Estructuras de Datos no Lineales

Práctica 7

Problemas de grafos I

PASOS A SEGUIR

1. Definir la clase `Grafo` (grafo no ponderado) y la clase genérica `GrafoP<tCoste>` (grafo ponderado). Para ello escribir cuatro ficheros de cabecera (.h) con las siguientes representaciones de grafos:
    a. Matriz de adyacencia (para grafos no ponderados, clase `Grafo`).
    b. Listas de adyacencia (para grafos no ponderados, clase `Grafo`).
    c. Matriz de costes (para grafos ponderados, plantilla de clase `GrafoP<tCoste>`).
    d. Listas de adyacencia (para grafos ponderados, plantilla de clase `GrafoP<tCoste>`).
2. Crear un fichero de cabecera (.h) con la definición de una sencilla clase genérica `matriz<T>` para matrices cuadradas.
3. Implementa un módulo `alg_grafos.h` con la implementación de plantillas de funciones para los algoritmos de Dijkstra, Floyd y Warshall.
4. Escribir un módulo (fichero .cpp) que contengan las implementaciones de las funciones requeridas en cada problema.
5. Para cada uno de los problemas, escribir un programa de prueba donde se realicen las llamadas a las funciones correspondientes, definidas en el paso anterior, comprobando el resultado de salida para una batería suficientemente amplia de casos de prueba. Esto se puede hacer de dos maneras: Incluyendo la función `main()` en el fichero .cpp del paso anterior; o bien, creando un nuevo fichero .cpp para la función `main()`, que se compilará por separado y se enlazará con con el .cpp anterior.

PROBLEMAS

1. Sobrecargar el operador de extracción `>>` para leer de un flujo de entrada de tipo `istream` la matriz de costes de un grafo ponderado. El grafo estará definido previamente y el operador se limitará a sustituir las aristas que pueda tener por los pesos extraídos de la entrada.

Para facilitar la introducción de grafos en los programas de prueba utilizaremos la función `leerFicheroGrafo()`, cuyo parámetro es el nombre de un fichero de texto del cual lee la matriz de costes de un grafo. En el fichero debe aparecer el número de nodos del grafo seguido de los pesos de las aristas en orden creciente de filas y columnas de la matriz de costes, por conveniencia podemos escribir una fila de la matriz en cada línea del fichero.

```
// Lectura de un grafo desde un fichero de texto de nombre nf,
// que contiene el número de vértices seguido de los pesos
// de las aristas.
template <typename tCoste>
GrafoP<tCoste> leerFicheroGrafo(const std::string& nf)
{
    std::ifstream fg(nf.c_str()); // apertura del fichero
    unsigned n;                   // núm. vértices
    fg >> n;
    GrafoP<tCoste> g(n);        // grafo ponderado de n vértices
    fg >> g;                    // lectura de aristas
    fg.close();                 // cierre del fichero
    return g;
}
```

2. Sobrecargar el operador de inserción `<<` para la clase genérica `GrafoP<tCoste>`, el cual escribirá la matriz de costes de un grafo en un flujo de salida de tipo `ostream`.

3. Escribir una función genérica que implemente el algoritmo de Dijkstra usando un grafo ponderado representado mediante listas de adyacencia.

4. Se define el *pseudocentro* de un grafo conexo como el nodo del mismo que minimiza la suma de las distancias mínimas a sus dos nodos más alejados.

Se define el *diámetro* del grafo como la suma de las distancias mínimas a los dos nodos más alejados del centro del grafo.

Dado un grafo conexo representado mediante matriz de costes, implementa un subprograma que devuelva la longitud de su diámetro.

5. Tu empresa de transportes “PEROTRAVEZUNGRAFO S.A.” acaba de recibir la lista de posibles subvenciones del Ministerio de Fomento, en la que una de las más jugosas se concede a las empresas cuyo grafo asociado a su matriz de costes sea acíclico. ¿Puedes pedir esta subvención?.

Implementa un subprograma que a partir de la matriz de costes, nos indique si tu empresa tiene derecho a dicha subvención.

6. Se necesita hacer un estudio de las distancias mínimas necesarias para viajar entre dos ciudades cualesquiera de un país llamado Zuelandia. El problema es sencillo pero hay que tener en cuenta unos pequeños detalles:

a) La orografía de Zuelandia es un poco especial, las carreteras son muy estrechas y por tanto solo permiten un sentido de la circulación.
b) Actualmente Zuelandia es un país en guerra. Y de hecho hay una serie de ciudades del país que han sido tomadas por los rebeldes, por lo que no pueden ser usadas para viajar.
c) Los rebeldes no sólo se han apoderado de ciertas ciudades del país, sino que también han cortado ciertas carreteras, (por lo que estas carreteras no pueden ser usadas).

d) Pero el gobierno no puede permanecer impasible ante la situación y ha exigido que absolutamente todos los viajes que se hagan por el país pasen por la capital del mismo, donde se harán los controles de seguridad pertinentes.

Dadas estas cuatro condiciones, se pide implementar un subprograma que dados

- el grafo (matriz de costes) de Zuelandia en situación normal,
- la relación de ciudades tomadas por los rebeldes,
- la relación de carreteras cortadas por los rebeldes
- y la capital de Zuelandia,

calcule la matriz de costes mínimos para viajar entre cualesquiera dos ciudades zuelandesas en esta situación.