



Método del Triangulo Para la Suma de Vectores

(Programa en java)

Julio Federico Meléndez Peña 23SIC024

Física

Ing. Vanesa Tenopala Zavala

29/02/2024

Índice

Introducción.....	3
Suma de vectores “Método del triángulo”	4
Descripción del programa.	4
Lenguaje y entorno de programación.	4
Código Java.	4
Capturas de ejecución.....	9
Observaciones del programa.	10
Conclusión.....	11
Referencias bibliográficas.....	12

Índice de imágenes.

Ilustración 1 Ventana emergente para agregar medias	9
Ilustración 2 Medidas insertadas en campos correspondientes	9
Ilustración 3 Resultados del calculo.....	10

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 triángulo, 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.*;

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 Graficos");
        setSize(800, 600); // Tamaño más grande
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        setLocationRelativeTo(null);

        // Componentes de la interfaz
        JPanel panel = new JPanel();
        panel.setLayout(new GridLayout(5, 2));
```

```

        lblCantidad1 = new JLabel("VALOR DE X:");
        txtCantidad1 = new JTextField();
        panel.add(lblCantidad1);
        panel.add(txtCantidad1);

        lblCantidad2 = new JLabel("VALOR DE Y:");
        txtCantidad2 = new JTextField();
        panel.add(lblCantidad2);
        panel.add(txtCantidad2);

        String[] unidades = {"Milimetro", "Centimetro",
"Metro", "Kilometro", "Milla", "Yarda"};
        cbUnidad = new JComboBox<>(unidades);
        panel.add(new JLabel("Unidad:"));
        panel.add(cbUnidad);

        btnCalcular = new JButton("CALCULAR");
        btnCalcular.addActionListener(this);
        panel.add(btnCalcular);

        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, "Centimetro");
        String resultadoMilimetros =
convertirDeMetros(resultado, "Milimetro");
        String resultadoKilometros =
convertirDeMetros(resultado, "Kilometro");
        String resultadoMillas =
convertirDeMetros(resultado, "Milla");
        String resultadoYardas =
convertirDeMetros(resultado, "Yarda");

        lblResultado.setText("<html>Resultado: " +
resultadoMetros + "<br>" +
        "Centimetros: " + resultadoCentimetros +
"<br>" +
        "Milimetros: " + resultadoMilimetros +
"<br>" +
        "Kilometros: " + resultadoKilometros +
"<br>" +
        "Millas: " + resultadoMillas + "<br>" +
        "Yardas: " + resultadoYardas + "</html>");

        // Dibujar el triángulo en el plano cartesiano
Graphics g = panelGrafico.getGraphics();
dibujarTriangulo(g, cantidad1Metros,
cantidad2Metros);

```

```

    }
}

    private double convertirAMetros(double cantidad, String
unidad) {
    switch (unidad) {
        case "Milimetro":
            return cantidad / 1000.0;
        case "Centimetro":
            return cantidad / 100.0;
        case "Metro":
            return cantidad;
        case "Kilometro":
            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 "Milimetro":
            return String.format("%.2f", cantidadMetros *
1000.0);
        case "Centimetro":
            return String.format("%.2f", cantidadMetros *
100.0);
        case "Metro":
            return String.format("%.2f", cantidadMetros);
        case "Kilometro":
            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 = 10;

    // 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

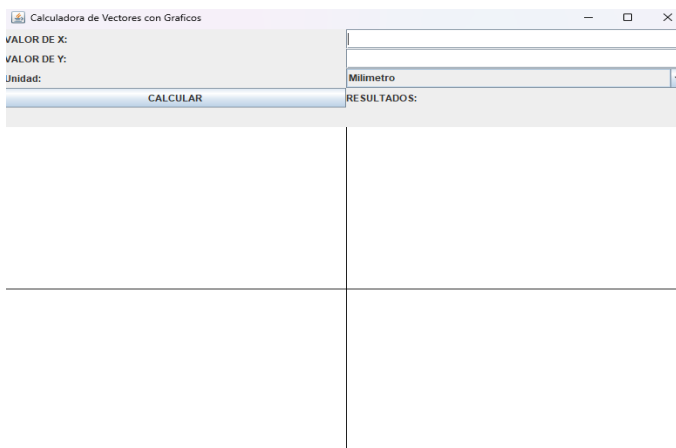
    // Dibujar los vÃ©rtices
    g.setColor(Color.red);
    g.fillOval(width / 2 - 3, height / 2 - 3, 6, 6); //
Origen
    g.fillOval(width / 2 + (int)(x * escala) - 3, height /
2 - (int)(y * escala) - 3, 6, 6); // Extremo
}

public static void main(String[] args) {
    SwingUtilities.invokeLater(() -> {
        CalculadoraVectoresGrafica calc = new
CalculadoraVectoresGrafica();
        calc.setVisible(true);
    });
}
}

```


Capturas de ejecución.

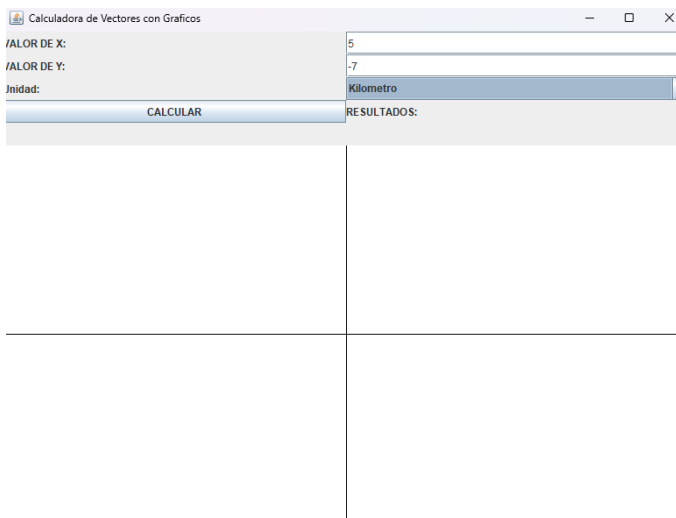
1.- El programa al ejecutarse te arroja una ventana emergente, donde en la parte inferior, te pedirá ingresar las medidas del recorrido, las cuales se manejan en dirección al plano cartesiano con números positivos y negativos.



The screenshot shows a window titled "Calculadora de Vectores con Graficos". It contains input fields for "VALOR DE X:" and "VALOR DE Y:", a unit dropdown menu currently set to "Milimetro", and a "CALCULAR" button. Below these inputs is a large empty area labeled "RESULTADOS:" which serves as a Cartesian coordinate system for plotting.

Ilustración 1 Ventana emergente para agregar medias

2.- Posteriormente se ingresan los valores requeridos en los recuadros de texto asignados por "Vector de x" y "Vector de y", después de eso se establece la unidad de medida deseada, en este caso es en Kilómetros.



The screenshot shows the same window as before, but now the "VALOR DE X:" field contains the number "5" and the "VALOR DE Y:" field contains the number "-7". The unit dropdown menu has been changed to "Kilometro". The "CALCULAR" button and the empty "RESULTADOS:" area remain the same.

Ilustración 2 Medidas insertadas en campos correspondientes

3.- Una vez ingresados los valores se le dará a calcular, este nos arrojará el resultado en todas las medidas relacionadas al kilómetro, en tipo conversión, y en la parte inferior nos arrojará la línea trazada del punto de partida al punto final en base a los datos ingresados.

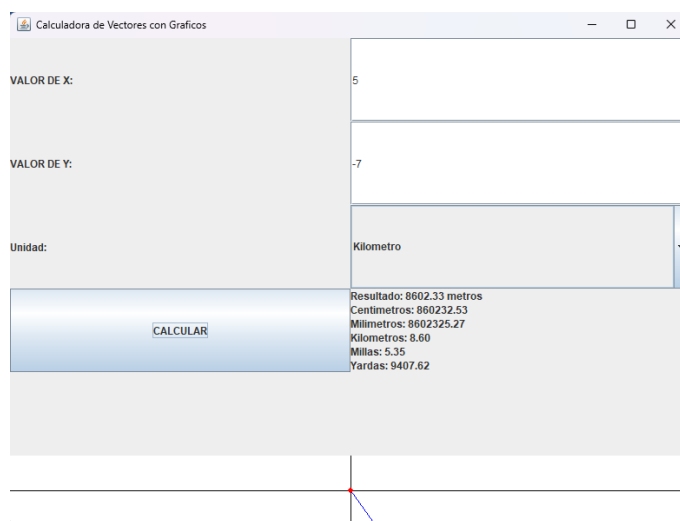


Ilustración 3 Resultados del calculo

Observaciones del programa.

El programa cuenta con algunas deficiencias en su funcionamiento, como el hecho de que el plano cartesiano donde se representa el triángulo es muy pequeño, además de no trazar completamente el triángulo, sino que solo traza la línea resultante.

Otro detalle es el hecho de no contar con un botón el cual borre la línea de cada dato, es decir que las líneas se van creando cada que se ingresan nuevas medidas, haciendo que todas se junten en el mismo plano, y el plano no cuenta con medidas, solo son las líneas que lo trazan.

Tiene margen de mejora, pero no se pudieron implementar de la mejor manera.

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