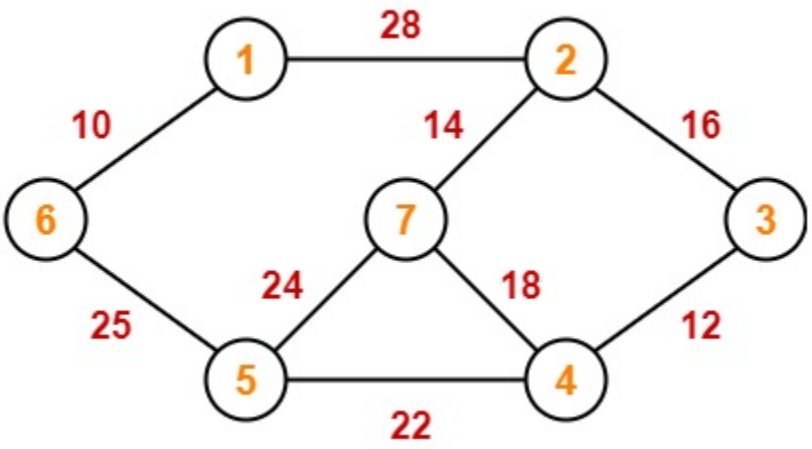
Handling Minimum Spanning Trees with Prim's Algorithms

Context

We have seen that graph theory is applied in computer science and mostly in computer networks. One of the computational tasks around graphs is finding the minimum spanning trees of networked devices. Computers will handle the computations in a way that is different from how humans do, and hence some algorithms they follow to accomplish the designated task.

Prim's algorithm is among the greedy approaches used to determine the MST from a given graph.



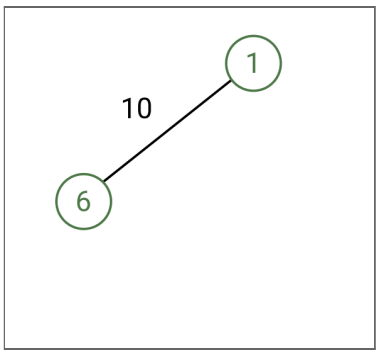
Questions:

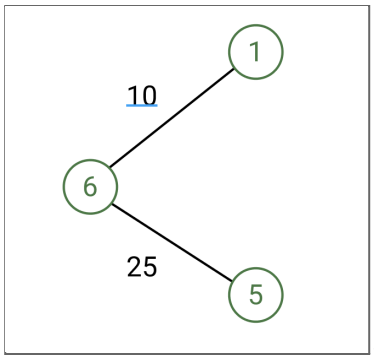
1. Use the step-by-step procedure to generate the minimum spanning tree.

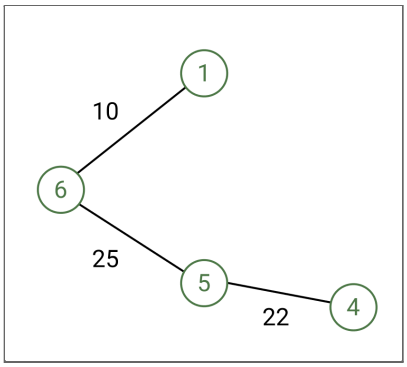
Answer:

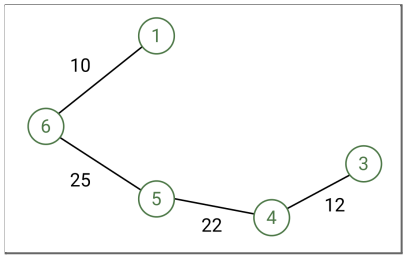
Using Prim's Algorithm - an algorithm used to find the MCS tree where you select the smallest edge as your initial point and always select the MC edge but make sure it's connected to the minimum spanning tree.

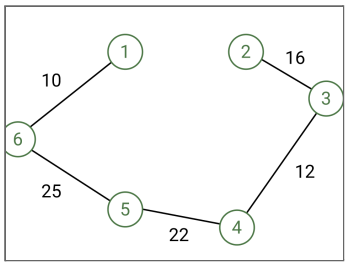
Step 1: Start with Edge 10, vertices 1 - 6 because it's the smallest edge.
Step 2: Look at connecting edgest to 6 or 1 and select the smallest edge among those. For this case, it's Edge 25 so vertices 6 - 5.
Step 3: Repeat step 2. Now it's edge 22, vertices 4 - 5
Step 4: Same logic, now it's edge 12
Step 5: For this, it's edge 16
Step 6: This is edge 14 and now we have a Spanning Tree

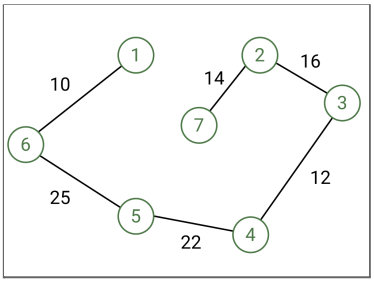
1. 

2. 

3. 

4. 

5. 

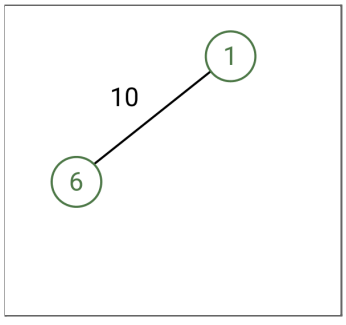
6. 

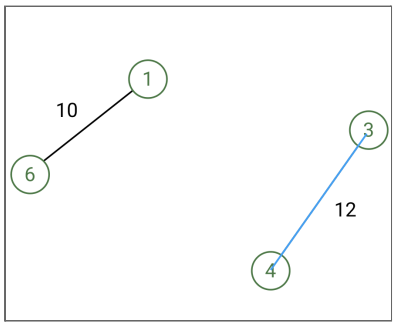
Now that we have a spanning tree we need to sum all the edges to get MCS

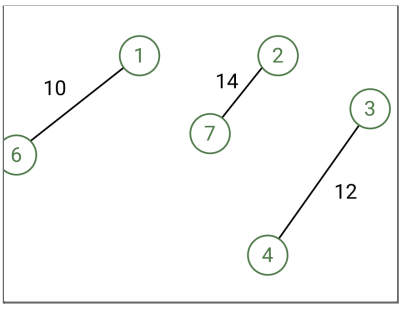
Therefor MCS = 99

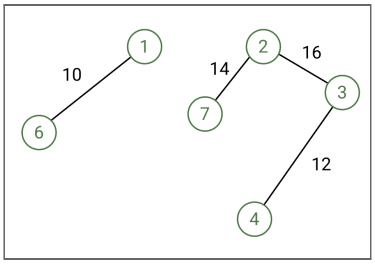
Using Kruskal's algorithm - It always follows a greedy approach where we always select minimum cost edges and connect them at the end.

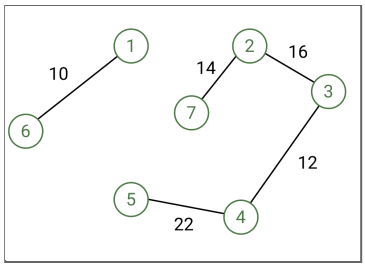
Step 1: the minimum edge is 10 so 1 - 6. But since it's not connected to the smallest we just proceed
Step 2: Next is edge 12 which is 4 - 3
Step 3: The next smallest edge is 14 so 2 - 7
Step 4: Next is edge 16 which is 2 - 3 but since they are connected to edge 14 we will join them together.
Step 5: Next is edge 18 but selecting that will form a circle so we ignore it and continue to edge 22 which is 4 - 5.
Step 6: Next is edge 24 but it will form a circle so we leave it and continue with edge 25 which is 6 - 5. Now it's complete.

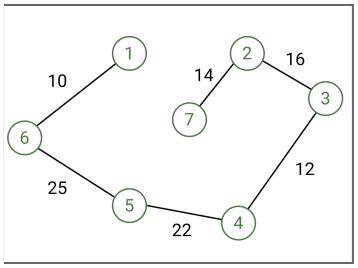
1. 

2. 

3. 

4. 

5. 

6. 

Therefor MCS = 99

Question 2:

Use pseudocode to write down Prim's algorithm to solve the problem. Use the pseudo code to analyze the running time complexity and express the running time in asymptotic analysis.

Pseudocode

1. Initialize a list to store the edges and weights of the vertices only - this will be an empty array
2. Initialize a variable to store MST (bool) so that we avoid duplicates and make sure we are not repeating the same vetics
3. Initialize a set (S) for all seen vertices.
4. Add initial starting vertex to the set of seen vertices. For this case it's {6}
5. Initialize the parent array to store the edges of the MST structure to point to it at the end - will use a parent array (type int)
6. Loop through the adjacent matrix from 0 to (v-1)
    a. Initialize a variable (cost) to store the minimum weight
    b. Initialize a start vertex, and end vertex and set their values to -1
    c. Loop over vertices we have in set Y -> (V1):
        i. If V2 in seen:
        ii. Continue
            1. If (the edge weight between v1 and v2 > 0) and (edge weight < cost):
                a. Start_vertex = V1
                b. end _vertex = V2
    d. Add end_vertex to seen set
    e. STORE (START) (END) (WEIGHT)
7. Print the weights and sum of edges.

Time complexity of the Algorithm = n^2 - This is because we have nested loops that checks for the inner most element.

Weakness of the algorithm

1. This will not work for a directed graph because it will force the algorithm to follow a path that is not desired by the algorithm.
2. When you have two edges which are the same the algorithm will give different MST according to the first edge being considered greater than the other.
3. When new edges are being added the algorithm will have to search again from the beginning.

Question 3:

Use a programming language of your choice to implement the algorithm (Prim's algorithm). Present the correctness of the implementation.

Link - https://github.com/Wenseslaus/PrimsAlgo