Project – Graphs

This project revolves around Graphs and specifically how to identify the shortest path between two vertices in a Graph. Your task is to implement Dijkstra's algorithm.

The class Graph is yours to implement, but the following functions must be defined:

```
Graph();
~Graph();
void addEdge(std::string v, std::string w, int weight); // Adds edges (v, w, weight)
int shortestPath(std::string v, std::string w);   // Performs Dijkstras and returns the path cost
std::string getShortestPath();                    // Returns last performed shortestPath as a string
void readGraphFromFile(std::string pathToFile) // Reads a graph from a file
```

You can assume that all graphs that are given are connected, undirected, weighted, and non-empty. All other concerns must be addressed. I recommend you use std::vectors to make your adjacency list or adjacency matrix, but you are free to choose however you seem fit.

The Graph class should function in such a way that when you have read a graph from file, one calls shortestPath(..) and gets the path cost. It should then be possible to get the shortest path as a string containing all vertices v_1, …, v_k from v to w using the getShortestPath function.

You are given test files, named graph1.txt to graph4.txt. The structure present in these files is to be read by the function **readGraphFromFile**.

Requirements:

- The class Graph shall be implemented in the files Graph.h and Graph.cpp
- No memory leaks are allowed.
- Your implementation **must** follow the proven complexity for Dijkstra's algorithm