"Heaven's Light is Our Guide"

Department of Computer Science & Engineering

RAJSHAHI UNIVERSITY OF ENGINEERING & TECHNOLOGY

Lab Report

Course No: CSE 2202

Course Name: Sessional Based on CSE 2201

Submitted to:

Biprodip Pal

Assistant Professor

Department of Computer Science & Engineering

Submitted by:

Md. Al Siam
Department of Computer Science & Engineering
Session: 2016-2017
Section: A
Roll No.: 1603008

Problem

Find the minimum spanning tree of a graph using Kruskal's algorithm.

Also output the the state of tree(s) in every step.

Solution in C++

```
#include <bits/stdc++.h>
using namespace std;

struct Edge{
    int u, v, wt;
    Edge(int _u, int _v, int _wt){
        u = _u;
        v = _v;
        wt = _wt;
    }
    bool operator < (const Edge &e2) const{
        return wt < e2.wt;
    }
};

vector<Edge> e;
int parent[100000];
int n;

vector<Edge> taken;

int num_of_edges;
```

```
void add_edge(int u, int v, int w){
    e.push_back(Edge(u, v, w));
}

int find_parent(int nd){
    if(parent[nd] == nd) return nd;
    return parent[nd] =
find_parent(parent[nd]);
}

void Kruskal_MST(){
    sort(e.begin(), e.end());

    int siz = e.size();

    for(int i = 0; i < n; i++){
        parent[i] = i;
    }

    int kount = 0;
    int ans = 0;

    for(int i = 0; i < siz; i++){
        int uu = find_parent(e[i].u);
        int vv = find_parent(e[i].v);
        if(uu != vv){
            parent[uu] = vv;
            ans += e[i].wt;
            kount++;
            taken.push_back(e[i]);

            ///Task 2
            printf("\n\n");
            int tsz = taken.size();
            set <int> parent_of_taken;
```

```
            for(int i = 0; i < tsz; i++){
                printf("%d --- %d\n",
taken[i].u, taken[i].v);

parent_of_taken.insert(find_parent(taken[i].u
));

parent_of_taken.insert(find_parent(taken[i].v
));

            }

            cout << "Current Representatives:
";
            set <int>::iterator it;
            for( it =
parent_of_taken.begin(); it !=
parent_of_taken.end(); it++ ){
                cout << *it << " ";
            }
            cout << endl;
            printf("Current Number of Tree:
%d\n", parent_of_taken.size());
            parent_of_taken.clear();

            if(kount == num_of_edges-1)
break;
        }
    }

    printf("\nMininum weight = %d\n\n", ans);

    printf("MST:\n====\n");
```

```
    for(int i = 1; i < n; i++){
        printf("Node:  %d, Parent of this:
%d\n", i, parent[i]);
    }
}

int main(){
    add_edge(1, 2, 7);
    add_edge(1, 3, 4);
    add_edge(1, 4, 1);
    add_edge(3, 4, 3);
    add_edge(2, 4, 8);
    add_edge(2, 5, 6);
    add_edge(5, 4, 6);

    num_of_edges = 7;
    n = 6; ///Number of Nodes

    Kruskal_MST();
}
```

Output

```
1 --- 4
Current Representatives: 4
Current Number of Tree: 1


1 --- 4
3 --- 4
Current Representatives: 4
Current Number of Tree: 1
```

1 --- 4
3 --- 4
2 --- 5
Current Representatives: 4 5
Current Number of Tree: 2

1 --- 4
3 --- 4
2 --- 5
5 --- 4
Current Representatives: 4
Current Number of Tree: 1

Mininum weight = 16

MST:
====
Node: 1, Parent of this: 4
Node: 2, Parent of this: 4
Node: 3, Parent of this: 4
Node: 4, Parent of this: 4
Node: 5, Parent of this: 4