Skill Development Lab-II (2018-2019)

BRACT's

VISHWAKARMA INSTITUTE OF INFORMATION TECHNOLOGY, PUNE – 48

An Autonomous Institute Affiliated to Savitribai Phule Pune University, Pune

SD(LP-II) ASSIGNMENT (S.Y.B. Tech. – DIV: C)

Name.: Prathamesh Gaikwad;
Roll no.: 223019;
GR. No.: 17u416; Batch: C1

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.

ALGORITHM:

Algorithm Prims(E,cost,n,t)

{

1.Let (k,l) be the edge of minimum cost

2.mincost=cost(k,l)

3.t[1,1]=k;t[1,2]=l;

4.for i=1 to n

do

If(cost[i,l]<cost[i,k]

then near[i]=l

Else near[i]=k;

5.near[k]=near[l]=0

6.for i=2 to n-1 do

6.1Let j be the index such that near[j]!=0 and Cost[j,near[j]] is minimum
6.2t[i,1]=j ;t[i ,2]=near[j]

6.3 mincost=mincost+cost[j,near[j]];

6.4 near[j]=0 6.5for k=1 to n do

if ((near[k]!=0) and (cost[k,near[k]]>cost[k,j]))

then near[k]=j

}

Return mincost

}

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";
```

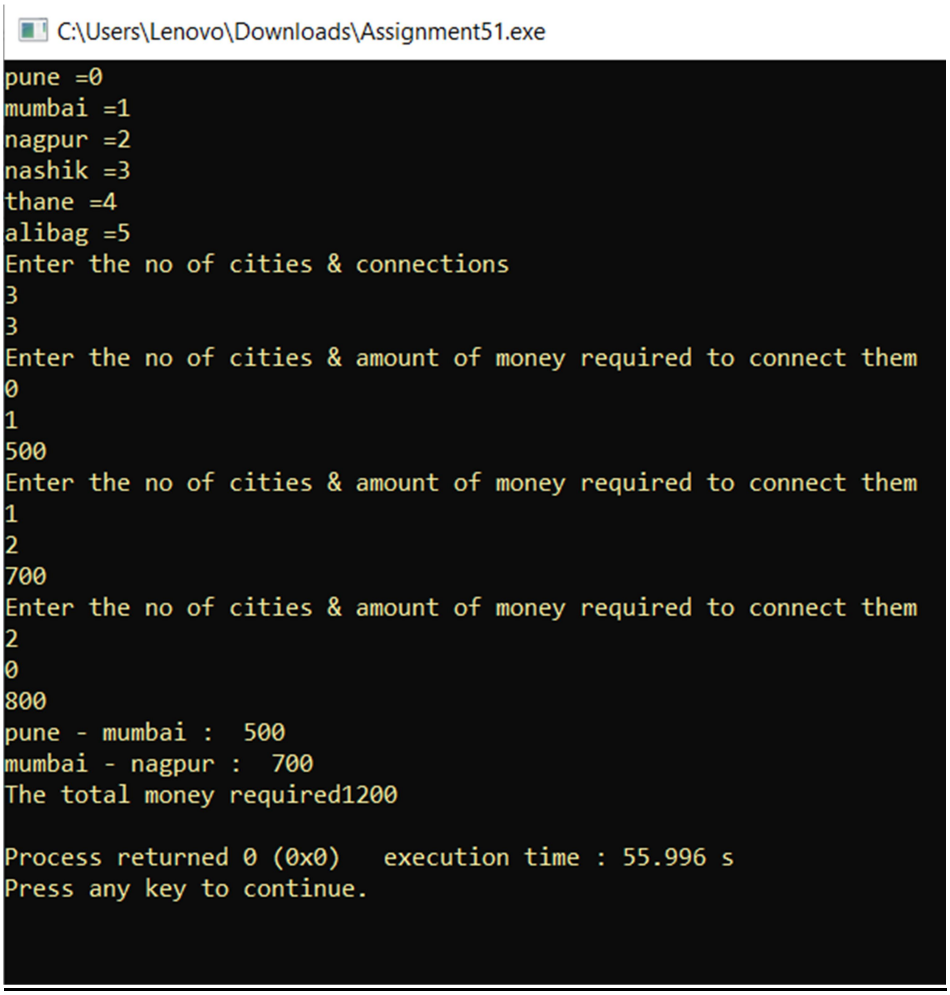
```
r[4]="thane";
r[5]="alibag";
for(i=0;i<6;i++)
{
   cout<<r[i]<<" ="<<i<<endl;
}
cout<<"Enter the no of cities & connections\n";
cin>>a>>b;
if(a>b)
{
   cout<<"Error\n";
}
else
{
for(i=0;i<a;i++)
{
   for(j=0;j<a;j++)
   {
      ad[i][j]=0;
   }
}
```

```
        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();
}
```

Output :



Conclusion :

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