Experiment No: 7

Kruskal's MST Algorithm using Greedy Method.

CODE:

```
#include <stdio.h>

#include <stdlib.h>

int comparator(const void* p1, const void* p2)

{

	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);

        return 0;
}
```

OUTPUT:

Output

```
/tmp/MndA9fPvSK.o
Following are the edges in the constructed MST
2 -- 3 == 4
0 -- 3 == 5
0 -- 1 == 10
Minimum Cost Spanning Tree: 19


=== Code Execution Successful ===
```