PROJECT (2)

a) required algorithm needed to find MST using Kruskal's Algorithm.

1. Main method

```
Main()
    vertices ← 5
    edges ← [
        Edge(0, 1, 10),
        Edge(0, 2, 6),
        Edge(0, 3, 5),
        Edge(1, 3, 15),
        Edge(2, 3, 4)
    ]
    mst ← Kruskal(edges, vertices)
    Print "Edges in the MST:"
    for each edge in mst do
        Print edge.Source + " -- " + edge.Destination + " == " + edge.Weight
    end for
End Main
```

2. Kruskal Method

```
Kruskal(edges, vertices)
    Sort edges by Weight
    parent ← new array of size vertices
    rank ← new array of size vertices, initialized to 0
    for i ← 0 to vertices - 1 do
        parent[i] ← i
```

```
  end for
mst ← empty list
  for each edge in edges do
    root1 ← Find(parent, edge.Source)
    root2 ← Find(parent, edge.Destination)
    if root1 ≠ root2 then
      Add edge to mst
      Union(parent, rank, root1, root2)
    end if
  end for
  return mst
End Kruskal
```

3. Find Method

```
Find(parent, vertex)
  if parent[vertex] ≠ vertex then
    parent[vertex] ← Find(parent, parent[vertex])
  end if
  return parent[vertex]
End Find
```

4. Union Method

```
Union(parent, rank, root1, root2)
  if rank[root1] > rank[root2] then
    parent[root2] ← root1
  else if rank[root1] < rank[root2] then
    parent[root1] ← root2
  else
    parent[root2] ← root1
    rank[root1] ← rank[root1] + 1
  end if
End Union
```

b) Analyze written algorithms in Part (a).

1. Sorting Edges:

$O(E\log E)$, where E is the number of edges.

2. Union-Find Operations:

Each (Find) and (Union) operation has a constant time complexity of $O(\alpha(V))$.

For E edges, total complexity: $O(E\cdot\alpha(V))$

-> **Overall Time Complexity:** $O(E\log E+E\cdot\alpha(V))$.