| | Carátula para entrega de prácticas |
|---|---|
| Facultad de Ingeniería | Laboratorio de docencia |

# Laboratorios de computación salas A y B

*Profesor:* René Adrián Dávila Pérez

*Asignatura:* Programación Orientada a Objetos

*Grupo:* 01

*No. de práctica(s):* 04

*Integrante(s):*
322276824
322258516
425037384
320108116
322221415

*No. de brigada:* 01

*Semestre:* 2026-1

*Fecha de entrega:* 19 de septiembre de 2025

*Observaciones:*

CALIFICACIÓN: _____

# Índice

# 1. Introducción

- **Planteamiento del problema:**
  Diseñar una aplicación en Java capaz de calcular la distancia entre dos puntos en el plano cartesiano a partir de valores recibidos desde los argumentos del método `main`. Además, integrar una interfaz gráfica que permita mostrar los resultados en una ventana emergente por medio de un botón.

- **Motivación:**
  La inclusión de interfaces gráficas en los programas mejora la interacción con el usuario y representa un paso importante en la formación práctica dentro de la programación. En este contexto, la GUI representa un concepto esencial para relacionar la lógica interna del programa y su presentación.

- **Objetivos:**
  Introducir y aplicar los principios de las Interfaces Gráficas de Usuario en Java, creando una aplicación funcional que combine la lógica del programa con una presentación interactiva y afianzar los conceptos de comunicación y colaboración entre distintas clases, mostrando cómo los objetos interactúan para lograr el resultado final.

# 2. Marco Teórico

## Interfaz Gráfica de Usuario (GUI)

Una *Graphical User Interface* es el medio a través del cual un usuario interactúa con un programa mediante elementos visuales como ventanas, botones o menús. De acuerdo con [1] "El objetivo de esta interfaz gráfica es representar el código del backend de un sistema de la forma más clara posible para el usuario para simplificarle las tareas diarias." En Java, una de las bibliotecas más empleadas para este propósito es `Swing`, que forma parte de la *Java Foundation Classes* (JFC) y provee componentes gráficos reutilizables y soporte para la gestión de eventos. [2]

## Distancia entre dos puntos

En el plano cartesiano, la distancia entre dos puntos $A(x_1, y_1)$ y $B(x_2, y_2)$ se define matemáticamente como [5]:

$$d(A, B) = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$$

Este cálculo deriva del teorema de Pitágoras y constituye una de las operaciones más elementales en geometría analítica.

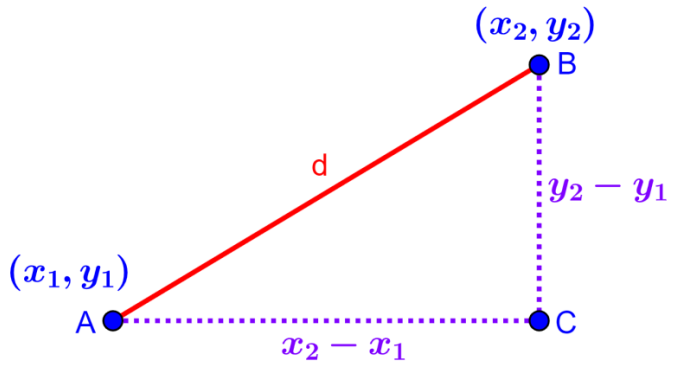


Figura 1: Representación geométrica de la distancia entre dos puntos. [4]

En Java, la clase `Math` proporciona el método `hypot`, que permite obtener directamente el valor de la hipotenusa a partir de las diferencias de coordenadas. [3]

# 3. Desarrollo

La implementación se desarrolló en cuatro clases: `Practica04`, `Punto`, `Mensajes` y `Ventana`. Cada una de ellas cumple una función específica dentro del programa, manteniendo un diseño modular y organizado.

## Clase Punto

La clase `Punto` se diseñó para modelar las coordenadas de un punto en el plano cartesiano. Posee dos atributos enteros, `x` y `y`, que representan las coordenadas.
Cuenta con un constructor con parámetros para inicializar de forma específica las coordenadas al momento de crear el objeto, donde se usa el operador `this` para hacer referencia a los atributos del mismo. Además, redefine el método `toString`, que devuelve un `String` con formato legible del punto, mostrando los valores de `x` y `y`.

## Clase Mensajes

La clase `Mensajes` tiene la función de preparar el texto que será mostrado en la interfaz gráfica. Incluye un método `mensajes`, que recibe como parámetros los dos objetos de tipo `Punto` y la distancia calculada. Este método devuelve una cadena con la información organizada: los valores de los puntos A y B, seguidos de la distancia entre ellos.

## Clase Ventana

La clase `Ventana` extiende de `JFrame`, lo que le permite crear una interfaz gráfica con una ventana emergente. Dentro de la clase se declara un botón de tipo `JButton`,

cuyo propósito es mostrar los resultados al ser presionado.
Cuenta con un constructor que recibe como parámetros los dos objetos `Punto`, la distancia entre ellos y el objeto `Mensajes`, donde se crea la estructura de la ventana mediante los métodos `setTitle` para el título, `setSize` para el tamaño y se establece que el programa termine al presionar el botón de cerrar con `setDefaultCloseOperation`. Posteriormente se inicializa el botón para que muestre un mensaje relativo a su función, mediante la adición de un `ActionListener` al botón, se establece el comportamiento que debe ejecutarse al detectar el clic. En este caso, se invoca el método de la clase `Mensajes` para generar la cadena de salida y se muestra en pantalla a través de un cuadro de diálogo `JOptionPane`. Finalmente se añade el botón.
De esta manera, la clase `Ventana` gestiona la parte gráfica del programa y conecta la interacción del usuario con los cálculos y datos previamente obtenidos.

### Clase Principal

En el método principal `main` se reciben como argumentos las coordenadas de dos puntos en el plano al momento de ejecutar el programa, estos valores se convierten a enteros mediante el método `parseInt` y se guardan en variables en el orden `x1`, `y1`, `x2`, `y2`. La distancia entre ambos puntos se calcula mediante el método `hypot` de la clase `Math`, el cual recibe las diferencias entre coordenadas y devuelve la magnitud de la hipotenusa para guardarla en una variable de tipo `double`. Luego se utilizan las coordenadas para crear dos objetos de tipo `Punto`, que representan a los puntos A y B.
Posteriormente se instancia un objeto de la clase `Mensajes`, que se encargará de generar la cadena de texto con la información a mostrar. Con estos elementos se crea un objeto de la clase `Ventana`, que se inicializa pasándole como parámetros los dos puntos, la distancia calculada y el mensaje. Finalmente, se establece que la ventana sea visible dándole un valor `true` al método `setVisible`, completando así la ejecución del programa.

## 4. Resultados


Figura 2: Ventana principal con el botón para mostrar distancia.

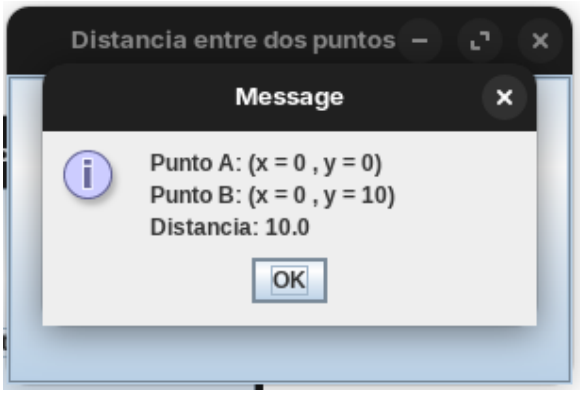


Figura 3: Ventana emergente que muestra los puntos y la distancia calculada.

# 5. Conclusiones

La práctica permitió consolidar conocimientos teóricos mediante la implementación de un programa en Java que combina cálculo matemático y desarrollo de interfaces gráficas.
El uso de Swing hizo posible integrar elementos gráficos, en particular mediante el empleo de un botón que desencadena la visualización de resultados en una ventana emergente. Con ello se reforzó el entendimiento de GUIs y la importancia de ofrecer una interacción clara con el usuario.
Se logró mostrar cómo un problema matemático sencillo puede trasladarse a un programa con una interfaz amigable, destacando la utilidad de combinar programación modular y elementos gráficos en el desarrollo de aplicaciones.

# Referencias

[1] *¿Qué es una interfaz gráfica de usuario (GUI)?* Ene. de 2021. URL: https://www.ionos.mx/digitalguide/paginas-web/desarrollo-web/que-es-una-gui/.

[2] *About the JFC and Swing*. Sep. de 2025. URL: https://docs.oracle.com/javase/tutorial/uiswing/start/about.html.

[3] *Class Math*. Sep. de 2025. URL: https://docs.oracle.com/javase/8/docs/api/java/lang/Math.html.

[4] *Distancia entre dos puntos – Fórmula y ejemplos*. Sep. de 2025. URL: https://www.neurochispas.com/matematicas/distancia-entre-dos-puntos-formula-y-ejemplos/.

[5] Charles H. Lehmann. *Geometría analítica*. Limusa, 1989.