Assignment no.7

Problem Statement:

You have a business with several offices; you want to lease phone lines to connect them up with each other; and the phone company charges different amounts of money to connect different pairs of cities. You want a set of lines that connects all your offices with a minimum total cost. Solve the problem by suggesting appropriate data structures.

INPUT:

```
#include <iostream>

#include <limits.h> using

namespace std;


class Office {

   int n;

   int adjacent[10][10];

string office[10];


public:

   void input ();

void display ();

void Prims ();

};


void Office::input () {     cout <<

"\nEnter no. of offices: ";     cin >>

n;

   cout << "\nEnter the names of offices: ";

   for (int i = 0 ; i < n ; i++)

cin >> office[i];


   cout << "\nEnter the cost to connect the offices: \n";
   for (int i = 0 ; i < n ; i++)
for (int j = i ; j < n ; j++) {
          if (i == j) {
adjacent[i][j] = 0;
continue;
          }

          cout << "Enter the cost to connect " << office[i] <<" and " << office[j]<< " : ";
cin >> adjacent[i][j];           adjacent[j][i] = adjacent[i][j];
       }
}

void Office::display () {

   for (int i = 0 ; i < n ; i++) {        cout
<< "\n";        for (int j = 0 ; j < n ; j++)
{            cout << adjacent[i][j] <<
"\t";
       }
  }
}

void Office::Prims () {     int visit[n], minCost = 0, count = n -
1, minIndex, cost = 0;
   for (int i = 0 ; i < n ; i++)
visit[i] = 0;

   cout << "\n\nShortest path: ";
visit[0]=1;
   cout << office[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 <<
office[minIndex] << " -> ";       cost =
cost + minCost;
   }
   cout << "End";

   cout << "\nMinimum cost: "<<cost;

}

int main () {
Office o1;    int
choice;    do {
      cout << "\n\nMINIMUM SPANNING TREE"
          << "\n1. Input data"
          << "\n2. Display data"
          << "\n3. Calculate minimum cost"
          << "\nEnter your choice: ";
      cin >> choice;
switch (choice) {
case 1:
o1.input ();
break;           case 2:
o1.display ();
break;           case 3:
o1.Prims ();
break;
       }
   } while (choice != 4);
return 0;
}
```

OUTPUT:

```
/*
MINIMUM SPANNING TREE
1. Input data
2. Display data
3. Calculate minimum cost
Enter your choice: 1

Enter no. of offices: 4

Enter the names of offices: A B C D

Enter the cost to connect the offices:
Enter the cost to connect A and B : 8
Enter the cost to connect A and C : 4
Enter the cost to connect A and D : 5
Enter the cost to connect B and C : 0
Enter the cost to connect B and D : 3
Enter the cost to connect C and D : 6


MINIMUM SPANNING TREE
```

```
1. Input data

2. Display data

3. Calculate minimum cost

Enter your choice: 2


0     8     4     5

8     0     0     3

4     0     0     6

5     3     6     0


MINIMUM SPANNING TREE

1. Input data

2. Display data

3. Calculate minimum cost

Enter your choice: 3



Shortest path: A -> C -> D -> B -> End

Minimum cost: 12


MINIMUM SPANNING TREE

1. Input data

2. Display data

3. Calculate minimum cost

Enter your choice: 4

*/
```