Data Structures and Algo in Java - Day 17

Completed some Medium Level Array Questions such as

1. Find the Leader
2. Find the Longest Consecutive Sequence

* Tried Brute Better and Optimal Solution

1. Set Matrix to Zero

* Tried Brute Better and Optimal Solution

1. Rotate the Matrix by 90 Degree

Here are the Codes

import java.util.\*;

public class day17

{

public static void main (String [] args)

{

// int arr [] = {10,22,12,3,0,6};

// leader(arr);

// int arr [] = {0,1,100,2,101,7,301,3,8,4,9,102,103,302,303,5,10,11,12};

// longestConsecutiveSequencebetter(arr);

// int arr [][] = {

// {1,1,1,1},

// {1,0,0,1},

// {1,1,0,1},

// {0,1,1,1}

// };

// matrixToZeroOptimal(arr,4,4);

int arr [][] = {

{1,2,3,4},

{5,6,7,8},

{9,10,11,12},

{13,14,15,16}

};

rotateOptimized90(arr);

}

public static void leader(int arr [])

{

int n = arr.length;

int maxi = Integer.MIN\_VALUE;

ArrayList<Integer> list = new ArrayList<>();

for(int i=n-1;i>=0;i--)

{

if(arr[i]>maxi)

{

list.add(arr[i]);

}

maxi = Math.max(maxi,arr[i]);

}

Collections.reverse(list);

System.out.println(list);

}

public static void longestConsecutiveSequencebrute(int arr[])

{

int n = arr.length;

int longest = 1;

int count;

for(int i=0;i<n;i++)

{

int x=arr[i];

count = 1;

while(linearSearch(arr,x+1)==true)

{

x=x+1;

count=count+1;

}

longest=Math.max(longest,count);

}

System.out.println(longest);

}

public static boolean linearSearch(int arr[],int x)

{

int n = arr.length;

for(int i=0;i<n;i++)

{

if(arr[i]==x)

{

return true;

}

}

return false;

}

public static void longestConsecutiveSequencebetter(int arr[])

{

int n = arr.length;

int longest = 1;

int currentCount = 1;

Arrays.sort(arr);

int lastSmall=arr[0];

for(int i=1;i<n;i++)

{

if(arr[i]==lastSmall)

{

continue;

}

if(arr[i]==lastSmall+1)

{

currentCount = currentCount + 1;

}

else

{

currentCount =1;

}

lastSmall=arr[i];

longest = Math.max(longest,currentCount);

}

System.out.println(longest);

}

public static void longestConsecutiveSequenceOptimal(int arr [])

{

int n = arr.length;

int count = 0;

int longest = 1;

HashSet<Integer> set = new HashSet<>();

for(int i=0;i<n;i++)

{

set.add(arr[i]);

}

for(int iruken : set)

{

if(!set.contains(iruken-1))//illai

{

count = 1;

int x = iruken;

while(set.contains(x+1))

{

x=x+1;

count = count +1;

}

longest = Math.max(longest,count);

}

}

System.out.println(longest);

}

public static void matrixToZero(int arr[][],int n,int m)

{

for(int i=0;i<n;i++)

{

for(int j=0;j<m;j++)

{

if(arr[i][j]==0)

{

changeRow(arr,i,j,n,m);

changeColumn(arr,i,j,n,m);

}

}

}

for(int i=0;i<n;i++)

{

for(int j=0;j<m;j++)

{

if(arr[i][j]==-1)

{

arr[i][j]=0;

}

}

}

for(int i=0;i<n;i++)

{

for(int j=0;j<m;j++)

{

System.out.print(arr[i][j]+" ");

}

System.out.println();

}

}

public static void changeRow(int arr [] [] ,int i,int j,int n,int m)

{

for(j=0;j<m;j++)

{

if(arr[i][j]!=0)

{

arr[i][j]=-1;

}

}

}

public static void changeColumn(int arr [] [] ,int i,int j,int n, int m)

{

for(i=0;i<n;i++)

{

if(arr[i][j]!=0)

{

arr[i][j]=-1;

}

}

}

public static void matrixToZeroBetter(int arr[] [],int n , int m)

{

int row [] = new int [n];

int column [] = new int [m];

for(int i=0;i<n;i++)

{

for(int j=0;j<m;j++)

{

if(arr[i][j]==0)

{

row[i] = 1;

column[j] = 1;

}

}

}

//change 1's to 0's

for(int i=0;i<n;i++)

{

for(int j=0;j<m;j++)

{

if(row[i]==1 || column[j]==1)

{

arr[i][j]=0;

}

}

}

for(int i=0;i<n;i++)

{

for(int j=0;j<m;j++)

{

System.out.print(arr[i][j]+ " ");

}

System.out.println();

}

}

public static void matrixToZeroOptimal(int arr[] [],int n,int m)

{

int col0 = 1;

for(int i=0;i<n;i++)

{

for(int j=0;j<m;j++)

{

if(arr[i][j]==0)

{

arr[i][0]=0;

if(j!=0)

{

arr[0][j]=0;

}

else

{

col0 = 0;

}

}

}

}

for(int i=1;i<n;i++)

{

for(int j=1;j<m;j++)

{

if(arr[i][j]!=0) // in the previous we had row[i] = 1 or col[j] = 1 which is the same as arr[i][j] !=0

{

if(arr[i][0]==0 || arr[0][j]==0)

{

arr[i][j]=0;

}

}

}

}

if(arr[0][0]==0)

{

for(int j=0;j<m;j++)

{

arr[0][j]=0;

}

}

if(col0==0)

{

for(int i=0;i<n;i++)

{

arr[i][0]=0;

}

}

for(int i=0;i<n;i++)

{

for(int j=0;j<m;j++)

{

System.out.print(arr[i][j]+ " ");

}

System.out.println();

}

}

public static void rotateOptimized90(int arr[][])

{

int n = arr.length;

int m = arr[0].length;

for(int i=0;i<n;i++)

{

for(int j=i+1;j<m;j++)

{

swap(arr,i,j);

}

}

for(int i=0;i<n;i++)

{

reverse(arr[i]);

}

for(int i=0;i<n;i++)

{

for(int j=0;j<m;j++)

{

System.out.print(arr[i][j]+" ");

}

System.out.println();

}

}

public static void swap ( int arr [][] , int i , int j)

{

int temp = arr [i][j];

arr[i][j] = arr[j][i];

arr[j][i] = temp;

}

public static void reverse ( int arr [])

{

int left = 0;

int right = arr.length -1 ;

while(left<right)

{

int temp = arr[left] ;

arr[left] = arr[right];

arr[right] = temp;

left++;

right--;

}

}

}