Algorithmic Graph Theory (Practical)

Semester VII (B.Tech. in Computer Science and Engineering)

1a) Implementation of undirected unweighted graph using Adjacency Matrix.
1b) Implementation of undirected weighted graph using Adjacency Matrix.
1c) Implementation of directed unweighted graph using Adjacency Matrix.
1d) Implementation of directed weighted graph using Adjacency Matrix.

2a) Implementation of undirected unweighted graph using Incidence Matrix.
2b) Implementation of undirected weighted graph using Incidence Matrix.
2c) Implementation of directed unweighted graph using Incidence Matrix.
2d) Implementation of directed weighted graph using Incidence Matrix.

3a) Implementation of undirected unweighted graph using Adjacency List.
3b) Implementation of undirected weighted graph using Adjacency List.
3c) Implementation of directed unweighted graph using Adjacency List.
3d) Implementation of directed weighted graph using Adjacency List.

4a) Consider an undirected graph, do DFS on it, and compute the DFS tree.
4b) Consider a directed graph, do DFS on it, and compute the DFS tree.
4c) Compute the number of components of a given undirected graph using DFS.
4d) Consider an undirected graph. Do DFS on it and differentiate its edges based on your sequence of visiting the vertices.
4e) Consider a directed graph. Do DFS on it and differentiate its edges based on your sequence of visiting the vertices.

5a) Consider an undirected graph, do BFS on it, and compute the BFS tree.
5b) Consider a directed graph, do BFS on it, and compute the BFS tree.
5c) Do BFS for computing the shortest distance and the path between a pair of vertices of an undirected graph, if one exists.
5d) Do BFS for computing the shortest distance and the path between a pair of vertices of a directed graph, if one exists.

6) Devise a scheme to represent a sparse matrix A and transpose this representation of A in lexicographic order.

7) Perform topological sort on a directed graph and report if there are any cycles. If it is a DAG, report a deterministic ordering of vertices of the given graph.

8) Expand the set of edges of a directed graph to obtain its transitive closure.

9) Report the set of strongly connected components (SCCs) in a given directed graph.

10) Report the set of bi-connected components (BCCs) in a given undirected graph.

11) Compute a minimum spanning tree (MST) of a suitable graph instance.

12) Devise an algorithm to compute a matching set of edges of a suitable graph instance.

13) Recognize triangulated graphs.

14) Recognize split graphs.