**Seguimiento 5**

# FIGURAS

**Cilindro** package Figuras;

/\*\*

* Esta clase denominada Cilindro es una subclase de FiguraGeométrica
* que cuenta con un radio y una altura.
* @version 1.2/2020

\*/ public class Cilindro extends FiguraGeometrica { private double radio; // Atributo que establece el radio de un cilindro private double altura; // Atributo que establece la altura de un cilindro

/\*\*

* Constructor de la clase Cilindro
* @param radio Parámetro de define el radio de un cilindro
* @param altura Parámetro de define la altura de un cilindro

\*/

public Cilindro(double radio, double altura) {

this.radio = radio; this.altura = altura; this.setVolumen(calcularVolumen()); /\* Calcula el volumen y establece su atributo \*/ this.setSuperficie(calcularSuperficie()); /\* Calcula la superficie y establece su atributo \*/

}

/\*\*

* Método para calcular el volumen de un cilindro
* @return El volumen de un cilindro

\*/

public double calcularVolumen() { double volumen = Math.PI \* altura \* Math.pow(radio, 2.0); return volumen;

}

/\*\*

* Método para calcular la superficie de un cilindro
* @return La superficie de un cilindro

\*/

public double calcularSuperficie() { double áreaLadoA = 2.0 \* Math.PI \* radio \* altura; double áreaLadoB = 2.0 \* Math.PI \* Math.pow(radio, 2.0); return áreaLadoA + áreaLadoB;

}

}

## Esfera

package Figuras;

/\*\*

* Esta clase denominada Esfera es una subclase de FiguraGeométrica
* que cuenta con un radio.
* @version 1.2/2020

\*/ public class Esfera extends FiguraGeometrica { private double radio; // Atributo que identifica el radio de una esfera

/\*\*

* Constructor de la clase Esfera
* @param radio Parámetro de define el radio de una esfera
* \*/

public Esfera(double radio) {

this.radio = radio; this.setVolumen(calcularVolumen()); /\* Calcula el volumen y establece su atributo \*/ this.setSuperficie(calcularSuperficie()); /\* Calcula la superficie y establece su atributo \*/

}

/\*\*

* Método para calcular el volumen de una esfera
* @return El volumen de una esfera

\*/

public double calcularVolumen() { double volumen = 1.333 \* Math.PI \* Math.pow(this.radio, 3.0); return volumen;

}

/\*\*

* Método para calcular la superficie de una esfera
* @return La superficie de una esfera

\*/

public double calcularSuperficie() { double superficie = 4.0 \* Math.PI \* Math.pow(this.radio, 2.0); return superficie;

}

}

## FiguraGeometrica package Figuras;

/\*\*

* Esta clase denominada FiguraGeométrica modela un figura \* geométrica que cuenta con un volumen y una superficie a ser
* calculados de acuerdo al tipo de figura geométrica.
* @version 1.2/2020

\*/ public class FiguraGeometrica { private double volumen; /\* Atributo que identifica el volumen de una figura geométrica \*/ private double superficie; /\* Atributo que identifica la superficie de una figura geométrica \*/

/\*\*

* Método para establecer el volumen de una figura geométrica
* @param volumen Parámetro que define el volumen de una figura
* geométrica

\*/

public void setVolumen(double volumen) { this.volumen = volumen;

}

/\*\*

* Método para establecer la superficie de una figura geométrica
* @param superficie Parámetro que define la superficie de una
* figura geométrica

\*/

public void setSuperficie(double superficie) { this.superficie = superficie;

}

/\*\*

* Método para obtener el volumen de una figura geométrica
* @return El volumen de una figura geométrica

\*/

public double getVolumen() { return this.volumen;

}

/\*\*

* Método para obtener la superficie de una figura geométrica
* @return La superficie de una figura geométrica

\*/

public double getSuperficie() { return this.superficie;

}

}

**Piramide** package Figuras;

/\*\*

* Esta clase denominada Pirámide es una subclase de FiguraGeométrica
* que cuenta con una base, una altura y un apotema.
* @version 1.2/2020

\*/ public class Piramide extends FiguraGeometrica { private double base; /\* Atributo que identifica la base de una pirámide \*/ private double altura; /\* Atributo que identifica la altura de unapirámide \*/ private double apotema; /\* Atributo que identifica el apotema de una pirámide \*/

/\*\*

* Constructor de la clase Pirámide
* @param base Parámetro de define la base de una pirámide
* @param altura Parámetro de define la altura de una pirámide
* @param apotema Parámetro de define el apotema de una pirámide

\*/

public Piramide(double base, double altura, double apotema) { this.base = base; this.altura = altura; this.apotema = apotema; this.setVolumen(calcularVolumen()); /\* Calcula el volumen y establece su atributo \*/ this.setSuperficie(calcularSuperficie()); /\* Calcula la superficie y establece su atributo \*/

}

/\*\*

* Método para calcular el volumen de una pirámide
* @return El volumen de una pirámide

\*/

public double calcularVolumen() { double volumen = (Math.pow(base, 2.0) \* altura) / 3.0; return volumen;

}

/\*\*

* Método para calcular la superficie de una pirámide
* @return La superficie de una pirámide

\*/

public double calcularSuperficie() { double áreaBase = Math.pow(base, 2.0); double áreaLado = 2.0 \* base \* apotema; return áreaBase + áreaLado;

}

}

## WindowCilindro

package Figuras;

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

/\*\*

* Esta clase denominada VentanaCilindro define una ventana para
* ingresar los datos de un cilindro y calcular su volumen y superficie.
* @version 1.2/2020

\*/ public class WindowCilindro extends JFrame implements ActionListener {

// Un contenedor de elementos gráficos private Container contenedor;

/\* Etiquetas estáticas para identificar los campos de texto a ingresar y calcular \*/ private JLabel radio, altura, volumen, superficie;

// Campos de texto a ingresar private JTextField campoRadio, campoAltura; // Botón para realizar los cálculos numéricos private JButton calcular;

/\*\*

\* Constructor de la clase VentanaCilindro

\*/

public WindowCilindro() {

inicio();

setTitle("Cilindro"); // Establece el título de la ventana setSize(280,210); // Establece el tamaño de la ventana setLocationRelativeTo(null); /\* La ventana se posiciona en el centro de la pantalla \*/ setResizable(false); /\* Establece que el botón de cerrar permitirá

salir de la aplicación \*/

}

/\*\*

* Método que crea la ventana con sus diferentes componentes
* gráficos \* \*/ private void inicio() {

contenedor = getContentPane(); /\* Obtiene el panel de contenidos de la ventana \*/ contenedor.setLayout(null); /\* Establece que el contenedor no

tiene un layout \*/

// Establece la etiqueta y campo de texto para el radio del cilindro radio = new JLabel(); radio.setText("Radio (cms):"); radio.setBounds(20, 20, 135, 23); /\* Establece la posición de la etiqueta de radio del cilindro \*/ campoRadio = new JTextField();

// Establece la posición del campo de texto de radio del cilindro campoRadio.setBounds(100, 20, 135, 23);

// Establece la etiqueta y campo de texto para la altura del cilindro altura = new JLabel(); altura.setText("Altura (cms):"); altura.setBounds(20, 50, 135, 23); /\* Establece la posición de la etiqueta de altura del cilindro \*/ campoAltura = new JTextField();

// Establece la posición del campo de texto de altura del cilindro campoAltura.setBounds(100, 50, 135, 23);

/\* Establece el botón para calcular el volumen y superficie del

cilindro \*/

calcular = new JButton(); calcular.setText("Calcular"); calcular.setBounds(100, 80, 135, 23); /\* Establece la posición del botón calcular \*/

/\* Agrega al botón un ActionListener para que gestione eventos

del botón \*/ calcular.addActionListener(this);

// Establece la etiqueta y el valor del volumen del cilindro volumen = new JLabel(); volumen.setText("Volumen (cm3):");

// Establece la posición de la etiqueta de volumen del cilindro volumen.setBounds(20, 110, 135, 23);

// Establece la etiqueta y el valor de la superficie del cilindro superficie = new JLabel(); superficie.setText("Superficie (cm2):");

// Establece la posición de la etiqueta de superficie del cilindro superficie.setBounds(20, 140, 135, 23);

// Se añade cada componente gráfico al contenedor de la ventana contenedor.add(radio); contenedor.add(campoRadio); contenedor.add(altura); contenedor.add(campoAltura); contenedor.add(calcular); contenedor.add(volumen); contenedor.add(superficie);

}

/\*\*

* Método que gestiona los eventos generados en la ventana del
* cilindro throws Exception Excepción al ingresar un campo nulo o
* error en formato de número

\*/

public void actionPerformed(ActionEvent event) { // Se inicializan el radio y la altura del cilindro boolean error = false; /\* Se inicializa variable para determinar si ocurre un error \*/ double radio = 0; double altura = 0;

try {

// Se obtiene el radio del cilindro ingresado radio = Double.parseDouble(campoRadio.getText()); // Se obtiene la altura del cilindro ingresada altura = Double.parseDouble(campoAltura.getText()); Cilindro cilindro = new Cilindro(radio, altura); /\* Se crea un objeto Cilindro \*/

// Se calcula y muestra el volumen volumen.setText("Volumen (cm3): " + String.format("%.2f",cilindro.calcularVolumen()));

// Se calcula y muestra la superficie superficie.setText("Superficie (cm2): " + String.format("%.2f",cilindro.calcularSuperficie()));

}

catch (Exception e){ error = true; // Si ocurre una excepción

}

finally {

if(error) { /\* Si ocurre una excepción, se muestra un mensaje

de error \*/

JOptionPane.showMessageDialog(null,"Campo nulo o error en formato de numero", "Error", JOptionPane.ERROR\_MESSAGE);

}

}

}

}

**WindowEsfera** package Figuras; import javax.swing.\*; import java.awt.\*; import java.awt.event.\*; import java.awt.event.ActionEvent; import java.awt.event.ActionListener;

/\*\*

* Esta clase denominada VentanaEsfera define una ventana para
* ingresar los datos de una esfera y calcular su volumen y superficie.
* @version 1.2/2020

\*/

public class WindowEsfera extends JFrame implements ActionListener {

// Un contenedor de elementos gráficos private Container contenedor;

/\* Etiquetas estáticas para identificar los campos de texto a ingresar

y calcular \*/ private JLabel radio, volumen, superficie; private JTextField campoRadio; // Campo de texto a ingresar private JButton calcular; /\* Botón para realizar los cálculos numéricos \*/

/\*\*

\* Constructor de la clase VentanaEsfera

\*/ public WindowEsfera() {

inicio();

setTitle("Esfera"); // Establece el título de la ventana setSize(280,200); // Establece el tamaño de la ventana setLocationRelativeTo(null); /\* La ventana se posiciona en el centro de la pantalla \*/ setResizable(false); /\* Establece que el botón de cerrar permitirá salir de la aplicación \*/

}

/\*\*

* Método que crea la ventana con sus diferentes componentes
* gráficos \*/

private void inicio() { contenedor = getContentPane(); /\* Obtiene el panel de contenidos de la ventana \*/ contenedor.setLayout(null); /\* Establece que el contenedor no

tiene un layout \*/

// Establece la etiqueta y campo de texto para el radio de la esfera radio = new JLabel(); radio.setText("Radio (cms):"); radio.setBounds(20, 20, 135, 23); /\* Establece la posición de la etiqueta de radio de la esfera \*/ campoRadio = new JTextField();

// Establece la posición del campo de texto de radio de la esfera campoRadio.setBounds(100, 20, 135, 23);

/\* Establece el botón para calcular el volumen y superficie de la

esfera \*/

calcular = new JButton(); calcular.setText("Calcular"); calcular.setBounds(100, 50, 135, 23); /\* Establece la posición del botón calcular \*/

/\* Agrega al botón un ActionListener para que gestione eventos del botón \*/ calcular.addActionListener(this);

// Establece la etiqueta y el valor del volumen de la esfera volumen = new JLabel(); volumen.setText("Volumen (cm3):");

// Establece la posición de la etiqueta de volumen de la esfera volumen.setBounds(20, 90, 135, 23);

// Establece la etiqueta y el valor de la superficie de la esfera superficie = new JLabel(); superficie.setText("Superficie (cm2):");

// Establece la posición de la etiqueta de superficie de la esfera superficie.setBounds(20, 120, 135, 23);

// Se añade cada componente gráfico al contenedor de la ventana contenedor.add(radio); contenedor.add(campoRadio); contenedor.add(calcular); contenedor.add(volumen); contenedor.add(superficie);

}

/\*\*

* Método que gestiona los eventos generados en la ventana de la
* esfera throws Exception Excepción al ingresar un campo nulo o
* error en formato de número

\*/

public void actionPerformed(ActionEvent evento) { if (evento.getSource() == calcular) { /\* Si se pulsa el botón Calcular \*/ boolean error = false;

try {

// Se obtiene y convierte el valor numérico del radio double radio = Double.parseDouble(campoRadio.

getText());

Esfera esfera = new Esfera(radio); /\* Se crea un objeto

Esfera \*/

// Se muestra el volumen volumen.setText("Volumen (cm3): " + String.format("%.2f", esfera.calcularVolumen()));

// Se muestra la superficie superficie.setText("Superficie (cm2): " + String.format("%.2f",esfera.calcularSuperficie()));

}

catch (Exception e) { error = true; // Si ocurre una excepción

}

finally {

if(error) { /\* Si ocurre una excepción, se muestra un mensaje de error \*/

JOptionPane.showMessageDialog(null,"Campo nulo o error en formato de número",

"Error", JOptionPane.ERROR\_MESSAGE);

}

}

}

}

}

## WindowPiramide

package Figuras; import javax.swing.\*; import java.awt.\*; import java.awt.event.\*; import java.awt.event.ActionEvent; import java.awt.event.ActionListener;

/\*\*

* Esta clase denominada VentanaPirámide define una ventana para
* ingresar los datos de una pirámide y calcular su volumen y superficie.
* @version 1.2/2020

\*/

public class WindowPiramide extends JFrame implements ActionListener {

// Un contenedor de elementos gráficos private Container contenedor;

/\* Etiquetas estáticas para identificar los campos de texto a ingresar

y calcular \*/ private JLabel base, altura, apotema, volumen, superficie;

// Campos de texto a ingresar private JTextField campoBase, campoAltura, campoApotema;

// Botón para realizar los cálculos numéricos private JButton calcular;

/\*\*

\* Constructor de la clase VentanaPirámide

\*/

public WindowPiramide() {

inicio();

setTitle("Pirámide"); // Establece el título de la ventana setSize(280,240); // Establece el tamaño de la ventana setLocationRelativeTo(null); /\* La ventana se posiciona en el centro de la pantalla \*/ setResizable(false); /\* Establece que el botón de cerrar permitirá salir de la aplicación \*/

}

/\*\*

* Método que crea la ventana con sus diferentes componentes
* gráficos \*/

private void inicio() { contenedor = getContentPane(); /\* Obtiene el panel de contenidos de la ventana \*/ contenedor.setLayout(null); /\* Establece que el contenedor no tiene un layout \*/

/\* Establece la etiqueta y campo de texto para la base de la pirámide \*/ base = new JLabel(); base.setText("Base (cms):");

// Establece la posición de la etiqueta de la base de la pirámide base.setBounds(20, 20, 135, 23); campoBase = new JTextField();

/\* Establece la posición del campo de texto de la base de la pirámide \*/

campoBase.setBounds(120, 20, 135, 23);

/\* Establece la etiqueta y campo de texto para la altura de la pirámide \*/ altura = new JLabel(); altura.setText("Altura (cms):");

// Establece la posición de la etiqueta de la altura de la pirámide altura.setBounds(20, 50, 135, 23); campoAltura = new JTextField();

/\* Establece la posición del campo de texto de la altura de la pirámide \*/ campoAltura.setBounds(120, 50, 135, 23);

/\* Establece la etiqueta y campo de texto para el apotema de la

pirámide \*/ apotema = new JLabel(); apotema.setText("Apotema (cms):");

// Establece la posición de la etiqueta del apotema de la pirámide apotema.setBounds(20, 80, 135, 23); campoApotema = new JTextField();

/\* Establece la posición del campo de texto del apotema de la pirámide \*/

campoApotema.setBounds(120, 80, 135, 23);

/\* Establece el botón para calcular volumen y superficie de la pirámide \*/ calcular = new JButton(); calcular.setText("Calcula"); calcular.setBounds(120, 110, 135, 23); /\* Establece la posición del botón calcular \*/

/\* Agrega al botón un ActionListener para que gestione eventos del botón \*/

calcular.addActionListener(this);

// Establece la etiqueta y el valor del volumen de la pirámide volumen = new JLabel(); volumen.setText("Volumen (cm3):");

// Establece la posición de la etiqueta de volumen de la pirámide volumen.setBounds(20, 140, 135, 23);

// Establece la etiqueta y el valor de la superficie de la pirámide superficie = new JLabel(); superficie.setText("Superficie (cm2):");

// Establece la posición de la etiqueta de superficie de la pirámide superficie.setBounds(20, 170, 135, 23);

// Se añade cada componente gráfico al contenedor de la ventana contenedor.add(base); contenedor.add(campoBase); contenedor.add(altura); contenedor.add(campoAltura); contenedor.add(apotema); contenedor.add(campoApotema);

contenedor.add(calcular); contenedor.add(volumen); contenedor.add(superficie);

}

/\*\*

* Método que gestiona los eventos generados en la ventana de la
* esfera throws Exception Excepción al ingresar un campo nulo o
* error en formato de número
* \* \*/

public void actionPerformed(ActionEvent event) {

Piramide piramide; boolean error = false; double base = 0; double altura = 0; double apotema = 0;

try {

// Se obtiene y convierte el valor numérico de la base base = Double.parseDouble(campoBase.getText()); // Se obtiene y convierte el valor numérico de la altura altura = Double.parseDouble(campoAltura.getText()); // Se obtiene y convierte el valor numérico del apotema apotema = Double.parseDouble(campoApotema.getText());

// Se crea un objeto Pirámide piramide = new Piramide(base, altura, apotema);

// Se muestra el volumen volumen.setText("Volumen (cm3): " + String.format("%.2f",piramide.calcularVolumen()));

// Se muestra la superficie superficie.setText("Superficie (cm2): " + String.format("%.2f",piramide.calcularSuperficie()));

}

catch (Exception e) { error = true; // Si ocurre una excepción

}

finally {

if (error) { /\* Si ocurre una excepción, se muestra un mensaje

de error \*/

JOptionPane.showMessageDialog(null, "Campo nulo o error en formato de número",

"Error",JOptionPane.ERROR\_MESSAGE);

}

}

}

}

**Main** package Figuras;

/\*\*

* Esta clase define el punto de ingreso al programa de figuras
* geométricas. Por lo tanto, cuenta con un método main de acceso al
* programa.
* @version 1.2/2020

\*/ public class main {

/\*\*

* Método main que sirve de punto de entrada al programa

\*/ public static void main(String[] args) { mainWindow miVentanaPrincipal; /\* Define la ventana

principal \*/

miVentanaPrincipal= new mainWindow(); /\* Crea la ventana

principal \*/ miVentanaPrincipal.setVisible(true); /\* Establece la ventana como visible \*/

// Establece que la ventana no puede cambiar su tamaño miVentanaPrincipal.setResizable(false);

}

}

## mainWindow

package Figuras; import java.awt.\*; import javax.swing.\*; import java.awt.event.\*; import java.awt.event.ActionEvent; import java.awt.event.ActionListener;

/\*\*

* Esta clase denominada VentanaPrincipal define una interfaz gráfica
* que permitirá consultar un menú principal con tres figuras
* geométricas.
* @version 1.2/2020

\*/ public class mainWindow extends JFrame implements ActionListener {

// Un contenedor de elementos gráficos private Container contenedor;

// Botones para seleccionar una figura geométrica determinada private JButton cilindro, esfera, pirámide;

/\*\*

* Constructor de la clase VentanaPrincipal

\*/

public mainWindow(){

inicio();

setTitle("Figura"); // Establece el título de la ventana setSize(350,160); // Establece el tamaño de la ventana setLocationRelativeTo(null); /\* La ventana se posiciona en el centro de la pantalla \*/

// Establece que el botón de cerrar permitirá salir de la aplicación setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

}

/\*\*

* Método que crea la ventana con sus diferentes componentes
* gráficos \*/ private void inicio() {

contenedor = getContentPane(); /\* Obtiene el panel de contenidos de la ventana \*/ contenedor.setLayout(null); /\* Establece que el contenedor no

tiene un layout \*/

// Establece el botón del cilindro cilindro = new JButton(); cilindro.setText("Cilindro"); cilindro.setBounds(20, 50, 80, 23); /\* Establece la posición del botón del cilindro \*/

/\* Agrega al botón un ActionListener para que gestione eventos

del botón \*/ cilindro.addActionListener(this); // Establece el botón de la esfera esfera = new JButton(); esfera.setText("Esfera"); esfera.setBounds(125, 50, 80, 23); /\* Establece la posición del botón de la esfera \*/

/\* Agrega al botón un ActionListener para que gestione eventos

del botón \*/ esfera.addActionListener(this); // Establece el botón de la pirámide pirámide = new JButton(); pirámide.setText("Pirámide"); pirámide.setBounds(225, 50, 100, 23); /\* Establece la posición del botón de la pirámide \*/

/\* Agrega al botón un ActionListener para que gestione eventos del botón \*/ pirámide.addActionListener(this);

// Se añade cada componente gráfico al contenedor de la ventana contenedor.add(cilindro); contenedor.add(esfera); contenedor.add(pirámide);

}

/\*\*

\* Método que gestiona los eventos generados en la ventana principal

\*/

public void actionPerformed(ActionEvent evento) { if (evento.getSource() == esfera) { // Si se pulsa el botón esfera WindowEsfera esfera = new WindowEsfera(); /\* Crea la

ventana de la esfera \*/ esfera.setVisible(true); /\* Establece que se visualice la ventana de la esfera \*/

}

if (evento.getSource() == cilindro) { /\* Si se pulsa el botón

cilindro \*/

WindowCilindro cilindro = new WindowCilindro(); /\* Crea la ventana del cilindro \*/ cilindro.setVisible(true); /\* Establece que se visualice la ventana del cilindro \*/

}

if (evento.getSource() == pirámide) { /\* Si se pulsa el botón

pirámide \*/

WindowPiramide pirámide = new WindowPiramide(); /\* Crea la ventana de la pirámide \*/ pirámide.setVisible(true); /\* Establece que se visualice la ventana de la pirámide \*/

}

}

}

**NOTAS Main** package Notas;

/\*\*

\*

\* @author Mario Cañas

\*/ public class main {

public static void main(String[] args) {

mainWindow win = new mainWindow();

win.setVisible(true); win.setResizable(false);

}

}

## mainWindow

package Notas; import java.awt.\*; import java.awt.event.ActionEvent; import java.awt.event.ActionListener; import javax.swing.\*;

/\*\*

* Esta clase denominada mainWindow define una interfaz gráfica
* que permitirá crear un array de notas. Luego, se puede calcular el \* promedio de notas, la desviación, la nota mayor y la nota menor del
* array.

\*/

public class mainWindow extends JFrame implements ActionListener {

// Un contenedor de elementos gráficos private Container contenedor; // Etiquetas estáticas de cada nota private JLabel nota1, nota2, nota3, nota4, nota5, promedio, desviación, mayor, menor; // Campos de ingreso de cada nota private JTextField campoNota1, campoNota2, campoNota3, campoNota4, campoNota5;

// Botones para realizar cálculos y para borrar las notas private JButton calcular, limpiar;

/\*\*

\* Constructor de la clase VentanaPrincipal

\*/

public mainWindow(){

inicio();

setTitle("Notas"); // Establece el título de la ventana setSize(280,380); // Establece el tamaño de la ventana setLocationRelativeTo(null); /\* La ventana se posiciona en el centro de la pantalla \*/

// Establece que el botón de cerrar permitirá salir de la aplicación setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE); setResizable(false); /\* Establece que el tamaño de la ventana no se puede cambiar \*/

}

/\*\*

* Método que crea la ventana con sus diferentes componentes
* gráficos

\*/

private void inicio() { contenedor = getContentPane(); /\* Obtiene el panel de contenidos de la ventana \*/ contenedor.setLayout(null); /\* Establece que el contenedor no

tiene un layout \*/

// Establece la etiqueta y el campo de texto de la nota 1 nota1 = new JLabel(); nota1.setText("Nota 1:"); nota1.setBounds(20, 20, 135, 23); /\* Establece la posición de la etiqueta nota 1 \*/ campoNota1 = new JTextField();

// Establece la posición del campo de texto de la nota 1 campoNota1.setBounds(105, 20, 135, 23);

// Establece la etiqueta y el campo de texto de la nota 2 nota2 = new JLabel(); nota2.setText("Nota 2:"); nota2.setBounds(20, 50, 135, 23); /\* Establece la posición de la etiqueta nota 2 \*/ campoNota2 = new JTextField();

// Establece la posición del campo de texto de la nota 2 campoNota2.setBounds(105, 50, 135, 23);

// Establece la etiqueta y el campo de texto de la nota 3 nota3 = new JLabel(); nota3.setText("Nota 3:"); nota3.setBounds(20, 80, 135, 23); /\* Establece la posición de la etiqueta nota 3\*/

campoNota3 = new JTextField();

// Establece la posición del campo de texto de la nota 3 campoNota3.setBounds(105, 80, 135, 23);

// Establece la etiqueta y el campo de texto de la nota 4 nota4 = new JLabel(); nota4.setText("Nota 4:"); nota4.setBounds(20, 110, 135, 23); /\* Establece la posición de la

etiqueta nota 4 \*/ campoNota4 = new JTextField();

// Establece la posición del campo de texto de la nota 4 campoNota4.setBounds(105, 110, 135, 23); // Establece la etiqueta y el campo de texto de la nota 5 nota5 = new JLabel(); nota5.setText("Nota 5:"); nota5.setBounds(20, 140, 135, 23); /\* Establece la posición de la etiqueta nota 5 \*/ campoNota5 = new JTextField();

// Establece la posición del campo de texto de la nota 5 campoNota5.setBounds(105, 140, 135, 23);

// Establece el botón Calcular calcular = new JButton(); calcular.setText("Calcular"); calcular.setBounds(20, 170, 100, 23); /\* Establece la posición del botón Calcular \*/

/\* Agrega al botón un ActionListener para que gestione eventos del botón \*/ calcular.addActionListener(this); // Establece el botón Limpiar limpiar = new JButton();

limpiar.setText("Limpiar"); limpiar.setBounds(125, 170, 80, 23); /\* Establece la posición del botón Limpiar \*/

/\* Agrega al botón un ActionListener para que gestione eventos del botón \*/ limpiar.addActionListener(this);

// Establece la etiqueta del promedio con su valor numérico

promedio = new JLabel(); promedio.setText("Promedio = "); promedio.setBounds(20, 210, 135, 23); /\* Establece la posición de la etiqueta promedio \*/

// Establece la etiqueta de la desviación con su valor numérico desviación = new JLabel(); desviación.setText("Desviacion = "); desviación.setBounds(20, 240, 200, 23); /\* Establece la posición de la etiqueta desviación \*/

// Establece la etiqueta de la nota mayor con su valor numérico mayor = new JLabel(); mayor.setText("Nota mayor = "); mayor.setBounds(20, 270, 120, 23); /\* Establece la posición de la etiqueta nota mayor \*/

// Establece la etiqueta de la nota menor con su valor numérico menor = new JLabel(); menor.setText("Nota menor = "); menor.setBounds(20, 300, 120, 23); /\* Establece la posición de la etiquete nota menor \*/

// Se añade cada componente gráfico al contenedor de la ventana contenedor.add(nota1); contenedor.add(campoNota1);

contenedor.add(nota2); contenedor.add(campoNota2); contenedor.add(nota3); contenedor.add(campoNota3); contenedor.add(nota4); contenedor.add(campoNota4); contenedor.add(nota5);

contenedor.add(campoNota5);

contenedor.add(calcular); contenedor.add(limpiar); contenedor.add(promedio); contenedor.add(desviación); contenedor.add(mayor); contenedor.add(menor);

}

/\*\*

\* Método que gestiona los eventos generados en la ventana principal

\*/

public void actionPerformed(ActionEvent evento) { if (evento.getSource() == calcular) { /\* Si se pulsa el botón Calcular \*/ notas nota = new notas(); // Se crea un objeto Notas // Se obtiene y convierte el valor numérico de la nota 1 nota.listaNotas[0] = Double.parseDouble(campoNota1.getText()); // Se obtiene y convierte el valor numérico de la nota 2 nota.listaNotas[1] = Double.parseDouble(campoNota2.getText()); // Se obtiene y convierte el valor numérico de la nota 3 nota.listaNotas[2] = Double.parseDouble(campoNota3.getText()); // Se obtiene y convierte el valor numérico de la nota 4 nota.listaNotas[3] = Double.parseDouble(campoNota4.getText()); // Se obtiene y convierte el valor numérico de la nota 5 nota.listaNotas[4] = Double.parseDouble(campoNota5.getText()); nota.calcularPromedio(); // Se calcula el promedio nota.calcularDesviacion(); // Se calcula la desviación

// Se muestra el promedio formateado

promedio.setText("Promedio = " + String.valueOf(String.format("%.2f", nota.calcularPromedio()))); double desv = nota.calcularDesviacion(); // Se muestra la desviación formateada desviación.setText("Desviación estándar = " + String.format("%.2f", desv));

// Se muestra el valor mayor formateado mayor.setText("Valor mayor = " + String.valueOf(nota.calcularMayor()));

// Se muestra el valor menor formateado menor.setText("Valor menor = " + String.valueOf(nota.calcularMenor()));

}

/\* Se se pulsa el botón Limpiar se dejan en blanco los campos de

notas \*/ if (evento.getSource() == limpiar) { campoNota1.setText(""); campoNota2.setText(""); campoNota3.setText(""); campoNota4.setText(""); campoNota5.setText("");

}

}

}

**Notas** package Notas;

public class notas { double[] listaNotas;

public notas() {

listaNotas = new double[5]; // Crea un array de 5 notas

}

double calcularPromedio() { double suma = 0; for(int i=1; i < listaNotas.length; i++) { // Se recorre el array suma = suma + listaNotas[i]; // Suma las notas del array

}

return (suma / listaNotas.length);

}

double calcularDesviacion() { double prom = calcularPromedio(); double suma = 0; for(int i=0; i < listaNotas.length; i++) { suma += Math.pow(listaNotas[i] - prom, 2 );

}

return Math.sqrt (suma/listaNotas.length ); /\* Retorna el cálculo final de la desviación \*/

}

double calcularMenor() { double menor = listaNotas[0]; for(int i=0; i < listaNotas.length; i++) { if (listaNotas[i] < menor) { menor = listaNotas[i];

}

}

return menor;

}

double calcularMayor() { double mayor = listaNotas[0]; for(int i=0; i < listaNotas.length; i++) { // Se recorre el array if (listaNotas[i] > mayor) { mayor = listaNotas[i];

}

}

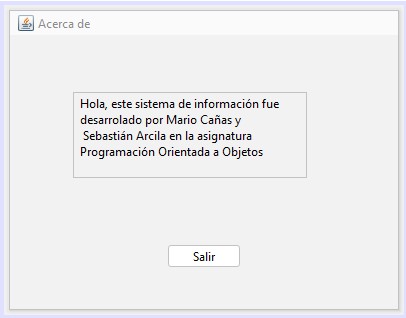
return mayor;

}

}

# SEGUIMIENTO5

## Acercade



**Corriente** package seguimiento5;

public class Corriente {

public static double calcular\_corriente(double voltaje, double resistencia){ double corriente; corriente = voltaje / resistencia; return corriente;

}

}

**Pitágoras** package seguimiento5;

public class Pitagoras { public static double calcular\_hipotenusa(double a, double b){ double hipotenusa; hipotenusa = Math.sqrt(Math.pow(a,2) + Math.pow(b,2));

return hipotenusa;

}

}

**Seguimiento** package seguimiento5; import javax.swing.JFrame;

public class Seguimiento5 {

/\*\*

\* @param args the command line arguments

\*/

public static void main(String[] args) { // TODO code application logic here mainForm mfrm = new mainForm(); mfrm.setExtendedState(JFrame.MAXIMIZED\_BOTH);

mfrm.setVisible(true);

}

}

**Voltaje** package seguimiento5;

public class Voltaje {

public static double calcular\_voltaje(double corriente, double resistencia){ double voltaje; voltaje = corriente + resistencia; return voltaje;

}

}

**formCorriente** package seguimiento5;

/\*\*

\*

* @author Mario Cañas

\*/

public class formCorriente extends javax.swing.JInternalFrame {

/\*\*

* Creates new form formCorriente

\*/

public formCorriente() { initComponents();

}

/\*\*

* 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"> private void initComponents() {

jLabel1 = new javax.swing.JLabel(); jLabel2 = new javax.swing.JLabel(); jLabel3 = new javax.swing.JLabel(); btnCalcular = new javax.swing.JButton(); btnClear = new javax.swing.JButton(); btnExit = new javax.swing.JButton(); txtV = new javax.swing.JTextField(); txtR = new javax.swing.JTextField(); lblCorriente = new javax.swing.JLabel();

setTitle("Calcular Corriente");

jLabel1.setText("Voltaje");

jLabel2.setText("Resistencia");

jLabel3.setText("Corriente");

btnCalcular.setText("Calcular"); btnCalcular.addActionListener(new java.awt.event.ActionListener() { public void actionPerformed(java.awt.event.ActionEvent evt) { btnCalcularActionPerformed(evt);

}

});

btnClear.setText("Borrar"); btnClear.addActionListener(new java.awt.event.ActionListener() { public void actionPerformed(java.awt.event.ActionEvent evt) { btnClearActionPerformed(evt);

}

});

btnExit.setText("Salir"); btnExit.addActionListener(new java.awt.event.ActionListener() { public void actionPerformed(java.awt.event.ActionEvent evt) { btnExitActionPerformed(evt);

}

});

javax.swing.GroupLayout layout = new javax.swing.GroupLayout(getContentPane()); getContentPane().setLayout(layout); layout.setHorizontalGroup( layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(layout.createSequentialGroup()

.addGap(46, 46, 46)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.TRAILING)

.addGroup(layout.createSequentialGroup()

.addComponent(jLabel1)

.addGap(65, 65, 65)

.addComponent(txtV, javax.swing.GroupLayout.PREFERRED\_SIZE, 140, javax.swing.GroupLayout.PREFERRED\_SIZE))

.addGroup(layout.createSequentialGroup()

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addComponent(jLabel2)

.addComponent(jLabel3))

.addGap(44, 44, 44)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addComponent(txtR, javax.swing.GroupLayout.DEFAULT\_SIZE, 140, Short.MAX\_VALUE)

.addComponent(lblCorriente, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, Short.MAX\_VALUE))))

.addGap(106, 106, 106))

.addGroup(layout.createSequentialGroup()

.addGap(31, 31, 31)

.addComponent(btnCalcular)

.addGap(55, 55, 55)

.addComponent(btnClear)

.addGap(45, 45, 45)

.addComponent(btnExit)

.addContainerGap(javax.swing.GroupLayout.DEFAULT\_SIZE, Short.MAX\_VALUE))

);

layout.setVerticalGroup( layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(layout.createSequentialGroup()

.addGap(32, 32, 32)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)

.addComponent(jLabel1)

.addComponent(txtV, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE))

.addGap(28, 28, 28)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)

.addComponent(jLabel2)

.addComponent(txtR, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE))

.addGap(40, 40, 40)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)

.addComponent(jLabel3)

.addComponent(lblCorriente))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED, 56, Short.MAX\_VALUE)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)

.addComponent(btnCalcular)

.addComponent(btnClear)

.addComponent(btnExit))

.addGap(36, 36, 36))

);

pack();

}// </editor-fold>

private void btnClearActionPerformed(java.awt.event.ActionEvent evt) {

// TODO add your handling code here:

double v, r, corriente; v=0;r=0;corriente=0; txtV.setText(""); txtR.setText(""); lblCorriente.setText("");

}

private void btnExitActionPerformed(java.awt.event.ActionEvent evt) {

// TODO add your handling code here:

this.dispose();

}

private void btnCalcularActionPerformed(java.awt.event.ActionEvent evt) {

// TODO add your handling code here: double v, r, corriente; v = Double.parseDouble(txtV.getText()); r = Double.parseDouble(txtR.getText()); corriente = Pitagoras.calcular\_hipotenusa(v, r); lblCorriente.setText(String.valueOf(corriente));

}

// Variables declaration - do not modify

private javax.swing.JButton btnCalcular; private javax.swing.JButton btnClear; private javax.swing.JButton btnExit; private javax.swing.JLabel jLabel1; private javax.swing.JLabel jLabel2; private javax.swing.JLabel jLabel3; private javax.swing.JLabel lblCorriente; private javax.swing.JTextField txtR; private javax.swing.JTextField txtV;

// End of variables declaration

}

**formPitagoras** package seguimiento5;

/\*\*

\*

* @author Mario Cañas

\*/

public class formPitagoras extends javax.swing.JInternalFrame {

/\*\*

* Creates new form formPitagoras

\*/

public formPitagoras() { initComponents();

}

/\*\*

* 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"> private void initComponents() {

jLabel1 = new javax.swing.JLabel(); jLabel2 = new javax.swing.JLabel(); jLabel3 = new javax.swing.JLabel(); btnCalcular = new javax.swing.JButton(); btnClear = new javax.swing.JButton(); btnSalir = new javax.swing.JButton(); txtA = new javax.swing.JTextField(); txtB = new javax.swing.JTextField(); lblHipotenusa = new javax.swing.JLabel();

setTitle("Calcular Hipotenusa");

jLabel1.setText("Lado A");

jLabel2.setText("Lado B");

jLabel3.setText("Hipotenusa");

btnCalcular.setText("Calcular"); btnCalcular.addActionListener(new java.awt.event.ActionListener() { public void actionPerformed(java.awt.event.ActionEvent evt) { btnCalcularActionPerformed(evt);

}

});

btnClear.setText("Borrar");

btnClear.addActionListener(new java.awt.event.ActionListener() { public void actionPerformed(java.awt.event.ActionEvent evt) { btnClearActionPerformed(evt);

}

});

btnSalir.setText("Salir");

btnSalir.addActionListener(new java.awt.event.ActionListener() { public void actionPerformed(java.awt.event.ActionEvent evt) { btnSalirActionPerformed(evt);

}

});

javax.swing.GroupLayout layout = new javax.swing.GroupLayout(getContentPane()); getContentPane().setLayout(layout); layout.setHorizontalGroup( layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(layout.createSequentialGroup()

.addGap(38, 38, 38)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.TRAILING)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING) .addComponent(btnCalcular)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.TRAILING)

.addComponent(jLabel3)

.addComponent(jLabel2, javax.swing.GroupLayout.Alignment.LEADING)))

.addGroup(layout.createSequentialGroup()

.addComponent(jLabel1)

.addGap(39, 39, 39)))

.addGap(43, 43, 43)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(layout.createSequentialGroup()

.addComponent(btnClear)

.addGap(42, 42, 42)

.addComponent(btnSalir))

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.TRAILING, false)

.addComponent(lblHipotenusa, javax.swing.GroupLayout.Alignment.LEADING, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, Short.MAX\_VALUE)

.addComponent(txtB, javax.swing.GroupLayout.Alignment.LEADING, javax.swing.GroupLayout.DEFAULT\_SIZE, 147, Short.MAX\_VALUE)

.addComponent(txtA, javax.swing.GroupLayout.Alignment.LEADING))) .addContainerGap(51, Short.MAX\_VALUE))

);

layout.setVerticalGroup( layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(layout.createSequentialGroup()

.addGap(37, 37, 37)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.TRAILING) .addGroup(layout.createSequentialGroup()

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)

.addComponent(jLabel1)

.addComponent(txtA, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE))

.addGap(49, 49, 49))

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)

.addComponent(txtB, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE) .addComponent(jLabel2)))

.addGap(37, 37, 37)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)

.addComponent(jLabel3)

.addComponent(lblHipotenusa))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED, 51, Short.MAX\_VALUE)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)

.addComponent(btnCalcular)

.addComponent(btnClear)

.addComponent(btnSalir))

.addGap(40, 40, 40))

);

pack();

}// </editor-fold>

private void btnSalirActionPerformed(java.awt.event.ActionEvent evt) {

// TODO add your handling code here:

this.dispose();

}

private void btnClearActionPerformed(java.awt.event.ActionEvent evt) {

// TODO add your handling code here:

double a, b, hipotenusa; a=0;b=0;hipotenusa=0; txtA.setText(""); txtB.setText(""); lblHipotenusa.setText(""); }

private void btnCalcularActionPerformed(java.awt.event.ActionEvent evt) {

// TODO add your handling code here: double a, b, hipotenusa; a = Double.parseDouble(txtA.getText()); b = Double.parseDouble(txtB.getText()); hipotenusa = Pitagoras.calcular\_hipotenusa(a, b); lblHipotenusa.setText(String.valueOf(hipotenusa));

}

// Variables declaration - do not modify private javax.swing.JButton btnCalcular; private javax.swing.JButton btnClear; private javax.swing.JButton btnSalir; private javax.swing.JLabel jLabel1; private javax.swing.JLabel jLabel2; private javax.swing.JLabel jLabel3; private javax.swing.JLabel lblHipotenusa; private javax.swing.JTextField txtA; private javax.swing.JTextField txtB;

// End of variables declaration

}

**formVoltaje** package seguimiento5;

/\*\*

\*

* @author Mario Cañas

\*/

public class formVoltaje extends javax.swing.JInternalFrame {

/\*\*

* Creates new form formVoltaje

\*/

public formVoltaje() { initComponents();

}

/\*\*

* 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"> private void initComponents() {

jLabel1 = new javax.swing.JLabel(); jLabel2 = new javax.swing.JLabel(); jLabel3 = new javax.swing.JLabel(); btnCalcular = new javax.swing.JButton(); btnClear = new javax.swing.JButton(); btnExit = new javax.swing.JButton(); txtC = new javax.swing.JTextField(); txtR = new javax.swing.JTextField(); lblVoltaje = new javax.swing.JLabel(); setTitle("Calcular Voltaje");

jLabel1.setText("Corriente");

jLabel2.setText("Resistencia");

jLabel3.setText("Votlaje");

btnCalcular.setText("Calcular"); btnCalcular.addActionListener(new java.awt.event.ActionListener() { public void actionPerformed(java.awt.event.ActionEvent evt) { btnCalcularActionPerformed(evt);

}

});

btnClear.setText("Borrar"); btnClear.addActionListener(new java.awt.event.ActionListener() { public void actionPerformed(java.awt.event.ActionEvent evt) { btnClearActionPerformed(evt);

}

});

btnExit.setText("Salir"); btnExit.addActionListener(new java.awt.event.ActionListener() { public void actionPerformed(java.awt.event.ActionEvent evt) { btnExitActionPerformed(evt);

}

});

txtC.addActionListener(new java.awt.event.ActionListener() { public void actionPerformed(java.awt.event.ActionEvent evt) { txtCActionPerformed(evt);

}

});

javax.swing.GroupLayout layout = new javax.swing.GroupLayout(getContentPane()); getContentPane().setLayout(layout); layout.setHorizontalGroup( layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(layout.createSequentialGroup()

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(layout.createSequentialGroup()

.addGap(33, 33, 33)

.addComponent(btnCalcular))

.addGroup(layout.createSequentialGroup()

.addGap(49, 49, 49)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addComponent(jLabel2)

.addComponent(jLabel3)

.addComponent(jLabel1))))

.addGap(40, 40, 40)

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

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.TRAILING, false)

.addComponent(lblVoltaje, javax.swing.GroupLayout.Alignment.LEADING, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, Short.MAX\_VALUE)

.addComponent(txtR, javax.swing.GroupLayout.Alignment.LEADING, javax.swing.GroupLayout.DEFAULT\_SIZE, 139, Short.MAX\_VALUE)

.addComponent(txtC, javax.swing.GroupLayout.Alignment.LEADING))

.addContainerGap(javax.swing.GroupLayout.DEFAULT\_SIZE, Short.MAX\_VALUE))

.addGroup(layout.createSequentialGroup()

.addGap(13, 13, 13)

.addComponent(btnClear)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED, 47, Short.MAX\_VALUE)

.addComponent(btnExit)

.addGap(43, 43, 43))))

);

layout.setVerticalGroup( layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(layout.createSequentialGroup()

.addGap(33, 33, 33)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)

.addComponent(jLabel1)

.addComponent(txtC, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE))

.addGap(25, 25, 25)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)

.addComponent(jLabel2)

.addComponent(txtR, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE))

.addGap(32, 32, 32)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)

.addComponent(jLabel3)

.addComponent(lblVoltaje))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED, 58, Short.MAX\_VALUE) .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)

.addComponent(btnCalcular)

.addComponent(btnClear)

.addComponent(btnExit))

.addGap(44, 44, 44))

);

pack();

}// </editor-fold>

private void txtCActionPerformed(java.awt.event.ActionEvent evt) {

// TODO add your handling code here:

}

private void btnExitActionPerformed(java.awt.event.ActionEvent evt) {

// TODO add your handling code here:

this.dispose();

}

private void btnClearActionPerformed(java.awt.event.ActionEvent evt) {

// TODO add your handling code here:

double c, r, voltaje; c=0;r=0;voltaje=0; txtC.setText(""); txtR.setText(""); lblVoltaje.setText("");

}

private void btnCalcularActionPerformed(java.awt.event.ActionEvent evt) { // TODO add your handling code here: double c, r, voltaje;

c = Double.parseDouble(txtC.getText()); r = Double.parseDouble(txtR.getText()); voltaje = Pitagoras.calcular\_hipotenusa(c, r); lblVoltaje.setText(String.valueOf(voltaje));

}

// Variables declaration - do not modify private javax.swing.JButton btnCalcular; private javax.swing.JButton btnClear; private javax.swing.JButton btnExit; private javax.swing.JLabel jLabel1; private javax.swing.JLabel jLabel2; private javax.swing.JLabel jLabel3; private javax.swing.JLabel lblVoltaje; private javax.swing.JTextField txtC; private javax.swing.JTextField txtR;

// End of variables declaration

}

**Mainform** package seguimiento5;

/\*\*

\*

* @author Mario Cañas

\*/

public class mainForm extends javax.swing.JFrame { /\*\*

* Creates new form mainForm

\*/

public mainForm() { initComponents();

}

/\*\*

* 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"> private void initComponents() {

dp = new javax.swing.JDesktopPane(); jMenuBar1 = new javax.swing.JMenuBar(); jMenu1 = new javax.swing.JMenu(); jRadioButtonMenuItem1 = new javax.swing.JRadioButtonMenuItem(); jMenu2 = new javax.swing.JMenu(); jRadioButtonMenuItem2 = new javax.swing.JRadioButtonMenuItem(); jMenu3 = new javax.swing.JMenu(); jRadioButtonMenuItem3 = new javax.swing.JRadioButtonMenuItem(); jRadioButtonMenuItem4 = new javax.swing.JRadioButtonMenuItem(); jMenu4 = new javax.swing.JMenu(); jRadioButtonMenuItem5 = new javax.swing.JRadioButtonMenuItem(); setDefaultCloseOperation(javax.swing.WindowConstants.EXIT\_ON\_CLOSE); setTitle("Universidad Nacional de Colombia");

dp.setBackground(new java.awt.Color(0, 153, 204));

javax.swing.GroupLayout dpLayout = new javax.swing.GroupLayout(dp); dp.setLayout(dpLayout); dpLayout.setHorizontalGroup( dpLayout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGap(0, 400, Short.MAX\_VALUE)

);

dpLayout.setVerticalGroup( dpLayout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGap(0, 277, Short.MAX\_VALUE)

);

jMenu1.setText("Archivo");

jRadioButtonMenuItem1.setSelected(true); jRadioButtonMenuItem1.setText("Salir"); jRadioButtonMenuItem1.addActionListener(new java.awt.event.ActionListener() { public void actionPerformed(java.awt.event.ActionEvent evt) { jRadioButtonMenuItem1ActionPerformed(evt);

}

});

jMenu1.add(jRadioButtonMenuItem1);

jMenuBar1.add(jMenu1);

jMenu2.setText("Pitagoras");

jRadioButtonMenuItem2.setSelected(true); jRadioButtonMenuItem2.setText("Calcular Pitagoras"); jRadioButtonMenuItem2.addActionListener(new java.awt.event.ActionListener() { public void actionPerformed(java.awt.event.ActionEvent evt) { jRadioButtonMenuItem2ActionPerformed(evt);

}

});

jMenu2.add(jRadioButtonMenuItem2);

jMenuBar1.add(jMenu2);

jMenu3.setText("Ohm");

jRadioButtonMenuItem3.setSelected(true); jRadioButtonMenuItem3.setText("Calcular Voltaje"); jRadioButtonMenuItem3.addActionListener(new java.awt.event.ActionListener() { public void actionPerformed(java.awt.event.ActionEvent evt) { jRadioButtonMenuItem3ActionPerformed(evt);

}

});

jMenu3.add(jRadioButtonMenuItem3);

jRadioButtonMenuItem4.setSelected(true); jRadioButtonMenuItem4.setText("Calcular Corriente"); jRadioButtonMenuItem4.addActionListener(new java.awt.event.ActionListener() { public void actionPerformed(java.awt.event.ActionEvent evt) { jRadioButtonMenuItem4ActionPerformed(evt);

}

});

jMenu3.add(jRadioButtonMenuItem4);

jMenuBar1.add(jMenu3);

jMenu4.setText("Acerca");

jRadioButtonMenuItem5.setSelected(true); jRadioButtonMenuItem5.setText("Acerca de"); jRadioButtonMenuItem5.addActionListener(new java.awt.event.ActionListener() { public void actionPerformed(java.awt.event.ActionEvent evt) { jRadioButtonMenuItem5ActionPerformed(evt);

}

});

jMenu4.add(jRadioButtonMenuItem5);

jMenuBar1.add(jMenu4);

setJMenuBar(jMenuBar1);

javax.swing.GroupLayout layout = new javax.swing.GroupLayout(getContentPane()); getContentPane().setLayout(layout); layout.setHorizontalGroup( layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addComponent(dp, javax.swing.GroupLayout.Alignment.TRAILING)

);

layout.setVerticalGroup( layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING) .addComponent(dp, javax.swing.GroupLayout.Alignment.TRAILING)

);

pack();

}// </editor-fold>

private void jRadioButtonMenuItem1ActionPerformed(java.awt.event.ActionEvent evt) { // TODO add your handling code here:

this.dispose();

}

private void jRadioButtonMenuItem4ActionPerformed(java.awt.event.ActionEvent evt) {

// TODO add your handling code here:

formCorriente fcorriente = new formCorriente(); dp.add(fcorriente); fcorriente.setClosable(true); fcorriente.setVisible(true);

}

private void jRadioButtonMenuItem2ActionPerformed(java.awt.event.ActionEvent evt) {

// TODO add your handling code here:

formPitagoras fpitagora = new formPitagoras(); dp.add(fpitagora); fpitagora.setClosable(true); fpitagora.setVisible(true);

}

private void jRadioButtonMenuItem3ActionPerformed(java.awt.event.ActionEvent evt) { // TODO add your handling code here:

formVoltaje fvoltaje = new formVoltaje(); dp.add(fvoltaje); fvoltaje.setClosable(true); fvoltaje.setVisible(true);

}

private void jRadioButtonMenuItem5ActionPerformed(java.awt.event.ActionEvent evt) { // TODO add your handling code here: Acercade acerca = new Acercade(); dp.add(acerca); acerca.setClosable(true); acerca.setVisible(true);

}

/\*\*

* @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(mainForm.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);

} catch (InstantiationException ex) {

java.util.logging.Logger.getLogger(mainForm.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);

} catch (IllegalAccessException ex) {

java.util.logging.Logger.getLogger(mainForm.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);

} catch (javax.swing.UnsupportedLookAndFeelException ex) {

java.util.logging.Logger.getLogger(mainForm.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 mainForm().setVisible(true);

}

});

}

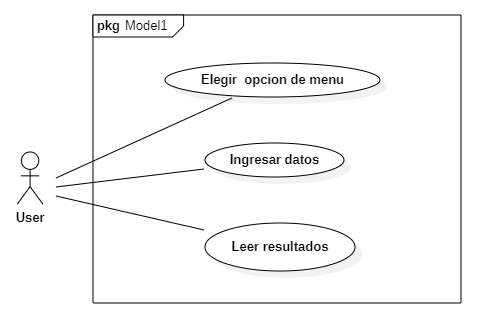
// Variables declaration - do not modify private javax.swing.JDesktopPane dp; private javax.swing.JMenu jMenu1;

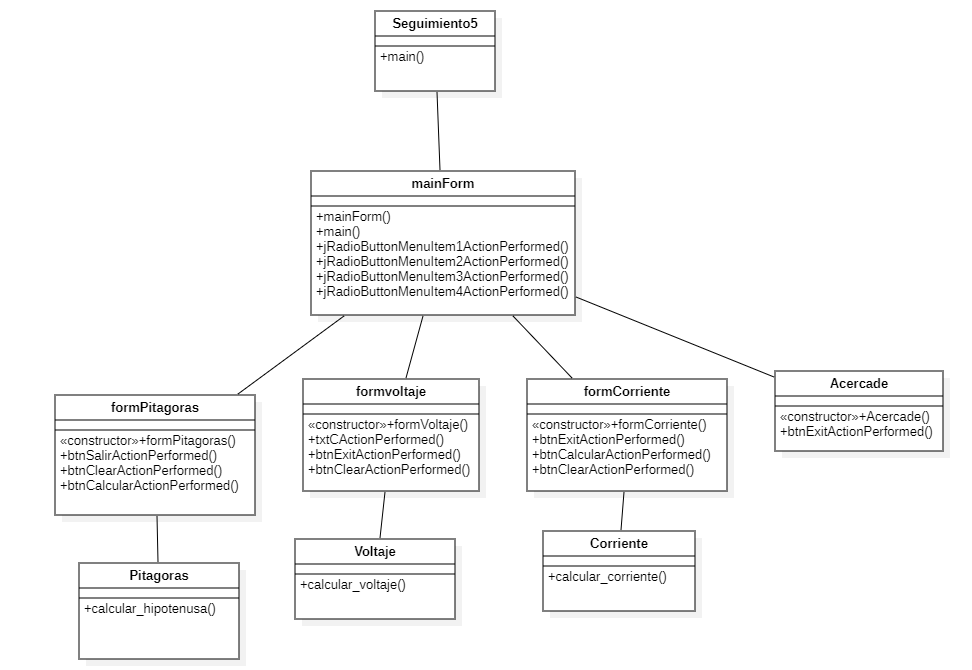
private javax.swing.JMenu jMenu2; private javax.swing.JMenu jMenu3; private javax.swing.JMenu jMenu4; private javax.swing.JMenuBar jMenuBar1; private javax.swing.JRadioButtonMenuItem jRadioButtonMenuItem1; private javax.swing.JRadioButtonMenuItem jRadioButtonMenuItem2; private javax.swing.JRadioButtonMenuItem jRadioButtonMenuItem3; private javax.swing.JRadioButtonMenuItem jRadioButtonMenuItem4; private javax.swing.JRadioButtonMenuItem jRadioButtonMenuItem5;

// End of variables declaration

}

EJERCICIO DE VIDEO DE CLASE

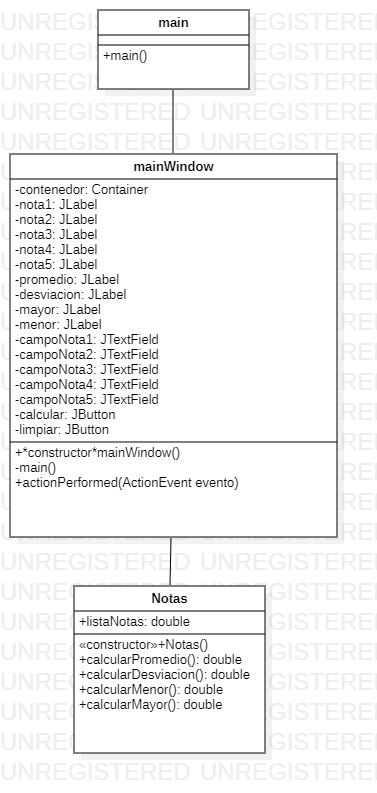




EJERCICIO NOTAS

Diagrama

Descripción generada automáticamente



EJERCICIO FIGURAS

Diagrama

Descripción generada automáticamente

Diagrama

Descripción generada automáticamente

Github: <https://github.com/MarioCa20/Seguimiento5>