TOPIC 5.12 : VERIFY UNIQUENESS OF MINIMUM SPANNING TREE

Problem Statement  
Given a graph with weights and a potential Minimum Spanning Tree (MST), verify if the given MST is unique. If it is not unique, provide another possible MST.

Test Case 1  
Input:  
n = 4  
m = 5  
edges = [(0, 1, 10), (0, 2, 6), (0, 3, 5), (1, 3, 15), (2, 3, 4)]  
given\_mst = [(2, 3, 4), (0, 3, 5), (0, 1, 10)]  
Output:  
Is the given MST unique? True

Test Case 2  
Input:  
n = 5  
m = 6  
edges = [(0, 1, 1), (0, 2, 1), (1, 3, 2), (2, 3, 2), (3, 4, 3), (4, 2, 3)]  
given\_mst = [(0, 1, 1), (0, 2, 1), (1, 3, 2), (3, 4, 3)]  
Output:  
Is the given MST unique? False  
Another possible MST: [(0, 1, 1), (0, 2, 1), (2, 3, 2), (3, 4, 3)]  
Total weight of MST: 7

Aim  
To write a program that checks whether the given MST is unique, and if not, generates another possible MST of the same total weight.

Algorithm

1. Start
2. Compute the MST of the given graph using Kruskal’s Algorithm
3. Compare the weight of the computed MST with the given MST
4. If the computed MST weight differs from the given MST weight, the given MST is invalid
5. If equal, check for alternative MSTs:
   * If two or more edges with the same weight could be swapped while maintaining MST weight, then MST is not unique
6. Return whether the MST is unique and, if not, provide another MST
7. Stop

Input and Output  
A screenshot of a computer

AI-generated content may be incorrect.

Result  
The program successfully verifies the uniqueness of the given MST and provides another possible MST when it is not unique.

Performance Analysis  
Time Complexity: O(m log m) for Kruskal’s Algorithm  
Space Complexity: O(n) for disjoint set data structure