Name: Rishi Tiku

Class: SE DS

UID: 2021700067

Subject: DAA LAB

Experiment: 6

Aim: To Implement Prim's Algorithm for finding Minimum Spanning Tree.

Problem:

Given a graph of 'v' vertices, find the corresponding minimum spanning tree using Prim's greedy Algorithm.

Theory:

Spanning tree - A spanning tree is the subgraph of an undirected connected graph.

Minimum Spanning tree - Minimum spanning tree can be defined as the spanning tree in which the sum of the weights of the edge is minimum. The weight of the spanning tree is the sum of the weights given to the edges of the spanning tree.

Prim's Algorithm for MST

Prim's algorithm is a Greedy algorithm. This algorithm always starts with a single node and moves through several adjacent nodes, in order to explore all of the connected edges along the way.

The algorithm 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, and 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.

The working of Prim's algorithm can be described by using the following steps:

Step 1: Determine an arbitrary vertex as the starting vertex of the MST.
Step 2: Follow steps 3 to 5 till there are vertices that are not included in the MST

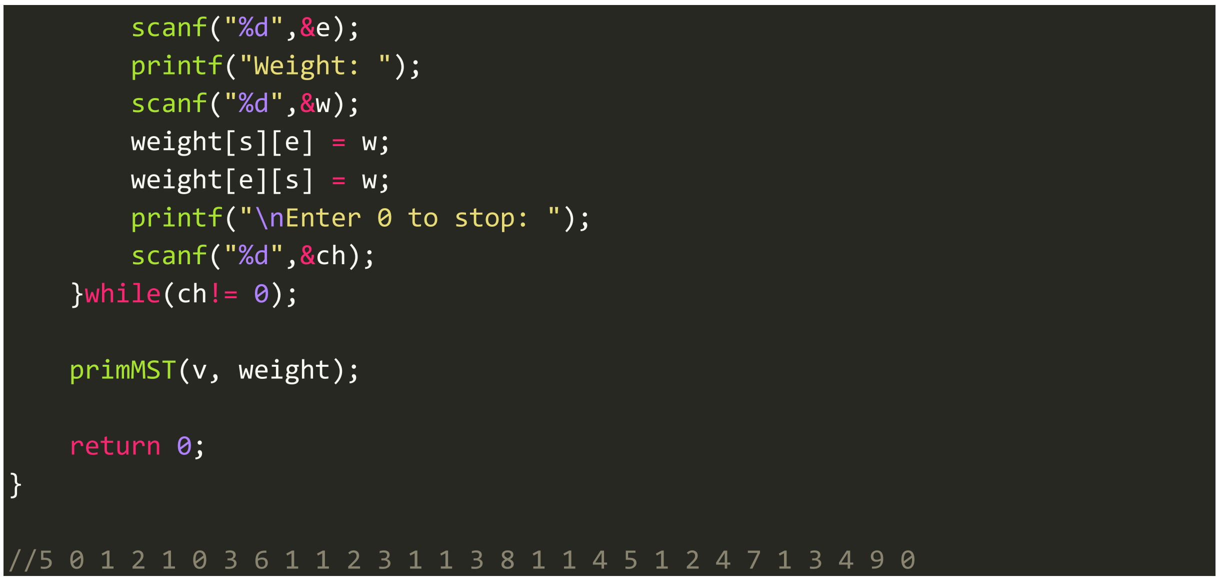
(known as fringe vertex).
Step 3: Find edges connecting any tree vertex with the fringe vertices.
Step 4: Find the minimum among these edges.
Step 5: Add the chosen edge to the MST if it does not form any cycle.
Step 6: Return the MST and exit

Time Complexity: O(V^2), If the input graph is represented using an adjacency list, then the time complexity of Prim's algorithm can be reduced to O(E * logV) with the help of a binary heap. In this implementation, we are always considering the spanning tree to start from the root of the graph
Auxiliary Space: O(V)

Code

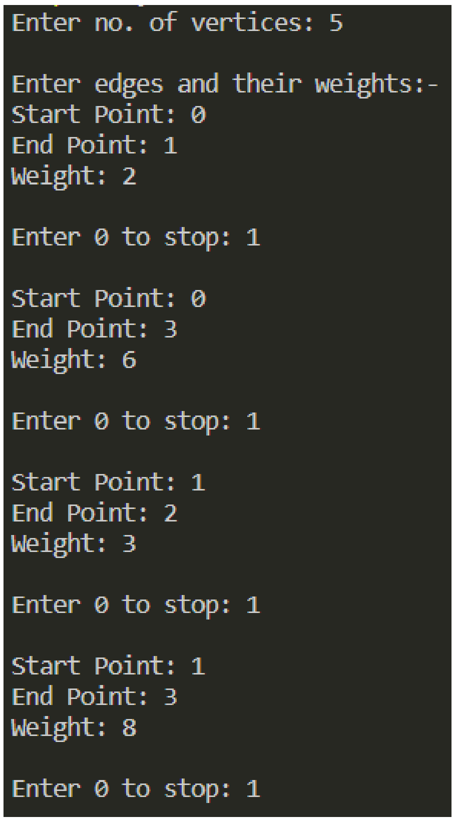
```c
#include <limits.h>
#include <stdbool.h>
#include <stdio.h>

int minKey(int V, int key[], bool used[])
{
    int min = INT_MAX, min_index;

    for (int ver = 0; ver < V; ver++)
    {
        if (used[ver] == false && key[ver] < min)
        {
            min = key[ver];
            min_index = ver;
        }
    }
    return min_index;
}

int printMST(int V, int parent[], int weight[V][V])
{
    int cost = 0;
    printf("\nEdge \t|\tWeight\n----------------------\n");
    for (int i = 1; i < V; i++)
    {
        printf("%d - %d \t|\t%d \n", parent[i], i, weight[parent[i]][i]);
        cost += weight[parent[i]][i];
    }
    printf("\nTotal cost = %d\n", cost);
}
```

```
void primMST(int V, int weight[V][V])
{
    int parent[V];
    int key[V];
    bool used[V];

    for (int i = 0; i < V; i++)
    {
        key[i] = INT_MAX, used[i] = false;
    }

    key[0] = 0;

    parent[0] = -1;

    for (int count = 0; count < V - 1; count++)
    {
        int u = minKey(V, key, used);
        used[u] = true;

        for (int v = 0; v < V; v++)
        {
            if (weight[u][v] && used[v] == false && weight[u][v] < key[v])
            {
                parent[v] = u;
                key[v] = weight[u][v];
            }
        }
    }

    printMST(V, parent, weight);
}

int main()
{
    int s, e, w, v, ch;

    printf("Enter no. of vertices: ");
    scanf("%d", &v);

    int weight[v][v];

    printf("\nEnter edges and their weights:-");
    do{
        printf("\nStart Point: ");
        scanf("%d",&s);
        printf("End Point: ");
```

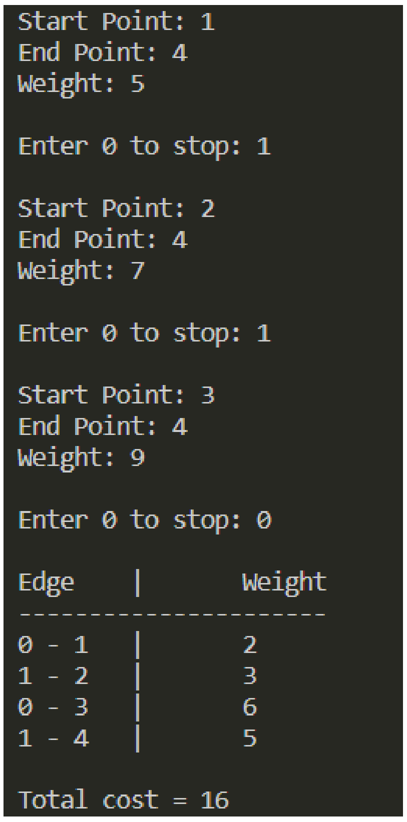


```c
        scanf("%d",&e);
        printf("Weight: ");
        scanf("%d",&w);
        weight[s][e] = w;
        weight[e][s] = w;
        printf("\nEnter 0 to stop: ");
        scanf("%d",&ch);
    }while(ch!= 0);

    primMST(v, weight);

    return 0;
}

//5 0 1 2 1 0 3 6 1 1 2 3 1 1 3 8 1 1 4 5 1 2 4 7 1 3 4 9 0
```

Output



```
Enter no. of vertices: 5

Enter edges and their weights:-
Start Point: 0
End Point: 1
Weight: 2

Enter 0 to stop: 1

Start Point: 0
End Point: 3
Weight: 6

Enter 0 to stop: 1

Start Point: 1
End Point: 2
Weight: 3

Enter 0 to stop: 1

Start Point: 1
End Point: 3
Weight: 8

Enter 0 to stop: 1
```



```
Start Point: 1
End Point: 4
Weight: 5

Enter 0 to stop: 1

Start Point: 2
End Point: 4
Weight: 7

Enter 0 to stop: 1

Start Point: 3
End Point: 4
Weight: 9

Enter 0 to stop: 0

Edge    |       Weight
------------------------
0 - 1   |       2
1 - 2   |       3
0 - 3   |       6
1 - 4   |       5

Total cost = 16
```

Conclusion

Prim's Algorithm has been used to obtain the minimum spanning tree of a given graph.