



Carátula para entrega de prácticas

Facultad de Ingeniería

Laboratorios de docencia

Laboratorio de Computación Salas A y B

Profesor(a): René Adrián Dávila Pérez

Asignatura: Programación Orientada a Objetos

Grupo: 7

No de Práctica(s): 4

Integrante(s): 322044339

322094152

322114461

425093384

322150984

No. de brigada: 6

Semestre: 2026-1

Fecha de entrega: 17 de septiembre de 2025

Observaciones:

CALIFICACIÓN: _____

Índice

1. Introducción	2
2. Marco Teórico	2
2.1. Clase	2
2.2. Objeto	3
2.3. Comunicación entre clases y objetos	3
2.4. Paquete Swing	3
3. Desarrollo	3
3.1. Mensajes.java	4
3.2. Ventana.java	4
3.3. Punto.java	4
3.4. Practica4.java	5
4. Resultados	5
4.1. Compilación y ejecución	5
4.2. Ventana emergente	6
4.3. Distancia entre los puntos	6
5. Conclusiones	7
6. Bibliografía	8

1. Introducción

Supongamos que deseamos calcular la distancia entre dos puntos en un mapa. Si lo hacemos de forma visual o con una regla, es probable que tengamos un margen de error elevado, especialmente si los lugares a medir están muy separados. Asimismo, al programar, si calculamos distancias entre puntos de manera manual o poco amigable, puede suceder que se produzcan errores en los cálculos, confusiones con las salidas, un código que no es claro y una interacción escasa con el usuario, lo que al momento de resolver problemas o analizar aplicaciones que requieren de estos cálculos podría causar muchos inconvenientes.

Cuando se crea un programa con una interfaz gráfica que posibilita calcular automáticamente las distancias entre dos puntos y presentarlas de manera clara, no solamente se erradican los errores humanos, sino que además se proporciona una opción más intuitiva y fácil de entender para el usuario. Con este objetivo en mente, utilizar Swing es una opción adecuada para lograr nuestra meta. Esto se debe a que Swing posibilita la creación de ventanas, botones y mensajes emergentes que guían al usuario y presentan los resultados de manera inmediata y sin ambigüedades. Esto hace que la programación esté más cerca de nuestro mundo real, en el que la interactividad y la presentación son tan relevantes como lo es la precisión de los cálculos.

El objetivo de este ejercicio es implementar un programa en Java que determine la distancia entre dos puntos en el plano cartesiano, [2] ingresando las coordenadas correspondientes. Luego, se realizará el cálculo utilizando el método `Math.hypot`. Sin embargo, para permitir que el usuario pueda visualizar los resultados de forma más accesible e interactiva, se incorporará una interfaz gráfica con Swing a través de ventanas emergentes. Se empleará la entrada de datos desde la terminal para obtener las coordenadas, las cuales serán verificadas y procesadas. Después, se exhibirá el resultado al pulsar un botón, de manera que se satisfagan los principios de modularidad, claridad y utilidad que caracterizan a la programación orientada a objetos.

2. Marco Teórico

2.1. Clase

Una clase es una abstracción de un modelo de la realidad que representa en un único elemento un conjunto de objetos, definen el comportamiento y la estructura de los objetos que se van a instanciar a partir de la clase. El ejemplo clásico al momento de tratar de explicar lo que es una clase, es verlo como un molde de galletas y las galletas que se crean a partir del molde serían los objetos. [1]

Entre algunos de los conceptos fundamentales de las clases están:

- **Atributos:** También conocidos como variables de instancia representan las características o propiedades de un objeto.

- Métodos: Estos representan el comportamiento de los objetos, son funciones que pueden realizar acciones o manipular los atributos de los objetos.
- Constructores: Son métodos especiales utilizados para inicializar los objetos cuando estos se crean. Estos métodos especiales llevan el mismo nombre que la clase en la que se encuentran y pueden tener diferentes parámetros para determinar el estado en el que se inicializa el objeto.

2.2. Objeto

Son una abstracción de una parte del mundo real, así como conceptos abstractos con características y comportamientos, cuentan con una estructura interna que combina variables, funciones y estructuras de datos. [4]

2.3. Comunicación entre clases y objetos

Para que los objetos instanciados puedan comunicarse entre sí, debe existir una especie de sistema de paso de mensajes, este sistema se logra por medio de llamadas a métodos de las clases. Un objeto llama a un método mediante la sintaxis "objeto.nombreDelMétodo" para poder interactuar con otros objetos. [5]

2.4. Paquete Swing

Es un paquete que proporciona clases y objetos de interfaz de usuario para configurar la apariencia y funcionamiento de la misma, se caracteriza por ser un paquete ligero y porque el funcionamiento de sus aplicaciones es independiente de la plataforma. [3]

3. Desarrollo

Para esta práctica teníamos como objetivo utilizar Swing Graphical Usual Interface a la cual accedemos por Java Swing para presentar ventanas, botones y mensajes, como entrada recolectar cuatro argumentos del main que usaremos para construir los puntos y como salida mandar una ventana emergente con los puntos y la distancia.

Por lo que para ello creamos cuatro documentos aunque dos fueron dados con anterioridad y solo tuvieron cambios breves, los documentos en cuestión son: Mensajes.java, Ventana.java, Punto.java y Practica4.java. Se abordará uno a uno para una descripción más organizada.

3.1. Mensajes.java

Para este fragmento del código se crea la clase Mensajes en la cuál declaramos una variable pública de tipo double llamada "distanciaPuntos", además de un x1, x2, y1, y2, todos de tipo int y un método público de nombre mensaje y que retornará una cadena en la que se concatena el texto y los valores de las variables x's y y's dándole un formato específico.

Por lo que suponiendo que el usuario ingresará 2, 3, 5, 7, su distancia entre puntos sería igual a 5 y se retornaría un mensaje como este "La distancia entre los puntos (2,5) y (3,7) es: 5"

3.2. Ventana.java

Primero es importante importar las clases necesarias para el manejo de la interfaz gráfica GUI, se importa java.awt.event.* para manejar eventos como clics en botones y javax.swing.* para usar JFrame, JButton, JOptionPane entre otros. Se crea la clase ventana extensión de JFrame y se crea un botón con JButton y un controlador, instancia de Mensajes, después se crea un constructor de la clase ventana que recibe a controlador de la clase Mensajes como parámetro. Para configurar la ventana como tal se utilizan los comandos setTitle para el título el cual fue "Distancia entre puntos", setSize para su tamaño en píxeles siendo de 300X200, setDefaultCloseOperation, este sirve para cuando al cerrar la ventana se termine la operación, y setLocationRelativeTo(null) que centra la ventana. Después se crea un botón y se efectúa comunicación entre el botón y un addActionListener, esto hace que cuando se pulse el botón se ejecuta un código, en este caso llama al método mensaje() del objeto controlador y se muestra una ventana con el mensaje esto gracias a JOptionPane. Por último se agrega el botón a la ventana.

Entonces esta parte del código prácticamente está dedicada a la parte visual en el uso de ventanas y botones por lo que al terminar de ejecutar nos saltará un botón y tras presionarlo nos aparecerá otra ventana con los datos que queremos.

3.3. Punto.java

Esta clase tiene un código bastante pequeño, primero se define la clase Punto con dos atributos enteros e inicializados en cero, después un constructor vacío por si es que no se da un punto, en automático se asume hacia el punto (0,0). Después se realiza otro constructor, pero para cuando se dan valores específicos, esto mediante this.x para x y this.y para y.

Por lo que entonces en esta clase creamos los objetos puntos que podemos manejar con más comodidad, dependiendo de la entrada se crearán un p1(x,y) y si no se recibe

entrada de uno este podrá ser reemplazado con (0,0)

3.4. Practica4.java

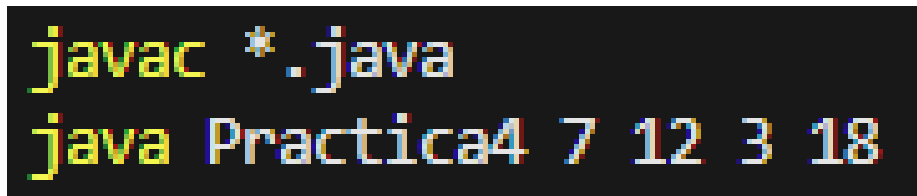
En este código se encuentra el método main por lo que sería el primero en ejecutarse, este método recibe los argumentos de la línea de comandos en el arreglo de "args". Dentro de main se crean dos objetos Punto, p1 y p2, el primero contendrá a los argumentos 0 y 1 tras pasar por un parseInt que los vuelve enteros, el segundo contendrá a los argumentos 2 y 3 que también pasarán por un parseInt. Después se calcula la distancia entre los dos puntos usando Math.hypot(p2.x -p1.x, p2.y -p1.y). Posteriormente se crea un objeto de la clase Mensaje llamado controlador, este tendrá la función de servir como puente entre las clases, habiendo comunicación entre este objeto controlador con el objeto distancia puntos o con "x1", "x2", "y1" y "y2". Por último se crea un objeto ventana que tiene como argumento al controlador y se hace uso de setVisible para que la ventana aparezca.

Entonces tomará cuatro argumentos del usuario tales como 2, 3, 5, 7 por medio de la línea de comandos, los hará pasar por un parseInt y los acomodará en objetos Punto, calculará la distancia entre ellos con hypot y mandará a Mensaje toda la información, además de crear un ventana que podrá ver el usuario.

Para nuestro ejercicio se ejecutó en programa con los siguientes puntos: (7,3) y (12,18)

4. Resultados

4.1. Compilación y ejecución



```
javac *.java
java Practica4 7 12 3 18
```

Figura 1: Ingreso de los valores desde la línea de comandos

Se compilan todos los archivos con formato ".java" y al momento de ejecutarse el programa, ahí mismo se proporcionan los números con los que el programa va trabajar.

4.2. Ventana emergente

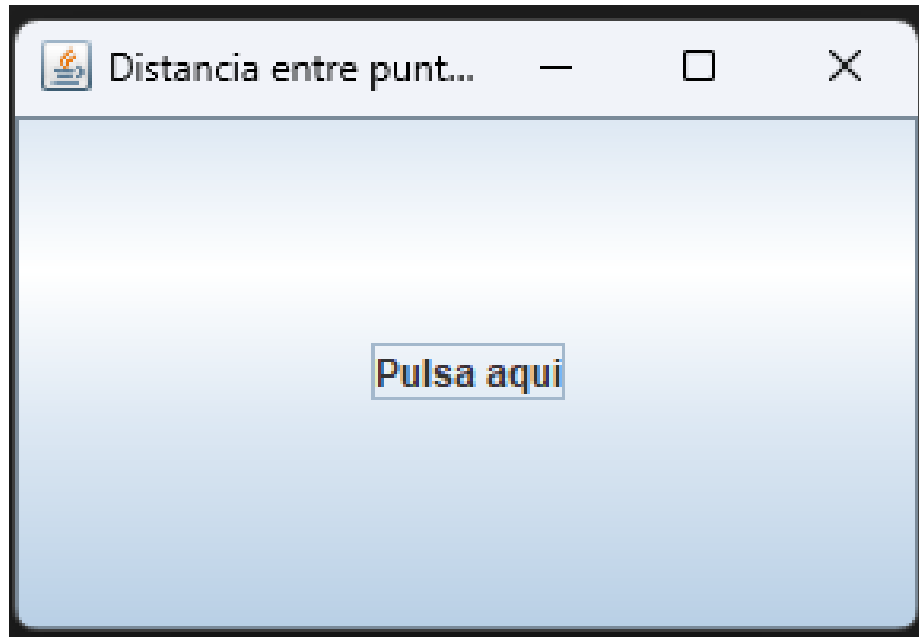


Figura 2: Ventana emergente y botón

Al momento de ejecutarse, el programa muestra automáticamente una ventana emergente y un botón, el cual una vez presionado mostrará un mensaje.

4.3. Distancia entre los puntos

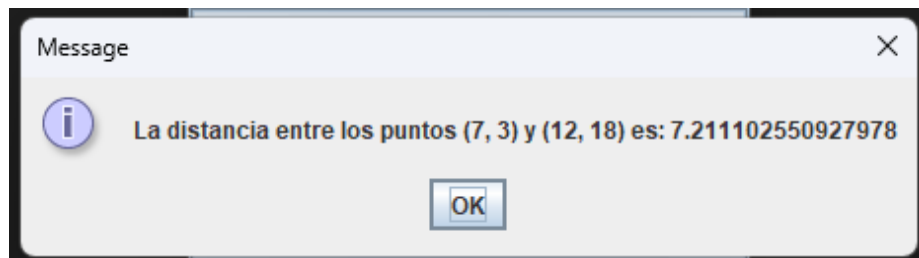


Figura 3: Mensaje con la distancia entre los puntos

Una vez se da clic en el botón, se muestra un mensaje que nos menciona los puntos con los que se trabajó y la distancia entre ellos una vez se realizaron las operaciones internamente.

5. Conclusiones

La práctica permitió comprender la relación entre clases y objetos, así como la interacción entre ellos mediante métodos. La implementación de la interfaz gráfica con el paquete *Swing* mostró cómo los conceptos teóricos de encapsulamiento, comunicación entre instancias y reutilización de código se aplican en programas prácticos con propósitos varios, en nuestro caso, distancia entre dos puntos.

El diseño dividido en múltiples clases (Punto, Mensajes, Ventana y Practica4) ejemplifica la importancia de la separación de responsabilidades, lo que nos brinda una estructura clara que favorece el mantenimiento y la depuración de errores. Asimismo, la práctica nos recordó que es importante tener en mente nuestros fundamentos matemáticos y los recursos de la biblioteca estándar de Java para resolver problemas como el abordado anteriormente.

6. Bibliografía

Referencias

- [1] Jorge López Blasco. *Introducción a POO en Java: Objetos y clases*. Accedido: 14 de septiembre de 2025. Oct. de 2023. URL: <https://openwebinars.net/blog/introduccion-a-poo-en-java-objetos-y-clases/>.
- [2] UNAM Colegio de Ciencias y Humanidades. *Distancia entre dos puntos*. Accedido: 15 de septiembre de 2025. s.f. URL: https://portalacademico.cch.unam.mx/materiales/prof/matdidac/sitpro/mate/mate/mate3/matemaIII/7U2Distancia_entre_dos_puntos.pdf.
- [3] Oracle Corporation. *Package javax.swing (Java Platform SE 8)*. Accedido: 14 de septiembre de 2025. s.f. URL: https://docs-oracle-com.translate.goog/javase/8/docs/api/javax/swing/package-summary.html?_x_tr_sl=en&_x_tr_tl=es&_x_tr_hl=es&_x_tr_pto=tc.
- [4] EBAC. *Objeto en programación: qué es y para qué sirve, características, ejemplos y tipos*. Accedido: 14 de septiembre de 2025. Sep. de 2023. URL: <https://ebac.mx/blog/objeto-en-programacion>.