Universidad Simón Bolívar
Departamento de Computación y Tecnología
De la Información
CI-2693

INFORME PROYECTO #1

Estudiante:
Pietro Iaia. Carnet: 15-1071
Manuel Guillermo Gil: 14-10397

OBJETIVO:

El objetivo de este proyecto es la familiarización con las operaciones básicas de los Tipos Abstractos de Datos (TADs) Grafo, Grafo No Dirigido y Grafo Dirigido. Para ello se desea que implemente los siguientes TADs usando el lenguaje de programación JAVA y también que desarrolle una aplicación cliente que permita probar los TADs.

Para la implementación se creará una clase abstracta llamada GRAFO que contendrá las operaciones y estructura de datos asociados a un grafo, sea este dirigido o no. Los grafos podrán tener lados múltiples y bucles.
La clase GRAFO tendrá dos clases concretas derivadas: Grafo no Dirigido y Grafo dirigido. Se desea que implemente el TAD GRAFO utilizando una Lista de Adyacencias.

TADS:

TAD Vértice<V>:
El TAD Vértice tiene en su representación un identificador de tipo String, un dato almacenado del tipo especificado por el constructor genérico y un atributo de tipo double que es el peso asociado al vértice.

El primer método llamado CrearVertice, el cual nos sirve como constructor de la clase, recibe id, El tipo genérico V y peso como parámetros y luego inicializa los atributos de la clase como tal.
Luego tenemos tres métodos getters para obtener el peso, id, y V. Finalmente otro método ToString el cual retorna una cadena de caracteres con los datos del objeto vertice.

TAD Lado<E>:
El TAD Lado está formado en su representación por un identificador de tipo String, un dato almacenado del tipo especificado por el constructor genérico y un peso de tipo double. Este TAD debe ser implementado

como una clase abstracta. El TAD Lado tiene dos subtipos el TAD Arco y el TAD Arista.
El método CrearLado, el cual nos sirve como constructor de la clase, recibe id, El tipo genérico E y peso como parámetros y luego inicializa los atributos de la clase como tal.
Luego los getters de los tres atributos arriba mencionados y finalmente un método ToString que devuelve una cadena de caracteres con los datos.

TAD Arco<E>:
Subtipo del TAD Lado<E> que representa a los lados que componen al TAD Grafo Dirigido. Es implementado como una clase concreta derivada de la clase abstracta Lado.

Los atributos definidos en su clase son Vi y Vf que corresponden al vértice inicial y vértice final del Arco.
CrearArco sirve de constructor, tomando los atributos de su clase padre. Luego los métodos getters para obtener los atributos y ToString para devolver una cadena de caracteres con los atributos.

TAD Arista<E>:
Subtipo del TAD Lado<E> que representa a los lados que componen al TAD Grafo No Dirigido. Es implementado como una clase concreta derivada de la clase abstracta Lado.

Los atributos definidos en su clase son V1 y V2 que corresponden a dos vértices extremos de la Arista.
CrearArista sirve de constructor, tomando los atributos de su clase padre. Luego los métodos getters y finalmente el método ToString() para devolver una cadena de caracteres con los atributos.

TAD Grafo<V, E>:
Este TAD contendrá las operaciones asociados a un grafo, sea dirigido o no dirigido. Los grafos podrán tener lados múltiples y bucles. El TAD

Grafo<V, E> debe ser implementado como una interfaz de JAVA llamada Grafo.

Todos los identificadores de los vértices que componen a un grafo deben ser únicos. De la misma manera no debe haber identificadores repetidos de los lados componen a un grafo.

El tipo genérico V especifica el tipo de los datos que contendrán los vértices y E especifica la clase de los datos que contendrán los lados.
Para garantizar el dar cumplimiento de esto, se observa que las Lists son colecciones iterables

TAD Grafo no Dirigido<V, E>:
Este TAD es un subtipo del TAD Grafo. Debe ser implementado como una clase concreta que implementa los métodos de la interfaz Grafo. El tipo de lado con el que está constituido esta representación del TAD Grafo es la Arista<E>

El atributo definido en esta clase es la Lista principal la cual contiene todos los Vertices del grafo ArrayList<Vetice<V>> Grafo.

Se implementa el constructor llamado CrearGrafoNoDirigido, luego los dos métodos AgregarArista uno en el cual toma como atributo la arista ya creada y otro en el cual como atributos se le pasa los atributos de una Arista y este crea la Arista y luego la agrega. EliminarArista que elimina la arista del grafo y ObtenerArista que obtiene una arista pedida por el usuario.

Luego se implementan los métodos de la interfaz Grafo para GrafoNoDirigido.

TAD Grafo Dirigido<V, E>:
Este TAD es un subtipo del TAD Grafo. Es implementado como una clase concreta que implementa los métodos de la interfaz Grafo. El tipo de

lado con el que está constituido esta representación del TAD Grafo es el Arco.

El atributo definido en esta clase es la Lista principal la cual contiene todos los Vértices del grafo ArrayList<Vetice<V>> Grafo.

Se implementa el constructor llamado CrearGrafoDirigido, luego los métodos GradoExterior que retorna el Grado exterior del Vértice dado, GradoInterior que retorna el Grado interior del Vértice dado, Predecesores que retorna una lista con todos los Vértices predecesores al Vertice dado, dos métodos AgregarArco uno en el cual toma como atributo el arco ya creado y otro en el cual como atributos se le pasa los atributos de un Arco y este crea el arco y lo agrega, EliminarArco que elimina el arco del Grafo, y ObtenerArco que obtiene un Arco pedido por el Usuario.

Luego se implementan los métodos de la interfaz Grafo para GrafoDirigido.

Cliente:

En esta clase es donde se implementará la parte grafica del proyecto. Primero empieza con abrir el archivo .txt y lee las primeras 5 lineas del archivo, donde obtiene el tipo de Dato de los Vertices, el tipo de Dato de los lados, si el Grafo es dirigido o no dirigido y el Numero de Lados y de Vertices. Luego de esto crea el Grafo dependiendo de lo leído anteriormente y cada uno (Dirigido o no dirigido) tendrá su propio menú para que el Usuario pueda probar las funciones del grafo.