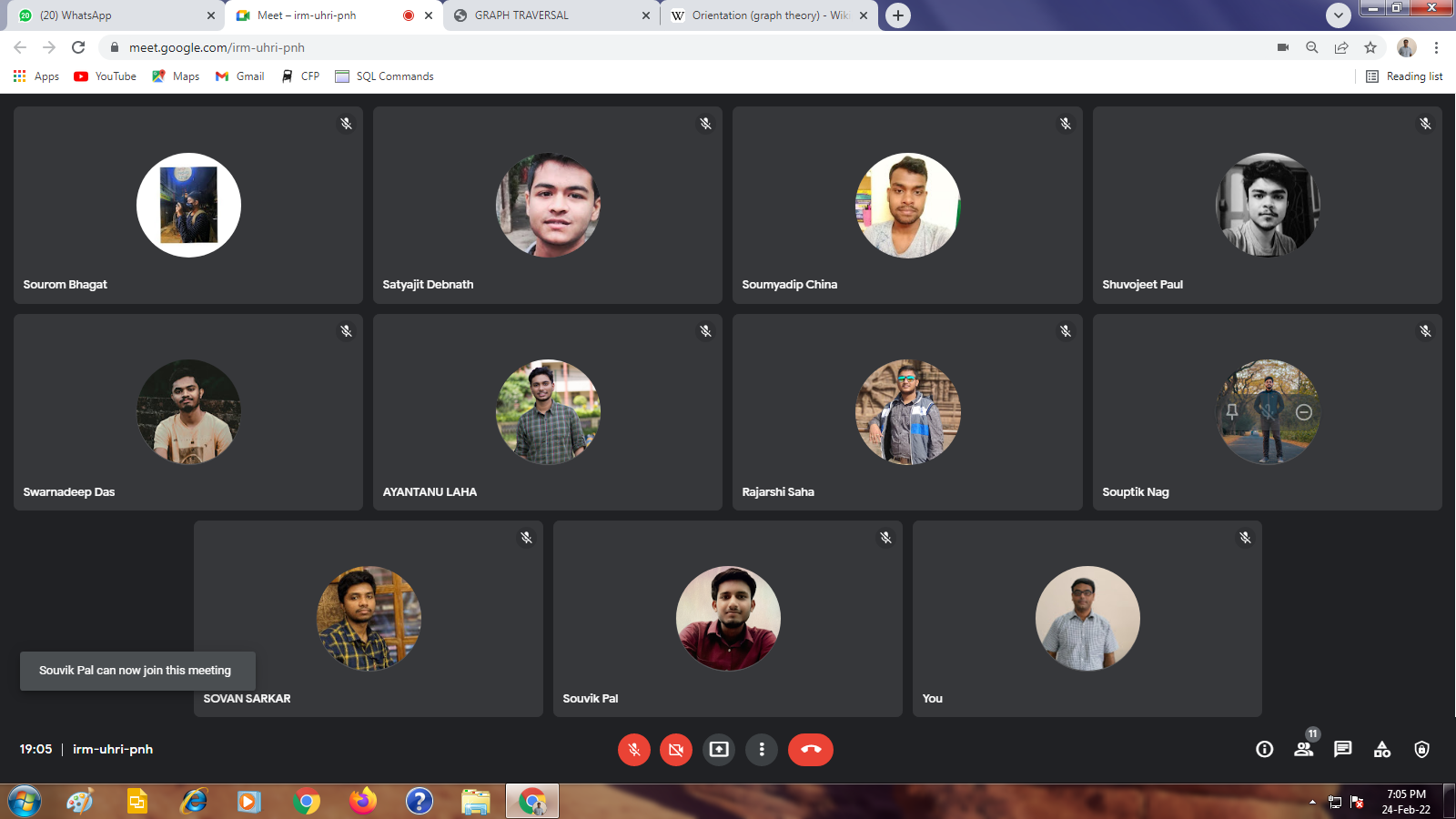
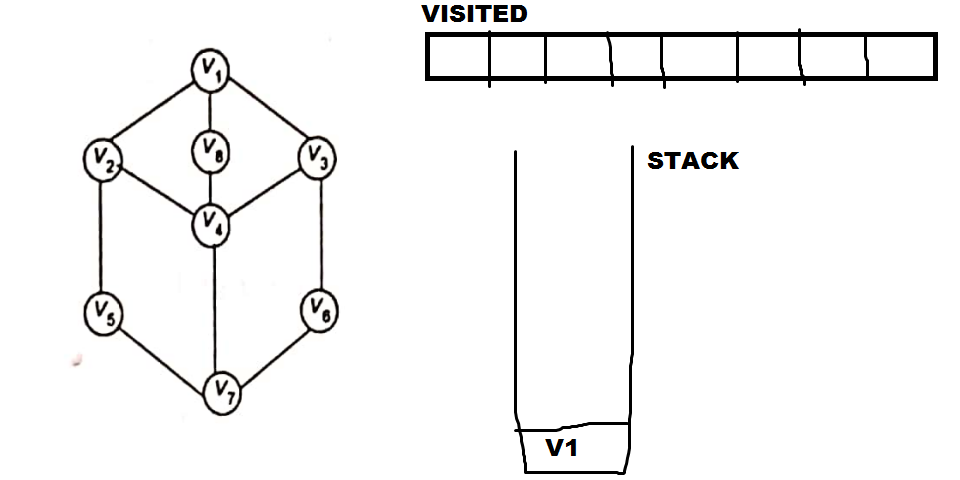
**CLASS 24/02/2022**

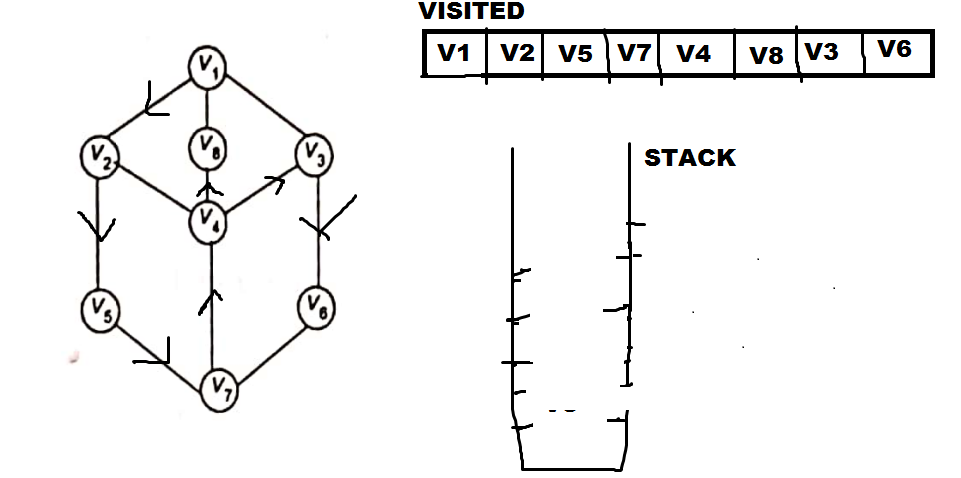
**UG SEMESTER-3**

**GRAPH THEORY**

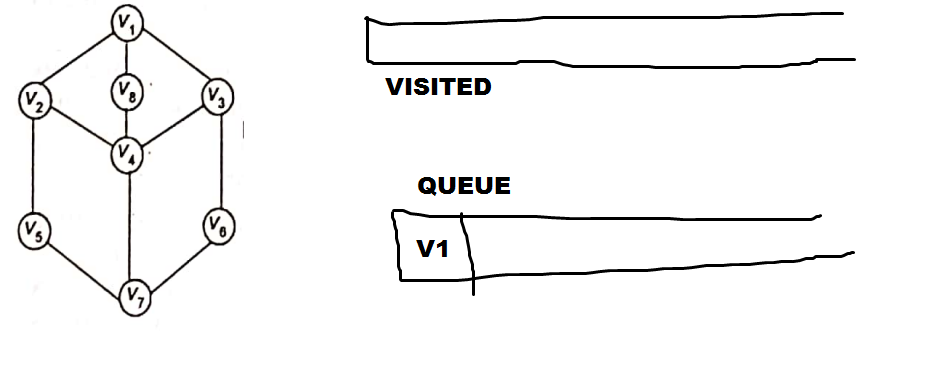


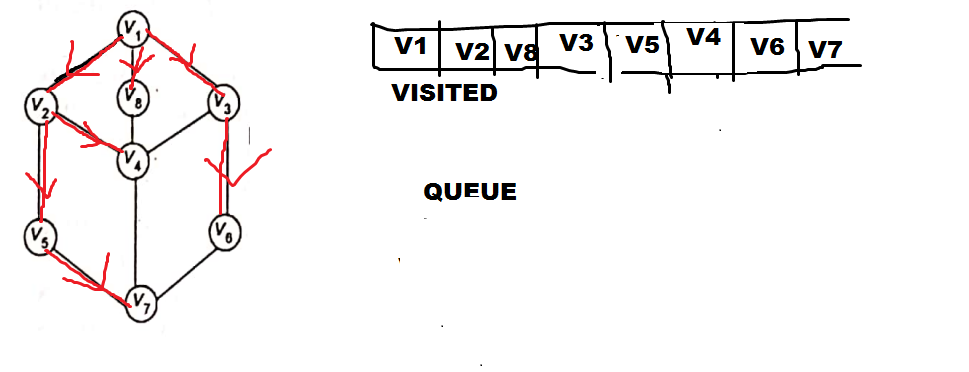
**DFS**





**BFS**





**Graph theory practical**

1. **Conversion of adjacency matrix to incidence matrix and adjacency list**

**Graph theory practical 2018**

1. Generate a random positive weighted graph of size nxn. The weights are positive real numbers. Now take a source vertex and find shortest paths to all the other vertices.

2. Generate a 2D matrix of size nxn. Each value of the matrix may be in the range (0-255). Now generate an undirected weighted graph G, taking each point of matrix as vertex. Each vertex has 8 –neighbours. An edge is there in between neighbours if their value difference (edge weight) is >0. Now apply prim’s algorithm to find the minimum spanning tree T. Now delete from T , (s-1) costliest edges to segment T into s different connected components.

3. Take input a 2D nxn matrix of strings of English alphabet. Now create a graph with these strings as vertices and there is an edge between neighbouring strings if they have one or more letters as common. Now find the connected components in this graph.

4. Take input a weighted undirected graph. Test by DFS algorithm whether it is a connected graph or not. Find the connected components, rank and nullity of this graph.

5. Implement Kruskal algorithm on a weighted(real number) undirected graph.

6. Take input a 2D nxn matrix of strings of English alphabet. Now create a graph with these strings as vertices and there is an edge between neighbouring strings if they have one or more letters as common. The edge weight between two vertices is defined as the difference in number of letters between two neighbouring strings. Now apply Floyd-warshall algorithm to find all pair of shortest paths.

7. Implement Breadth First Search algorithm on a weighted undirected graph.

8. Generate a random positive weighted graph of size nxn. The weights are positive real numbers. Now take a source vertex and find shortest paths to all the other vertices.

9. Generate a 2D matrix of size nxn. Each value of the matrix may be in the range (0-255). Now generate an undirected weighted graph G, taking each point of matrix as vertex. Each vertex has 8 –neighbours. An edge is there in between neighbours if their value difference (edge weight) is >0. Now apply prim’s algorithm to find the minimum spanning tree T. Now delete from T , (s-1) costliest edges to segment T into s different connected components.

10. Take input a 2D nxn matrix of strings of English alphabet. Now create a graph with these strings as vertices and there is an edge between neighbouring strings if they have one or more letters as common. Now find the connected components in this graph.

11. Take input a weighted undirected graph. Test by DFS algorithm whether it is a connected graph or not. Find the connected components, rank and nullity of this graph.

12. Implement Kruskal algorithm on a weighted(real number) undirected graph.

13. Take input a 2D nxn matrix of strings of English alphabet. Now create a graph with these strings as vertices and there is an edge between neighbouring strings if they have one or more letters as common. The edge weight between two vertices is defined as the difference in number of letters between two neighbouring strings. Now apply Floyd-warshall algorithm to find all pair of shortest paths.

14. Implement Breadth First Search algorithm on a weighted undirected graph.

15. Take input a set of numbers A. Generate all subsets of A except null set. Now represent the subsets of A as vertices. The two such subsets have an edge if they have common elements. The edge weight is the number of common elements between two subsets. Implement Dijkstra’s algorithm in such a graph.

16. Take input a set of Real numbers. Now create a graph with these numbers as vertices. There is an edge between two numbers n1 and n2 if |n1-n2|>=2. The edge weights between two vertices n and n2 is defined as |n1-n2|. Now apply Floyd Warshall algorithm to find all pair of shortest paths.