Experiment No.7

To implement Kruskal’s MST Algorithm using Greedy Method.

#include <stdio.h>

#include <stdlib.h>

struct Edge {

int src, dest, weight;

};

struct Subset {

int parent;

int rank;

};

struct Edge\* createGraph(int V, int E) {

struct Edge\* graph = (struct Edge\*)malloc(E \* sizeof(struct Edge));

return graph;

}

int find(struct Subset subsets[], int i) {

if (subsets[i].parent != i)

subsets[i].parent = find(subsets, subsets[i].parent);

return subsets[i].parent;

}

void Union(struct Subset subsets[], int x, int y) {

int xroot = find(subsets, x);

int yroot = find(subsets, y);

if (subsets[xroot].rank < subsets[yroot].rank)

subsets[xroot].parent = yroot;

else if (subsets[xroot].rank > subsets[yroot].rank)

subsets[yroot].parent = xroot;

else {

subsets[yroot].parent = xroot;

subsets[xroot].rank++;

}

}

int compare(const void\* a, const void\* b) {

struct Edge\* edge1 = (struct Edge\*)a;

struct Edge\* edge2 = (struct Edge\*)b;

return edge1->weight - edge2->weight;

}

void KruskalMST(struct Edge\* graph, int V, int E) {

struct Edge resultMST[V];

int e = 0;

qsort(graph, E, sizeof(graph[0]), compare);

struct Subset\* subsets = (struct Subset\*)malloc(V \* sizeof(struct Subset));

for (int v = 0; v < V; v++) {

subsets[v].parent = v;

subsets[v].rank = 0;

}

while (e < V - 1 && E > 0) {

struct Edge nextEdge = graph[--E];

int x = find(subsets, nextEdge.src);

int y = find(subsets, nextEdge.dest);

if (x != y) {

resultMST[e++] = nextEdge;

Union(subsets, x, y);

}

}

printf("Edges of Minimum Spanning Tree:\n");

for (int i = 0; i < e; ++i)

printf("%d -- %d == %d\n", resultMST[i].src, resultMST[i].dest, resultMST[i].weight);

free(subsets);

}

int main() {

int V = 4;

int E = 5;

struct Edge\* graph = createGraph(V, E);

graph[0].src = 0;

graph[0].dest = 1;

graph[0].weight = 10;

graph[1].src = 0;

graph[1].dest = 2;

graph[1].weight = 6;

graph[2].src = 0;

graph[2].dest = 3;

graph[2].weight = 5;

graph[3].src = 1;

graph[3].dest = 3;

graph[3].weight = 15;

graph[4].src = 2;

graph[4].dest = 3;

graph[4].weight = 4;

KruskalMST(graph, V, E);

return 0;

}

Output:

Edges of Minimum Spanning Tree:

2 -- 3 == 4

0 -- 3 == 5

0 -- 1 == 10