Seat No.: T191094350 Name : Aryan Sirdesai

Problem Statement : Implement Greedy search algorithm for : V. Kruskal's Minimal Spanning Tree Algorithm

In [1]:
```
class Graph:

    def __init__(self, vertices):
        self.V = vertices
        self.graph = []

    def addEdge(self, u, v, w):
        self.graph.append([u, v, w])

    def find(self, parent, i):
        if parent[i] != i:
            parent[i] = self.find(parent, parent[i])
        return parent[i]

    def union(self, parent, rank, x, y):
        if rank[x] < rank[y]:
            parent[x] = y
        elif rank[x] > rank[y]:
            parent[y] = x
        else:
            parent[y] = x
            rank[x] += 1

    def KruskalMST(self):
        result = []
        i = 0
        e = 0
        self.graph = sorted(self.graph, key=lambda item: item[2])
        parent = []
        rank = []

        for node in range(self.V):
            parent.append(node)
            rank.append(0)

        while e < self.V - 1:
            u, v, w = self.graph[i]
            i = i + 1
            x = self.find(parent, u)
            y = self.find(parent, v)

            if x != y:
                e = e + 1
                result.append([u, v, w])
                self.union(parent, rank, x, y)

        minimumCost = 0
        print("Edges in the constructed MST")
        for u, v, weight in result:
            minimumCost += weight
            print("%d -- %d == %d" % (u, v, weight))
        print("Minimum Spanning Tree", minimumCost)


if __name__ == '__main__':
    vertices = int(input("Enter the number of vertices: "))
```

```
    g = Graph(vertices)
    edges = int(input("Enter the number of edges: "))
    print("Enter the edges and their weights:")
    for i in range(edges):
        u, v, w = map(int, input().split())
        g.addEdge(u, v, w)
    g.KruskalMST()
```

```
Enter the number of vertices: 5
Enter the number of edges: 7
Enter the edges and their weights:
0 1 4
0 2 1
1 2 2
1 3 5
2 3 1
2 4 6
3 4 8
Edges in the constructed MST
0 -- 2 == 1
2 -- 3 == 1
1 -- 2 == 2
2 -- 4 == 6
Minimum Spanning Tree 10
```

In [ ]: