Week 8:

I. Assume that a project of road construction to connect some cities is given to your friend. Map of these cities and roads which will connect them (after construction) is provided to him in the form of a graph. Certain amount of rupees is associated with construction of each road. Your friend has to calculate the minimum budget required for this project. The budget should be designed in such a way that the cost of connecting the cities should be minimum and number of roads required to connect all the cities should be minimum (if there are N cities then only N-1 roads need to be constructed). He asks you for help. Now, you have to help your friend by designing an algorithm which will find minimum cost required to connect these cities. (use Prim's algorithm)

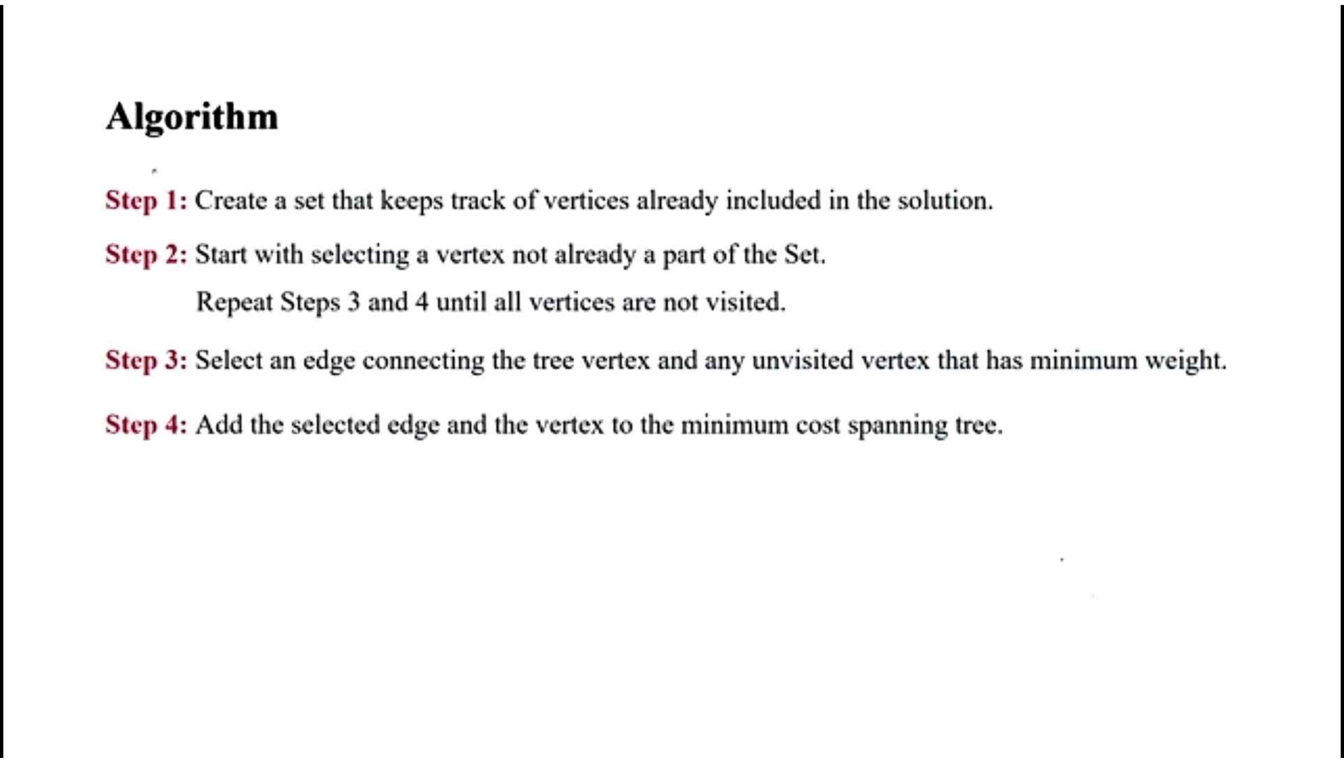
Input Format:
The first line of input takes number of vertices in the graph.
Input will be the graph in the form of adjacency matrix or adjacency list.

Output Format:
Output will be minimum spanning weight

Sample I/O Problem I and II:

Input:	**Output:**

7
0 0 7 5 0 0 0
0 0 8 5 0 0 0
7 8 0 9 7 0 0
5 0 9 0 15 6 0
0 5 7 15 0 8 9
0 0 0 6 8 0 11
0 0 0 0 9 11 0
	Minimum Spanning Weight: 39



Algorithm

Step 1: Create a set that keeps track of vertices already included in the solution.

Step 2: Start with selecting a vertex not already a part of the Set.

Repeat Steps 3 and 4 until all vertices are not visited.

Step 3: Select an edge connecting the tree vertex and any unvisited vertex that has minimum weight.

Step 4: Add the selected edge and the vertex to the minimum cost spanning tree.

II. Implement the previous problem using Kruskal's algorithm.

Input Format:
The first line of input takes number of vertices in the graph.
Input will be the graph in the form of adjacency matrix or adjacency list.

Output Format:
Output will be minimum spanning weight

Sample I/O Problem I and II:

Input:	**Output:**

7
0 0 7 5 0 0 0
0 0 8 5 0 0 0
7 8 0 9 7 0 0
5 0 9 0 15 6 0
0 5 7 15 0 8 9
0 0 0 6 8 0 11
0 0 0 0 9 11 0
	Minimum Spanning Weight: 39

Algorithm

```
Step 1: Sort all the edges in non-decreasing order of their weight.

Step 2: Pick the edge with the minimum weight.
        Check if it forms a cycle with the spanning tree formed so far.
            If cycle is not formed, include this edge.
            Else, discard it.

Step 3: Repeat step 2 until there are (N-1) edges in the spanning tree.
```

III. Assume that same road construction project is given to another person. The amount he will earn from this project is directly proportional to the budget of the project. This person is greedy, so he decided to maximize the budget by constructing those roads who have highest construction cost. Design an algorithm and implement it using a program to find the maximum budget required for the project.

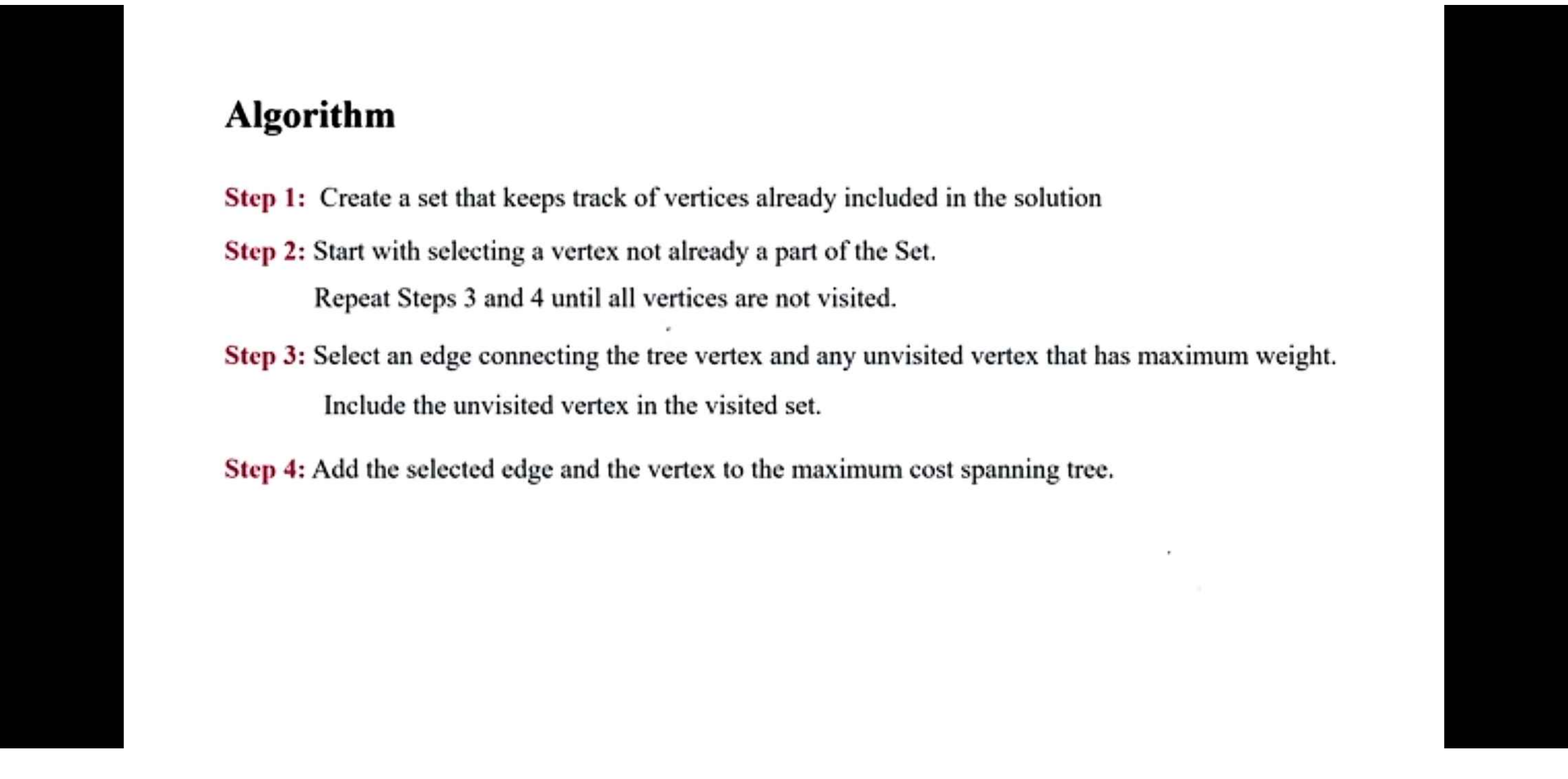
Input Format:
The first line of input takes number of vertices in the graph.
Input will be the graph in the form of adjacency matrix or adjacency list.

Output Format:
Out will be maximum spanning weight.

Sample I/O Problem III:

Input:	**Output:**

7
0 0 7 5 0 0 0
0 0 8 5 0 0 0
7 8 0 9 7 0 0
5 0 9 0 15 6 0
0 5 7 15 0 8 9
0 0 0 6 8 0 11
0 0 0 0 9 11 0
	Maximum Spanning Weight: 59



Algorithm

Step 1: Create a set that keeps track of vertices already included in the solution

Step 2: Start with selecting a vertex not already a part of the Set.

Repeat Steps 3 and 4 until all vertices are not visited.

Step 3: Select an edge connecting the tree vertex and any unvisited vertex that has maximum weight.

Include the unvisited vertex in the visited set.

Step 4: Add the selected edge and the vertex to the maximum cost spanning tree.