



**THE UNIVERSITY
OF LAHORE
ISLAMABAD
CAMPUS**

Data Structures & Algorithms (CS09203)

Lab Report

Name: Muhammad Umer
Registration #: CSU-F16-104
Lab Report #: 12
Dated: 25-06-2018
Submitted To: Mr. Usman Ahmed

The University of Lahore, Islamabad Campus
Department of Computer Science & Information Technology

Experiment # 1

Implement Prim MST (Minimum Spanning Tree) on the given graph.

Objective

To understand and implement the Prim Minimum Spanning Tree on the graph with different cycles.

Software Tool

1. Sublime Text Editor
2. Dev C++
3. Window 7 (32 Bit)

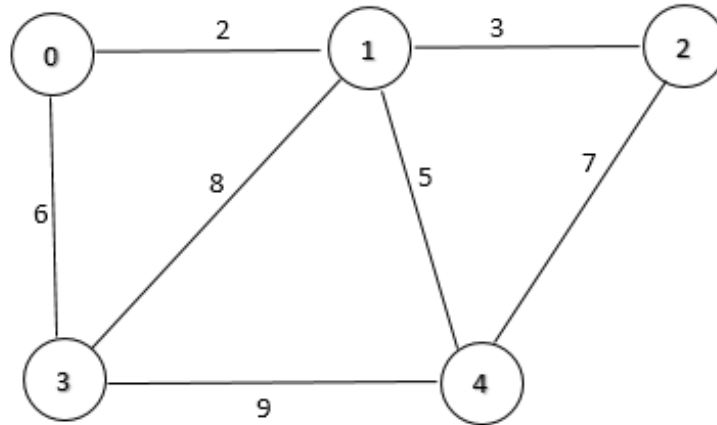
1 Theory

We have discussed Kruskals algorithm for Minimum Spanning Tree. Like Kruskals algorithm, Prims 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.

The idea behind Prims 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.

1.1 Procedure: Task 1 Implement Prim MST on graph

Implement the Prim MST Minimum Spanning Tree on the following graph:



```

#include <stdio.h>
#include <limits.h>
#define V 5
int minKey(int key[], bool mstSet[])
{
    int min = INT_MAX, min_index;

    for (int v = 0; v < V; v++)
        if (mstSet[v] == false && key[v] < min)
            min = key[v], min_index = v;
    return min_index;
}
int printMST(int parent[], int n, int graph[V][V])
{
    printf("Edge\t\tWeight\n");
    for (int i = 1; i < V; i++)
        printf("%d\t-%d\t\t\t%d\n", parent[i], i, graph[i][parent[i]]);
}
void primMST(int graph[V][V])
{
    int parent[V]; // Array to store constructed MST
    int key[V];    // Key values used to pick minimum weight edge in cut
    bool mstSet[V]; // To represent set of vertices not yet included in MST

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

```

```

    key[0] = 0;          // Make key 0 so that this vertex is picked as first
    parent[0] = -1;
    for (int count = 0; count < V-1; count++)
    {
        int u = minKey(key, mstSet);
        mstSet[u] = true;
        for (int v = 0; v < V; v++)
            if (graph[u][v] && mstSet[v] == false && graph[u][v] <
key[v])
                parent[v] = u, key[v] = graph[u][v];
        }
        printMST(parent, V, graph);
    }
}
int main()
{
    int graph[V][V] = {{0, 2, 0, 6, 0},
                        {2, 0, 3, 8, 5},
                        {0, 3, 0, 0, 7},
                        {6, 8, 0, 0, 9},
                        {0, 5, 7, 9, 0},
                        };

    // Print the solution
    primMST(graph);

    return 0;
}

```

Output: Consider the Figure 1 for the output of the above code in the end of this document.

Source Code

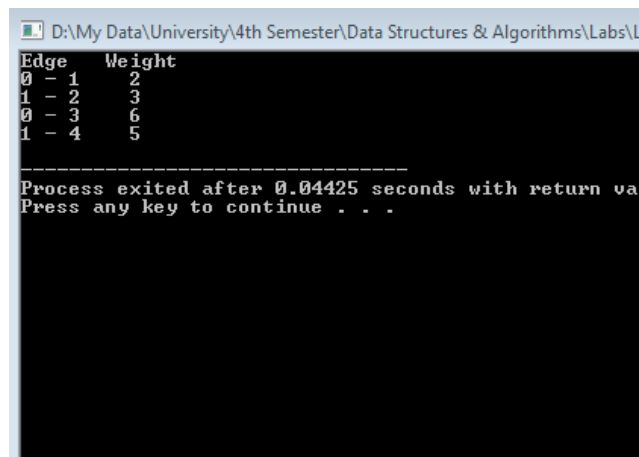
<https://goo.gl/ccBvqK>

2 Conclusion

Graphs are used to represent many real life applications: Graphs are used to represent networks. The networks may include paths in a city or telephone

network or circuit network. Prim Minimum Spanning Tree algorithm can be implemented on graphs to visit every node in the shortest path, graphs may contain cycles, so we may come to the same node again. To avoid processing a node more than once, we use a boolean visited array.

(Concerned Teacher/Lab Engineer)



```
D:\My Data\University\4th Semester\Data Structures & Algorithms\Labs\l
Edge  Weight
0 - 1  2
1 - 2  3
0 - 3  6
1 - 4  5
-----
Process exited after 0.04425 seconds with return va
Press any key to continue . . .
```

Figure 1: Minimum Spanning Tree implementation on graph