Tecnológico de Monterrey

Análisis y diseño de algoritmos avanzados

TC2038.603

Actividad : **Actividad 4.1 Implementación Intersección y proximidad aplicando geometría computacional**

Fecha de entrega: 16 de octubre de 2023

Profesor: **Felipe Castillo Rendón**

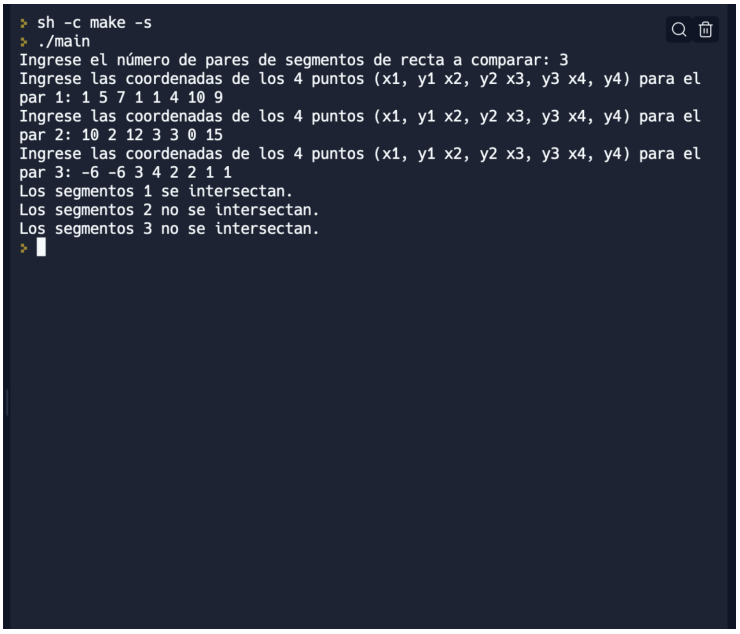
5to Semestre

Nombres	Apellidos	Matrícula
María Fernanda	Argueta Wolke	A00830194

Casos de prueba

Este programa permite ingresar coordenadas de múltiples pares de segmentos de recta en un plano cartesiano y determina si dichos segmentos se intersectan. Utiliza un algoritmo de detección de intersección que considera la orientación de los puntos y verifica si los segmentos se cruzan o son colineales. Tomando como base la geometría computacional.

Prueba 1.
Funcionamiento normal donde se ingresa la n que corresponde a la cantidad de pares de rectas a comparar. Luego se observa cómo se ingresan las coordenadas y el resultado de si se intersectan o no.



Prueba 2.
Cuando se ingresa una letra en vez de un número el programa no lanza errores sino que únicamente se termina automáticamente.

Prueba 3.
Al probar rectas que no se intersectan se observa el resultado correctamente.

Prueba 3.
Al realizar otra prueba con dos rectas que se intersectan se obtiene el resultado correcto.

Complejidad del programa

La complejidad de este programa es lineal en función del número de pares de segmentos a verificar (numPairs), lo que se refleja en el bucle principal del programa. La función crítica, `doSegmentsIntersect`, tiene una complejidad constante (O(1)), ya que el número de operaciones realizadas no depende del número de segmentos, sino de la cantidad fija de cálculos matemáticos necesarios para determinar si dos segmentos se intersectan.

En resumen, la complejidad del programa es O(numPairs) debido al bucle principal y O(1) para la función de verificación de intersección de segmentos. Esto significa que el programa escala de manera eficiente en función del número de segmentos a comparar y es adecuado para la mayoría de los casos prácticos.