```cpp
#include <iostream>
#include <limits.h>
using namespace std;

class FlightNetwork {
    int n;  // Number of cities
    int adjacent[10][10];  // Adjacency matrix for flight costs
    string city[10];  // City names

public:
    void input();
    void display();
    void Prims();
};

// Function to input flight details
void FlightNetwork::input() {
    cout << "\nEnter the number of cities: ";
    cin >> n;

    cout << "\nEnter the names of the cities:\n";
    for (int i = 0; i < n; i++)
        cin >> city[i];

    cout << "\nEnter the flight time (or cost) between cities:\n";
    for (int i = 0; i < n; i++)
        for (int j = i; j < n; j++) {
            if (i == j) {
                adjacent[i][j] = 0;  // No self-loop
                continue;
            }
            cout << "Enter the cost to connect " << city[i] << " and " << 
city[j] << ": ";
            cin >> adjacent[i][j];
            adjacent[j][i] = adjacent[i][j];  // Undirected graph (bi-
directional flights)
        }
}

// Function to display the adjacency matrix
void FlightNetwork::display() {
    cout << "\nFlight Cost Adjacency Matrix:\n";
    for (int i = 0; i < n; i++) {
        for (int j = 0; j < n; j++)
            cout << adjacent[i][j] << "\t";
        cout << "\n";
    }
}

// Function to find the Minimum Spanning Tree using Prim's Algorithm
void FlightNetwork::Prims() {
    int visit[n], minCost = 0, count = n - 1, minIndex, cost = 0;

    for (int i = 0; i < n; i++)
        visit[i] = 0;

    cout << "\n\nOptimal Flight Route (Minimum Spanning Tree):\n";
    visit[0] = 1;
    cout << city[0] << " -> ";

    while (count--) {
        minCost = INT_MAX;

        for (int i = 0; i < n; i++) {
            for (int j = 0; j < n; j++) {
                if (visit[i] == 1 && adjacent[i][j] != 0 && adjacent[i][j] < minCost && visit[j] == 0) {
                    minCost = adjacent[i][j];
                    minIndex = j;
                }
            }
        }

        visit[minIndex] = 1;
        cout << city[minIndex] << " -> ";
        cost += minCost;
    }

    cout << "End\n";
    cout << "Minimum Total Flight Cost: " << cost << "\n";
}

// Main function
int main() {
    FlightNetwork network;
    int choice;

    do {
        cout << "\n\nFLIGHT NETWORK (Minimum Spanning Tree Algorithm)";
        cout << "\n1. Input Flight Data";
        cout << "\n2. Display Flight Cost Matrix";
        cout << "\n3. Find Optimal Flight Connections (MST)";
        cout << "\n4. Exit";
        cout << "\nEnter your choice: ";
        cin >> choice;

        switch (choice) {
            case 1:
                network.input();
                break;
            case 2:
                network.display();
                break;
            case 3:
                network.Prims();
                break;
        }
    } while (choice != 4);

    return 0;
}

```