UNIDAD TEMÁTICA 7 – GRAFOS DIRIGIDOS

PRACTICOS DOMICILIARIOS INDIVIDUALES - 2

EJERCICIO 1

Ahora que hemos introducido el TDA Grafo Dirigido, y empezamos a conocer las operaciones que el mismo debería soportar, deseamos analizar diferentes formas de diseñar su estructura a efectos de lograr la mayor eficiencia en las operaciones, consumo de memoria razonable, y uso avanzado de las colecciones y clases disponibles en la librería del lenguaje.

Se requiere:

1. Diseñar a alto nivel la arquitectura del TDA Grafo, indicando las operaciones a implementar, las estructuras a usar y el orden del tiempo de ejecución correspondiente. Se sugiere dibujar las estructuras (clases) y relaciones entre las mismas. Evaluar la aplicabilidad de diferentes **colecciones** disponibles en la **API de colecciones de JAVA** para lograr un diseño más eficiente y compacto (es decir, dadas las operaciones que son requeridas, comparar la implementación usando diferentes combinaciones de colecciones disponibles en la API de JAVA – a la luz del consumo de memoria y tiempo de ejecución).En años anteriores hemos desarrollado el TDA Grafo usando sólamente el TDA LISTA del curso. Se facilita el código correspondiente ***sólo como recurso ilustrativo para refinar el diseño*** ("FuentesGrafosconListas").
2. Dada la interfaz provista para el TDA Grafo ("Interfaz básica TDA Grafo"), implementarla de acuerdo a las estructuras seleccionadas en el punto **1**. El **constructor** de la clase TGrafoDirigido deberá recibir como parámetros **dos colecciones**: "**vertices**" y "**aristas**".
3. Un vértice **(TVertice)** tendrá: etiqueta(comparable), visitado(Boolean) y datos (object)
4. Una arista **(TArista)** tendría origen(TVertice), destino(TVertice), costo(int)
5. Una adyacencia **(TAdyacencia)** debería tener destino(TVertice), costo(int)
6. En la clase "UtilGrafos" existente una funcionalidad que permite, a partir de la lista de vértices del grafo, crear una matriz de costos. Será necesario reescribir este método para que funcione con las nuevas estructuras del TDA GrafoDirigido.

Una vez completado el código:

a) Crear una instancia de TGrafoDirigido
b) Insertar los siguientes vértices:

   Artigas, Canelones, Durazno, Florida, Montevideo, Punta del Este y Rocha.

c) Insertar las siguientes aristas:

   Artigas, Rocha, 400; Canelones, Artigas, 500; Canelones, Colonia, 200; Canelones, Durazno, 170; Canelones, Punta del Este, 90; Colonia, Montevideo, 180; Florida, Durazno, 60; Montevideo, Artigas, 700; Montevideo, Canelones, 30; Montevideo, Punta del Este, 130; Punta del Este, Rocha, 90; Rocha, Montevideo, 270; Florida, Durazno, 60

d) Utilizando la clase provista "UtilGrafos", a partir de la instancia del Grafo Dirigido creado, producir la matriz de adyacencias e imprimirla en pantalla. Verificar que es correcta de acuerdo al Grafo planteado.

EJERCICIO 2

- Agregar al TDA Grafo Dirigido un método “floyd()” que devuelva una matriz cuadrada de Comparable. Esta matriz se inicializará en base a la matriz de adyacencias del grafo, y al terminar contendrá los costos de caminos mínimos de acuerdo al algoritmo de Floyd.
- Imprimir el resultado (matriz) de ejecutar el método “floyd()” sobre el grafo dirigido creado en el Ejercicio 1. Verificar los resultados
- Subir un archivo “Floyd.txt” con el código implementado.

EJERCICIO 3

En base al código fuente de TDA Grafo Dirigido, desarrollar:

- Un método para obtener la excentricidad de un cierto vértice del grafo pasado como parámetro (se puede pasar la etiqueta como parámetro) “***obtenerExcentricidad(comparable Etiqueta)”,*** que devuelve un valor entero positivo o **-1** si es infinito.
- Un método para indicar qué vértice es el Centro del Grafo *“**centroDelGrafo()**”* devuelve la etiqueta del vértice centro del grafo.
- Probar estos métodos con el grafo indicado en el **ejercicio 1**
- Subir un archivo **“centro.txt”** con el código implementado (ambos métodos).