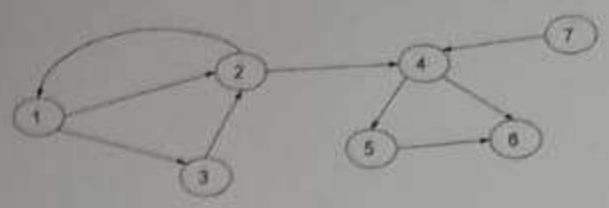
Parcial de Programación 3 - TUDAI

26/10/2024

Ejercicio 1



a) Dado el siguiente grafo y un DFS que sabemos que comienza por el **vértice 1** y toma como primer arco, el **arco (1,2)**. Listar los arcos tree, los arcos back y el **recorrido DFS** que se obtendrá.

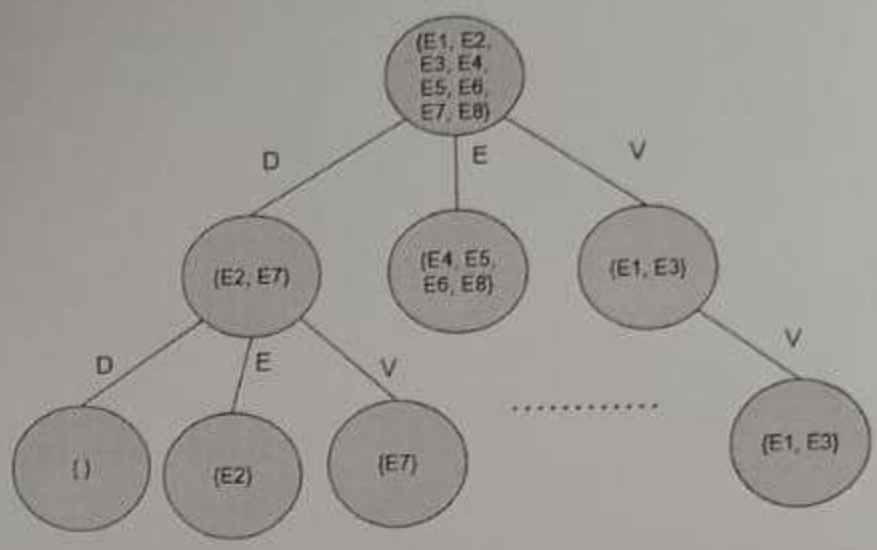
b) Si sabemos que un grafo dirigido es cíclico ¿podemos asegurar que siempre vamos a encontrar un arco back o depende del recorrido DFS que elijamos? Justificar.

c) Un grafo no dirigido de grado 7 compuesto por 7 vértices ¿Puede ser no conexo? Justifique su respuesta.

d) Y un grafo dirigido con igual número de vértices y grado, ¿puede ser no conexo? Justifique su respuesta.

Ejercicio 2

Se tiene un árbol ternario como el de la imagen para llevar el registro de los resultados de una competencia por equipos. El nodo raíz contendrá el conjunto de los 10 equipos que participan de la competencia, mientras que cada nodo del árbol sólo contendrá los equipos que dconsiguieron los resultados sabiendo que el nodo izquierdo representa una derrota, el nodo central un empate y el nodo derecho una victoria.



Cada nivel del árbol representa una fecha de competencia de los equipos.

Implementar un algoritmo en Java que permita obtener los equipos, sin repetidos, que consiguieron al menos 2 victorias cuando se jugaron 4 fechas de competencia.

Nota: Puede asumir la existencia de la clase TreeNode:

```
public class TreeNode {
        private List<Equipo> equipos;
        private TreeNode left, center, right;
}
```

Ejercicio 3

Se desea implementar un recorrido sobre un grafo dirigido mediante la técnica de DFS pero intentando minimizar la cantidad de veces que se llama al método auxiliar DFS_visit.

Para esto, se propone modificar el método DFS para incorporar una estrategia Greedy en busca del objetivo deseado.

Explique qué **estrategia Greedy** se podría utilizar para intentar minimizar la cantidad de llamados al método DFS_visit.

Implementar en Java el método DFS dónde se vea la estrategia Greedy elegida.

Ejercicio 4

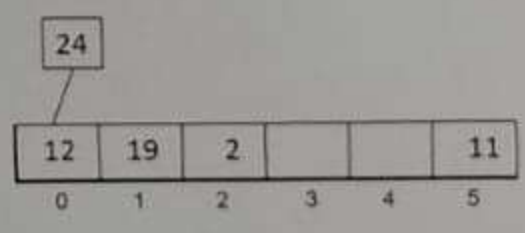
Dado un **conjunto de productos N** donde cada producto tiene un peso Pi y un valor económico Vi, y una mochila con capacidad máxima de K kilos. Se desea encontrar el **subconjunto de productos** a ubicar en la mochila que maximice el valor económico total, sin superar los K kilos disponibles. Se sabe que los productos no pueden ser fraccionados, es decir, si se elige poner un producto X en la mochila deberá hacerse en su totalidad.

a) Dibuje el árbol de exploración del algoritmo, indicando qué decisiones se toman en cada paso y qué información se lleva en los estados.

b) **Escriba un pseudo-java** del algoritmo que resuelva el problema mediante **Backtracking**.

Ejercicio 5

Dada la tabla de hashing de la derecha, que se comporta como el HashTable de JAVA, y se encuentra cargada de la manera que se muestra. Si M=6 y $\rho_d = 1.2$.



a) Indique cómo quedará **la tabla luego** de cada una de las siguientes inserciones de claves: 13, 3 y 38.

b) **Responda y justifique**: Si se tiene una estructura de hashing abierto separado, con M=15 y como función de hashing se elige la función h(x)=10, ¿Qué impacto tendrá esto en la estructura y en la operación de búsqueda?