PONTIFICIA UNIVERSIDAD JAVERIANA
Departamento de Ingeniería de Sistemas
Estructuras de Datos.

Taller 4: Grafos, 2024-10

Evaluación

La entrega se hará a través de la correspondiente asignación de BrightSpace. Se debe entregar un único archivo comprimido, nombrado con el número del grupo correspondiente, seguido del primer apellido de cada integrante del grupo y finalizando con la palabra "T4", estos datos deben ir separados por "_" ejemplo:

"G1_apellido1_apellido2_apellido3_T4.zip"

Este archivo debe contener el documento de diseño y las justificaciones (.pdf) y el código fuente de los programas (.h, .hxx, .cxx, .cpp).

- La inclusión de archivos en cualquier otro tipo de formato diferente a los especificados (como archivos de proyecto de Dev-C++, CodeBlocks, replit, etc) o el envío del archivo comprimido en otro formato diferente a los especificados, resultará en una calificación de 0 (cero) sobre 5 (cinco) para la entrega respectiva. **Los archivos enviados DEBEN poderse compilar y ejecutar SIN errores usando los comandos vistos en el taller 0.**
- **Para calificar el taller se deben cumplir todos los requerimientos solicitados, si no se cumplen, se obtiene una calificación de 0.**

No olvide aplicar correctamente el tema visto de **composición** para hacer un diseño e implementación adecuado.

Problema aerolíneas:

Se le pide implementar un sistema sencillo para una aerolínea en estados unidos con las siguientes funcionalidades iniciales:

1. El camino más rápido.
2. La ruta más corta.
3. El camino más barato.

Datos:

Se le entrega unos archivos en formato csv, los cuales debe leer para insertar toda la información en un grafo.

- Datos de aeropuertos:
    - *Ciudad / código_aeropuerto / latitud / longitud*

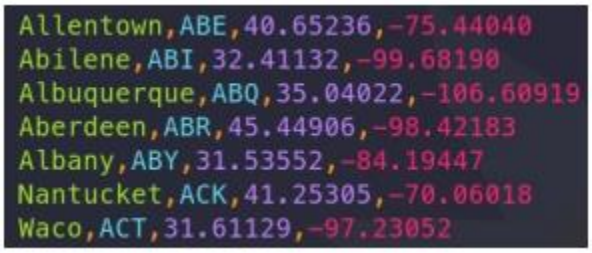

Imagen 1: Aeropuertos

- Datos de vuelos:
    - aeropuerto i / aeropuerto j / tiempo promedio / precio / cant vuelos entre aeropuertos

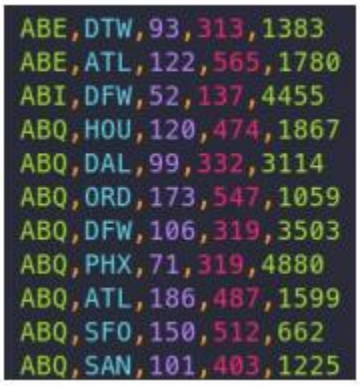

Imagen 2: Vuelos

Desarrollo:

1. Crear un archivo “command.txt “, con los comandos para compilar el código y ejecutarlo.

2. Documento con los TAD’s utilizados, casos de prueba, resultados de cada operación y conclusiones.

3. Cree una consola sencilla (similar a la del proyecto) con las siguientes operaciones:

a) *camino_corto:*

Parámetros de entrada: origen y destino.

La funcionalidad debe imprimir el camino más corto de la siguiente forma:

SAN− > DEN− > JFK

b) *componentes_conectados*:

Parámetros de entrada: todo el grafo.

Se deben generar la lista de componentes conectados e imprimir los aeropuertos que pertenecen a cada componente conectado. En caso que su respuesta sea 1 solo

componente conectado, modifique algunos nodos o aristas del grafo para que como mínimo tenga 2 componentes conectados.

c) *recorrido_grafo*:

Parámetros de entrada: origen, tipo recorrido [**BFS, DFS]**

Dado un nodo origen y un tipo de recorrido, se debe imprimir como quedaría el recorrido correspondiente.

NOTA: Tener en cuenta que tanto la ciudad origen como la destino pueden tener más de un aeropuerto, y nos interesa la mejor forma (rápida o barata) entre todas las combinaciones posibles. Los datos de aeropuertos se encuentran en coordenadas geográficas, usted debe buscar como calcular la distancia entre coordenadas en KM.

```
1       double calcularDistancia(int x1, int y1, int x2, int y2)
2       {
3         /*
4         Aplicar la f rmula que dice:
5         distancia = raiz-cuadrada-de(elevar-al-cuadrado(x1-x2) + elevar-al-cuadrado(
    y1-y2))
6         Nota: no importa el orden de los puntos ni si la distancia al restar es
    negativa
7         */
8       }
9
```

Calificación:

- (17%) Diagrama de TAD's que muestre el diseño del sistema.
- (5%) Archivo txt que indica como compilar y ejecutar cada una de las pruebas que se pide. La idea es que se usen los comandos que se indiquen en este documento y se pueda probar todas las funcionalidades.
- (15%) Lectura de archivos csv y creación de grafo.
- (17%) Implementación de *camino_corto*.
- (17%) Implementación de *componentes_conectados*.
- (17%) Implementacion de *recorrido_grafo*.
- (12%) Plan de pruebas (DEBE incluir imágenes y casos de prueba para cada una de las 3 operaciones que se piden, debe incluir varios casos como mínimo 3 casos por cada operación. En la operación de "*recorrido_grafo*" debe comparar el mismo caso con los dos tipos de recorrido). En total debe tener como mínimo 9 pruebas.

NOTA: si el código no compila por línea de comandos usando versión c++ 11, el taller no se califica y tiene nota de 0.