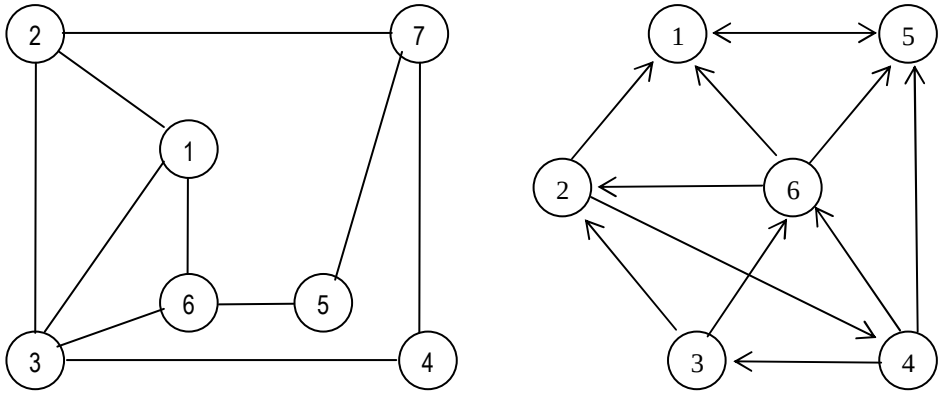
Práctico 5

Grafos

EJERCICIO 1 (I)

Dados los siguientes grafos, determine su representación en las formas:

1. Matriz de adyacencia
2. Listas de adyacencia



Discuta en que casos es preferible cada forma de representación.

EJERCICIO 2 (I)

Implemente en C/C++ el TAD grafo de enteros

CrearGrafo : $\varnothing \rightarrow$ Grafo
AgregarVertice : Grafo $\times$ Vertice $\rightarrow$ Grafo
QuitarVertice : Grafo $\times$ Vertice $\rightarrow$ Grafo
AgregarArista : Grafo $\times$ Arista $\rightarrow$ Grafo
QuitarArista : Grafo $\times$ Arista $\rightarrow$ Grafo
Vertices : Grafo $\rightarrow$ ConjuntoDeVertices
Adyacentes : Grafo $\times$ Vertice $\rightarrow$ ConjuntoDeVertices

Utilizando:

1. Matriz de adyacencia
2. Listas de adyacencia

Notar que debe definirse tipos para Vértice, Arista y ConjuntoDeVertices. Esos tipos deben ser comunes a ambas representaciones.

EJERCICIO 3 (I)

Supongo que el identificador de cada Vértice es un nombre en lugar de un entero.

¿Como modificaría la implementación mediante listas de adyacencia para soportar este nuevo requerimiento? Haciendo una comparación contra la implementación estándar con listas de adyacencia, ¿varía el orden de ejecución (promedio) de alguna operación del tipo abstracto? En caso afirmativo: ¿se le ocurre alguna manera de implementarlo para que el tiempo promedio no se vea modificado?

EJERCICIO 4 (R)

Las recorridas DFS y BFS numeran los vértices de un Grafo, partiendo de un Vértice dado.

a) Constuya las recorridas DFS y BFS para los grafos del ejercicio 1 a partir del nodo número 6.
b) Implemente las funciones

```
ListaVertice dfs_preorden (Grafo, Vertice)
ListaVertice dfs_postorden (Grafo, Vertice)

ListaVertice bfs_preorden (Grafo, Vertice)
ListaVertice bfs_postorden (Grafo, Vertice)
```

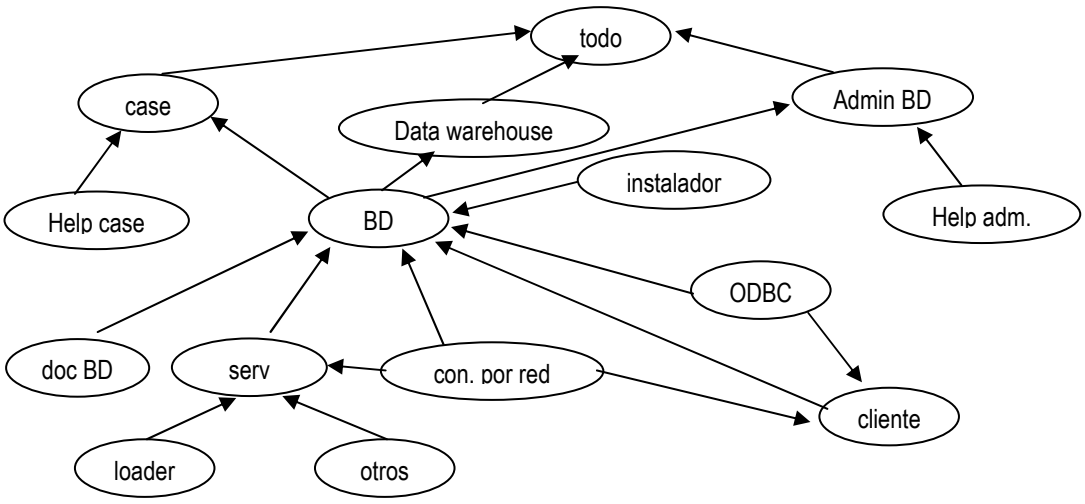
Que dados un Grafo y un Vértice devuelven la lista de Vértices resultante de las recorridas DFS y BFS en preorden y postorden respectivamente. Analice en cada caso si es conveniente una implementación recursiva o iterativa, así como la utilización de estructuras auxiliares.

EJERCICIO 5 (R)

Escriba un algoritmo basado en DFS para encontrar el camino dirigido más largo en un Grafo dirigido acíclico.

EJERCICIO 6 (I)

Dado el siguiente grafo que indica la precedencia entre tareas de instalación de componentes:



Determine un orden posible para las tareas que respete las restricciones de orden. ¿En qué casos no es posible determinar una ordenación de este tipo?

EJERCICIO 7 (R)

a) Defina una función que determine si un Grafo dirigido contiene un camino euleriano.
b) Analice el tiempo de ejecución del algoritmo si el Grafo se representa mediante:

1. Listas de adyacencia
2. Matriz de adyacencia

EJERCICIO 8 (R)

Implemente la función

```
ListaVertice puntosArt (Grafo)
```

Que dado un grafo no dirigido conexo, devuelve sus puntos de articulación.

EJERCICIO 9 (C)

a) Defina una función que determine las componentes fuertemente conexas de un Grafo dirigido.
b) Analice el tiempo de ejecución del algoritmo si el Grafo se representa mediante:

3. Listas de adyacencia
4. Matriz de adyacencia

EJERCICIO 10 (C)

Se dice que un Grafo es bipartito si sus vértices se pueden dividir en dos conjuntos (clases de equivalencia) tal que los que caigan dentro de un mismo conjunto no tengan aristas que los conecten.

Defina un algoritmo basado en BFS que determine si un Grafo es bipartito. En caso de serlo, retornar dos listas con las clases de equivalencia del Grafo.

EJERCICIO 11 (C)

Escriba un algoritmo basado en DFS que tome como entrada un Grafo dirigido y dos de sus vertices y determine todos los caminos simples de un vértice al otro. Calcule el orden de su algoritmo para el peor caso.