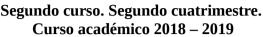


Universidad de Córdoba Escuela Politécnica Superior de Córdoba

ESTRUCTURAS DE DATOS

GRADO EN INGENIERÍA INFORMÁTICA





CUARTA PRÁCTICA

GRAFOS, CAMINOS Y DISTANCIAS MÍNIMAS.

- **OBJETIVO.** El objetivo de esta práctica es doble:
 - Implementar el tipo grafo utilizando una matriz de adyacencias.
 - Implementar un algoritmo que permita obtener el camino y la distancia mínima entre todos pares de nodos en un grafo.
- **ENUNCIADO**. Implemente el tipo abstracto grafo y el algoritmo de Floyd para buscar el camino más corto entre todos los pares de nodos.

PRIMERA PARTE. Implementación de la clase grafo

- La implementación se realizará en el fichero *grafo.hpp* y *funciones.hpp*.
- Observación:
 - Las funciones o fragmentos de código a completar vienen indicados en el código por medio del comentario "// TODO"
 - La clase grafo debe utilizar dos plantillas (templates): (1) *G_Nodo*, indica el tipo genérico utilizado para representar un nodo y (2) *G_Lado*, indica el tipo genérico utilizado para representar un lado.
 - Para realizar las pruebas se suministra un ejemplo, correspondiente a la red de carreteras andaluza. La matriz de conexión está almacenada en el archivo matrizAndalucia.txt, y los nombres de las capitales están en el archivo Andalucia.txt.
 - El valor 32000 indica que no existe una conexión directa entre los dos nodos correspondientes.

Atributos:

- el grafo deberá tener
 - Un vector de un tipo parametrizado que representa los nodos del grafo (_nodos)
 - Una matriz de un tipo parametrizado que representa los lados del grafo (*_lados*)
- Constructores y destructor
 - Grafo ()
 - Crea un nuevo grafo vacío
 - Grafo (n: int)
 - Crea un nuevo grafo inicializando las estructuras para almacenar *n* nodos
 - Grafo (g: Grafo)
 - Crea un nuevo grafo a partir de otro grafo.
 - ~Grafo()
 - Elimina el grafo
- Métodos
 - Void borrarGrafo ()
 - Borra el grafo liberando la memoria
 - Grafo operador = (g: Grafo)
 - Operador de asignación. Operador que copia el grafo "g" en el grafo actual
 - Bool cargarGrago (g: Grafo)

- Carga un grafo desde fichero
- **IMPORTANTE**: es responsabilidad del alumno identificar cualquier otro método que sea necesario como, por ejemplo, accesores o modificadores.

· SEGUNDA PARTE. Implementación del algoritmo de Floyd

• La implementación se realizará en el fichero *algoritmosgrafos.hpp*.

Observación:

 Las funciones o fragmentos de código a completar vienen indicados en el código por medio del comentario "// TODO"

Verificación:

Se proporciona un fichero Makefile responsable de compilar *main.cpp* (**no debe ser modificado**). Se generará el ejecutable *main.exe*, el cual servirá para probar la implementación realizada por el alumno.

• Métodos:

 Se deberá de codificar la función (o funciones) necesaria para la implementación del algoritmo de Floyd.

ENTREGA Y EVALUACIÓN

- Duración de la práctica nº 4: tres sesiones de dos horas cada una.
- o Plazo máximo de entrega
 - 23:55 horas del domingo 19 de mayo de 2019
- Se proporciona un fichero comprimido denominado "practica-4.zip" que contiene los siguientes ficheros

Practica-4.pdf

Enunciado de la práctica 4 (este documento)

Makefile

- make:
 - Compila el código y crea un programa ejecutable denominado *main.exe* para probar la implementación del algoritmo de Floyd.
- · make clean:
 - Borra ficheros superfluos
- Todos los ficheros de código (.hpp y .cpp) descritos anteriormente.

Al terminar la práctica,

- se deberá subir un fichero **comprimido** denominado "practica-4-usuario.zip",
- donde usuario es el login de cada estudiante.
- y que contenga todos los ficheros de la práctica.

Observaciones

- Se debe usar el espacio de nombres de la asignatura: ed
- Se debe comentar el código entre líneas.

Evaluación

- o La calificación de la práctica se basará
 - en la calidad y completitud del trabajo realizado.
 - y en la **defensa presencial de cada estudiante**.

Se valorará

- La correcta implementación del tipo grafo.
- La correcta implementación del algoritmo de Floyd.
- El correcto funcionamiento del programa principal propuesto.
- La claridad del código, así como de sus comentarios.