**SHORTEST DISTANCE BETWEEN THE CITIES OF PAKISTAN- DSA LAB PROJECT REPORT**

GROUP MEMBERS:

1. HASAN ABBAS – 2022204
2. MUHAMMAD MURTAZA – 2022402
3. HASSAN AHMED – 2022211

INTRODUCTION:

This project implements a C++ program that models a graph and efficiently finds the shortest path between two vertices using the Breadth-First Search (BFS) algorithm. The program reads a graph from a text file, allowing for flexible representation of various graph structures. It provides a user-friendly interface to inquire about the shortest path between specified vertices.

Language Choice:

C++ was chosen for this project due to its:

Object-oriented capabilities:

Encapsulating graph data and algorithms within classes.

Efficiency in handling collections and algorithms:

Well-suited for graph-related operations.

Standard Template Library (STL):

Offers versatile data structures like unordered\_map, list, and queue, essential for graph implementation.

File I/O capabilities:

Enables reading graph data from external files.

Key Features

Graph Class:

Represents the graph using an unordered\_map to store vertices and their adjacent edges.

Methods:

Add\_edge():

Adds an edge between two vertices with a specified weight.

Print\_Graph():

Prints a visual representation of the graph for debugging and understanding.

Read\_Graph\_From\_File():

Reads graph data from a text file, ensuring flexibility in graph creation.

BFS ():

Implements the Breadth-First Search algorithm to find the shortest path between two vertices.

Main Function:

Creates a graph object.

Reads the graph data from a file.

Prompts the user for the starting and destination vertices.

Calls the BFS () function to find and display the shortest path.

Data Structures:

Unordered map:

Efficiently maps vertices (strings) to their adjacency lists.

List:

Stores each vertex's adjacent edges as pairs of (neighboring vertex, weight).

Queue:

Facilitates the BFS traversal, ensuring vertices are visited in a level-by-level manner.

Algorithm:

Breadth-First Search (BFS):

Systematically explores the graph level by level, ensuring the shortest path is found first.

User Interface:

Prompts the user for the starting and destination vertices.

Displays the shortest path (if found) along with its total distance.

File Handling:

Reads graph data from a text file, allowing for flexibility in graph creation and modification.

Conclusion:

The Graph Traversal and Shortest Path Finder program successfully demonstrates graph representation and BFS implementation in C++. It provides a practical tool for exploring graph structures and finding shortest paths between vertices.

USER-INTERFACE:

