Name :- Abhay Rawat

Course :- B.Tech (CSE) Sec :- B

Sem :- V Roll No :- 01

University Roll No :- 1961002

Design & Analysis of Algorithm

Tutorial Sheet 5

Ques What is difference between DFS & BFS Please write the applications of both the algorithm.

Ans

BFS	DFS
I BFS stands for Breadth First search	DFS stands for Depth first search
II BFS ~~can be~~ use Queue data structure	DFS use Stack data structure.
III Siblings are visited before the children	Children are visited before the siblings
IV BFS consider all neighbours first & therefore not suitable for decision making tree used in games or puzzel	DFS is more suitable for game or puzzle problem. We make a decision, then explore all paths through this decision. And if this decision leads to win situtions, we stop

(V)	BFS is more suitable for searching vertices which are closest to the given source.	DFS is more suitable when there are solutions away from source

Applications

DFS

I Detect a cycle in a graph
II To find out the path between cities
III Topological sort
IV It also check if a graph is bipartite or not.
V The bridge in a graph can be found using DFS
VI Planarity testing

BFS

I Social Networking Websites
II GPS navigation system
III Broadcasting in the network
IV Peer to Peer Network
V Used in Garbage collection
VI Crawlers in Search Engines
VII Path Finding
VIII Ford-Fulkerson Algorithm

Ques 2 Which Data Structure are used to implement BFS & DFS & why?

Ans Queue Data structure use in BFS. BFS used to find a single source shortest path in an unweighted graph, because in BFS, we

reach a veerten with minimum number of edges from a source veerten.

Queue data structure are considered inherently "fair" - the fifo concept that underlies a queue will ensure that those things that veere discoveered first will be explored first, before exploring those that veere discoveered subsequently. The "first fie thing first" fairness approach is a fundamental tend behind BFS. like veaves outveard from a Pebble tossed in a pond. A BFS search expand systematical predictable.

DFS is depth first search. so you have to tranveerse a vehole branch of tree then you can traverse the adjacent nodes. So for keeping tracking on the current node. It reavired last in first out approch vehich can be implemented by stack. After it reaches the depth of a node then all the nodes will be poped out of Stack Next it searched for adjacent nodes vehich are not visited get yet.

Ques 3 what do you mean by sparse & dense graph? vehich representation of graph is better for sparse & dense graphs?

Ans

Sparsh graph

- A graph in which the number of edges is much less than the possible number of edges
- Sparsh graph can be a disconnected graph.
- Generally if edges < [V] log|V|.

Dense graph

- A graph in which the number of edges is close to the maximal number of edges.

Sparsh graph — Adjacency list
Dense graph — Adjacency matrix

Ques4 How can you detect in a cycle in a graph using BFS and DFS?

Ans

Detect cycle in a graph using BFS

You need maintain an array par[i] = parent of node i, now start BFS from node 1 & go on level wise. If a condition occurs when we are exploring neighbours of a node u and it is visited node but is not parent u then this is certainly a edge leading cycle.

Detect cycle in a graph using DFS

To detect cycle, check for a cycle in individual trees by checking back edges. To detect a back edge, keep track of vertices currently in the recursive stack of function for DFS

traversal. If a vertex is reached that is already in the recursion stack. Then there is a cycle in the ~~graph~~ tree.

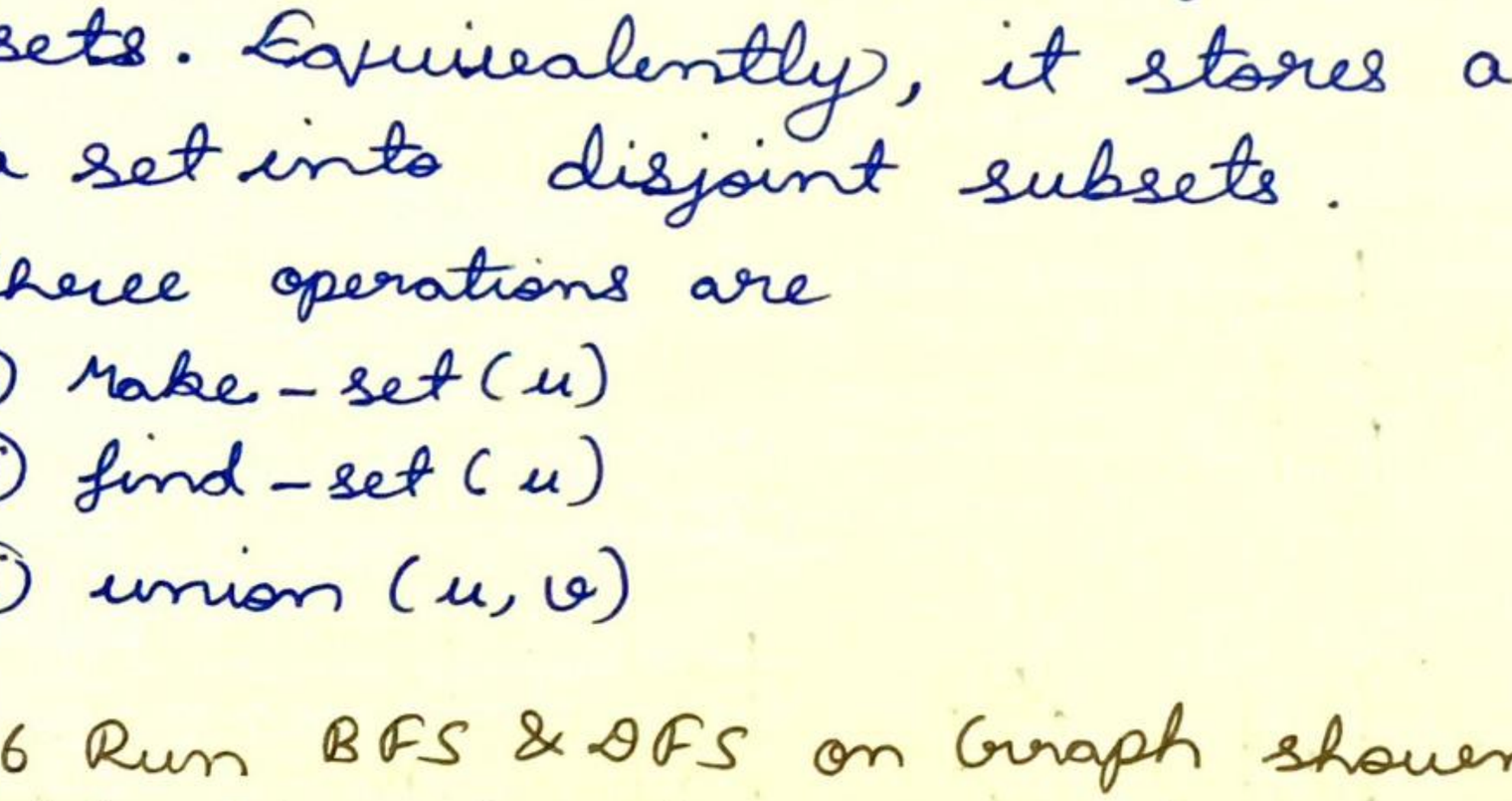
Ques5 What do you mean by disjoint set data structure? Explain 3 operations along with examples, which can be performed on disjoint sets.

Ans A disjoint set data structure, also called a union-find set, is a data structure that stores a collection of disjoint (non-overlapping) sets. Equivalently, it stores a partition of a set into disjoint subsets.

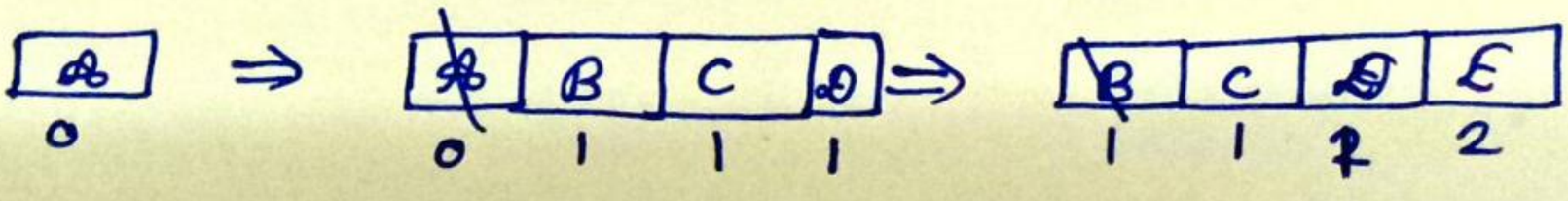
There operations are

(i) make-set(u)
(ii) find-set(u)
(iii) union(u, v)

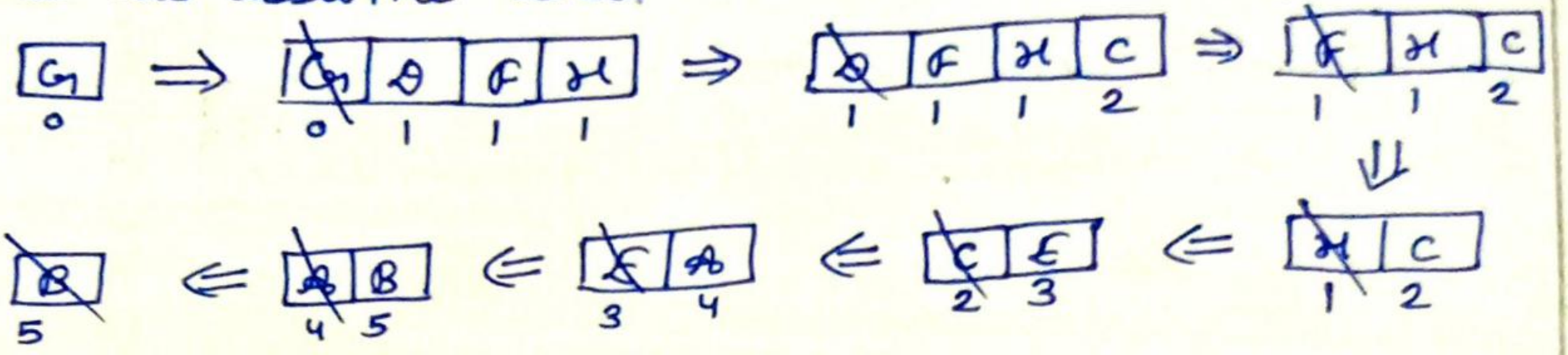
Ques6 Run BFS & DFS on Graph shown on right side (Graph with 8 vertex)

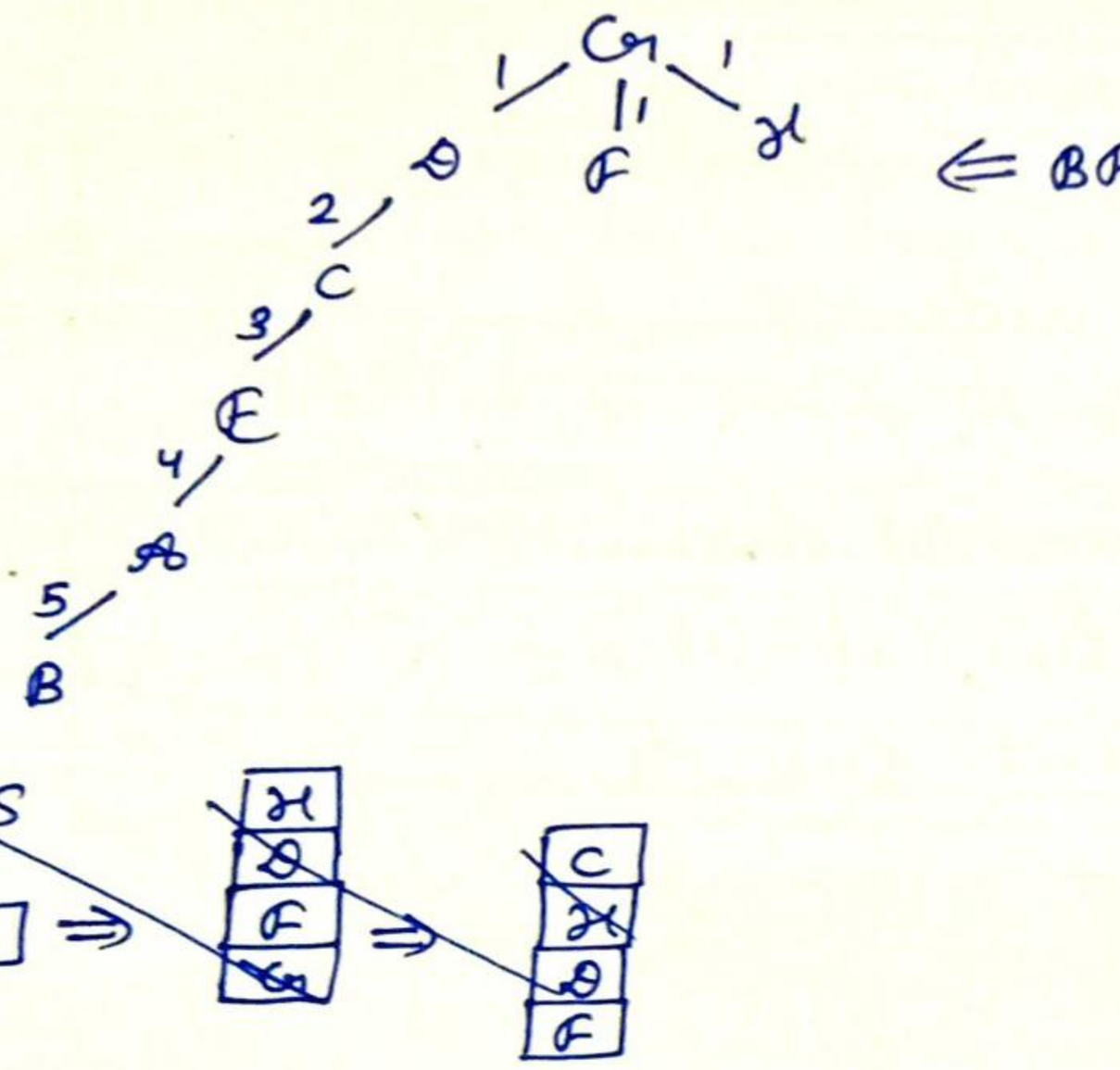


Sol Let us assume source node is A

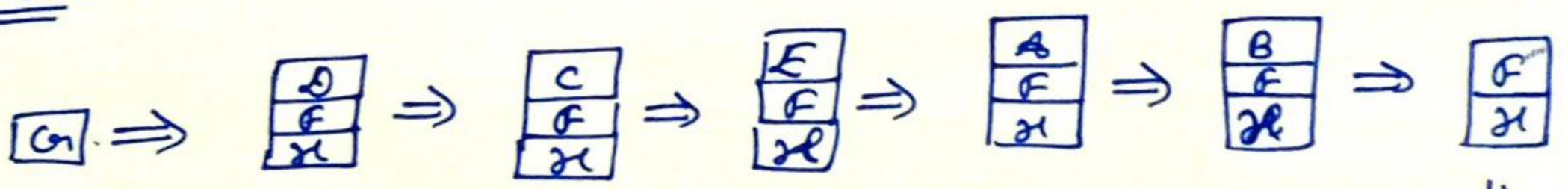


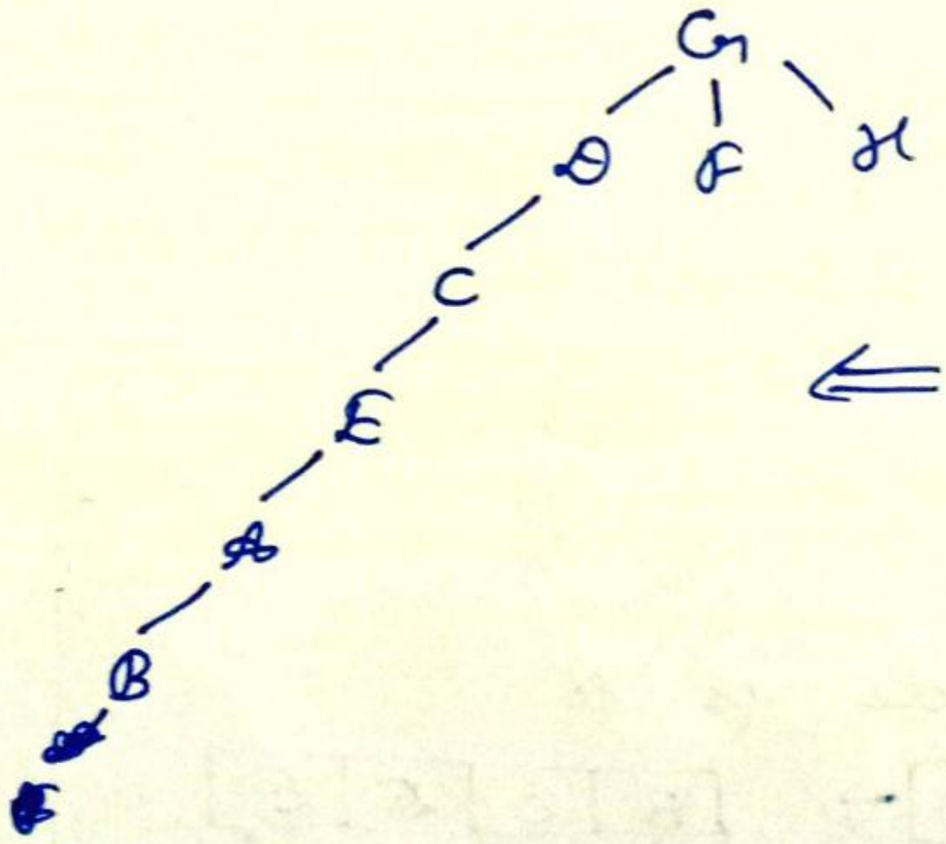
Sol Let us assume that G is our source node





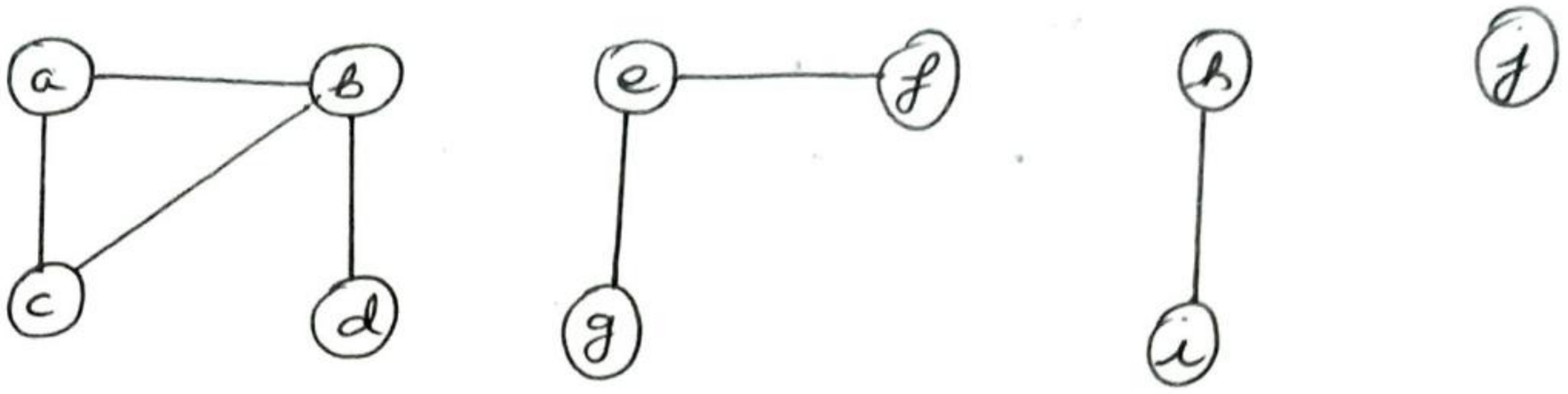
DFS





Ques 7 Find out the number of connected components & vertices in each component using dispint set data structure.

Ans



~~Ans~~ ~~find set~~

① make pair

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

② find set $u = \{a, b, c, d, e, f, g, h, i, j\}$

add edge (a, b)

$S_1 = \{a, b\}$

add edge = {(a, c)

$S_1 = \{a, b, c\}$

add edge = (b, d)

$S_1 = \{a, b, c, d\}$

add edge = (e, f)

$S_2 = \{e, f\}$

add edge = (e, g)

$S_2 = \{e, f, g\}$

add edge (h, i)

$S_3 = \{h, i\}$

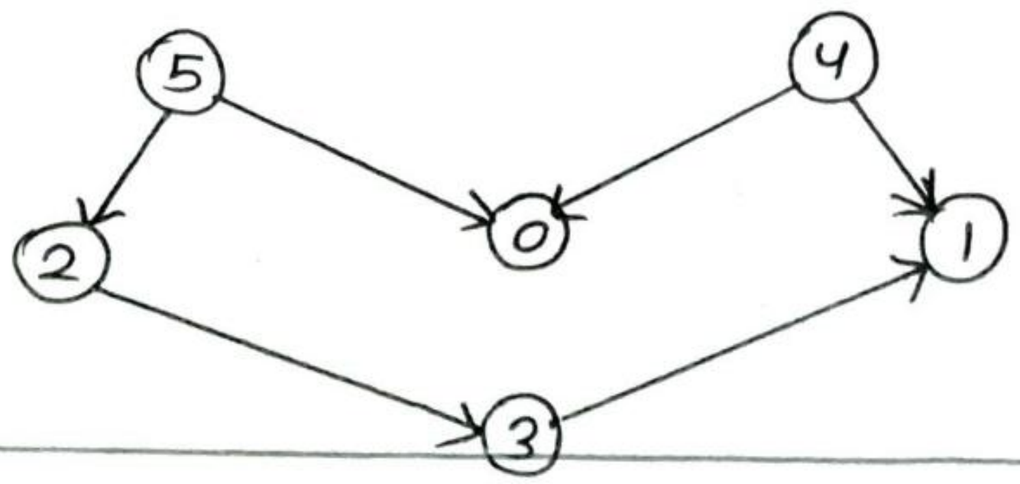
add edge (j)

$S_4 = \{j\}$

$S_5 = \{S_1, S_2, S_3, S_4, \}$

Hence, there have 4 connected graph

Ques 8 Apply topological sort & DFS on graph having vertices 0 to 5.



Ans

Stack

Step 1 We start with small no with 0, & 0 not have any other path so 0 will go in stack

Step 2 After that we go with 2 & 2 have one path to go 3

Step 2 After that we go with 1 in 1 there not have any other part. so 1 will go in stack.

Step 3 After that we go to 2 & 2 having one path to go 3 & in 3 there have not any unvisited node so will go in stack & come in 2 and also 2 don't have any further path so our 2 will go in stacks.

Step 4 And After that we go with 4 & in 4 we don't have any unvisited so our 4 will go in stacks

Step 5 After that we come with 5 & in 5 we don't have any unvisited so our 5 will go in stacks

Step 6 After that we don't have any further edge so stack will print & we have a topological sorting

| 5 4 2 3 1 0 |

Ans

Through DFS

| 5 4 2 3 1 0 |

Ques 9 Heap data structure can be used to implement prioority queue? Name few graph algorithm where

you need to use priority Queue & why?

Ans We can used heap to implement the priority Queue. It will take O(log n) time to insert & delete each element in the priority Queue, Based on heap structure, priority Queue also has two types max priority queue & min priority queue.

Priority Queue is used in many algorithms

(a) Dijkstra's Shortest Path Algs :- when the graph is stored in the form of adjacency list or matrix, priority queue can be used to extract minimum efficiently when implementation of Dijkstra's algorithm

(b) Prism's algorithm :- It is used to implement Prism's Algorithms to store keys of nodes & extract min. key node at every step.

(c) Huffman's algorithm :- constructing an optimum prefix-free encoding of an string

Ques 10 what is the difference b/w Max & Min Heap?

Ans

	Min Heap	Max Heap
I	In a min-heap the key present at the root node must ~~the key~~ be less than or equal to among the keys present at all of its children	In a max-heap the key present at the root node must be grater than or equal to among the keys present at all of its children

II	In a Min-Heap the minimum key element present at the root.	In a Max-Heap the maximum key element present at the root.
III	A min-heap used the accending priority	A Max-heap used the decending priority.
IV	In the construction of a ~~Heap~~ Min-Heap, the smallest element has priority	In the construction of a Max-heap, the largest element has priority
V	In a Min-Heap, the smallest element is the first to be popped from the heap.	In a Max-heap, the largest element is the first to be popped from the heap