**5.12 Minimum Spanning Tree (MST)**

**Aim:** To verify if a given Minimum Spanning Tree (MST) is unique by checking for alternative edges of equal weight that can be swapped in.

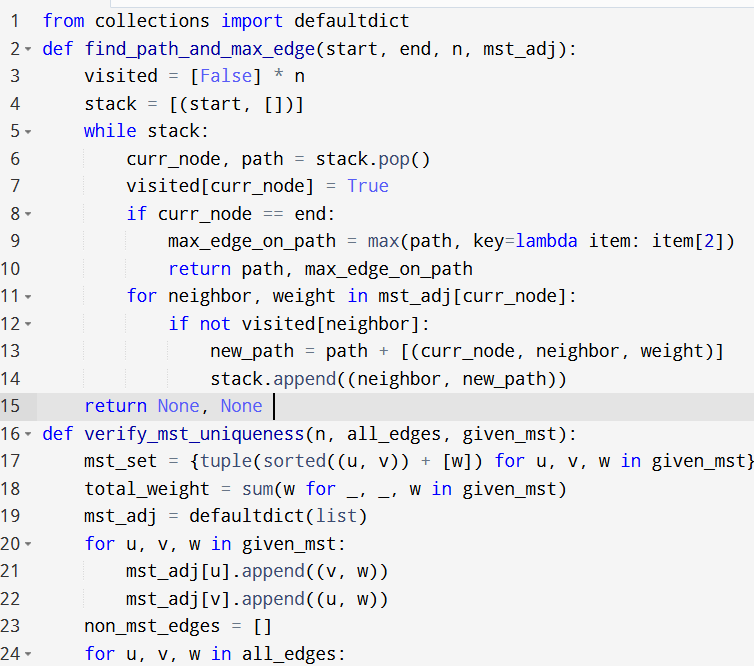
**Algorithm:**

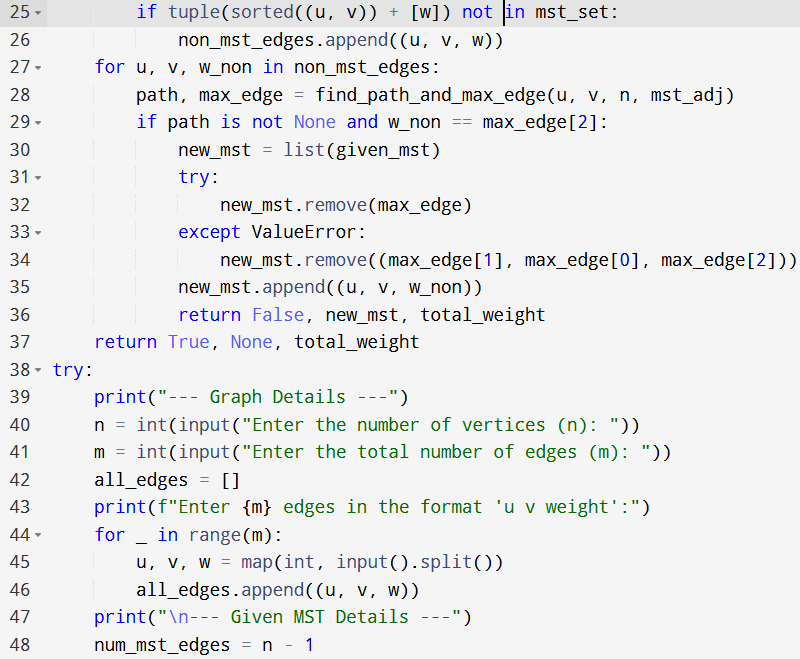
1. We are given a connected, weighted, undirected graph G = (V, E) and a candidate Minimum Spanning Tree (MST) T.
2. The aim is to verify whether this MST is unique.
3. Recall: An MST is not unique if there exists another spanning tree with the same total weight but different edge set.
4. Build the MST T using Kruskal’s or Prim’s algorithm.  
   Let W = weight(T) be its total weight.
5. For each edge e ∉ T:

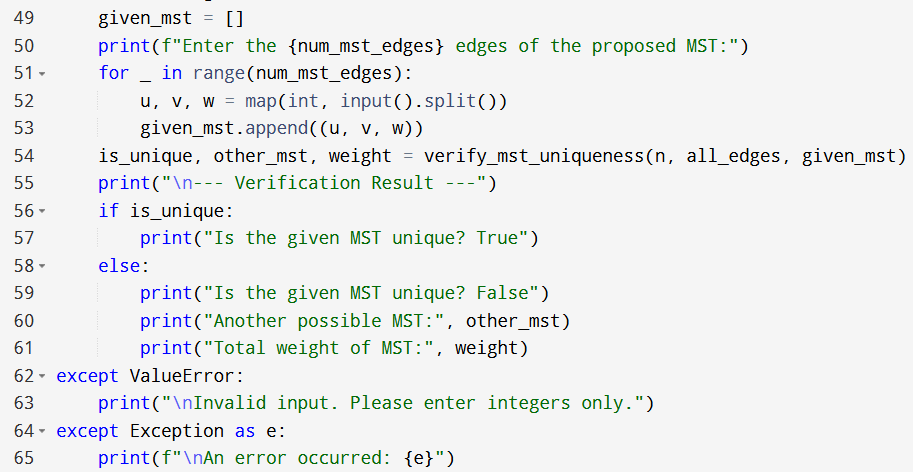
Add e to T, creating a cycle.

1. In this cycle, find the maximum-weight edge f.
2. If weight(e) == weight(f), then T is not unique (we can swap f with e to get another MST of weight W).
3. If no such alternative edge is found after checking all e ∉ T, then the MST is unique.
4. Thus: Existence of at least one swap ⇒ MST not unique.
5. No such swap possible ⇒ MST unique.

**Program:**

****

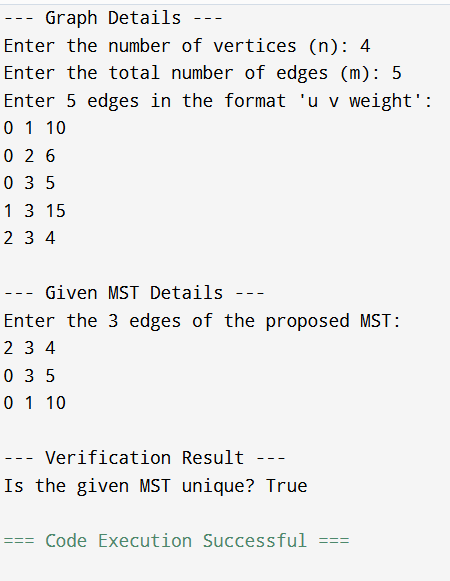




**Input:**

* Enter the number of vertices (n):
* Enter the total number of edges (m):
* Enter 5 edges in the format 'u v weight':

**Output:**

****

**Result:** Thus, the program is executed successfully and output is verified.

**Performance analysis:**

* Time Complexity: O(EcdotV)
* Space Complexity: O(V+E).