NAME - DEEPTI PANDEY
SECTION - CST SPL 1
CLASS ROLL NO - 21
UNIVERSITY ROLL NO - 2016724

Tutorial - 5

① What is difference between DFS and BFS. Please write the applications of both the algorithms.

DFS	BFS
DFS stands for Depth First Search	BFS stands for Breadth First Search
DFS uses Stack data Structure	BFS uses Queue data Structure
DFS algorithm is a recursive algorithm that uses the idea of backtracking	In BFS, there is no backtracking
DFS requires less memory	BFS requires more memory.
Here, children are visited before the siblings	Here siblings are visited before the siblings.

2) Which data structures are used to implement BFS and DFS and why?

Queue is used ~~to~~ for BFS. It can only add/pop by certain sequence. ~~Or~~ These data structures are considered inherently "fair". The FIFO concept that underlies a Queue will ensure that those things were discovered first will be explored first.

DFS algorithm is a recursive algorithm that uses the idea of backtracking.
It traverses a graph in a depthward motion and uses ~~a~~ stack to remember to get the next vertex to start a search when a dead end occurs in any iteration.

③ What do you mean by sparse and dense graphs? Which representation of graph is better for sparse and dense graphs?
Sparse graphs are the graphs which are sparsely connected for example ~~trees~~. In this the number of edges is in O(n) where n is the number of vertices.
Dense graphs on the other hand are densely connected. In these graphs, the number of edges is usually O(n^2), so, adjacency matrix is preferred.

(4) How can you detect a cycle in a graph using BFS and DFS ?

Using BFS to detect a cycle - There is a visited list, so when reading the neighbors of current node and if the neighbour node is found which was visited before that ~~mean~~ indicates that a loop has been found.

Using DFS to detect a cycle - The graph has a cycle if and only if there exists a back edge. A back edge is an edge which is from a node to itself or one of its ancestor in the tree produced by DFS forming a cycle.

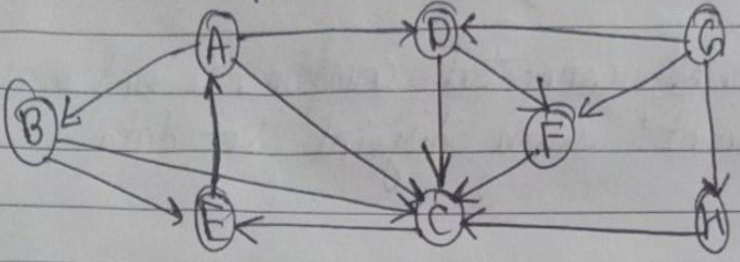
(5) What do you mean by disjoint set data Structure?

Partitioning the individuals into different sets according to the groups in which they fall This method is known as disjoint set data structure which maintains collection of disjoint sets and each set is represented by its representative which is one of its members.

The operations that can be performed on
(6) ~~Run~~ disjoint sets are union, find and MakeSet. These are as follows:

Intersection of two sets. As sets are disjoint its always empty unless these two coincide union of two sets - Supported out of the box. Enumerate the Set which is 'iterating' each element in given set. This depends again on implementation.

(4) Run BFS and DFS on graph shown on right side (Graph with 8 vertices).



BFS,

A

{B, C, D}   {A}

{D, C, E}   {A, B}

{C, E, F}   {A, B, D}

{E, F}   {A, B, D, C}

{F}   {A, B, D, C, E}

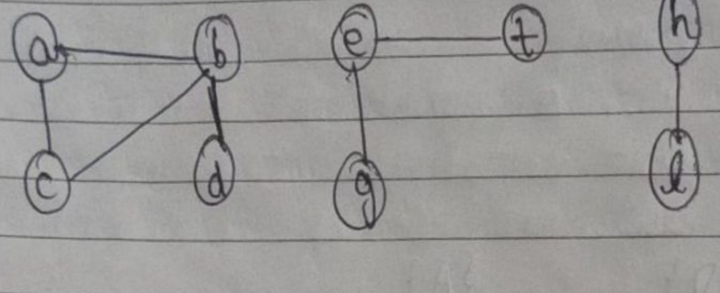
⇒ {A, B, D, C, E, F}

DFS,

Visited	A	B	C	D	E	F
Stack	~~B~~	E	F	~~E~~	F	
	~~D~~	~~D~~	C	F		
	~~C~~	C				

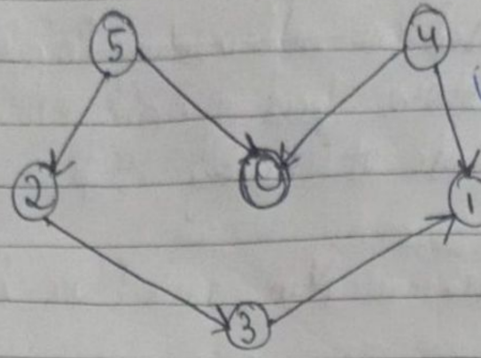
$\Rightarrow \{A, B, D, ~~C~~, E, F\}$

(7) Find out number of connected components and vertices in each component using disjoint set data structure.



Initial Sets	Collection of disjoint Sets								
	{a}	{b}	{c}	{d}	{e}	{f}	{g}	{h}	{i} {j}
(b,d)	{a}	{b,d}	{c}		{e,g}	{f}		{h} {i}	{j}
(e,g)	{a,[illegible]}	{b,d}			{e,g}	{f}		{h} {i}	{j}
(a,c)	{a,c}	{b,d}			{e,g}	{f}		{h,i}	{j}
(h,i)	{a,c} ~~{a,[illegible]}~~				{e,g}	{f}		{h,i}	{j}
(a,b)	{a,b,c,d}				{e,g}			{h,i}	{j}
(e,f)	{a,b,c,d}				{e,f,g}			{h,i}	{j}
(b,c)	{a,b,c,d}				{e,f,g}				

⑧ Apply topological sorting and DFS on graph having vertices from 0 to 5.



	0	1	2	3	4	5
visited	false	false	false	false	false	false

Stack (empty)

① Topological Sort (0) visited [0] = true

Stack | 0 |

② Topological Sort (1) visited [1] = true
Stack | 0 | 1 |

③ Topological Sort (2) visited [2] = true
Topological Sort (3) visited [3] = true
Stack | 0 | 1 | 2 | 3 |

④ Topological Sort (4) visited [4] = true
and (0 & 1) already visited

Stack | 0 | 1 | 2 | 3 | 4 |

Topological Sort (5) visited [5] = true and (2,0) already visited
=) | 0 | 1 | 2 | 3 | 4 | 5 |

⑨ Heap Data Structure can be used to implement priority queue? Name few graph algorithms where you need to use priority queue and why?

Heap Data Structures are used to implement priority queue. The binary heaps provides good asymptotic time complexities, with, under many reasonable sets of assumptions.

Dijkstra's Algorithm - when the graph is stored in the form of adjacency list or matrix, priority queue

Can be used to extract minimum efficiency when Implementing DijKstra's algorithm.

Prism's Algorithm - It is used to ~~implementing~~ implement prism's algorithm to store keys of nodes and extract minimum key node at every step.

10) What is the difference between Min and Max Heap?

Min Heap	Max Heap
In this, key is present at root node must be less than or equal to keys present	In this key present at root node must be greater than or equal to keys present
In this, minimum keys are present at root node	In this, maximum keys are present at root node
It uses ascending priority	It uses descending priority.