Name - [illegible] Singh
Class roll no - 04
Section - G

Tutorial 5

Ans 1

BFS	DFS
1) Stands for breadth first search	1) Stands for depth first search
2) It uses queue to find shortest path	2) It uses stack to find shortest path
3) BFS is better when target is closer to source	3) DFS is better when target is far away from source
4) It is ~~[illegible]~~ slower than DFS	4) It is much faster than BFS

DFS Application

- using DFS we can find path b/w three vertices
- we can perform topological sorting which is used to scheduling jobs
- we can use DFS to detect cycles

BFS Application

- BFS can also be used to detect cycle
- Finding shortest path and minimal spanning tree in unweighted graph

Scanned with CamScanner

Soln 2 BFS uses queue data structure. In BFS you mark any node in graph as source node and start traversing from it. BFS traverse all nodes in the graph and keeps dropping them as completed. BFS visited an adjacent unvisited node, marks it as done and insert it into queue.

DFS uses stack data structure because DFS traverse a graph in depthward motion and uses a stack to remember to get next vertex to start a search, when a dead end occurs in any iteration.

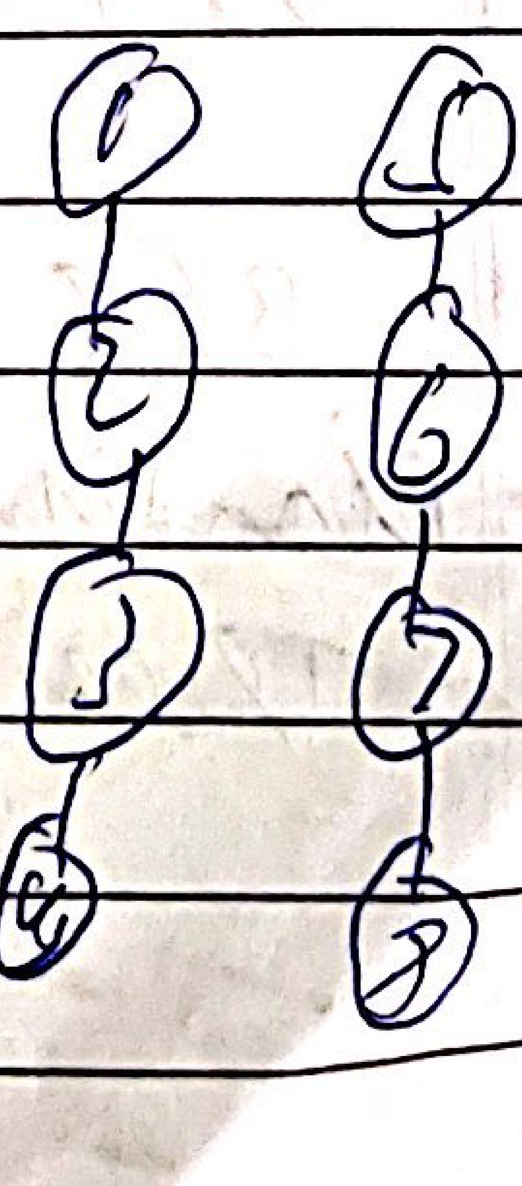
Soln 5 Disjoint set data structures

It allows to find out whether two elements are in same set or not efficiently.

• A disjoint set can be defined as subsets where there is no common element b/w two sets.

Eg $S_1 = \{1, 2, 3, 4\}$

$S_2 = \{5, 6, 7, 8\}$



Soln 6 BFS :

Node – B E C A D E

Parent – – B B E A D

Path B → E → A → D → E

DFS :

Node Processed – B B C E A D F

Stack – B CE EE AE DE FE

Path → B → C → E → A → D → F

Soln 8

Adjacency list

0 →

1 →

2 → 3

3 → 1

4 → 0, 1

5 → 2, 0

Visited

0	1	2	3	4	5
F	F	F	F	F	F

Step 1 Topological sort (0) ; visited [0] = True

Stack $\boxed{0}$

Step 2 Topological sort (1), visited [1] = True

Stack [0 1]

Scanned with CamScanner

Step 3 topological sort (2), visited (2) = true

Stack [0|1|3|2]

Step 4 Stack [0|1|3|2|4]

Step 5 Stack [0|1|3|2|4|5]

Step 6 Print all elements of stack from top to bottom

→ 5, 4, 2, 3, 1, 0

Sol 10

Min Heap	Max Heap
• In min heap the key present at root node must be less or equal to among keys present at all its children	• In max heap key present at root node must be greater or equal to key present at all child
• uses ascending priority	• uses descending priority
• The minimum key is present at the root node	• The maximum key is present at root node.