Module 7 – Greedy Algorithm

1. Minimum Spanning Tree (MST)

Concepts:

Definition: A spanning tree of a graph is a subgraph that is a tree and connects all the vertices together. A minimum spanning tree (MST) is a spanning tree with the minimum possible total edge weight.

Applications:

Network design (e.g., telephone, electrical, hydraulic, TV cable, computer networks).
Approximation algorithms for NP-hard problems like the Traveling Salesman Problem.

Algorithms:

Prim's Algorithm:

Description: Starts from an arbitrary vertex and grows the MST one edge at a time by adding the smallest edge that connects a vertex in the MST to a vertex outside it.

Steps:

Initialize a tree with a single vertex, chosen arbitrarily from the graph.

Grow the tree by one edge: of the edges that connect the tree to vertices not yet in the tree, find the minimum-weight edge, and transfer it to the tree.

Repeat step 2 until all vertices are included.

Kruskal's Algorithm:

Description: Sorts all the edges in the graph in non-decreasing order of their weight. It then adds the next smallest edge to the growing spanning tree if it doesn't form a cycle.

Steps:

Sort all the edges in non-decreasing order of their weight.

Pick the smallest edge. Check if it forms a cycle with the spanning tree formed so far. If not, include this edge. Else, discard it.

Repeat step 2 until there are (V-1) edges in the spanning tree.

Dijkstra's Algorithm:

Description: Finds the shortest path from a source vertex to all other vertices in a weighted graph with non-negative edge weights.

Steps:

Assign a tentative distance value to every node: set it to zero for the initial node and infinity for all other nodes.

Set the initial node as current. Mark all other nodes unvisited. Create a set of all unvisited nodes.

For the current node, consider all its unvisited neighbors and calculate their tentative distances through the current node. Compare the newly calculated tentative distance to the current assigned value and assign the smaller one.

When done considering all neighbors of the current node, mark it as visited. A visited node will never be checked again.

Select the unvisited node with the smallest tentative distance, and set it as the new "current node." Then go back to step 3.