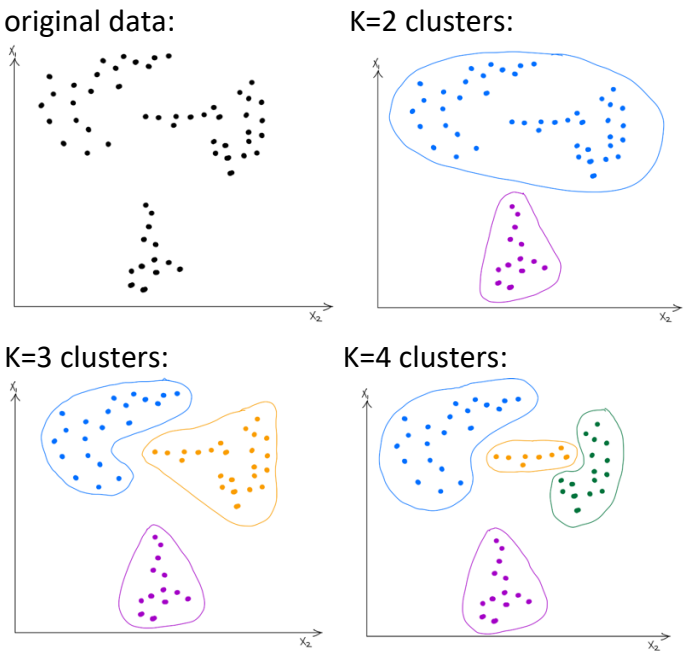
HARDIK KHARE | 70765344

HW 6.1 - Chained K-Nearest-Neighbor Clusters (50 points)

An epidemic is breaking out in a region containing M cities, and quarantine partitions are needed based on the likely spread of the disease. **You are asked to write a program that clusters the cities into K clusters (with K being a variable parameter indicating the number of quarantined partitions) such that if a city is closest to some other city, then those cities are part of the same quarantine group.**

Using Euclidean distance as a measure of closeness between points, the following cities would be clustered as follows (with varying K)…



Section 1: Successful compilation of program

```
[hardikkhare@Hardiks-MacBook-Pro AP HW 6 % javac -Xlint chainedKNN.java
[hardikkhare@Hardiks-MacBook-Pro AP HW 6 % java chainedKNN
```

Section 2: program running on the provided example from the assignment

```
hardikkhare@Hardiks-MacBook-Pro AP HW 6 % javac -Xlint chainedKNN.java
hardikkhare@Hardiks-MacBook-Pro AP HW 6 % java chainedKNN
Enter number of cities
15
Enter city coordinates
A 6 2
B 7 3
C 9 3
D 8 5
E 9 8
F 8 9
G 7 10
H 6 11
I 8 14
J 6 14
K 4 14
L 2 14
M 2 8
N 2 6
O 3 5


Enter number of cluster or 'quit'
8

Cluster 1: A, B
Cluster 2: C
Cluster 3: D
Cluster 4: E, F, G, H
Cluster 5: I
Cluster 6: J, K, L
Cluster 7: M
Cluster 8: N, O

Enter number of cluster or 'quit'
2

Cluster 1: A, B, C, D, E, F, G, H, I, J, K, L
Cluster 2: M, N, O

Enter number of cluster or 'quit'
3

Cluster 1: A, B, C, D
Cluster 2: E, F, G, H, I, J, K, L
Cluster 3: M, N, O

Enter number of cluster or 'quit'
1

Cluster 1: A, B, C, D, E, F, G, H, I, J, K, L, M, N, O

Enter number of cluster or 'quit'
```

Section 3: Provided test input

~ No Test Input provided on Piazza ~

Section 4: Edge Case #1
Description: Number of cluster is more than numbers of points

Input: Num points : 15; Num clusters 20

Expected Output: We should get 15 clusters as that is max number of cities

```
Enter number of cluster or 'quit'
20

Cluster 1: A
Cluster 2: B
Cluster 3: C
Cluster 4: D
Cluster 5: E
Cluster 6: F
Cluster 7: G
Cluster 8: H
Cluster 9: I
Cluster 10: J
Cluster 11: K
Cluster 12: L
Cluster 13: M
Cluster 14: N
Cluster 15: O
```

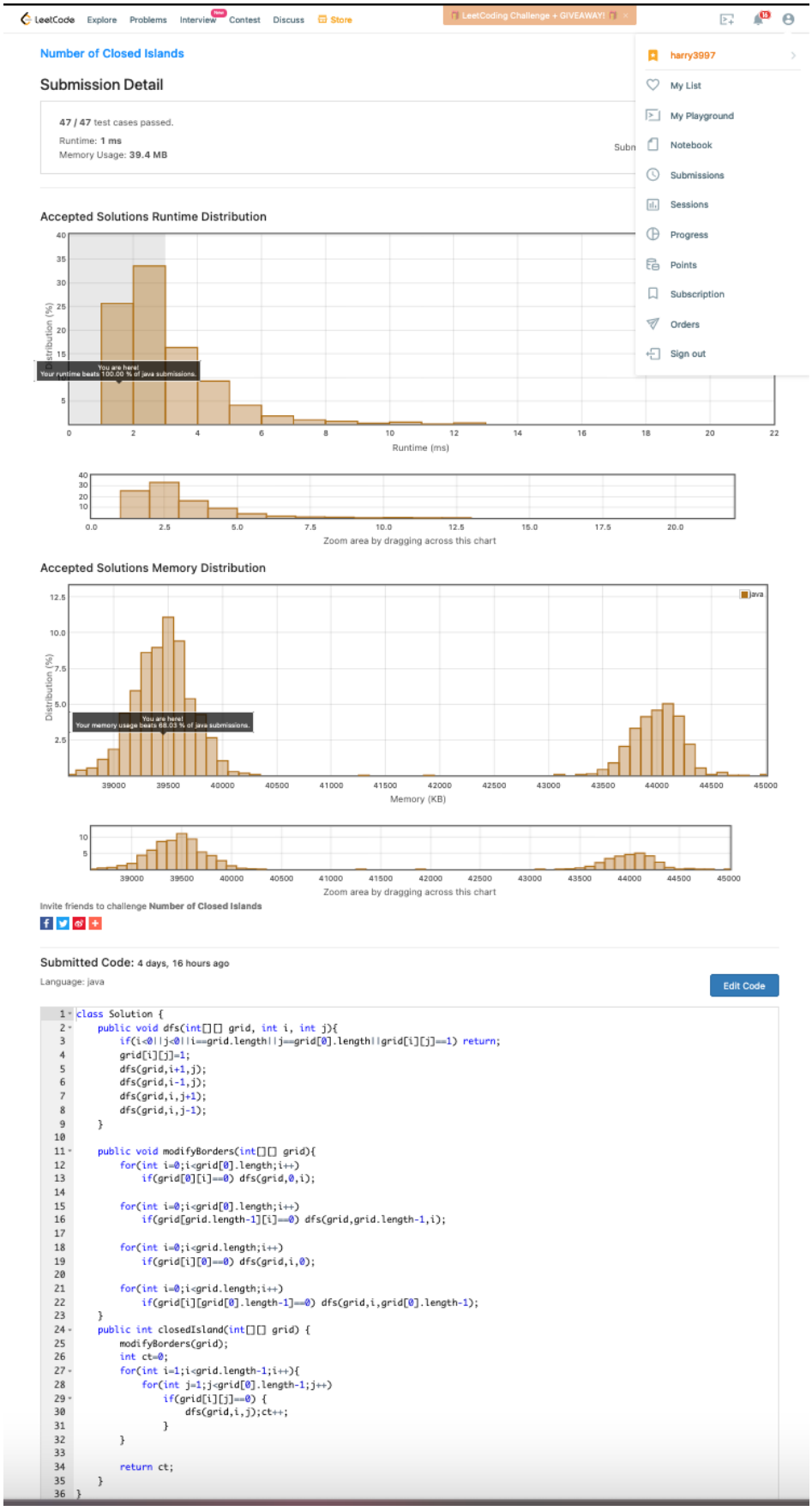
Section 5: Edge Case #2
Description: Number of points is 0.

Expected Output: There will be no clusters

Output:

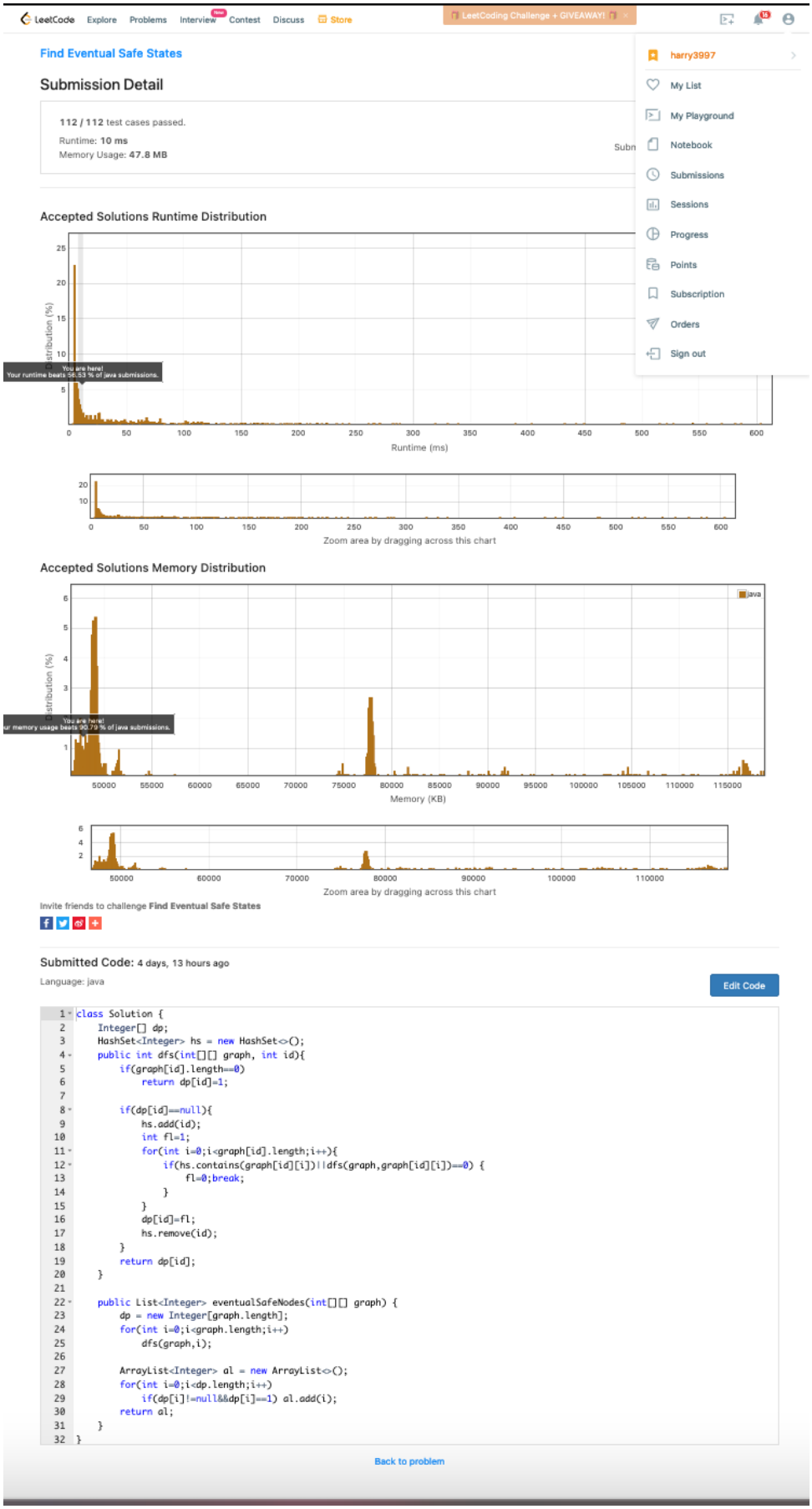
```
hardikkhare@Hardiks-MacBook-Pro AP HW 6 % java chainedKNN
Enter number of cities
0
Enter city coordinates


Enter number of cluster or 'quit'
2


Enter number of cluster or 'quit'
```

https://leetcode.com/problems/number-of-closed-islands/




```java
class Solution {
    public void dfs(int[][] grid, int i, int j){
        if(i<0||j<0||i==grid.length||j==grid[0].length||grid[i][j]==1) return;
        grid[i][j]=1;
        dfs(grid,i+1,j);
        dfs(grid,i-1,j);
        dfs(grid,i,j+1);
        dfs(grid,i,j-1);
    }

    public void modifyBorders(int[][] grid){
        for(int i=0;i<grid[0].length;i++)
            if(grid[0][i]==0) dfs(grid,0,i);

        for(int i=0;i<grid[0].length;i++)
            if(grid[grid.length-1][i]==0) dfs(grid,grid.length-1,i);

        for(int i=0;i<grid.length;i++)
            if(grid[i][0]==0) dfs(grid,i,0);

        for(int i=0;i<grid.length;i++)
            if(grid[i][grid[0].length-1]==0) dfs(grid,i,grid[0].length-1);
    }
    public int closedIsland(int[][] grid) {
        modifyBorders(grid);
        int ct=0;
        for(int i=1;i<grid.length-1;i++){
            for(int j=1;j<grid[0].length-1;j++)
                if(grid[i][j]==0) {
                    dfs(grid,i,j);ct++;
                }
        }

        return ct;
    }
}
```

https://leetcode.com/problems/find-eventual-safe-states/




```java
class Solution {
    Integer[] dp;
    HashSet<Integer> hs = new HashSet<>();
    public int dfs(int[][] graph, int id){
        if(graph[id].length==0)
            return dp[id]=1;

        if(dp[id]==null){
            hs.add(id);
            int fl=1;
            for(int i=0;i<graph[id].length;i++){
                if(hs.contains(graph[id][i])||dfs(graph,graph[id][i])==0) {
                    fl=0;break;
                }
            }
            dp[id]=fl;
            hs.remove(id);
        }
        return dp[id];
    }

    public List<Integer> eventualSafeNodes(int[][] graph) {
        dp = new Integer[graph.length];
        for(int i=0;i<graph.length;i++)
            dfs(graph,i);

        ArrayList<Integer> al = new ArrayList<>();
        for(int i=0;i<dp.length;i++)
            if(dp[i]!=null&&dp[i]==1) al.add(i);
        return al;
    }
}
```