SKILL DEVELOPMENT LAB-II(2018-19)

Assignment 5 :

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 pair 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 .

Objective :

We have to implement this using Minimum Spanning Tree with the use of graph data structure.

Theory :

Given a connected and undirected graph, a *spanning tree* of that graph is a subgraph that is a tree and connects all the vertices together. A single graph can have many different spanning trees. A *minimum spanning tree (MST)* or minimum weight spanning tree for a weighted, connected and undirected graph is a spanning tree with weight less than or equal to the weight of every other spanning tree. The weight of a spanning tree is the sum of weights given to each edge of the spanning tree.

Applications :

- Building a connected network.
- Clustering.
- Traveling salesman problem.
- Image registration and segmentation.

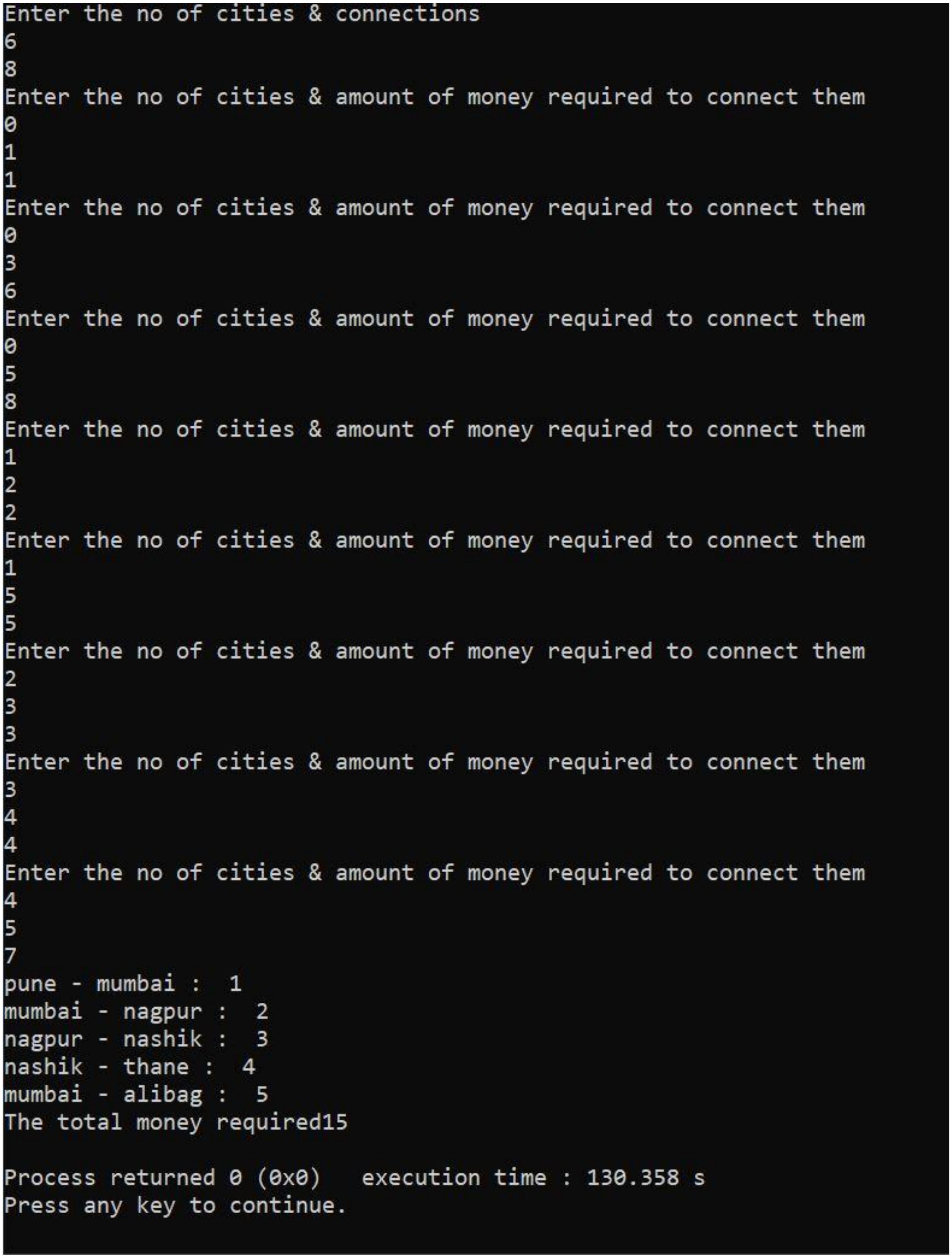
Program :

```
#include<iostream>

using namespace std;

class operation

{

   int ad[20][20],visited[20],i,j,a,b,c=0,w,k,l,s=0;

   string r[6];

public:
```

```
void inser()
{
  r[0]="pune";
  r[1]="mumbai";
  r[2]="nagpur";
  r[3]="nashik";
  r[4]="thane";
  r[5]="alibag";
  for(i=0;i<6;i++)
  {
    cout<<r[i]<<" ="<<i<<endl;
  }
  cout<<"Enter the no of cities & connections\n";
  cin>>a>>b;
  if(a>b)
  {
    cout<<"Error\n";
  }
  else
  {
  for(i=0;i<a;i++)
  {
    for(j=0;j<a;j++)
    {
      ad[i][j]=0;
```

```
        }
    }
     for(i=0;i<b;i++)
     {
        cout<<"Enter the no of cities & amount of money required to connect them\n";
        cin>>k>>l>>w;
        ad[k][l]=w;
     }
     prims();
    }
}
void prims()
{
    visited[0]=1;
    for(i=1;i<a;i++)
        visited[i]=0;
while (c<a-1)
{
   int min=9999,x=0,y=0;
   //for(i=0;i<a;i++)
    //{visited[i]=0;}
  for (int i = 0; i<a; i++)
    {
    if (visited[i]==1)
    {
      for (int j = 0; j <a; j++)
      {
        if (visited[j]==0 && ad[i][j])
        {
         if (min > ad[i][j])
          {
           min = ad[i][j];
           x = i;
           y = j;
          }
        }
      }
    }
   }
   s=s+ad[x][y];
   cout <<r[x] <<  " - " << r[y] << " :  " << ad[x][y];
   cout << endl;
   visited[y]=1;
   c++;
  }
  cout<<"The total money required"<<s<<endl;
  }
};
int main()
{
```

```
operation op;

op.inser();
```



```
Enter the no of cities & connections
6
8
Enter the no of cities & amount of money required to connect them
0
1
1
Enter the no of cities & amount of money required to connect them
0
3
6
Enter the no of cities & amount of money required to connect them
0
5
8
Enter the no of cities & amount of money required to connect them
1
2
2
Enter the no of cities & amount of money required to connect them
1
5
5
Enter the no of cities & amount of money required to connect them
2
3
3
Enter the no of cities & amount of money required to connect them
3
4
4
Enter the no of cities & amount of money required to connect them
4
5
7
pune - mumbai :  1
mumbai - nagpur :  2
nagpur - nashik :  3
nashik - thane :  4
mumbai - alibag :  5
The total money required15

Process returned 0 (0x0)   execution time : 130.358 s
Press any key to continue.
```

Conclusion :

Thus we have implemented minimum spanning tree using graph data structure.