■ Graph Mastery Worksheets (Day 1 - 5)

Day 1 – Graph Representation

1. Q1. Given a graph with 5 vertices, draw it and represent it using an adjacency matrix.
2. Q2. Convert the adjacency matrix of the above graph into an adjacency list.
3. Q3. Write a function to count the total number of edges in an undirected graph represented by an adjacency matrix.
4. Q4. Write a function to compute the average degree of the graph.

Day 2 – Depth First Search (DFS)

1. Q1. Implement DFS (recursive) for an undirected graph represented as an adjacency matrix.
2. Q2. Modify your DFS function to print all vertices reachable from a source node.
3. Q3. Write a function using DFS to check whether the given undirected graph is connected.
4. Q4. Implement iterative DFS using a stack for the same graph.

Day 3 – Breadth First Search (BFS)

1. Q1. Implement BFS for an undirected graph starting from node 0. Print the traversal order.
2. Q2. Write a function to check whether there exists a path between two given nodes using BFS.
3. Q3. Find the minimum number of edges required to go from node 0 to node 4 using BFS.
4. Q4. Compare the time complexity of BFS and DFS on the same graph.

Day 4 – Directed Graphs

1. Q1. Represent the following directed graph using adjacency matrix and adjacency list.
2. Q2. Implement DFS for a directed graph and print all reachable vertices from a given source.
3. Q3. Write a function to detect if a directed graph contains a cycle using DFS.
4. Q4. Implement topological sort using DFS for a directed acyclic graph (DAG).

Day 5 – Weighted Graphs

1. Q1. Represent the following weighted graph using an adjacency matrix.
2. Q2. Write a function to find the vertex with the maximum sum of edge weights.
3. Q3. Implement Dijkstra's algorithm to find the shortest path from a source node to all other nodes.
4. Q4. Explain why BFS cannot be used for shortest path in weighted graphs and give an example.