Programming Assignment 5 : Spanning Trees – Kruskal's and Prim's Algorithms

Objective

The goal of this assignment is to implement two classical algorithms discussed in class to find the **Minimum Spanning Tree (MST)** of a graph: **Kruskal's Algorithm** and **Prim's Algorithm**. You will understand the differences between these algorithms and learn how to apply them to connected, undirected, weighted graphs.

Instructions

You are required to implement the following tasks:

1 Kruskal's Algorithm

Implement Kruskal's algorithm to find the minimum spanning tree of a graph.

Steps

- **Input:**
  - The graph will be represented as an edge list. Each edge is a tuple (u, v, w) where u and v are the vertices connected by the edge, and w is the weight of the edge.
  - Example: [(0, 1, 4), (0, 2, 3), (1, 2, 1),...]
- **Output:**
  - Return the edges included in the MST and the total weight of the MST.

Functions to Implement

- Kruskal(graph_edges, num_vertices) −> Tuple[List[Tuple[int, int, int]], int]: Returns the edges in the MST and the total weight.

2 Prim's Algorithm

Implement Prim's algorithm to find the minimum spanning tree of a graph.

Steps

- **Input:**
  - The graph will be represented as an adjacency matrix. If two vertices are not directly connected, the weight will be represented as float ('inf').
  - Example:

$$\begin{bmatrix} 0 & 4 & 3 & \infty & \infty \\ 4 & 0 & 1 & 2 & \infty \\ 3 & 1 & 0 & 4 & 5 \\ \vdots & \vdots & \vdots & \vdots & \vdots \end{bmatrix}$$

- **Output:**
  - Return the edges included in the MST and the total weight of the MST.

Functions to Implement

- Prim(graph_matrix) −> Tuple[List[Tuple[int, int, int]], int]: Returns the edges in the MST and the total weight.

3 Spanning Tree Validation

Write a function to check if the given set of edges forms a valid spanning tree for a graph.

Steps

- **Input:**
  - A graph represented as an adjacency list or adjacency matrix.
  - A list of edges in the form [(u, v, w), (v, x, y)...] .
- **Output:**
  - Return True if the edges form a valid spanning tree, False otherwise.

Function to Implement

- Is_spanning_tree(graph_edges, edges_in_tree , num_vertices) −> bool: Returns whether the edges form a valid spanning tree.