Assignment 12

Sayak Ghorai || BT21GCS004 || B2 || Design and Analysis of Algorithm

Q: Write an code for Kruskal's Algorithm and N-Queen Problem.

Code for Kruskal's Algorithm Multiplication:

KruskalsMST.java

```java
import java.util.ArrayList;
import java.util.Comparator;
import java.util.List;
public class KruskalsMST {
    19 usages
    static class Edge {
        3 usages
        int src, dest, weight;
        14 usages
        public Edge(int src, int dest, int weight)
        {
            this.src = src;
            this.dest = dest;
            this.weight = weight;
        }
    }
    5 usages
    static class Subset {
        9 usages
        int parent, rank;
        1 usage
        public Subset(int parent, int rank)
        {
            this.parent = parent;
            this.rank = rank;
        }
    }
    private static void kruskals(int V, List<Edge> edges)
    {
        int j = 0;
        int noOfEdges = 0;
        Subset[] subsets = new Subset[V];
        Edge[] results = new Edge[V];
        for (int i = 0; i < V; i++) {
            subsets[i] = new Subset(i,  rank: 0);
        }
        while (noOfEdges < V - 1) {
            Edge nextEdge = edges.get(j);
            int x = findRoot(subsets, nextEdge.src);
            int y = findRoot(subsets, nextEdge.dest);
            if (x != y) {
                results[noOfEdges] = nextEdge;
                union(subsets, x, y);
                noOfEdges++;
            }
            j++;
        }
        System.out.println(
                "Edges in MST:");
        int minCost = 0;
        for (int i = 0; i < noOfEdges; i++) {
            System.out.println(results[i].src + " ====» "
                    + results[i].dest + "  --Weight= "
                    + results[i].weight);
            minCost += results[i].weight;
        }
        System.out.println("Total cost of MST: " + minCost);
    }

    1 usage
```

```java
    private static void union(Subset[] subsets, int x,
                              int y)
    {
        int rootX = findRoot(subsets, x);
        int rootY = findRoot(subsets, y);

        if (subsets[rootY].rank < subsets[rootX].rank) {
            subsets[rootY].parent = rootX;
        }
        else if (subsets[rootX].rank
                 < subsets[rootY].rank) {
            subsets[rootX].parent = rootY;
        }
        else {
            subsets[rootY].parent = rootX;
            subsets[rootX].rank++;
        }
    }
    5 usages
    private static int findRoot(Subset[] subsets, int i)
    {
        if (subsets[i].parent == i)
            return subsets[i].parent;

        subsets[i].parent
                 = findRoot(subsets, subsets[i].parent);
        return subsets[i].parent;
    }
    public static void main(String[] args)
    {
        int V = 9;
        List<Edge> graphEdges = new ArrayList<>(
                List.of(
                        new Edge( src: 0,  dest: 1,  weight: 4),
                        new Edge( src: 0,  dest: 7,  weight: 8),
                        new Edge( src: 1,  dest: 2,  weight: 8),
                        new Edge( src: 1,  dest: 7,  weight: 11),
                        new Edge( src: 2,  dest: 3,  weight: 7),
                        new Edge( src: 2,  dest: 8,  weight: 2),
                        new Edge( src: 2,  dest: 5,  weight: 4),
                        new Edge( src: 3,  dest: 4,  weight: 9),
                        new Edge( src: 3,  dest: 5,  weight: 14),
                        new Edge( src: 4,  dest: 5,  weight: 10),
                        new Edge( src: 5,  dest: 6,  weight: 2),
                        new Edge( src: 6,  dest: 7,  weight: 1),
                        new Edge( src: 6,  dest: 8,  weight: 6),
                        new Edge( src: 7,  dest: 8,  weight: 7)));
        graphEdges.sort(Comparator.comparingInt(o -> o.weight));
        kruskals(V, graphEdges);
    }
}
```

Output:

```
 /Users/sayakghorai/Desktop/DAA_Assignments/Assignment10/Assignment12/out/production/Assignment12 KruskalsMST
Edges in MST:
2 ====» 3  --Weight= 4
0 ====» 3  --Weight= 5
0 ====» 1  --Weight= 10
Total cost of MST: 19

Process finished with exit code 0
```

N-Queen Problem:

KruskalsMST.java | NQueen.java

```java
public class NQueen {
    12 usages
    private static int N;
    4 usages
    private static int[] rows;
    4 usages
    private static boolean[] usedCols;
    4 usages
    private static boolean[] usedDiagonals;
    4 usages
    private static boolean[] usedReverseDiagonals;

    public static void main(String[] args) {
        N = 4;
        rows = new int[N];
        usedCols = new boolean[N];
        usedDiagonals = new boolean[2 * N - 1];
        usedReverseDiagonals = new boolean[2 * N - 1];
        solveNQueens( row: 0);
    }
    private static void solveNQueens(int row) {
        if (row == N) {
            printBoard();
            return;
        }
        for (int col = 0; col < N; col++) {
            if (!isCellUnderAttack(row, col)) {
                placeQueen(row, col);
                solveNQueens( row: row + 1);
                removeQueen(row, col);
            }
        }
    }
    1 usage
    private static boolean isCellUnderAttack(int row, int col) {
        return usedCols[col] || usedDiagonals[row + col] || usedReverseDiagonals[N - 1 - row + col];
    }
    1 usage
    private static void placeQueen(int row, int col) {
        rows[row] = col;
        usedCols[col] = true;
        usedDiagonals[row + col] = true;
        usedReverseDiagonals[N - 1 - row + col] = true;
    }
    1 usage
    private static void removeQueen(int row, int col) {
        rows[row] = 0;
        usedCols[col] = false;
        usedDiagonals[row + col] = false;
        usedReverseDiagonals[N - 1 - row + col] = false;
    }
    private static void printBoard() {
        for (int row = 0; row < N; row++) {
            for (int col = 0; col < N; col++) {
                if (rows[row] == col) {
                    System.out.print("Q ");
                } else {
                    System.out.print(". ");
                }
            }
            System.out.println();
        }
        System.out.println();
    }
}
```

Output:
4x4

```
. Q . .
. . . Q
Q . . .
. . Q .

. . Q .
Q . . .
. . . Q
. Q . .
```

6x6

```
/Users/sayakghorai/Desktop/DAA_Assignments/Assignme
. Q . . . .
. . . Q . .
. . . . . Q
Q . . . . .
. . Q . . .
. . . . Q .

. . Q . . .
. . . . . Q
. Q . . . .
. . . . Q .
Q . . . . .
. . . Q . .

. . . Q . .
Q . . . . .
. . . . Q .
. Q . . . .
. . . . . Q
. . Q . . .

. . . . Q .
. . Q . . .
Q . . . . .
. . . . . Q
. . . Q . .
. Q . . . .


Process finished with exit code 0
```