Name: Ankit raj Roll No.:-11233036 Department/Semester/Group: CSE/4/G1

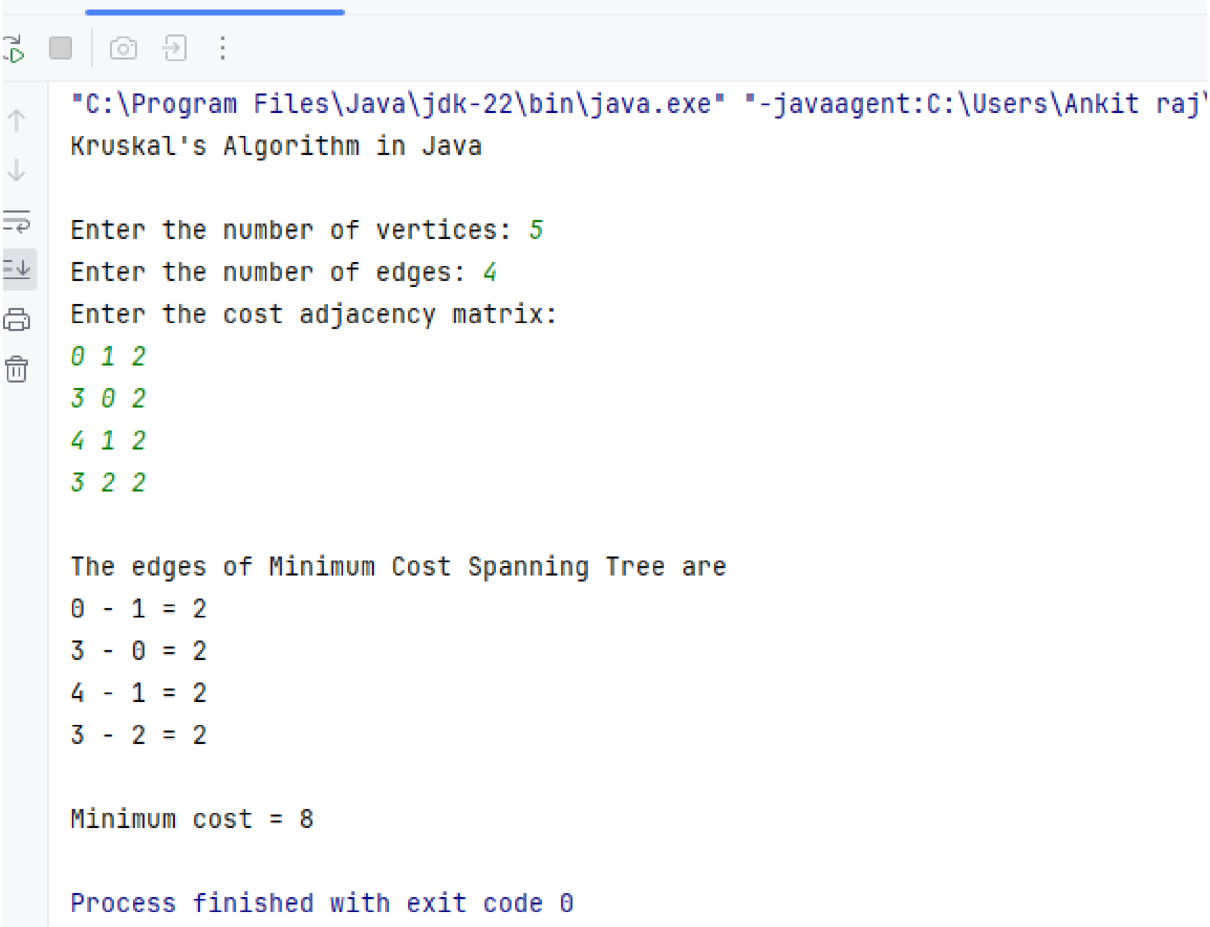
EXPERIMENT NO:-10

Aim Of Experiment:- To implement Kruskal's Algorithm for finding the Minimum Spanning Tree (MST) of a given graph using the Greedy approach.

```
class Edge implements Comparable<Edge> {
  int src, dest, weight;

  public Edge(int src, int dest, int weight) {
    this.src = src;
    this.dest = dest;
    this.weight = weight;
  }

  public int compareTo(Edge other) {
    return this.weight - other.weight;
  }
}

class Subset {
  int parent, rank;
}

public class KruskalMST {
  int vertices, edges;
  Edge[] edgeList;

  public KruskalMST(int vertices, int edges) {
    this.vertices = vertices;
    this.edges = edges;
    edgeList = new Edge[edges];
  }

  int find(Subset[] subsets, int i) {
    if (subsets[i].parent != i)
      subsets[i].parent = find(subsets, subsets[i].parent);
    return subsets[i].parent;
  }

  void union(Subset[] subsets, int x, int y) {
    int rootX = find(subsets, x);
    int rootY = find(subsets, y);

    if (subsets[rootX].rank < subsets[rootY].rank) {
      subsets[rootX].parent = rootY;
    } else if (subsets[rootX].rank > subsets[rootY].rank) {
      subsets[rootY].parent = rootX;
    } else {
      subsets[rootY].parent = rootX;
```

```
            subsets[rootX].rank++;
        }
    }

    void kruskalAlgorithm() {
        Arrays.sort(edgeList);
        Subset[] subsets = new Subset[vertices];

        for (int i = 0; i < vertices; i++) {
            subsets[i] = new Subset();
            subsets[i].parent = i;
            subsets[i].rank = 0;
        }

        Edge[] result = new Edge[vertices - 1];
        int e = 0, i = 0;

        System.out.println("\nThe edges of Minimum Cost Spanning Tree are");

        int minCost = 0;
        while (e < vertices - 1 && i < edges) {
            Edge nextEdge = edgeList[i++];
            int x = find(subsets, nextEdge.src);
            int y = find(subsets, nextEdge.dest);

            if (x != y) {
                result[e++] = nextEdge;
                union(subsets, x, y);
                System.out.println(nextEdge.src + " - " + nextEdge.dest + " = " + nextEdge.weight);
                minCost += nextEdge.weight;
            }
        }

        System.out.println("\nMinimum cost = " + minCost);
    }

    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);

        System.out.println("Kruskal's Algorithm in Java");
        System.out.print("\nEnter the number of vertices: ");
        int vertices = scanner.nextInt();
        System.out.print("Enter the number of edges: ");
        int edges = scanner.nextInt();

        KruskalMST graph = new KruskalMST(vertices, edges);

        System.out.println("Enter the cost adjacency matrix:");
    for (int i = 0; i < edges; i++) {
      int src = scanner.nextInt();
      int dest = scanner.nextInt();
      int weight = scanner.nextInt();
      graph.edgeList[i] = new Edge(src, dest, weight);
    }

    graph.kruskalAlgorithm();
    scanner.close();
  }
}
```

OUTPUT:-



```
"C:\Program Files\Java\jdk-22\bin\java.exe" "-javaagent:C:\Users\Ankit raj'
Kruskal's Algorithm in Java

Enter the number of vertices: 5
Enter the number of edges: 4
Enter the cost adjacency matrix:
0 1 2
3 0 2
4 1 2
3 2 2

The edges of Minimum Cost Spanning Tree are
0 - 1 = 2
3 - 0 = 2
4 - 1 = 2
3 - 2 = 2

Minimum cost = 8

Process finished with exit code 0
```