Problems Of the day: Date:11th Nov 2024

1. Anagram

class Solution {

public static boolean areAnagrams(String s1, String s2) {

char arr1[]=s1.toCharArray();

char arr2[]=s2.toCharArray();

Arrays.sort(arr1);

Arrays.sort(arr2);

String a=new String(arr1);

String b=new String(arr2);

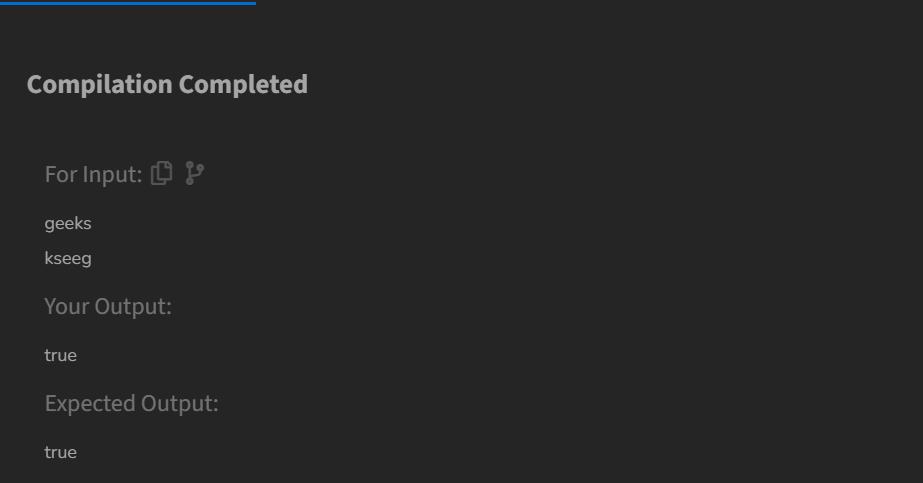
if(a.equals(b)) return true;

else return false;

}

}

Output:

****

2.Row With Max 1’s

Class Solution{

public int rowWithMax1s(int arr[][]) {

int maxrow=-1;

int j=arr[0].length-1;

for(int i=0;i<arr.length;i++)

{

while(j>=0 && arr[i][j]==1)

{

maxrow=i;

j--;

}

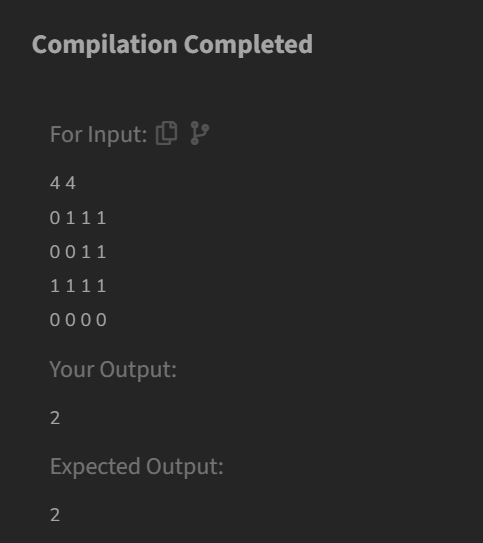
}

return maxrow;

}

}

Output:

****

**Longest Consecutive Subsequences:**

class Solution {

public int findLongestConseqSubseq(int[] arr) {

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

int max=0;

for(int num:arr)

{

Set.add(num);

}

for(int num:arr)

{

if(!Set.contains(num-1))

{

int curNum=num;

int curlen=1;

while(Set.contains(curNum+1))

{

curNum++;

curlen++;

}

max=Math.max(max,curlen);

}

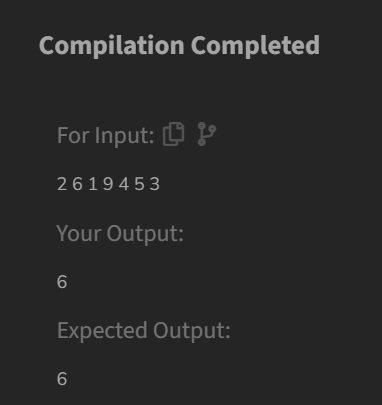
}

return max;

}

}

**Output:**

****

4.Longest Palindromic Substring:

class Solution{

String longestPalindrome(String S){

int n = S.length();

int max = 1, s = 0;

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

for (int j = i; j < n; j++) {

int l = i, h = j;

boolean isPalindrome = true;

while (l < h) {

if (S.charAt(l) != S.charAt(h)) {

isPalindrome = false;

break;

}

l++;

h--;

}

if (isPalindrome && (j - i + 1) > max) {

s = i;

max = j - i + 1;

}

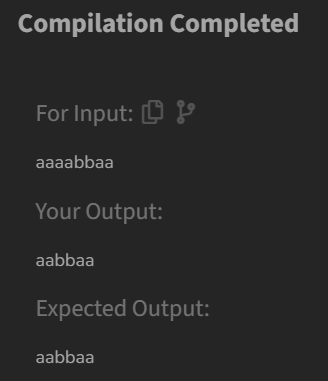
}

}

return S.substring(s, s + max);

}

}  
  
Output:



5. Rat in the Maze:

class Solution {

ArrayList<String> a=new ArrayList<>();

public ArrayList<String> findPath(int[][] mat) {

// Your code here

int n=mat.length;

if(mat[0][0]==0 || mat[n-1][n-1]==0)

{

a.add("-1");

}

String path="";

boolean vis[][]=new boolean[n][n];

dfs(mat,0,0,vis,n,path);

return a;

}

void dfs(int[][] mat,int i,int j, boolean [][] vis,int n,String path)

{

if(i<0||j<0 || i>=n ||j>=n)

{

return;

}

if(mat[i][j]==0||vis[i][j]==true)

{

return;

}

if(i==n-1 && j==n-1)

{

a.add(path);

return;

}

vis[i][j]=true;

dfs(mat,i+1,j,vis,n,path+'D');

dfs(mat,i-1,j,vis,n,path+'U');

dfs(mat,i,j+1,vis,n,path+'R');

dfs(mat,i,j-1,vis,n,path+'L');

vis[i][j]=false;

}

}  
  
Output:

