Greedy Technique:

1. Draw the weighted graph whose vertices are a-e, and whose edges-weights are given by {(a; b; 2); (a; c; 6); (a; e; 5); (a; d; 1); (b; c; 9); (b; d; 3); (b; e; 7); (c; d; 5); (c; e; 4); (d; e; 8)}

Perform Kruskal's algorithm to obtain a minimum spanning tree for G. Label each edge to indicate the order that it was added to the forest. When sorting, break ties based on the order that the edge appears in the above set.

2. Draw the weighted directed graph whose vertices are a-g, and whose edges-weights are given
by
{(a; b; 2); (b; g; 1); (g; e; 1); (b; e; 3); (b; c; 2); (a; c; 5); (c; e; 2); (c; d; 7); (e; d; 3); (e; f; 8); (d; f; 1)}
Perform Dijkstra's algorithm to determine the Dijkstra spanning tree that is rooted at source vertex a. Draw a table that indicates the distance estimates of each vertex in each of the rounds. Circle the vertex that is selected in each round.

3. Use Huffman's algorithm to provide an optimal average-bit-length code for the 7 elements {a, b,…. ,g} whose respective probabilities are
0.2; 0.2; 0.15; 0.15; 0.15; 0.1; 0.05:
Compute the average bit-length of a codeword.

4. Demonstrate Prim’s algorithm on the graph G = (V, E), where the weighted edges are given by E = {(a, b, 1),(a, c, 3),(b, c, 3),(c, d, 6),(b, e, 4),(c, e, 5),(d, f, 4),(d, g, 4), (e, g, 5),(f, g, 2),(f, h, 1),(g, h, 2)}.

5. Does Prim's and Kruskal's algorithm work if negative weights are allowed? Explain.

6. **Scheduling with Deadlines**. The input for this problem is a set of n tasks a1,….,an. The tasks are to be executed by a single processor starting at time t = 0. Each task ai requires one unit of processing time, and has an integer deadline di. Moreover, if the processor finishes executing ai at time t, where di <=t, then a profit pi is earned. For example, if task a1 has a deadline of 3 and a profit of 10, then it must be either the first, second, or third task executed in order to earn the profit of 10. Consider the following greedy algorithm for maximizing the total profit earned. Sort the tasks in decreasing order of profit. Then for each task ai in the ordering, schedule ai at time t<=di, where t is the latest time that does not exceed di, and for which no other task has yet to be scheduled at time t. If no such t exists, then skip ai and proceed to the next task in the ordering. Apply this algorithm to the following problem instance. If two tasks have the same profit, then ties are broken by alphabetical order. For example, Task b preceeds Task e in the ordering.

Task	a	b	c	d	e	f	g	h	i	j	k
Deadline	4	3	1	4	3	1	4	6	8	2	7
Profit	40	50	20	30	50	30	40	10	60	20	50

7. For the following instance of the Fractional Knapsack problem, determine the amount of each good that is placed in the knapsack by Fractional Knapsack Algorithm, and provide the total container profit. Assume M = 10.

Good	Weight	Profit
1	3	4
2	5	6
3	5	5
4	1	3
5	4	5

8. The **0-1** Knapsack problem is similar to Fractional Knapsack, except now, for each good g belongs to G, either all of g or none of g is placed in the knapsack. Consider the following modification of the Fractional Knapsack greedy algorithm. If the weight of the current good g exceeds the remaining capacity RC, then g is skipped and the algorithm continues to the next good in the ordering. Otherwise, it adds all of g to the knapsack and decrements RC by w(g), while incrementing TP by p(g). Verify that this modified algorithm does not produce an optimal knapsack for the problem instance of Question 7.