Estructuras de Datos

Paralelo 1 – PAO II 2020

Taller y Práctica sobre Grafos

Introducción

En esta práctica, usted implementará algunos de los algoritmos fundamentales del TDA Grafo. Para esto, utilizará el código adjunto, que provee una implementación en Java de los TDAs vértice (`Vertex.java`), arco (`Edge.java`), y grafo. Este último TDA está implementado tanto mediante una matriz de adyacencia (`GraphMA.java`) como mediante listas de adyacencias (`GraphLA.java`).

Antes de empezar con la implementación de los métodos solicitados a continuación, asegúrese de familiarizarse con el código provisto. En particular, revise el programa principal de la clase `Main`, que ilustra la forma de crear un grafo con el TDA adjunto.

Ejercicios de Programación

Los siguientes métodos deben ser incluidos dentro de la clase `GraphLA`, que utiliza listas de adyacencias para abstraer el concepto de grafos. El TDA `GraphMA` es incluido en el código de este practica únicamente con fines didácticos. Queda a criterio de cada estudiante explorarlo en mayor detalle.

Taller (se entrega hoy hasta las 11:00am):

1. Implemente un método que recorra el grafo por anchura.
2. Implemente un método que recorra el grafo por profundidad.
3. Implemente un método que retorne las componentes conexas de un grafo.

Tarea (se entrega hasta el 19 de enero):

4. Implemente un método que retorne las componentes fuertemente conexas de un grafo.
5. Implemente un método que ejecute el algoritmo de Dijkstra para calcular el camino más corto desde un vértice en particular a todos los demás vértices del grafo.
6. Implemente un método que, dados dos contenidos, retorne la longitud del camino más corto entre los nodos que almacenan dichos contenidos dentro del grafo.

Recomendaciones Generales

En esta práctica usted deberá tomar algunas decisiones de diseño respecto al tipo y número de argumentos que deben recibir y/o retornar los métodos que usted implementará. En general, es recomendable que sus métodos funcionen recibiendo **contenidos** que existen dentro del grafo. Por ejemplo, es recomendable que los métodos implementados en los puntos 1 y 2 reciban el contenido del vértice desde donde se desea recorrer el grafo. Su método deberá, por tanto, primero buscar el vértice del grafo que almacena este contenido. Para esto, utilice el método `buscarVertice`. Esta lógica aplica para los métodos requeridos en los puntos 5 y 6.

En los puntos 3 y 4, asegúrese de validar que estos métodos puedan ser ejecutados solo cuando el grafo correspondiente es del tipo apropiado. Por ejemplo, el concepto de “componentes **fuertemente** conexas” no existe a menos que el grafo sea dirigido. Su método debe informar de esto a quien lo use.