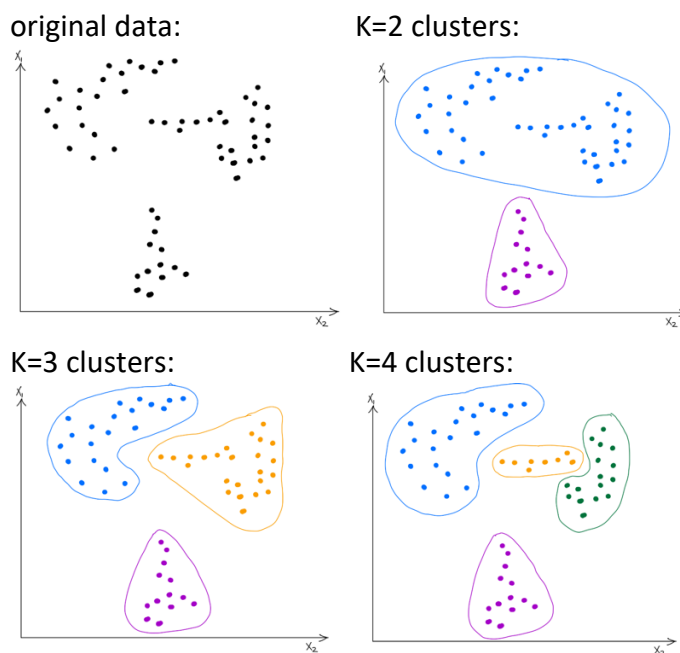


HARDIK KHARE | 70765344

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/>

LeetCode

Explore Problems Interview Contest Discuss Store

LeetCode Challenge + GIVEAWAY!

🔔

👤

🔒

Number of Closed Islands

Submission Detail

47 / 47 test cases passed.

Runtime: 1 ms

Memory Usage: 39.4 MB

Accepted Solutions Runtime Distribution

Accepted Solutions Memory Distribution

Invite friends to challenge Number of Closed Islands

Submitted Code: 4 days, 16 hours ago

Language: java

Edit Code

```
1 class Solution {
2     public void dfs(int[][] grid, int i, int j){
3         if(i<0||j<0||i==grid.length||j==grid[0].length||grid[i][j]==1) return;
4         grid[i][j]=1;
5         dfs(grid,i+1,j);
6         dfs(grid,i-1,j);
7         dfs(grid,i,j+1);
8         dfs(grid,i,j-1);
9     }
10
11     public void modifyBorders(int[][] grid){
12         for(int i=0;i<grid[0].length;i++){
13             if(grid[0][i]==0) dfs(grid,0,i);
14
15             for(int i=0;i<grid[0].length;i++){
16                 if(grid[grid.length-1][i]==0) dfs(grid,grid.length-1,i);
17
18                 for(int i=0;i<grid.length;i++){
19                     if(grid[i][0]==0) dfs(grid,i,0);
20
21                     for(int i=0;i<grid.length;i++){
22                         if(grid[i][grid[0].length-1]==0) dfs(grid,i,grid[0].length-1);
23                     }
24                 }
25             }
26             public int closedIsland(int[][] grid) {
27                 modifyBorders(grid);
28                 int ct=0;
29                 for(int i=1;i<grid.length-1;i++){
30                     for(int j=1;j<grid[0].length-1;j++){
31                         if(grid[i][j]==0) {
32                             dfs(grid,i,j);ct++;
33                         }
34                     }
35                 }
36                 return ct;
37             }
38         }
```

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

LeetCode

Explore

Problems

Interview

Contest

Discuss

Store

LeetCode Coding Challenge + GIVEAWAY!

Find Eventual Safe States

Submission Detail

112 / 112 test cases passed.
Runtime: 10 ms
Memory Usage: 47.8 MB

Submissions

My List

My Playground

Notebook

Submissions

Sessions

Progress

Points

Subscription

Orders

Sign out

Accepted Solutions Runtime Distribution

25
20
15
10
5
0

0 50 100 150 200 250 300 350 400 450 500 550 600

Runtime (ms)

You are here!
Your runtime beats 99.33 % of java submissions.

20
10
0

0 50 100 150 200 250 300 350 400 450 500 550 600

Zoom area by dragging across this chart

Accepted Solutions Memory Distribution

6
5
4
3
2
1
0

50000 55000 60000 65000 70000 75000 80000 85000 90000 95000 100000 105000 110000 115000

Memory (KB)

You are here!
Your memory usage beats 99.79 % of java submissions.

6
4
2
0

50000 60000 70000 80000 90000 100000 110000

Zoom area by dragging across this chart

Invite friends to challenge Find Eventual Safe States

Submitted Code: 4 days, 13 hours ago

Language: java

Edit Code

```
1 class Solution {
2     Integer[] dp;
3     HashSet<Integer> hs = new HashSet<>();
4     public int dfs(int[][] graph, int id){
5         if(graph[id].length==0)
6             return dp[id]=1;
7
8         if(dp[id]==null){
9             hs.add(id);
10            int fl=1;
11            for(int i=0;i<graph[id].length;i++){
12                if(hs.contains(graph[id][i])||dfs(graph,graph[id][i])==0) {
13                    fl=0;break;
14                }
15            }
16            dp[id]=fl;
17            hs.remove(id);
18        }
19        return dp[id];
20    }
21
22    public List<Integer> eventualSafeNodes(int[][] graph) {
23        dp = new Integer[graph.length];
24        for(int i=0;i<graph.length;i++)
25            dfs(graph,i);
26
27        ArrayList<Integer> al = new ArrayList<>();
28        for(int i=0;i<dp.length;i++)
29            if(dp[i]!=null&&dp[i]==1) al.add(i);
30        return al;
31    }
32 }
```

Back to problem

