

**UNIVERSIDAD DE SONORA**

**Unidad Regional Centro**

**División de Ingeniería**

# **Ingeniería en Sistemas de Información**

**Fundamentos de Computación 3**

**Maestro: IVAN DOSTOYEWSKI MEZA IBARRA**

**Grupo 1 – Hora 5 PM – 6 PM- Edif. 5G – L205**

**Nombre del Alumno:**

* **Vallejo Leyva Marcos**

**Actividad 6: Recorrido de grafo en anchura y profundidad.**

# **Reporte de Código Java - Recorrido de grafo en anchura y profundidad.**

**Índice**

1. **Introducción**
2. **Marco Teórico**
3. **Descripción General del Código**
4. **Métodos Implementados en el Código**
5. **Ejecución y código**
6. **Conclusiones**

**Introducción**

El código implementa un grafo y los algoritmos de recorrido en anchura y profundidad en Java. Los grafos son estructuras de datos fundamentales en ciencias de la computación que modelan relaciones entre objetos. Los algoritmos de recorrido en anchura (BFS) y en profundidad (DFS) son técnicas esenciales para explorar y visitar todos los nodos de un grafo. Este documento proporciona una descripción detallada del código, incluyendo su marco teórico, métodos implementados y ejecución.

**Marco Teórico**

**Grafo:** Un grafo es una estructura matemática que consiste en un conjunto de vértices (también llamados nodos) y un conjunto de aristas (también llamadas bordes) que conectan estos vértices. Los grafos se utilizan ampliamente en ciencias de la computación y en diferentes áreas para modelar una variedad de situaciones y relaciones entre objetos.

Recorridos en Anchura y Profundidad

Los recorridos en anchura y profundidad son dos técnicas fundamentales para explorar y visitar todos los nodos de un grafo.

* Recorrido en Anchura (BFS - Breadth-First Search): En este algoritmo, se visita primero el nodo inicial y luego se exploran todos los nodos vecinos antes de avanzar a los nodos siguientes. Se utiliza una estructura de datos cola (queue) para garantizar que los nodos se visiten en el orden en que se descubrieron.
* Recorrido en Profundidad (DFS - Depth-First Search): En este algoritmo, se sigue un camino desde el nodo inicial hasta que no hay más nodos por visitar en ese camino. Luego, se retrocede al nodo anterior y se elige otro camino no explorado. Se utiliza una estructura de datos pila (stack) para realizar un seguimiento de los nodos visitados y el orden en que se visitaron.

**Descripción General del Código**

El código consta de una clase Graph que representa un grafo y contiene métodos para agregar aristas, así como para realizar recorridos en anchura y profundidad. La clase Edge representa las aristas del grafo. La clase Main contiene el método main para la interacción con el usuario y la ejecución de los recorridos.

**Métodos Implementados en el Código**

El código implementa los siguientes métodos en la clase **Graph**:

* **addEdge(char src, char dest, double weight)**: Este método permite agregar una arista al grafo. Toma como parámetros el vértice de origen (**src**), el vértice de destino (**dest**) y el peso de la arista (**weight**). Crea una nueva arista con estos parámetros y la añade a la lista de adyacencia del vértice de origen.

Texto

Descripción generada automáticamente

* **BFS(char start)**: Este método realiza un recorrido en anchura del grafo comenzando desde el vértice dado como **start**. Utiliza una estructura de datos cola para mantener un seguimiento de los nodos a visitar. Durante el recorrido, visita cada nodo y luego explora todos sus nodos vecinos antes de pasar al siguiente nivel de nodos. Este proceso continúa hasta que se visitan todos los nodos alcanzables desde el nodo inicial.

Texto

Descripción generada automáticamente

* **DFS(char start)**: Este método realiza un recorrido en profundidad del grafo comenzando desde el vértice dado como **start**. Utiliza una estructura de datos pila para mantener un seguimiento de los nodos a visitar. Durante el recorrido, sigue un camino lo más profundo posible antes de retroceder y explorar otros caminos. Este proceso continúa hasta que se visitan todos los nodos alcanzables desde el nodo inicial.

Texto

Descripción generada automáticamente

**Ejecución y Código:**

**Clase Main**

La clase **Main** es la clase principal que contiene el método **main** donde se inicia la ejecución del programa. Esta clase se encarga de interactuar con el usuario a través de un menú y de llamar a los métodos adecuados de la clase **Graph** según la opción seleccionada.

El flujo principal del programa es el siguiente:

1. **Menú Interactivo**: Se muestra un menú que permite al usuario seleccionar entre varias opciones:
   * Crear grafo.
   * Realizar recorrido en anchura.
   * Realizar recorrido en profundidad.
   * Salir del programa.
2. **Selección de Opción**: Dependiendo de la opción seleccionada por el usuario, se ejecuta la acción correspondiente.
3. **Crear Grafo**: Si el usuario elige la opción de crear un grafo, se solicita al usuario ingresar el número de aristas y luego ingresar las aristas en el formato adecuado (**Origen Destino Peso**). Estos datos se utilizan para agregar las aristas al grafo utilizando el método **addEdge** de la clase **Graph**.
4. **Recorrido en Anchura**: Si el usuario elige la opción de realizar un recorrido en anchura, se solicita al usuario ingresar el nodo origen para el recorrido. Luego, se llama al método **BFS** de la clase **Graph** pasando el nodo origen como argumento. El recorrido en anchura se realiza y se muestra por consola.
5. **Recorrido en Profundidad**: Si el usuario elige la opción de realizar un recorrido en profundidad, se solicita al usuario ingresar el nodo origen para el recorrido. Luego, se llama al método **DFS** de la clase **Graph** pasando el nodo origen como argumento. El recorrido en profundidad se realiza y se muestra por consola.
6. **Salir del Programa**: Si el usuario elige la opción de salir del programa, la ejecución termina y el programa se cierra.

La clase **Main** utiliza objetos de la clase **Scanner** para interactuar con el usuario y un objeto de la clase **Graph** para manipular el grafo y realizar los recorridos.

**Código clase Main:**

Texto

Descripción generada automáticamenteTexto

Descripción generada automáticamente

**Ejecución del código:**

Texto

Descripción generada automáticamenteTexto

Descripción generada automáticamente

**Conclusiones:**

Este programa ofrece una implementación básica de un grafo y los algoritmos de recorrido en anchura y profundidad. Estos algoritmos son fundamentales en la teoría de grafos y tienen una amplia gama de aplicaciones en campos como la optimización, la inteligencia artificial y la búsqueda en grafos, entre otros. La comprensión de estos algoritmos es esencial para cualquier desarrollador que trabaje con estructuras de datos y algoritmos en el ámbito de la informática.