Bansilal Ramnath Agarwal Charitable Trust's

Vishwakarma Institute of Technology

(An Autonomous Institute affiliated to Savitribai Phule Pune University)

Data Structures

Assignment No. 07

Name: Gourav Balaji Suram

Class: SY-CS-SEDA

Roll No.: 39

PRN No.: 12220032

Problem statement:

Implement Minimum Spanning Tree using Prims and Kruskal's Algorithm.

Prims Algorithm.

```c
#include <limits.h>
#include <stdbool.h>
#include <stdio.h>
#define Vertices 5

int Least_Key(int key[], bool Min_Span_Tree[]){
    int least = INT_MAX, min_index;
    for (int v = 0; v < Vertices; v++)
        if (Min_Span_Tree[v] == false && key[v] < least)
            least = key[v], min_index = v;
    return min_index;
}

int print_Prims_MST(int parent[], int graph[Vertices][Vertices]){
    printf("Edge \tWeight\n");
    for (int i = 1; i < Vertices; i++){
        printf("%d - %d \t%d \n", parent[i], i, graph[i][parent[i]]);
    }

}

void prims_MST(int graph[Vertices][Vertices]){
    int parent[Vertices];
    int key[Vertices];
    bool Min_Span_Tree[Vertices];
    for (int i = 0; i < Vertices; i++){
        key[i] = INT_MAX, Min_Span_Tree[i] = false;
    }
    key[0] = 0;
    parent[0] = -1;
    for (int count = 0; count < Vertices - 1; count++) {
        int u = Least_Key(key, Min_Span_Tree);
        Min_Span_Tree[u] = true;
        for (int v = 0; v < Vertices; v++){
            if (graph[u][v] && Min_Span_Tree[v] == false && graph[u][v] <
key[v]){
                parent[v] = u, key[v] = graph[u][v];
            }
        }
    }

    printf("Created Spanning Tree for Given Graph is: \n");
    printf("\n");
    print_Prims_MST(parent, graph);
}
```

```
int main(){
    int graph[Vertices][Vertices] = {
                        {0, 3, 0, 6, 0},
                        {3, 0, 4, 8, 5},
                        {0, 4, 0, 0, 7},
                        {6, 8, 0, 0, 11},
                        {0, 5, 7, 11, 0}
                        };

    prims_MST(graph);
    return 0;
}
```

OUTPUT

```
PS F:\DS ASSIGN> .\prims.exe
Created Spanning Tree for Given Graph is:

Edge    Weight
0 - 1   3
1 - 2   4
0 - 3   6
1 - 4   5
PS F:\DS ASSIGN>
```

Kruskal's Algorithm.

```c
#include<stdio.h>
#include<stdlib.h>

int comparator(const void *p1,const void *p2)//used by qsort()
{
    const int (*x)[3]=p1;
    const int (*y)[3]=p2;

    return (*x)[2]-(*y)[2];
}

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

int findParent(int parent[],int component){
    if(parent[component]==component)
    return component;
    return parent[component]=findParent(parent,parent[component]);
}

void unionSet(int u,int v,int parent[],int rank[],int n){
    u=findParent(parent,u);
    v=findParent(parent,v);

    if(rank[u]<rank[v]){
        parent[u]=v;
    }
    else if(rank[u]<rank[v]){
        parent[v]=u;
    }
    else{
        parent[v]=u;
        rank[u]++;
    }
}

void kruskalAlgo(int n,int edge[n][3]){

    qsort(edge,n,sizeof(edge[0]),comparator);
    int parent[n];
    int rank[n];
    makeSet(parent,rank,n);
    int minCost=0;
```

```
    printf("Following are the edges in the constructed MST\n");
    for(int i=0;i<n;i++){
        int v1=findParent(parent,edge[i][0]);
        int v2=findParent(parent,edge[i][1]);
        int wt=edge[i][2];

        if(v1!=v2){
            unionSet(v1,v2,parent,rank,n);
            minCost+=wt;
            printf("%d -- %d == %d\n",edge[i][0],edge[i][1],wt);
        }
    }

    printf("Minimum Cost Spanning Tree: %d\n",minCost);
}

int main(){
    int edge[5][3]={
                    {0,1,10},
                    {0,2,6},
                    {0,3,5},
                    {1,3,15},
                    {2,3,4}
                    };
    kruskalAlgo(5,edge);
}
```

OUTPUT

```
PS F:\DS ASSIGN> .\Kruskal.exe
Following are the edges in the constructed MST
2 -- 3 == 4
0 -- 3 == 5
0 -- 1 == 10
Minimum Cost Spanning Tree: 19
PS F:\DS ASSIGN>
```