NAME:-RISHIKESH SHEDE
CLASS:- SYBTECH-C
BATCH:- C-2
ROLL NO:- 223053
GR NO:- 17U208

ASSIGNMENT NO.4.

Aim :- For a weighted graph G, find the minimum spanning tree using Prims
algorithm .

Objective:- To study the prims algorithm.

Theory:-

Prim's algorithm is also a Greedy algorithm. It starts with an empty
spanning tree. The idea is to maintain two sets of vertices. The first set contains the
vertices already included in the MST, the other set contains the vertices not yet
included. At every step, it considers all the edges that connect the two sets, and picks
the minimum weight edge from these edges. After picking the edge, it moves the other
endpoint of the edge to the set containing MST.
A group of edges that connects two set of vertices in a graph is called cut in graph
theory. So, at every step of Prim's algorithm, we find a cut (of two sets, one contains the
vertices already included in MST and other contains rest of the verices), pick the
minimum weight edge from the cut and include this vertex to MST Set (the set that
contains already included vertices)
.
How does Prim's Algorithm Work? The idea behind Prim's algorithm is simple, a
spanning tree means all vertices must be connected. So the two disjoint subsets
(discussed above) of vertices must be connected to make a Spanning Tree. And they
must be connected with the minimum weight edge to make it a Minimum Spanning
Tree.

Algorithm:-

1) Create a set mstSet that keeps track of vertices already included in MST.

2) Assign a key value to all vertices in the input graph. Initialize all key values as
INFINITE. Assign key value as 0 for the first vertex so that it is picked first.

3) While mstSet doesn't include all vertices
….**a)** Pick a vertex u which is not there in mstSet and has minimum key value.
….**b)** Include u to mstSet.

S.Y.-C,Department of Computer Engineering,VIIT,2018-19

NAME:-RISHIKESH SHEDE

CLASS:- SYBTECH-C

BATCH:- C-2

ROLL NO:- 223053

GR NO:- 17U208

….**c)** Update key value of all adjacent vertices of u. To update the key values, iterate through all adjacent vertices. For every adjacent vertex v, if weight of edge u-v is less than the previous key value of v, update the key value as weight of u-v

The idea of using key values is to pick the minimum weight edge from <u>cut</u>. The key values are used only for vertices which are not yet included in MST, the key value for these vertices indicate the minimum weight edges connecting them to the set of vertices included in MST.

Program Code:-

```cpp
#include<iostream>

#include<conio.h>

#include<stdlib.h>

using namespace std;

int cost[10][10],i,j,k,n,stk[10],top,v,visit[10],visited[10],u;

main()
{
        int m,c;

        cout <<"enterno of vertices";

        cin >> n;

        cout <<"ente no of edges";

        cin >> m;

        cout <<"\nEDGES Cost\n";
```

```cpp
for(k=1;k<=m;k++)

{

        cin >>i>>j>>c;

        cost[i][j]=c;

}

for(i=1;i<=n;i++)

for(j=1;j<=n;j++)

        if(cost[i][j]==0)

        cost[i][j]=31999;


cout <<"ORDER OF VISITED VERTICES";

k=1;

while(k<n)

{

        m=31999;

        if(k==1)

        {

                for(i=1;i<=n;i++)

                        for(j=1;j<=m;j++)

                        if(cost[i][j]<m)

                        {

                                m=cost[i][j];

                                u=i;
```

```
                    }
            }
            else
            {
                    for(j=n;j>=1;j--)
                    if(cost[v][j]<m && visited[j]!=1 && visit[j]!=1)
                    {
                            visit[j]=1;
                            stk[top]=j;
                            top++;
                            m=cost[v][j];
                            u=j;
                    }
            }
            cost[v][u]=31999;
            v=u;
            cout<<v << " ";
            k++;
            visit[v]=0; visited[v]=1;
        }
}
```
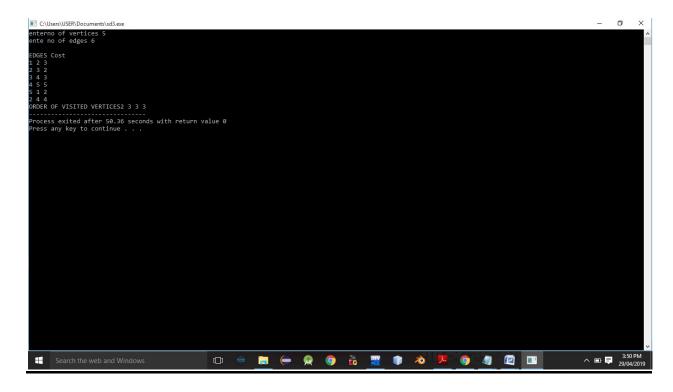
Output Screenshots:-

NAME:-RISHIKESH SHEDE

CLASS:- SYBTECH-C

BATCH:- C-2

ROLL NO:- 223053

GR NO:- 17U208

Conclusion:- Thus,we have studied prims algorithm.