

# Experiment N0. 8

**Aim :** 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.

**Code :**

```
#include<iostream>
using namespace std;

int main() {
    int n, i, j, k, row, col, mincost=0, min;
    char op;
    cout<<"Enter no. of vertices: ";
    cin>>n;
    int cost[n][n];
    int visit[n];
    for(i=0; i<n; i++)
        visit[i] = 0;
    for(i=0; i<n; i++)
        for(int j=0; j<n; j++)
            cost[i][j] = -1;
    for(i=0; i<n; i++) {
        for(j=i+1; j<n; j++) {
            cout<<"Do you want an edge between "<<i+1<<" and "<<j+1<<": ";
            cin>>op;
            if(op=='y' || op=='Y') {
                cout<<"Enter weight: ";
                cin>>cost[i][j];
                cost[j][i] = cost[i][j];
            }
        }
    }
    visit[0] = 1;
    for(k=0; k<n-1; k++) {
        min = 999;
        for(i=0; i<n; i++) {
            for(j=0; j<n; j++) {
                if(visit[i] == 1 && visit[j] == 0) {
                    if(cost[i][j] != -1 && min>cost[i][j]) {
                        min = cost[i][j];
                        row = i;
                        col = j;
                    }
                }
            }
        }
    }
}
```

```

    }
    }
    }
    mincost += min;
    visit[col] = 1;
    cost[row][col] = cost[col][row] = -1;
    cout<<row+1<<"->"<<col+1<<endl;
}
cout<<"\nMin. Cost: "<<mincost;
return 0;
}

```

## Output :

The screenshot shows a C++ IDE with the file 'prac8thdsal.cpp' open. The code implements a minimum spanning tree algorithm using Prim's algorithm. It prompts the user to enter the number of vertices and then asks for edges between vertices and their weights. The output window shows the execution results for 5 vertices and 10 edges, resulting in a minimum cost of 16.

```

//Aim: 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 amount
#include<iostream>
using namespace std;

int main() {
    int n, i, j, k, row, col, mincost=0, min;
    char op;
    cout<<"Enter no. of vertices: ";
    cin>>n;
    int cost[n][n];
    int visit[n];
    for(i=0; i<n; i++)
        visit[i] = 0;
    for(i=0; i<n; i++)
        for(j=0; j<n; j++)
            cost[i][j] = -1;
    for(i=0; i<n; i++) {
        for(j=i+1; j<n; j++) {
            cout<<"Do you want an edge between "<<i+1<<" and "<<j+1<<": ";
            cin>>op;
            if(op=='y' || op=='Y') {
                cout<<"Enter weight: ";
                cin>>cost[i][j];
                cost[j][i] = cost[i][j];
            }
        }
    }
    visit[0] = 1;
    for(k=0; k<n-1; k++) {
        min = 999;
        for(i=0; i<n; i++) {
            for(j=0; j<n; j++) {
                if(visit[i] == 1 && visit[j] == 0) {
                    if(cost[i][j] != -1 && min>cost[i][j]) {
                        min = cost[i][j];
                        row = i;
                        col = j;
                    }
                }
            }
        }
        mincost += min;
        visit[col] = 1;
        cost[row][col] = cost[col][row] = -1;
        cout<<row+1<<"->"<<col+1<<endl;
    }
    cout<<"\nMin. Cost: "<<mincost;
    return 0;
}

```

Output:

```

Enter no. of vertices: 5
Do you want an edge between 1 and 2: y
Enter weight: 6
Do you want an edge between 1 and 3: y
Enter weight: 3
Do you want an edge between 1 and 4: y
Enter weight: 8
Do you want an edge between 1 and 5: n
Do you want an edge between 2 and 3: y
Enter weight: 4
Do you want an edge between 2 and 4: n
Do you want an edge between 2 and 5: n
Do you want an edge between 3 and 4: 3
Do you want an edge between 3 and 5: y
Enter weight: 3
Do you want an edge between 4 and 5: y
Enter weight: 6
1->3
3->5
3->2
5->4

Min. Cost: 16

-----
Process exited after 173.1 seconds with return value 0
Press any key to continue . . .

```