

**Método del Triangulo Para la Suma de Vectores**

(Programa en java)

Jonathan Romero Leon 23SIC009

Física

Ing. Vanesa Tenopala Zavala

29/02/2024

Índice:

[Introducción. 3](#_Toc160051013)

[Suma de vectores “Método del triángulo”. 4](#_Toc160051014)

[Descripción del programa. 4](#_Toc160051015)

[Lenguaje y entorno de programación. 4](#_Toc160051016)

[Código Java. 4](#_Toc160051017)

[Capturas de ejecución. 4](#_Toc160051018)

[Conclusión. 5](#_Toc160051019)

[Referencias bibliográficas. 6](#_Toc160051020)

# Introducción.

El método del triángulo para la suma de vectores es una técnica geométrica que permite encontrar la resultante de dos o más vectores. Este método se basa en la ley del paralelogramo, que establece que la suma vectorial de dos vectores se puede representar por la diagonal de un paralelogramo construido a partir de esos dos vectores.

Sin embargo, en situaciones donde solo se tienen dos vectores, el método del triángulo simplifica este proceso al considerar únicamente un triángulo. Este triángulo está formado por los dos vectores dados y la resultante, siendo uno de los vectores la hipotenusa y los otros dos lados del triángulo.

La aplicación del método del triángulo para la suma de vectores implica lo siguiente:

Representar gráficamente los vectores dados con una escala adecuada en un plano cartesiano. Los vectores se dibujan como segmentos de flecha con una dirección y magnitud específicas.

Colocar la punta de un vector al origen del otro vector, formando así un triángulo con ambos vectores como lados. La resultante es la línea que conecta el origen del primer vector con la punta del segundo vector.

Medir las magnitudes de los dos vectores y los ángulos entre ellos utilizando instrumentos adecuados o técnicas trigonométricas.

Aplicar la ley de los cosenos o la ley de senos, dependiendo de la información disponible, para encontrar la magnitud y dirección de la resultante. La magnitud se determina mediante la fórmula de la suma de los cuadrados de las magnitudes de los dos vectores y el doble del producto de sus magnitudes y el coseno del ángulo entre ellos.

El método del triángulo para la suma de vectores es una herramienta visual y geométrica que facilita la comprensión de las relaciones entre vectores y puede ser útil en la resolución de problemas prácticos que involucren movimientos o fuerzas concurrentes.

# Suma de vectores “Método del triángulo”.

## Descripción del programa.

Se plantea realizar un programa para la suma de vectores mediante el método del triangulo, dicho programa contara con interfaz grafica de usuario (GUI) para una mejor visualización por parte del usuario, dicha interfaz contara con paneles y graficas para el uso y trazado del método del triángulo.

El programa utilizará una paquetería adecuada la cual permita el trazado y medición de ángulos, además de que se pondrá a prueba con las medidas vistas en clase, para asegurar su correcto funcionamiento.

## Lenguaje y entorno de programación.

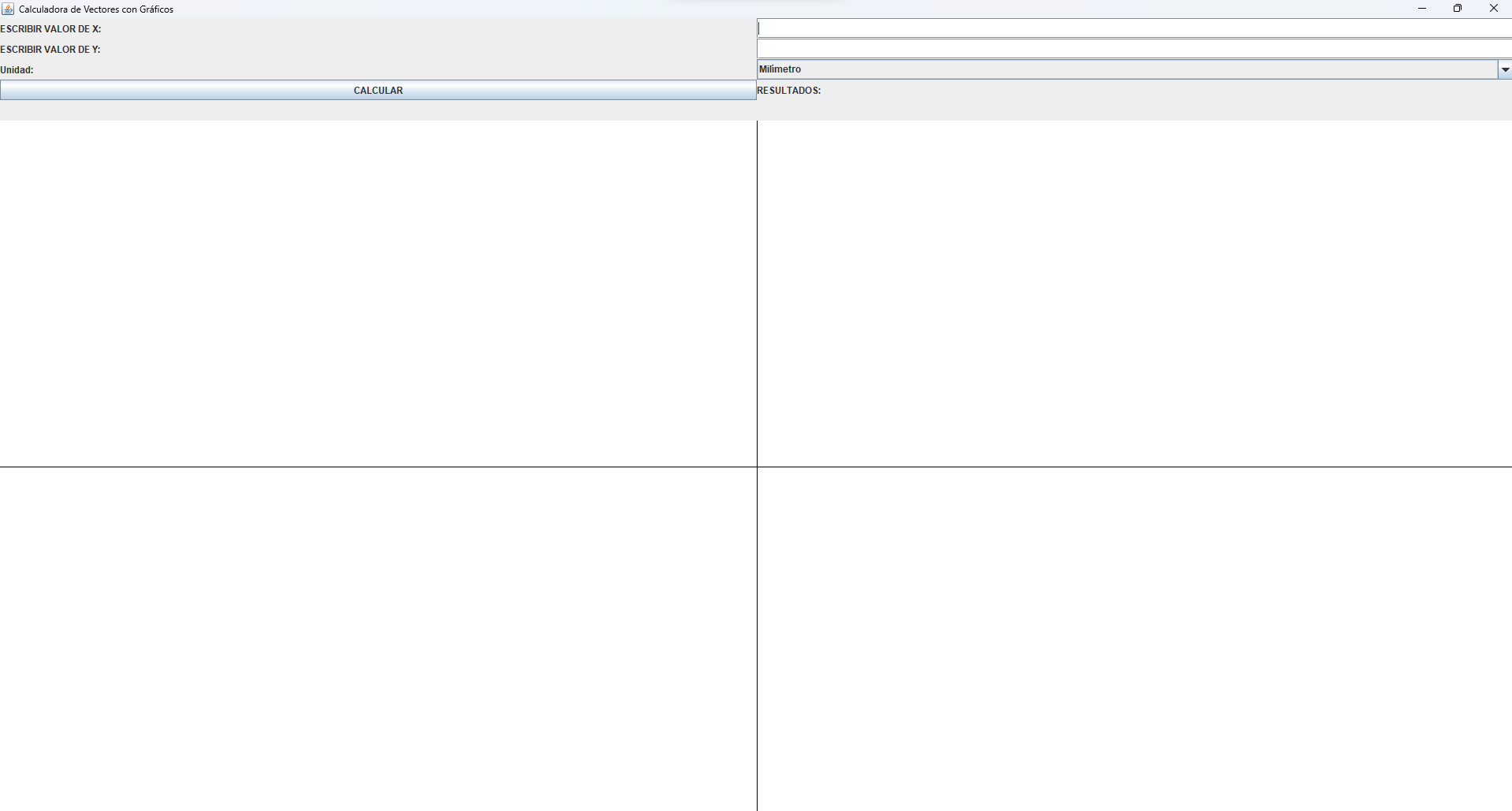
El programa se realizó en lenguaje Java el cual cuenta con una amplia versatilidad para este tipo de operaciones y ejecuciones.

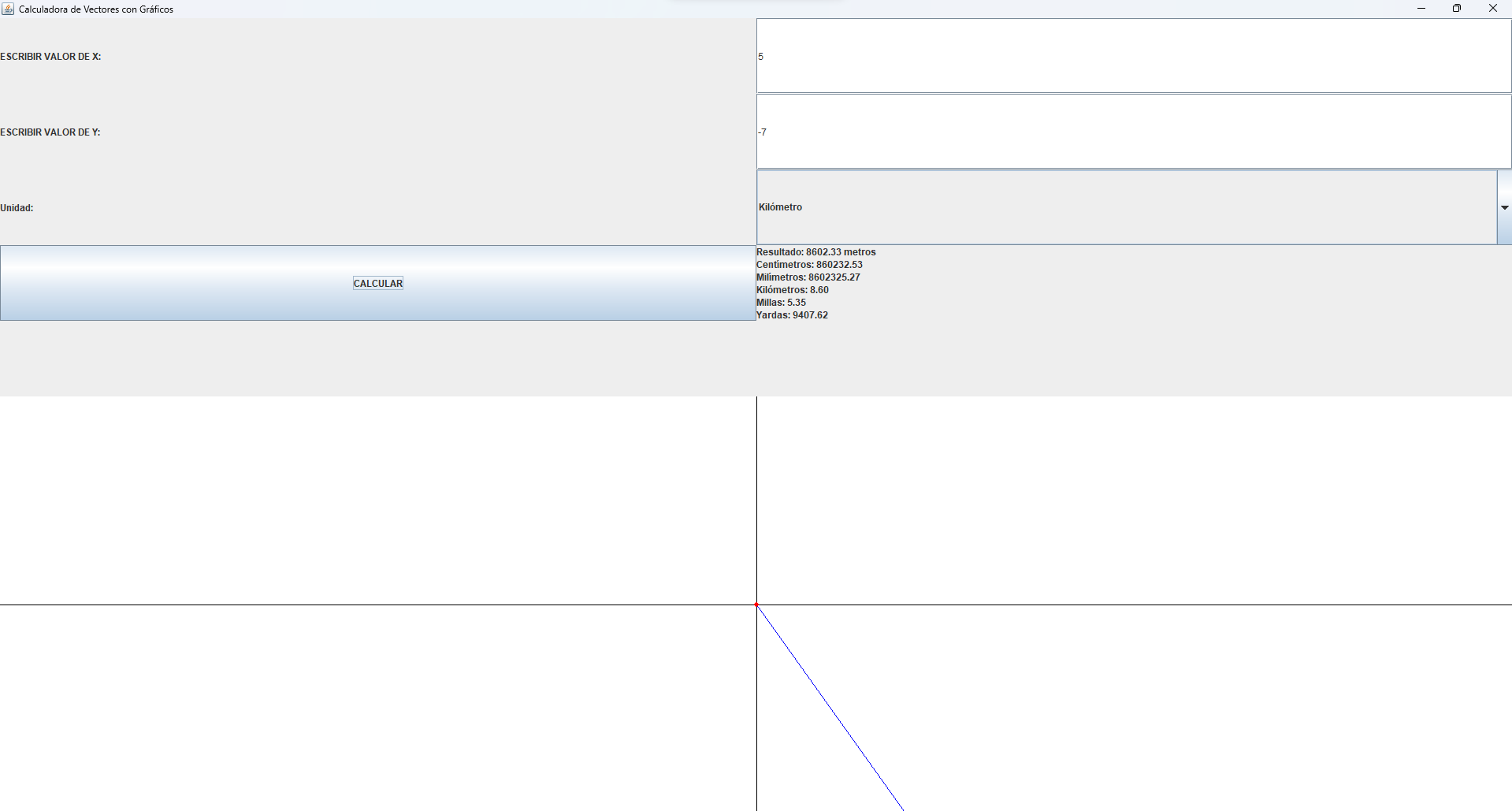
Al ser en lenguaje Java se eligió el programa JGrasp, el cual ofrece facilidades en la creación de programas, en conjunto con el JDK de Java, el cual contiene todas las librerías de Java, y simplemente se mandan a llamar en el código.

## Código Java.

import javax.swing.\*;  
import java.awt.\*;  
import java.awt.event.\*;  
  
//clase principal  
public class CalculadoraVectoresGrafica extends JFrame implements ActionListener {  
 private JLabel lblCantidad1, lblCantidad2, lblResultado;  
 private JTextField txtCantidad1, txtCantidad2;  
 private JButton btnCalcular;  
 private JComboBox<String> cbUnidad;  
 private JPanel panelGrafico;  
  
 public CalculadoraVectoresGrafica() {  
 setTitle("Calculadora de Vectores con Gráficos");  
 setSize(800, 600); // Tamaño más grande  
 setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);  
 setLocationRelativeTo(null);  
  
 // Componentes del jpanel  
 JPanel panel = new JPanel();  
 panel.setLayout(new GridLayout(5, 2));  
   
 //Etiquetas  
 lblCantidad1 = new JLabel("ESCRIBIR VALOR DE X:");  
 txtCantidad1 = new JTextField();  
 panel.add(lblCantidad1);  
 panel.add(txtCantidad1);  
  
 lblCantidad2 = new JLabel("ESCRIBIR VALOR DE Y:");  
 txtCantidad2 = new JTextField();  
 panel.add(lblCantidad2);  
 panel.add(txtCantidad2);  
   
 //lista desplegbles  
 String[] unidades = {"Milímetro", "Centímetro", "Metro", "Kilómetro", "Milla", "Yarda"};  
 cbUnidad = new JComboBox<>(unidades);  
 panel.add(new JLabel("Unidad:"));  
 panel.add(cbUnidad);  
   
 //boton  
  
 btnCalcular = new JButton("CALCULAR");  
 btnCalcular.addActionListener(this);  
 panel.add(btnCalcular);  
   
 //etiqueta de reesultado  
  
 lblResultado = new JLabel("RESULTADOS:");  
 panel.add(lblResultado);  
  
 // Panel para dibujar gráfico  
 panelGrafico = new JPanel() {  
 @Override  
 protected void paintComponent(Graphics g) {  
 super.paintComponent(g);  
 dibujarPlanoCartesiano(g);  
 }  
 };  
 panelGrafico.setBackground(Color.white);  
  
 // Agregar el panel al centro del BorderLayout  
 getContentPane().add(panel, BorderLayout.NORTH);  
 getContentPane().add(panelGrafico, BorderLayout.CENTER); // Agregado al centro  
  
 // Centrar la ventana  
 setLocationRelativeTo(null);  
 }  
  
 @Override  
 public void actionPerformed(ActionEvent e) {  
 if (e.getSource() == btnCalcular) {  
 // Obtener cantidades y unidad  
 double cantidad1 = Double.parseDouble(txtCantidad1.getText());  
 double cantidad2 = Double.parseDouble(txtCantidad2.getText());  
 String unidad = (String) cbUnidad.getSelectedItem();  
  
 // Convertir a metros  
 double cantidad1Metros = convertirAMetros(cantidad1, unidad);  
 double cantidad2Metros = convertirAMetros(cantidad2, unidad);  
  
 // Calcular resultado  
 double resultado = Math.sqrt(Math.pow(cantidad1Metros, 2) + Math.pow(cantidad2Metros, 2));  
  
 // Mostrar resultado  
 String resultadoMetros = String.format("%.2f", resultado) + " metros";  
 String resultadoCentimetros = convertirDeMetros(resultado, "Centímetro");  
 String resultadoMilimetros = convertirDeMetros(resultado, "Milímetro");  
 String resultadoKilometros = convertirDeMetros(resultado, "Kilómetro");  
 String resultadoMillas = convertirDeMetros(resultado, "Milla");  
 String resultadoYardas = convertirDeMetros(resultado, "Yarda");  
  
 lblResultado.setText("<html>Resultado: " + resultadoMetros + "<br>" +  
 "Centímetros: " + resultadoCentimetros + "<br>" +  
 "Milímetros: " + resultadoMilimetros + "<br>" +  
 "Kilómetros: " + resultadoKilometros + "<br>" +  
 "Millas: " + resultadoMillas + "<br>" +  
 "Yardas: " + resultadoYardas + "</html>");  
  
 // Dibujar el triángulo en el plano cartesiano  
 Graphics g = panelGrafico.getGraphics();  
 dibujarTriangulo(g, cantidad1Metros, cantidad2Metros);  
 }  
 }  
   
 //conversionew  
  
 private double convertirAMetros(double cantidad, String unidad) {  
 switch (unidad) {  
 case "Milímetro":  
 return cantidad / 1000.0;  
 case "Centímetro":  
 return cantidad / 100.0;  
 case "Metro":  
 return cantidad;  
 case "Kilómetro":  
 return cantidad \* 1000.0;  
 case "Milla":  
 return cantidad \* 1609.34;  
 case "Yarda":  
 return cantidad \* 0.9144;  
 default:  
 return 0.0;  
 }  
 }  
  
 private String convertirDeMetros(double cantidadMetros, String unidad) {  
 switch (unidad) {  
 case "Milímetro":  
 return String.format("%.2f", cantidadMetros \* 1000.0);  
 case "Centímetro":  
 return String.format("%.2f", cantidadMetros \* 100.0);  
 case "Metro":  
 return String.format("%.2f", cantidadMetros);  
 case "Kilómetro":  
 return String.format("%.2f", cantidadMetros / 1000.0);  
 case "Milla":  
 return String.format("%.2f", cantidadMetros / 1609.34);  
 case "Yarda":  
 return String.format("%.2f", cantidadMetros / 0.9144);  
 default:  
 return "Unidad no válida";  
 }  
 }  
  
 private void dibujarPlanoCartesiano(Graphics g) {  
 int width = panelGrafico.getWidth();  
 int height = panelGrafico.getHeight();  
  
 // Dibujar ejes X e Y  
 g.setColor(Color.black);  
 g.drawLine(0, height / 2, width, height / 2); // Eje X  
 g.drawLine(width / 2, 0, width / 2, height); // Eje Y  
 }  
  
 private void dibujarTriangulo(Graphics g, double x, double y) {  
 int width = panelGrafico.getWidth();  
 int height = panelGrafico.getHeight();  
  
 // Escalar para que quepa en el plano  
 int escala = 50;  
  
 // Dibujar el vector  
 g.setColor(Color.blue);  
 g.drawLine(width / 2, height / 2, width / 2 + (int)(x \* escala), height / 2 - (int)(y \* escala)); // Dibujar vector  
  
 // Dibujar los vértices  
 g.setColor(Color.red);  
 g.fillOval(width / 2 - 3, height / 2 - 3, 6, 6); // Origen  
 g.fillOval(width / 2 + (int)(x \* escala) - 3, height / 2 - (int)(y \* escala) - 3, 6, 6); // Extremo  
 }  
  
 public static void main(String[] args) {  
 SwingUtilities.invokeLater(() -> {  
 CalculadoraVectoresGrafica calc = new CalculadoraVectoresGrafica();  
 calc.setVisible(true);  
 });  
 }  
}

## Capturas de ejecución.





# Conclusión.

En conclusión, el Método del Triángulo para la suma de vectores es una técnica geométrica eficaz y visualmente intuitiva para encontrar la resultante de dos vectores. Su simplicidad y enfoque en la construcción de un triángulo con los vectores dados permiten una representación clara de las magnitudes y direcciones involucradas en la suma vectorial.

A través de este método, es posible visualizar y comprender fácilmente la relación entre los vectores, aprovechando la geometría del triángulo formado. Esto simplifica la resolución de problemas relacionados con fuerzas, desplazamientos u otras magnitudes vectoriales en situaciones bidimensionales.

Sin embargo, es importante destacar que el método del triángulo puede volverse menos práctico a medida que se enfrenta a un mayor número de vectores o situaciones tridimensionales, donde otras técnicas algebraicas, como la descomposición de vectores en componentes rectangulares, pueden resultar más eficientes.

En resumen, el Método del Triángulo es una herramienta valiosa para introducir y comprender la suma de vectores, proporcionando una base visual sólida que facilita el análisis y la resolución de problemas vectoriales en un plano bidimensional.

# Referencias bibliográficas.

El