.
- Facilita el mantenimiento al centralizar la gestión de colores.

FILELECTOR.JAVA:

public class FileLector extends javax.swing.JFrame

Extiende JFrame para crear una ventana de diálogo. Permite al usuario ingresar la ruta de un archivo.


Características principales:

- Ventana sin decoración (setUndecorated(true)).
- Interfaz minimalista con campo de texto y etiqueta.
- Manejo de eventos de teclado para capturar la ruta.
- Diseño personalizado con bordes redondeados.

FRACTIONPANEL.JAVA:

public class FractionPanel extends javax.swing.JPanel

Panel principal para la visualización y entrada de fracciones.





Componentes:

- Campos de texto personalizados para numeradores y denominadores.
- Símbolos de operación e igual.
- Barras divisorias para las fracciones.

Características:

- Layout usando GridBagLayout para organización precisa.
- Validación de entrada numérica.
- Diseño responsivo basado en el tamaño de la ventana.

ROUNDPANE.JAVA:

public class RoundPane extends javax.swing.JPanel

Panel con esquinas redondeadas.

Características:

- Dibujo personalizado usando Graphics2D.
- Efecto de borde con dos capas.
- Antialiasing para mejor calidad visual.

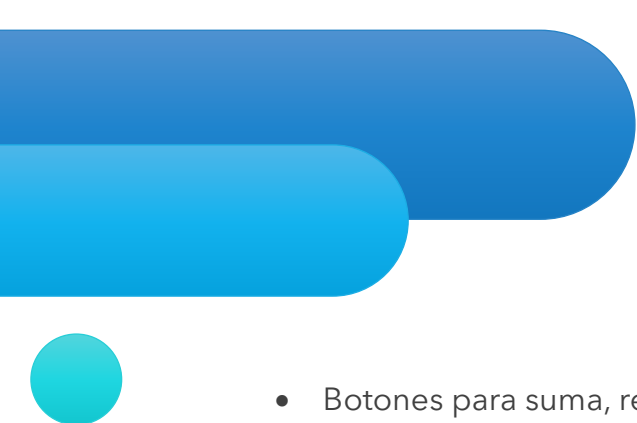
BUTTONPANEL.JAVA:

public class ButtonPanel extends javax.swing.JPanel

Panel de botones para operaciones.

Componentes:



- 
- Botones para suma, resta, multiplicación, división.
 - Botón para manejo de archivos.

Características:

- Diseño grid con espaciado uniforme.
- Botones personalizados con efectos hover.
- Manejo de eventos para cada operación.

TITLEBAR.JAVA:


public class TitleBar extends javax.swing.JPanel

Panel superior que actúa como barra de título personalizada.

Componentes:

- Etiqueta (title): Muestra el nombre de la aplicación ("Calculadora de fracciones").
- Botón (close): Permite cerrar la aplicación.

Características:

- Diseño personalizado con esquinas redondeadas y colores definidos.
 - Barra de título interactiva que permite mover la ventana arrastrándola.
 - Botón de cierre con un diseño circular:
 - Cambia de color al pasar el mouse.
 - Cierra la aplicación al hacer clic.
 - Etiqueta centrada con texto estilizado y colores de contraste.
 - Soporte para layouts responsivos con GridBagLayout.
 - Dibujo de fondo con antialiasing para una apariencia moderna.
- 

VENTANA.JAVA

public class Ventana extends javax.swing.JFrame

Clase principal que define la ventana donde se ejecuta el programa.

Componentes:

- Barra de título (titleBar):
 - Panel superior que muestra el nombre de la aplicación y un botón de cierre.
 - Permite mover la ventana mediante arrastre.
- Panel de fracciones (fractionPanel):
 - Espacio donde se ingresan y muestran las fracciones para realizar operaciones.
- Panel de botones (buttonPanel):
 - Contiene botones para realizar las operaciones (suma, resta, multiplicación, división) y manejo de archivos.

Métodos:

- getWidth() y getHeight():
 - Devuelven el ancho y alto disponible para los componentes internos.
- getFractionPanel():
 - Proporciona acceso al panel de fracciones para interactuar con sus datos desde otras partes del programa.

FRACTION.JAVA

public class Fraction

Propiedades (numerator y denominator):

- Ambas son públicas, lo cual no es la mejor práctica de encapsulación. Deberían ser privadas y manejarse exclusivamente a través de los métodos get y set.

Constructor:

- Recibe valores iniciales para el numerador y denominador.
- No valida que el denominador sea distinto de cero (lo que podría causar errores en cálculos).

Métodos principales:

- simplify:
 - Simplifica la fracción utilizando el máximo común divisor (GCD).
 - Crea una nueva instancia de Fraction con los valores simplificados.
- gcd:
 - Implementa el algoritmo de Euclides para calcular el GCD, lo cual es eficiente.
- toString:
 - Devuelve una representación de la fracción en formato de cadena.
 - Admite opciones para incluir un signo negativo o paréntesis.
- valid:
 - Verifica si el denominador es cero.
 - La lógica parece invertida; debería devolver false si el denominador es cero, ya que una fracción con denominador cero no es válida.
- checkZero:
 - Verifica si el numerador de una fracción es cero.

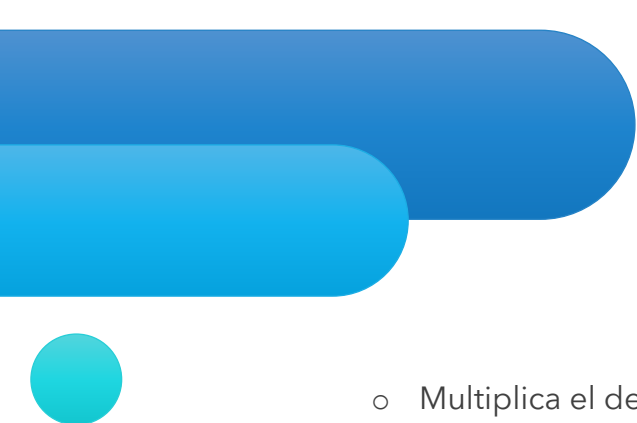
FRACTIONS.JAVA

public class Fractions

La clase Fractions es un conjunto de métodos estáticos para realizar operaciones matemáticas entre dos instancias de Fraction.


Implementa correctamente las siguientes operaciones:

- Suma (suma):
 - Calcula el denominador común multiplicando los denominadores de las dos fracciones.
 - Ajusta los numeradores multiplicando el numerador de cada fracción por el denominador de la otra.
 - Suma los numeradores ajustados para obtener el numerador del resultado.
 - Simplifica el resultado antes de retornarlo.
- Resta (resta):
 - Similar a la suma, pero resta los numeradores ajustados en lugar de sumarlos.
 - Simplifica el resultado antes de retornarlo.
- Multiplicación (multiplicacion):
 - Multiplica los numeradores de ambas fracciones para obtener el numerador del resultado.
 - Multiplica los denominadores de ambas fracciones para obtener el denominador del resultado.
 - Simplifica el resultado antes de retornarlo.
- División (division):
 - Invierte la segunda fracción (denominador pasa a ser numerador y viceversa).
 - Multiplica el numerador de la primera fracción por el denominador de la segunda fracción.

- 
- Multiplica el denominador de la primera fracción por el numerador de la segunda fracción.

CARACTERÍSTICAS CLAVE

Reutilización de la clase Fraction:

- Cada operación retorna una nueva instancia de la clase Fraction como resultado.
 - Se utiliza el método `simplify()` para garantizar que la fracción resultante esté en su forma más simplificada.
- 



CLASES INTERNAS IMPORTANTES

CustomText (en FractionPanel.java):

Campo de texto personalizado para fracciones.

Características:

- Validación de entrada numérica.
- Límite de 4 dígitos.
- Diseño visual personalizado.
- Manejo de eventos de teclado.

Bar (en FractionPanel.java):

- Componente visual para la línea divisoria de fracciones.

Características:

- Dibujo personalizado con bordes redondeados.
- Tamaño adaptativo.

CustomButton (en ButtonPanel.java):

- Botón personalizado con efectos visuales.

Características:



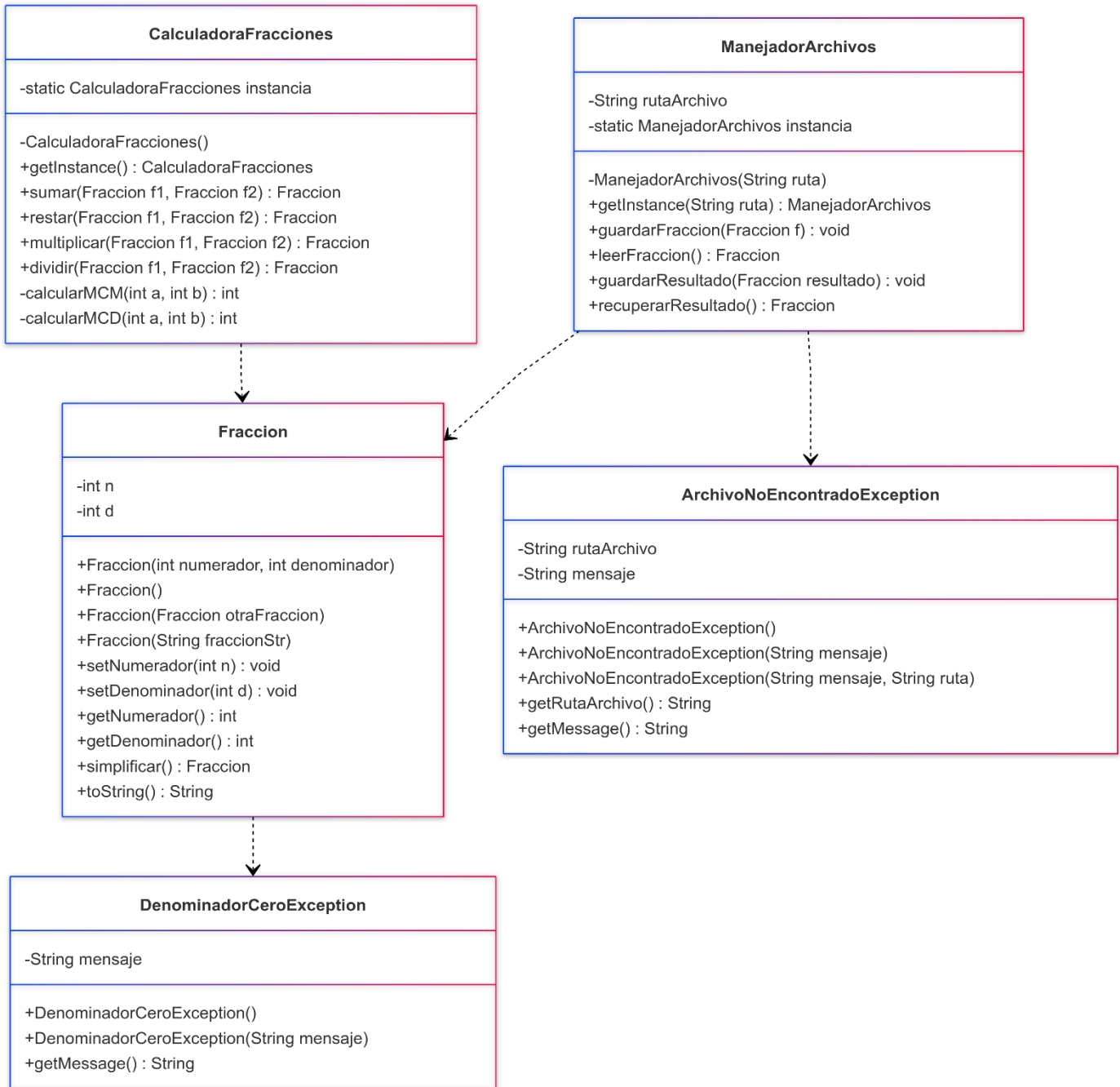
- Efecto hover.
 - Cursor personalizado.
 - Diseño visual personalizado.
- 
- 

DIAGRAMA DE CLASES



CÓDIGO FUENTE

FraccionReader.java

```
package logic;

import java.io.*;
import java.nio.file.*;

public class FraccionReader {

    public static Fraction resultado;

    // Método estático para leer y procesar el archivo
    public static void procesarArchivo(String rutaArchivo) throws
IOException {

        // Lee el archivo
        String contenido = new
String(Files.readAllBytes(Paths.get(rutaArchivo)));

        // Divide en líneas si hay múltiples operaciones
        String[] lineas = contenido.split("\\n");

        // Procesa cada línea
        for (String linea : lineas) {

            String[] partes = linea.split(" ");
            Fraction fraccion1 = parsearFraccion(partes[0]);
            Fraction fraccion2 = parsearFraccion(partes[2]);
            String operador = partes[1];

            // Operar las fracciones según el operador
            switch (operador) {
                case "+":
                    resultado = Fractions.suma(fraccion1, fraccion2);
```

```

        break;
    case "-":
        resultado = Fractions.resta(fraccion1, fraccion2);
        break;
    case "*":
        resultado = Fractions.multiplicacion(fraccion1,
fraccion2);
        break;
    case "/":
        resultado = Fractions.division(fraccion1, fraccion2);
        break;
    default:
        throw new IllegalArgumentException("Operador no
soportado: " + operador);
    }
    resultado.simplify();
}
guardarResultado();

}

// Convierte una cadena en Fraction
private static Fraction parsearFraccion(String texto) {
    String[] partes = texto.split("/");
    int numerador = Integer.parseInt(partes[0]);
    int denominador = Integer.parseInt(partes[1]);
    return new Fraction(numerador, denominador);
}

// Guarda el resultado en un archivo
private static void guardarResultado() throws IOException {

    String carpetaOutput = "src/output";

    // Crear la carpeta si no existe
    Files.createDirectories(Paths.get(carpetaOutput));

```

```
        // Crear y/o sobre-escribe el archivo
        String rutaResultado = carpetaOutput + "/resultado.txt";
        String textoResultado = resultado.getNumerator() + "/" +
resultado.getDenominator();
        Files.write(Paths.get(rutaResultado), textoResultado.getBytes());
    }
}
```

Fraction.java

```
package logic;

public class Fraction {
    public Integer numerator;
    public Integer denominator;

    public Fraction(int numerator, int denominator) {
        this.numerator = numerator;
        this.denominator = denominator;
    }

    public Integer getNumerator() {
        return this.numerator;
    }

    public Integer getDenominator() {
        return this.denominator;
    }

    public void setNumerator(int numerator) {
        this.numerator = numerator;
    }

    public void setDenominator(int denominator) {
        this.denominator = denominator;
    }
}
```

```

    }

    public Fraction simplify() throws ArithmeticException{ // Simplificar
    fracción con los datos que tenga actualmente

        Fraction fraction = new Fraction(0, 0);
        try {
            fraction = new Fraction(this.numerator, this.denominator);
            int gcd = this.gcd(fraction.getNumerator(),
fraction.getDenominator());
            fraction.setNumerator(fraction.getNumerator() / gcd);
            fraction.setDenominator(fraction.getDenominator() / gcd);
            return fraction;
        } catch (ArithmeticException e) {
            return fraction;
        }

    }

    private int gcd(int a, int b) {
        if (b == 0) {
            return a;
        }
        return gcd(b, a % b);
    }

    public String toString(boolean sign, boolean parenthesis) {
        String fraction = this.numerator + " / " + this.denominator;
        if(sign){
            fraction = "-" + fraction;
        }
        if (parenthesis) {
            fraction = "(" + fraction + ")";
        }
        return fraction;
    }

```

```

    }

    public boolean valid() {
        if (this.denominator == 0) {
            return true;
        }
        return false;
    }

    public boolean checkZero(Fraction fraction) {
        if (fraction.numerator == 0) {
            return true;
        }
        return false;
    }
}

```

Fractions.java

```

package logic;

public class Fractions {

    public static Fraction suma(Fraction fractionA, Fraction fractionB) {
        Fraction result = new Fraction(0, 0);
        result.numerator = fractionA.numerator * fractionB.denominator +
fractionB.numerator * fractionA.denominator;
        result.denominator = fractionA.denominator * fractionB.denominator;
        return result.simplify();
    }

    public static Fraction resta(Fraction fractionA, Fraction fractionB) {
        Fraction result = new Fraction(0, 0);
        result.numerator = fractionA.numerator * fractionB.denominator -
fractionB.numerator * fractionA.denominator;
        result.denominator = fractionA.denominator * fractionB.denominator;
    }
}

```



```

        return result.simplify();
    }

    public static Fraction multiplicacion(Fraction fractionA, Fraction
fractionB) {
        Fraction result = new Fraction(0, 0);
        result.numerator = fractionA.numerator * fractionB.numerator;
        result.denominator = fractionA.denominator * fractionB.denominator;
        return result.simplify();
    }

    public static Fraction division(Fraction fractionA, Fraction fractionB)
{
        Fraction result = new Fraction(0, 0);
        result.numerator = fractionA.numerator * fractionB.denominator;
        result.denominator = fractionA.denominator * fractionB.numerator;
        return result.simplify();
    }
}

```

ButtonPanel.java

```

package ui;

import javax.swing.*.*;
import java.awt.*.*;
import java.awt.Window.*;
import java.awt.event.*;
import java.awt.geom.RoundRectangle2D;

public class ButtonPanel extends javax.swing.JPanel {

    private Ventana ventana;
    private CustomButton plus, minus, mult, div, file;
}

```

```
public ButtonPanel(Ventana ventana) {

    // Llenado de variables
    this.ventana = ventana;

    // Configuración de panel
    int borderSize = ventana.getWidth() / 16;
    setPreferredSize(new Dimension(ventana.getUsableWidth(),
(ventana.getUsableWidth() - borderSize * 6) / 5 + borderSize));
    setMinimumSize(getPreferredSize());
    setMaximumSize(getPreferredSize());
    setBorder(BorderFactory.createEmptyBorder(0, borderSize, borderSize,
borderSize));
    setLayout(new GridLayout(1, 5, borderSize, borderSize));
    setBackground(ventana.getBackground());

    // Configuración de componentes
    plus = new CustomButton(ventana, "+") {
        {
            addActionListener(new ActionListener() {

                @Override
                public void actionPerformed(ActionEvent e) {

                    ventana.getFractionPanel().getSymbol().setText("+");
                    ventana.repaint();

                }

            });
        }
    };
    minus = new CustomButton(ventana, "-") {
        {
            addActionListener(new ActionListener() {
```

```
        @Override
        public void actionPerformed(ActionEvent e) {

            ventana.getFractionPanel().getSymbol().setText("-");
            ventana.repaint();

        }

    });
}

};
mult = new CustomButton(ventana, "x") {
    {
        addActionListener(new ActionListener() {

            @Override
            public void actionPerformed(ActionEvent e) {

                ventana.getFractionPanel().getSymbol().setText("x");
                ventana.repaint();

            }

        });
    }
};
div = new CustomButton(ventana, "÷") {
    {
        addActionListener(new ActionListener() {

            @Override
            public void actionPerformed(ActionEvent e) {

                ventana.getFractionPanel().getSymbol().setText("÷");
                ventana.repaint();

            }

        });
    }
};
```

```

        }

        });
    }
};

file = new CustomButton(ventana, "F") {
    {
        addActionListener(new ActionListener() {

            @Override
            public void actionPerformed(ActionEvent e) {

                FileLector lector = new FileLector(ventana);
                lector.setVisible(true);

            }

        });
    }
};

// Adicion de componentes
add(plus);
add(minus);
add(mult);
add(div);
add(file);

}
}

// Clase CustomButton

class CustomButton extends javax.swing.JButton {

    private boolean isMouseIn = false;

```

```
private Cursor normalCursor = new Cursor(Cursor.DEFAULT_CURSOR);
private Cursor handCursor = new Cursor(Cursor.HAND_CURSOR);
private Color defaultColor = Colors.BLACK;

public CustomButton(Ventana ventana, String title) {

    super(title);

    // Configuraciones
    setBackground(defaultColor);
    setFocusPainted(false);
    setBorderPainted(false);
    setContentAreaFilled(false);
    setFont(new Font("Arial Nova", Font.BOLD, ventana.getWidth() / 16));
    setForeground(Colors.WHITE);

    // Listeners
    addMouseListener(new MouseAdapter() {
        @Override
        public void mouseEntered(MouseEvent e) {
            isMouseIn = true;
            ventana.setCursor(handCursor);
        }

        @Override
        public void mouseExited(MouseEvent e) {
            isMouseIn = false;
            ventana.setCursor(normalCursor);
        }
    });
}

@Override
public void paintComponent(Graphics g) {

    Graphics2D g2 = (Graphics2D) g;
```

```
        g2.setRenderingHint(RenderingHints.KEY_ANTIALIASING,
RenderingHints.VALUE_ANTIALIAS_ON);

        RoundedRectangle2D rounded = new RoundedRectangle2D.Double(0, 0,
getWidth(), getHeight(), 20, 20);

        if (isMouseIn) {
            g2.setColor(Colors.GRAY);
        } else {
            g2.setColor(getBackground());
        }
        g2.fill(rounded);

        super.paintComponent(g);
    }
}
```

Colors.java

```
package ui;

import java.awt.Color;

/**
 * Clase que almacena la paleta de colores usada en el programa
 */

public class Colors {

    public static final Color BLACK = new Color(30, 30, 30);
    public static final Color DARK_GRAY = new Color(44, 44, 44);
    public static final Color GRAY = new Color(67, 67, 67);
    public static final Color WHITE = new Color(245, 245, 245);
    public static final Color RED = new Color(255, 45, 85);
    public static final Color BLUE = new Color(48, 176, 199);
}
```

```
}
```

FileLector.java

```
package ui;

import javax.swing.*.*;

import logic.FraccionReader;

import java.awt.*.*;
import java.awt.event.*.*;
import java.awt.event.*.*;
import java.awt.geom.*;
import java.io.IOException;

public class FileLector extends javax.swing.JFrame {

    private Ventana ventana;
    private JLabel text;
    private JTextField location;

    private String rute = "";

    public FileLector(Ventana ventana) {

        super();
        this.ventana = ventana;
        // Tamaño de pantalla
        Dimension size = new Dimension(ventana.getWidth() / 3 * 2,
ventana.getWidth() / 7);

        // Configuracion de ventana
        setSize(size);
    }
}
```

```
setLocationRelativeTo(ventana);
setUndecorated(true);
setType(Type.UTILITY);
setContentPane(new FilePane(this));
setBackground(new Color(0, 0, 0, 0));

// Configuración de componentes
text = new JLabel("Ingrese la ruta del archivo:") {
    {
        setFont(new Font("Arial Nova", Font.BOLD, ventana.getWidth()
/ 32));
        setForeground(Colors.WHITE);
    }
};

location = new JTextField() {
    {
        setFont(new Font("Arial Nova", Font.BOLD, ventana.getWidth()
/ 32));
        setEditable(true);
        setSelectedTextColor(Colors.WHITE);
        setSelectionColor(Colors.GRAY);
        setBackground(Colors.DARK_GRAY);
        setForeground(Colors.WHITE);
        setBorder(BorderFactory.createEmptyBorder(0, 10, 0, 10));
        setCaretColor(getForeground());

        addKeyListener(new KeyAdapter() {

            @Override
            public void keyTyped(KeyEvent e) {

                if (e.getKeyChar() == '\n') {
                    rute = getText();
                    dispose();
                }
            }
        });
    }
};
```



```

        try {
            FraccionReader.procesarArchivo(rute);
            ventana.getFractionPanel().getN3().setText("
" + FraccionReader.resultado.getNumerator());
            ventana.getFractionPanel().getD3().setText("
" + FraccionReader.resultado.getDenominator());
        } catch (IOException e1) {
            System.err.println("Archivo no encontrado");
        }
    }
}

});
}

};

// Adicion de los componentes
add(text);
add(location);

}

}

class FilePane extends javax.swing.JPanel {

    private FileLector fileLector;

    public FilePane(FileLector fileLector) {

        super();
        this.fileLector = fileLector;
        setLayout(new BorderLayout(this, BorderLayout.Y_AXIS));
        setBorder(BorderFactory.createEmptyBorder(10, 10, 10, 10));

    }
}

```

```
@Override
public void paintComponent(Graphics g) {

    super.paintComponent(g);
    Graphics2D g2 = (Graphics2D) g;
    g2.setRenderingHint(RenderingHints.KEY_ANTIALIASING,
RenderingHints.VALUE_ANTIALIAS_ON);

    RoundRectangle2D border = new RoundRectangle2D.Double(0, 0,
getWidth(), getHeight(), 22, 22);
    g2.setColor(Colors.GRAY);
    g2.fill(border);

    RoundRectangle2D rectangle = new RoundRectangle2D.Double(1, 1,
getWidth() - 2, getHeight() - 2, 20, 20);
    g2.setColor(Colors.BLACK);
    g2.fill(rectangle);

}

}
```

FractionPanel.java

```
package ui;

import javax.swing.*;

import logic.Fraction;
import logic.Fractions;

import java.awt.*;
import java.awt.RenderingHints.Key;
import java.awt.event.*;
import java.awt.event.KeyAdapter;
```

```
import java.awt.geom.RoundRectangle2D;

public class FractionPanel extends javax.swing.JPanel {

    private Ventana ventana;
    private CustomText n1, d1, n2, d2, n3, d3;
    private JLabel symbol, equal;

    // Values
    private Fraction fraction1 = new Fraction(1, 2);
    private Fraction fraction2 = new Fraction(2, 4);
    private Fraction fraction3 = new Fraction(1, 1);

    public FractionPanel(Ventana ventana) {

        // Llenado de variables
        this.ventana = ventana;

        // Configuración del panel
        int borderSize = ventana.getWidth() / 16;
        setBorder(BorderFactory.createEmptyBorder(borderSize * 2, borderSize
* 5 / 2, borderSize * 2, borderSize * 5 / 2));
        setLayout(new GridBagLayout());
        setBackground(ventana.getBackground());

        // Configuración de componentes
        n1 = new CustomText(ventana, "1", true, 'N', 1);
        n2 = new CustomText(ventana, "2", true, 'N', 2);
        n3 = new CustomText(ventana, "1", false, 'N', 3);
        d1 = new CustomText(ventana, "2", true, 'D', 1);
        d2 = new CustomText(ventana, "4", true, 'D', 2);
        d3 = new CustomText(ventana, "1", false, 'D', 3);

        n1.setPair(d1);
        n2.setPair(d2);
        n3.setPair(d3);
    }
}
```

```
d1.setPair(n1);
d2.setPair(n2);
d3.setPair(n3);

symbol = new JLabel("+");
symbol.setFont(new Font("Arial Nova", Font.BOLD, ventana.getWidth()
/ 14));
symbol.setHorizontalAlignment(SwingConstants.CENTER);
symbol.setForeground(Colors.WHITE);
symbol.setPreferredSize(getSize());

equal = new JLabel("=");
equal.setFont(new Font("Arial Nova", Font.BOLD, ventana.getWidth() /
14));
equal.setHorizontalAlignment(SwingConstants.CENTER);
equal.setForeground(Colors.WHITE);

// Adicion de componentes
GridBagConstraints gbc = new GridBagConstraints();
gbc.fill = GridBagConstraints.BOTH;
gbc.weightx = 1;
gbc.weighty = 1;

Insets bottom = new Insets(0, 0, 15, 0);
Insets top = new Insets(15, 0, 0, 0);
Insets none = new Insets(0, 0, 0, 0);

gbc.gridx = 0;
gbc.gridy = 0;
// gbc.insets = bottom;
add(n1, gbc);

gbc.gridx = 1;
gbc.gridheight = 3;
// gbc.insets = none;
add(symbol, gbc);
```

```
gbc.gridx = 2;
gbc.gridheight = 1;
// gbc.insets = bottom;
add(n2, gbc);

gbc.gridx = 3;
gbc.gridheight = 3;
// gbc.insets = none;
add(equal, gbc);

gbc.gridx = 4;
gbc.gridheight = 1;
// gbc.insets = bottom;
add(n3, gbc);

gbc.gridx = 0;
gbc.gridy = 1;
// gbc.insets = none;
add(new Bar(ventana), gbc);

gbc.gridx = 2;
add(new Bar(ventana), gbc);

gbc.gridx = 4;
add(new Bar(ventana), gbc);

gbc.gridx = 0;
gbc.gridy = 2;
// gbc.insets = top;
add(d1, gbc);

gbc.gridx = 2;
add(d2, gbc);

gbc.gridx = 4;
```

```
        add(d3, gbc);

    }

    public void setFraction1(Fraction value) {
        fraction1 = value;
    }

    public void setFraction2(Fraction value) {
        fraction2 = value;
    }

    public Fraction getFraction1() {
        return fraction1;
    }

    public Fraction getFraction2() {
        return fraction2;
    }

    public Fraction getFraction3() {
        return fraction3;
    }

    public void setFraction3(Fraction fraction) {

        fraction3 = fraction;
        n3.setText("" + fraction.getNumerator());
        n3.setValue(fraction.getNumerator());
        d3.setText("" + fraction.getDenominator());
        d3.setValue(fraction.getDenominator());

    }

    public JLabel getSymbol() {
        return symbol;
    }
}
```

```
}

    public CustomText getN3() {
        return n3;
    }

    public CustomText getD3() {
        return d3;
    }
}

// Clase CustomText

class CustomText extends javax.swing.JTextField {

    private Ventana ventana;

    private int value;
    private char type;
    private CustomText pair;
    private int number = 0;

    public CustomText(Ventana ventana, String text, Boolean editable, char
type, int number) {

        super(text);
        this.ventana = ventana;
        this.type = type;
        this.number = number;
        value = Integer.parseInt(text);

        // Configuraciones
        setFont(new Font("Arial Nova", Font.BOLD, ventana.getWidth() / 20));

        if (editable) {
```

```
        setBackground(Colors.GRAY);
        setCaretColor(Colors.WHITE);
    } else {
        setBackground(Colors.BLACK);
        setCaretColor(getBackground());
    }

    setEditable(editable);
    setSelectedTextColor(Colors.WHITE);
    setSelectionColor(Colors.DARK_GRAY);
    setForeground(Colors.WHITE);
    setHorizontalAlignment(SwingConstants.CENTER);
    setBorder(BorderFactory.createEmptyBorder());
    setOpaque(false);

    addKeyListener(new KeyAdapter() {

        @Override
        public void keyTyped(KeyEvent e) {

            char key = e.getKeyChar();

            if (key == '\n' && !getText().equals("")) {

                // Asigna la fraccion
                value = Integer.parseInt(getText());
                setFraction(pair, number);

                // Busca la operacion
                char simbolo =
                    ventana.getFractionPanel().getSymbol().getText().charAt(0);

                Fraction opera = new Fraction(0, 0);
                if (simbolo == '+') {
```



```

        opera =
Fractions.suma(ventana.getFractionPanel().getFraction1(),
ventana.getFractionPanel().getFraction2());

    } else if (simbolo == '-') {

        opera =
Fractions.resta(ventana.getFractionPanel().getFraction1(),
ventana.getFractionPanel().getFraction2());

    } else if (simbolo == 'x') {

        opera =
Fractions.multiplicacion(ventana.getFractionPanel().getFraction1(),
ventana.getFractionPanel().getFraction2());

    } else {

        opera =
Fractions.division(ventana.getFractionPanel().getFraction1(),
ventana.getFractionPanel().getFraction2());

    }

    // Opera las fracciones y asigna a la 3ra
    if (opera.valid() || (opera.getNumerator() == 0 &&
opera.getDenominator() == 0)) {
        ventana.getFractionPanel().getN3().setText("0_o");
        ventana.getFractionPanel().getD3().setText("Error");
    } else {
        ventana.getFractionPanel().setFraction3(opera);
    }

} else if (!Character.isDigit(key)) {
    e.consume();

```

```
        System.err.println("Se ingreso un caracter no
numerico");
    } else if (getText().length() >= 4) {
        e.consume();
        System.err.println("Se ingreso un numero muy grande");
    }
    }
});

}

public void setPair(CustomText pair) {
    this.pair = pair;
}

public int getValue() {
    return value;
}

public void setValue(int value) {
    this.value = value;
}

public void setFraction(CustomText pair, int id) {

    Fraction fraction;

    if (type == 'N') {
        fraction = new Fraction(value, pair.getValue());
    } else {
        fraction = new Fraction(pair.getValue(), value);
    }

    if (id == 1) {
        ventana.getFractionPanel().setFraction1(fraction);
    }
}
```

```

        } else {
            ventana.getFractionPanel().setFraction2(fraction);
        }
        // System.out.println(fraction.getNumerator() + " : " +
fraction.getDenominator());
        System.out.println(ventana.getFractionPanel().getFraction1().getNum
rator() + " : " +
ventana.getFractionPanel().getFraction1().getDenominator());
        System.out.println(ventana.getFractionPanel().getFraction2().getNum
rator() + " : " +
ventana.getFractionPanel().getFraction2().getDenominator());

    }

    @Override
    public void paintComponent(Graphics g) {

        Graphics2D g2 = (Graphics2D) g;
        g2.setRenderingHint(RenderingHints.KEY_ANTIALIASING,
RenderingHints.VALUE_ANTIALIAS_ON);

        RoundRectangle2D rounded = new RoundRectangle2D.Double(0, 0,
getWidth(), getHeight(), 20, 20);
        g2.setColor(getBackground());
        g2.fill(rounded);

        super.paintComponent(g);

    }

}

// Clase bar

class Bar extends javax.swing.JPanel {

```

```
Ventana ventana;

public Bar(Ventana ventana) {

    super();
    this.ventana = ventana;
    setOpaque(false);

}

@Override
public void paintComponent(Graphics g) {

    super.paintComponent(g);
    Graphics2D g2 = (Graphics2D) g;
    g2.setRenderingHint(RenderingHints.KEY_ANTIALIASING,
RenderingHints.VALUE_ANTIALIAS_ON);

    int height = ventana.getWidth() / 64;
    RoundedRectangle2D bar = new RoundedRectangle2D.Double(0, getHeight() /
2 - height / 2, getWidth(), height, height, height);
    g2.setColor(Colors.WHITE);
    g2.fill(bar);

}

}
```

RoundPane.java

```
package ui;

import javax.swing.*.*;
import java.awt.*.*;
import java.awt.geom.RoundRectangle2D;
```

```
public class RoundPane extends javax.swing.JPanel {

    public RoundPane(Ventana ventana) {

        setSize(ventana.getWidth(), ventana.getHeight());
        setLayout(new BorderLayout(this, BorderLayout.Y_AXIS));

    }

    @Override
    public void paintComponent(Graphics g) {

        super.paintComponent(g);
        Graphics2D g2 = (Graphics2D) g;
        g2.setRenderingHint(RenderingHints.KEY_ANTIALIASING,
RenderingHints.VALUE_ANTIALIAS_ON);

        RoundRectangle2D border = new RoundRectangle2D.Double(0, 0,
getWidth(), getHeight(), 22, 22);
        g2.setColor(Colors.BLACK);
        g2.fill(border);

        RoundRectangle2D rectangle = new RoundRectangle2D.Double(1, 1,
getWidth() - 2, getHeight() - 2, 20, 20);
        g2.setColor(Colors.DARK_GRAY);
        g2.fill(rectangle);

    }

}
```

TitleBar.java

```
package ui;

import javax.swing.*.*;
```

```
import java.awt.*;
import java.awt.event.*;
import java.awt.geom.Ellipse2D;
import java.awt.geom.Rectangle2D;
import java.awt.geom.RoundRectangle2D;

/**
 * JPanel donde se muestra el nombre de la aplicacion y el boton de salida
 */

public class TitleBar extends javax.swing.JPanel {

    private Ventana ventana;
    private JLabel title;
    private JButton close;

    public TitleBar(Ventana ventana) {

        // Llenado de variables
        this.ventana = ventana;

        // Configuracion de barra de titulo
        setPreferredSize(new Dimension(ventana.getUsableWidth(), 40));
        setMinimumSize(getPreferredSize());
        setMaximumSize(getPreferredSize());
        setBorder(BorderFactory.createEmptyBorder(9, 31, 9, 9));
        setLayout(new GridBagLayout());
        setBackground(Colors.BLACK);
        setOpaque(false);
        MouseAdapter drag = new MouseAdapter() {

            int mouseX, mouseY;

            @Override
            public void mousePressed(MouseEvent e) {
                mouseX = e.getX();
```

```
        mouseY = e.getY();
    }

    @Override
    public void mouseDragged(MouseEvent e) {
        int x = e.getXOnScreen() - mouseX;
        int y = e.getYOnScreen() - mouseY;

        ventana.setLocation(x, y);
    }
};
addMouseListener(drag);
addMouseMotionListener(drag);

// Configuracion de titulo
title = new JLabel("Calculadora de fracciones");
title.setPreferredSize(new Dimension(ventana.getUsableWidth() - 62,
22));
title.setFont(new Font("Arial Nova", Font.BOLD, 16));
title.setHorizontalAlignment(SwingConstants.CENTER);
title.setForeground(Colors.WHITE);

// Configuracion del boton close
close = new JButton() {

    boolean isMouseIn = false;
    {
        // Configuraciones
        setPreferredSize(new Dimension(22, 22));
        setBackground(Colors.RED);
        setFocusPainted(false);
        setBorderPainted(false);
        setContentAreaFilled(false);

        // Listeners
        addActionListener(new ActionListener() {
```

```
        @Override
        public void actionPerformed(ActionEvent e) {
            System.exit(0);
        }
    });

    addMouseListener(new MouseAdapter() {
        @Override
        public void mouseEntered(MouseEvent e) {
            isMouseIn = true;
        }

        @Override
        public void mouseExited(MouseEvent e) {
            isMouseIn = false;
        }
    });
}

@Override
public void paintComponent(Graphics g) {

    super.paintComponent(g);
    Graphics2D g2 = (Graphics2D) g;
    g2.setRenderingHint(RenderingHints.KEY_ANTIALIASING,
RenderingHints.VALUE_ANTIALIAS_ON);

    Ellipse2D circle = new Ellipse2D.Double(0, 0, 22, 22);

    if (isMouseIn) {
        g2.setColor(Colors.RED.brighter().brighter());
    } else {
        g2.setColor(getBackground());
    }
    g2.fill(circle);
}
```



```
    }  
};  
  
// Adicion de componentes  
GridBagConstraints gbc = new GridBagConstraints();  
gbc.fill = GridBagConstraints.BOTH;  
  
gbc.gridx = 0;  
gbc.gridy = 0;  
add(title, gbc);  
  
gbc.gridx = 1;  
gbc.gridy = 0;  
add(close, gbc);  
  
}  
  
@Override  
public void paintComponent(Graphics g) {  
  
    super.paintComponent(g);  
    Graphics2D g2 = (Graphics2D) g;  
    g2.setRenderingHint(RenderingHints.KEY_ANTIALIASING,  
RenderingHints.VALUE_ANTIALIAS_ON);  
  
    RoundRectangle2D rounded = new RoundRectangle2D.Double(0, 0,  
getWidth(), getHeight(), 20, 20);  
    Rectangle2D rectangle = new Rectangle2D.Double(0, getHeight() / 2,  
getWidth(), getHeight());  
  
    g2.setColor(getBackground());  
    g2.fill(rounded);  
    g2.fill(rectangle);  
  
}
```

```
}
```

Ventana.java

```
package ui;

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

/**
 * Ventana donde se ve la ejecucion principal del programa
 */

public class Ventana extends javax.swing.JFrame {

    private int usableWidth, usableHeight;
    private TitleBar titleBar;
    private FractionPanel fractionPanel;
    private ButtonPanel buttonPanel;

    // Constructor para la ventana
    public Ventana() {

        super();
        // Tamaño de pantalla
        Dimension screen = Toolkit.getDefaultToolkit().getScreenSize();
        screen.setSize(screen.width / 3, screen.width / 4);

        // Configuracion de ventana
        setSize(screen);
        setLocationRelativeTo(null);
        setUndecorated(true);
        setContentPane(new RoundPane(this));
        setBackground(new Color(0, 0, 0, 0));

        // Llenado de variables
    }
}
```

```
usableWidth = getWidth() - 2;
usableHeight = getHeight() - 2;

// Creacion de paneles
titleBar = new TitleBar(this);
fractionPanel = new FractionPanel(this);
buttonPanel = new ButtonPanel(this);

// Adicion de paneles
add(titleBar);
add(fractionPanel);
add(buttonPanel);

}

// Metodos
public int getUsableWidth() {
    return usableWidth;
}

public int getUsableHeight() {
    return usableHeight;
}


public FractionPanel getFractionPanel() {
    return fractionPanel;
}
}
```

Main.java

```
import ui.Ventana;

public class Main {

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

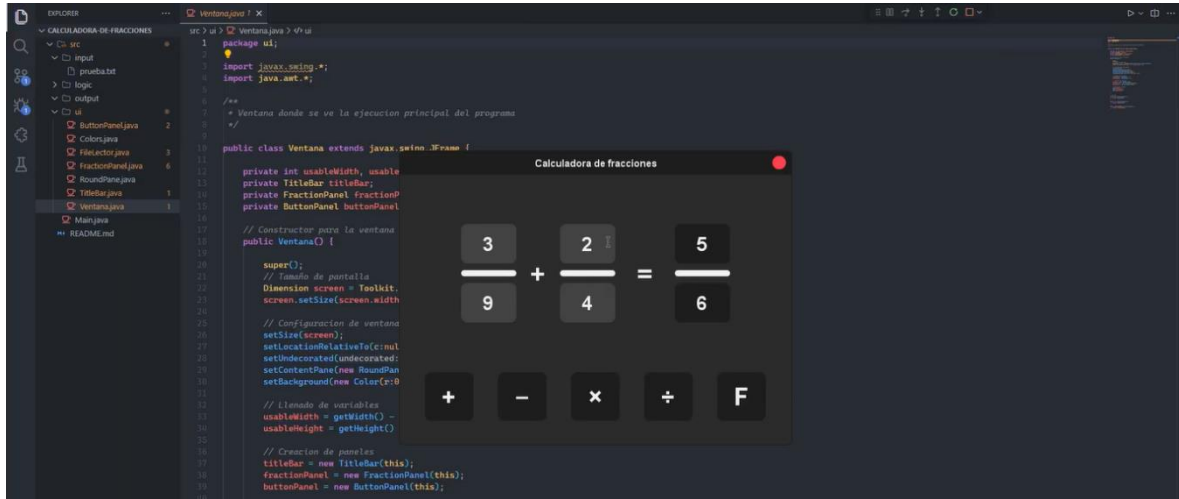


```
Ventana ventana = new Ventana();  
ventana.setVisible(true);  
  
}  
}
```

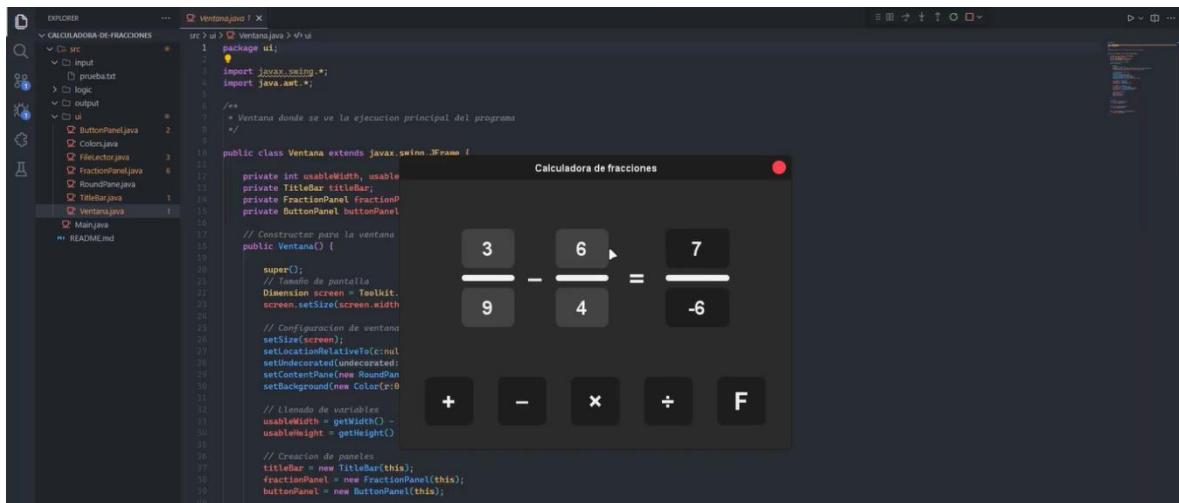


PRUEBAS DE EJECUCIÓN

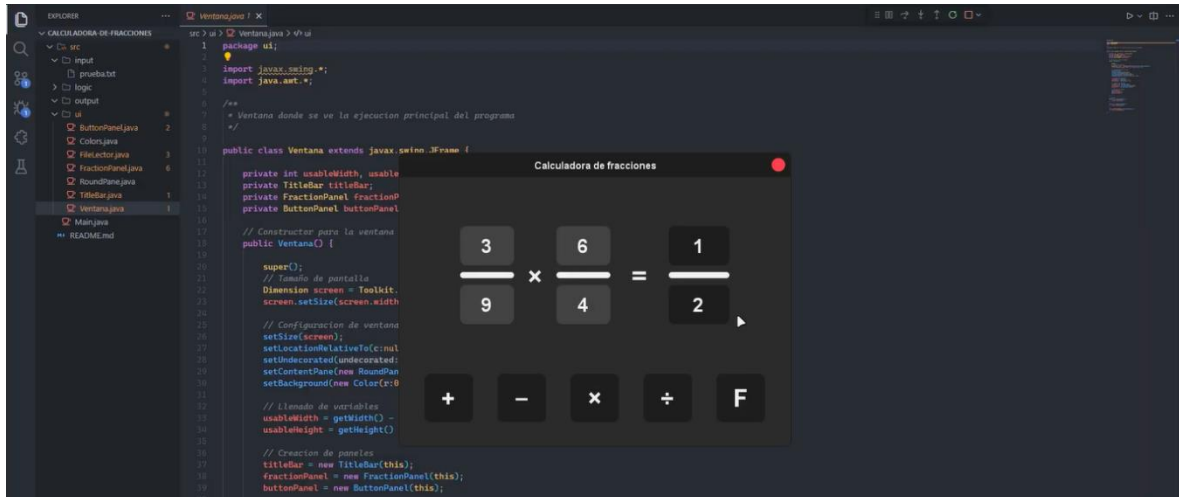
SUMA:



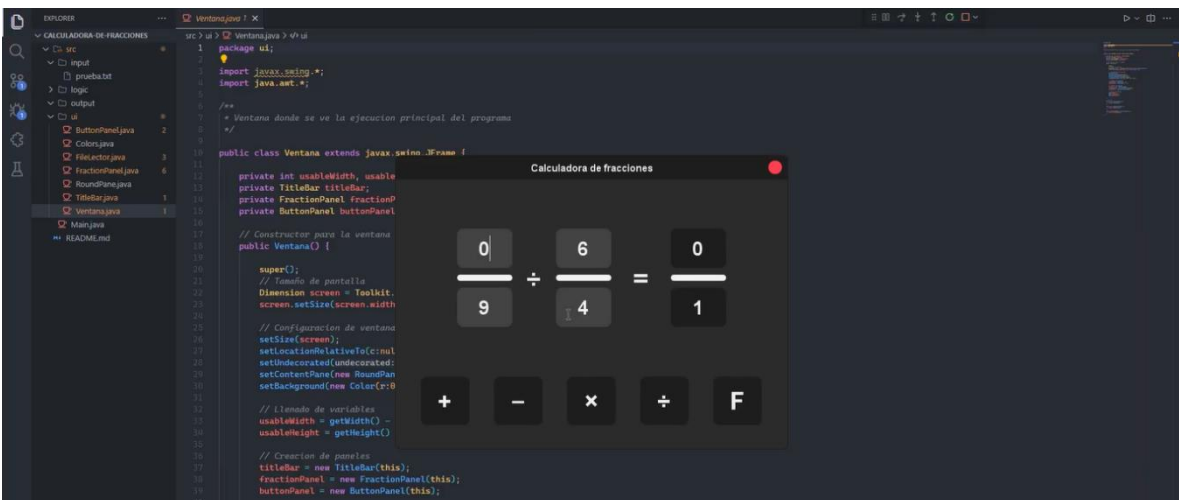
RESTA:



MULTIPLICACIÓN:



DIVISIÓN:





LINK DEL VIDEO DE EJECUCIÓN

https://drive.google.com/file/d/1Op0aON_3DjqgAhzvXBNJM45r6XUkgey9/view