Data Structures and Algorithms
(CS09203)

Lab Report

Name:	Farhan Naseer
Registration #:	SEU-F16-125
Lab Report #:	12
Dated:	28-06-2018
Submitted To:	Sir. Usman Ahmed

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

Experiment # 12
Implementation of Minimum spaning tree

Objective
The objective of this session is to implement and understand Minimum spaning tree.

Software Tool
1. I use Code Blocks with GCC compiler.

1 Theory

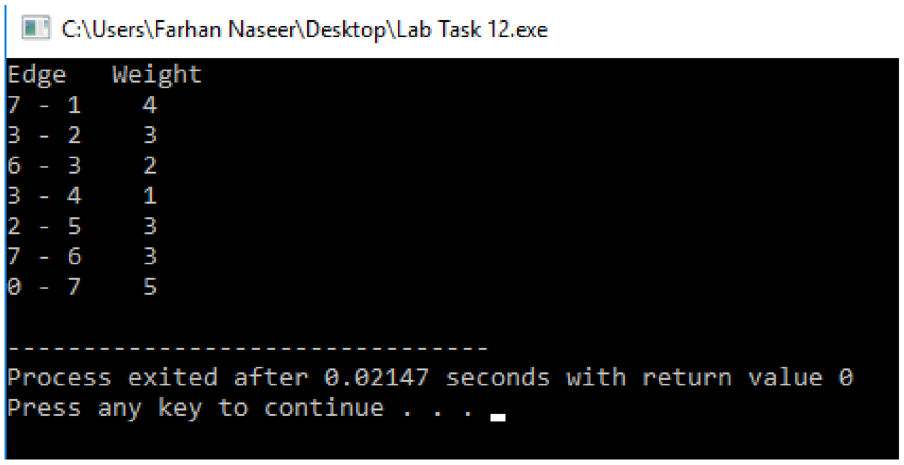
This section discusses how to create the graph and tell the number of edges and vertices . trees are used to model electrical circuits, chemical compounds, highway maps, and so on. They are also used in the analysis of electrical circuits, finding the shortest route, project planning, linguistics, genetics, social science, and so forth Undirected Edge - An undirected egde is a bidirectional edge. If there is a undirected edge between vertices A and B then edge (A , B) is equal to edge (B , A). Directed Edge - A directed egde is a unidirectional edge. If there is a directed edge between vertices A and B then edge (A , B) is not equal to edge (B , A). Weighted Edge - A weighted egde is an edge with cost on it.

2 Task

2.1 Procedure: Task 5

Write a C++ code using functions for the following operations. 1.Minimum Spaning Tree

2.2



```
C:\Users\Farhan Naseer\Desktop\Lab Task 12.exe
Edge   Weight
7 - 1    4
3 - 2    3
6 - 3    2
3 - 4    1
2 - 5    3
7 - 6    3
0 - 7    5

--------------------------------
Process exited after 0.02147 seconds with return value 0
Press any key to continue . . .
```

Figure 1: output

```
#include <stdio.h>
#include <limits.h>
 using namespace std;

#define V 8

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   Weight\n");
    for (int i = 1; i < V; i++)
        printf("%d - %d    %d \n", parent[i], i, graph[i][parent[i]]);
}
```

```
void primMST(int graph[V][V])
{
      int parent[V];
      int key[V];
      bool mstSet[V];
      for (int i = 0; i < V; i++)
         key[i] = INT_MAX, mstSet[i] = false;
      key[0] = 0;
      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] = {{1, 8, 0, 0, 0,10,0,5},
                      {8, 0, 4, 0, 4,4,0,4},
                      {0, 4, 0, 3, 0,3,0,0},
                      {0, 0, 3, 0, 1,6,2,0},
                      {0, 4, 0, 1, 0,0,3,0},
                         {10, 4, 3, 6, 0,0,0,0},
                      {0, 0, 0, 2, 3,0,0,3},
                      {5, 4, 0, 0, 0,0,3,0},
 };
        primMST(graph);

    return 0;
}
```

3 Conclusion

In today lab we have discussed how we can create Minimum Spaning tree and how to display it on a screen by a code.