Experiment 25

Code:

```
#include <stdio.h>
#include <stdlib.h>
struct Edge {
   int src, dest, weight;
};
struct Subset {
   int parent;
   int rank;
};
int compare(const void* a, const void* b) {
   return ((struct Edge*)a)->weight - ((struct Edge*)b)->weight;
}
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++;
```

```
    }
}
void KruskalMST(struct Edge edges[], int V, int E) {
    struct Edge result[V];  // Store MST
    int e = 0;
    int i = 0;
    qsort(edges, E, sizeof(edges[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 && i < E) {
        struct Edge next = edges[i++];
        int x = find(subsets, next.src);
        int y = find(subsets, next.dest);
        if (x != y) {
            result[e++] = next;
            Union(subsets, x, y);
        }
    }
    printf("Edges in MST:\n");
    int totalWeight = 0;
    for (i = 0; i < e; i++) {
        printf("%d -- %d == %d\n", result[i].src, result[i].dest, result[i].weight);
        totalWeight += result[i].weight;
    }
    printf("Total weight: %d\n", totalWeight);
    free(subsets);
```

```
}
int main() {
    int V = 4;
    int E = 5;
    struct Edge edges[] = {
        {0, 1, 10},
        {0, 2, 6},
        {0, 3, 5},
        {1, 3, 15},
        {2, 3, 4}
    };
    KruskalMST(edges, V, E);
    return 0;
}
```