TUTORIAL:→05

Name → Mohd Nasir
Sec → 'F'
Roll → '14'
Uni Roll → 2016055
Student → 20021595
Subject → Design And Analysis of Algorithms.

Ques→1. What is difference between DFS and BFS. Write application both the algorithm.

BFS	DFS
(1). It's stands for Breath First Search.	(1) It's stand for Depth first search.
(2). In this searching we use Queue data structure.	(2) In this searching we use stack datastructure.
(3). It give 100% result.	(3) It does not give 100% result.
(4) It is more suitable for searching vertices are closer to given source.	(4). It is more suitable when there as away solution away from source.
(5). There is no concept of backtracking.	(5). It is a recursive algorithm that use backtracking.

Applications:→

(a) BFS:→ Bipartite graph and minimum no of Nodes Path, networking, and GPS.

(b) DFS:→ acyclic graph, topological Sort Scheduling Problems.

Ques→02 Which data structure are used to implement BFS and DFS and Why?

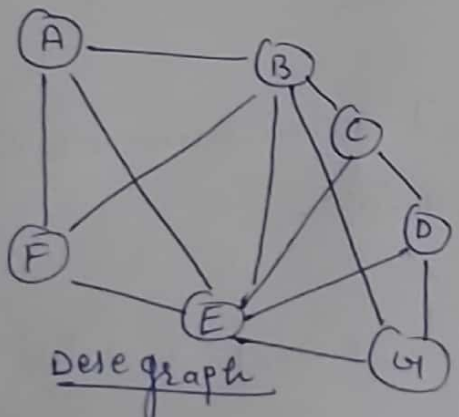
↳ For implementing BFS We use a Queue data-structure for finding minimum no. of nodes path between source node to destination node. We use Queue because things don't have to be processed immediately. but have to be processed in FIFO order. Like BFS :. Searching for nodes level traverse. Ex it search nodes w.r.t their distance from source. For this Queue is better in ~~any~~ ito use BFS.

For implemeting DFS We use →stack data-structure it traverse a graph in depthward motion and stack-10 remember to get the next nodes to start a search when dead end occurse in any iteration.

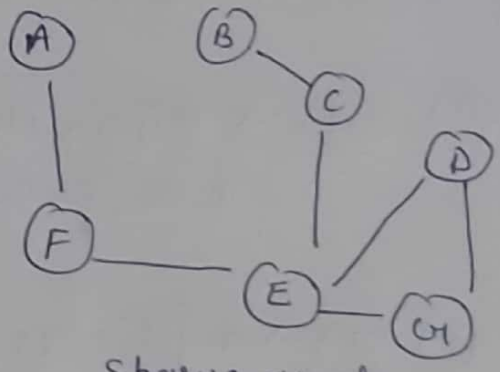
Ques→03 What do you mean by Spars and dense graphs? Which representation of graph is better for Sparse and dense graph?

↳ Dense graph is a graph in which no. of nodes is close to maximal no of edges.

↳ sparse graph is a graph in which no of edges in very less.



Dese graph



Sparse graph.

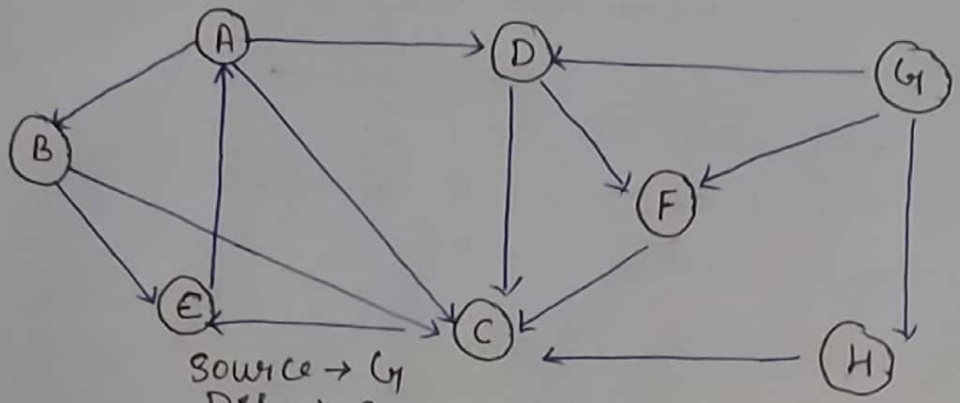
·) For Sparse graph it is prefered to use Adjency list.

·) For dense graph it is prefered to Adjency Matrix.

(2) Union :→ It takes two elements as input and find representation of this sets using the find operation. and finally Puts either one of the tree under Set node of other tree effectively Merging the trees and set.

```
void union(int a, int b)
{
    arr[a] = b
}
```

Ques → 6 Run BFS and DFS on graph.



BFS :→

Child	G	H	D	F	C	E	A	B
Parent	-	G	G	G	H	C	E	A

Path → G → H → C → E → A → B

DFS :→

PUSH	POP
~~G~~	G
D	F
H	C
~~F~~	E
~~C~~	A
~~E~~	B
~~A~~	
~~B~~	

Path → G → F → C → E → A → B.

Ques→04 How can you detect a cycle in graph using BFS and DFS.

Ans→ For detecting cycle in a graph using BFS we need to use Kahn's algorithm for topological sorting.

The steps required are:-

(1). Compute in-degree (no of incoming edges) for each of vertex present in graph and initialize count of visited nodes as 0.

(2) Pick all vertices with in degree as 0 and add them in queue.

(3) Remove a vetex from queue and then.
- increment count of visited nodes by 1.
- Decrease in-degree by 1 for all neighbouring nodes.
- If in-degree of neighbouring nodes reduced to zero then add in queue.

(4). Repeat 3. until queue is Empty.

(5). If count of visited nodes is not equal to no of nodes in graph. has cycle, otherwise not.

Ques→05 What do you mean by disjoint set data structure? Explain and ~~representations~~ operations along with example which can be performed on disjoint set.

Ans→ A disjoint set is a data structure that keeps track of set of element partitioned into several disjoint subsets. In other words. a disjoint set in a group of sets where no item can be in more than one set.

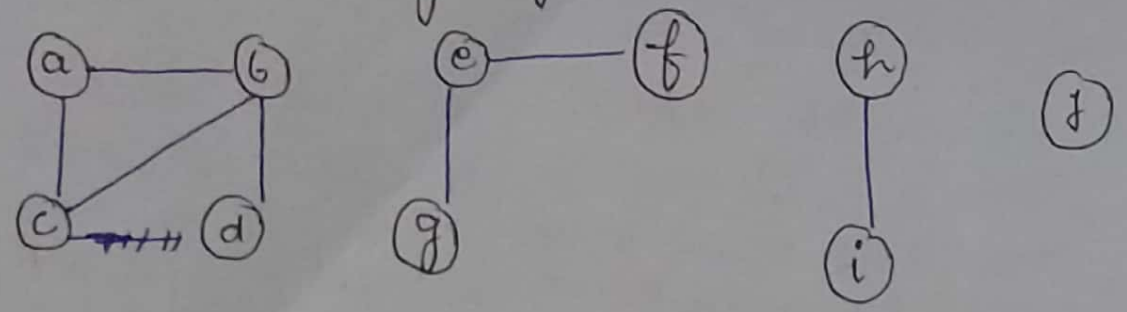
3 operations→

(i) Find → can be implemented by recursively traversing the parent array until we hit a node who is parent to itself.

```
int find (int V) {
  if (V == arr[V])
    return V;
  else return find(arr[V]);
}
```

Ques → 7 find out no of connected components and vertices in each component using disjoint set data structure.

Ans →



V = {a} {b} {c} {d} {e} {f} {g} {h} {i} {j}

= {a,b} {a,c}, {b,c}, {b,d}, {e,f}, {e,g} {h,i}, {j}

(a,b) {a,b} {c} {e} {f} {g} {h} {i} {j}.

(a,c) {a,b,c} {d} {e} {f} {g} {h} {i} {j}.

(b,c) {a,b,c} {d}, {e}, {f}, {g}, {h}, {i}, {j}.

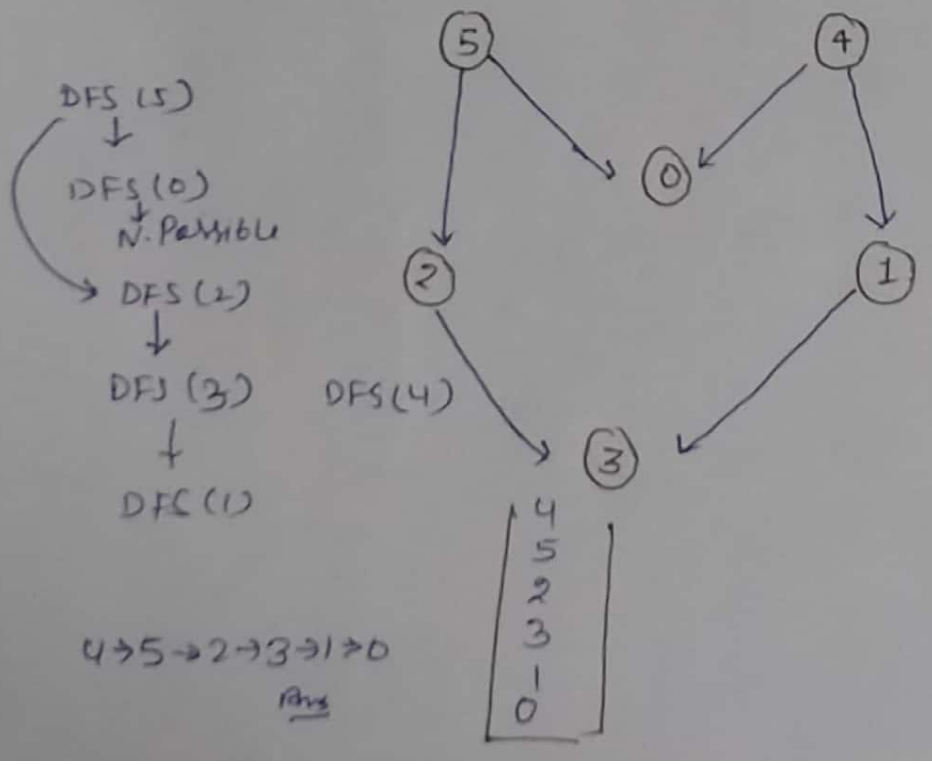
(b,d) {a,b,c,d} {e} {f} {g} {h} {i} {j}.

(e,g) {a,b,c,d} {e,f} {g} {h} {i} {j}

(h,i) {a,b,c,d} {e,f,g} {h,i} {j}

no. of connected components → 03.

Ques → 8 Apply topological sort and DFS on graph having vertices from 0 to 5.



Ques→9. Heap data structure can be used to implement priority queue Name few graph algorithm where you need to use Priority queue and why?

Ans Yes, Heap data structure can be used to implemented priority queue, It will take O(logn) time to insert and delete each element in priority queue. Based on heap structure priority queue has two types max priority queue based on max heap and min heap priority queue based on min heap Heaps provide better performance compare to array etc.