DSA PRACTICE – 6

Name:Deepak S

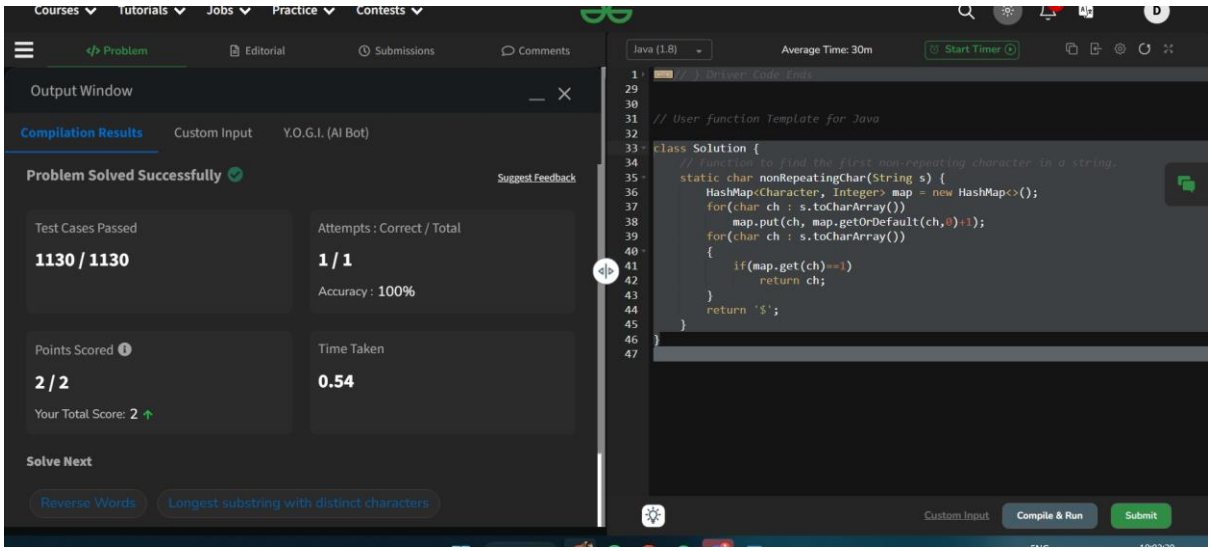
Register No: 22IT018

Non Repeating Character

Code:

```
class Solution {
   // Function to find the first non-repeating character in a string.
   static char nonRepeatingChar(String s) {
      HashMap<Character, Integer> map = new HashMap<>();
      for(char ch : s.toCharArray())
         map.put(ch, map.getOrDefault(ch,0)+1);
      for(char ch : s.toCharArray())
      {
         if(map.get(ch)==1)
            return ch;
      }
      return '$';
   }
}
```

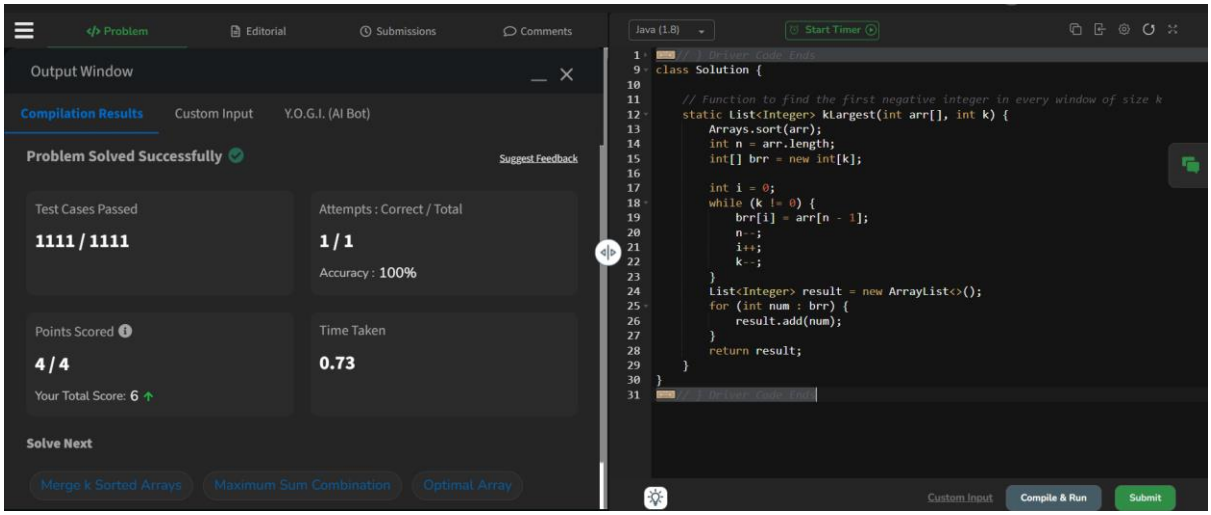
output:



k largest elements

output:

```
Class Solution {
   // Function to find the first negative integer in every window of size k
   static List<Integer> kLargest(int arr[], int k) {
      Arrays.sort(arr);
      int n = arr.length;
      int[] brr = new int[k];

      int i = 0;
      while (k != 0) {
         brr[i] = arr[n - 1];
         n--;
         i++;
         k--;
      }
      List<Integer> result = new ArrayList<>();
      for (int num : brr) {
         result.add(num);
      }
      return result;}
```

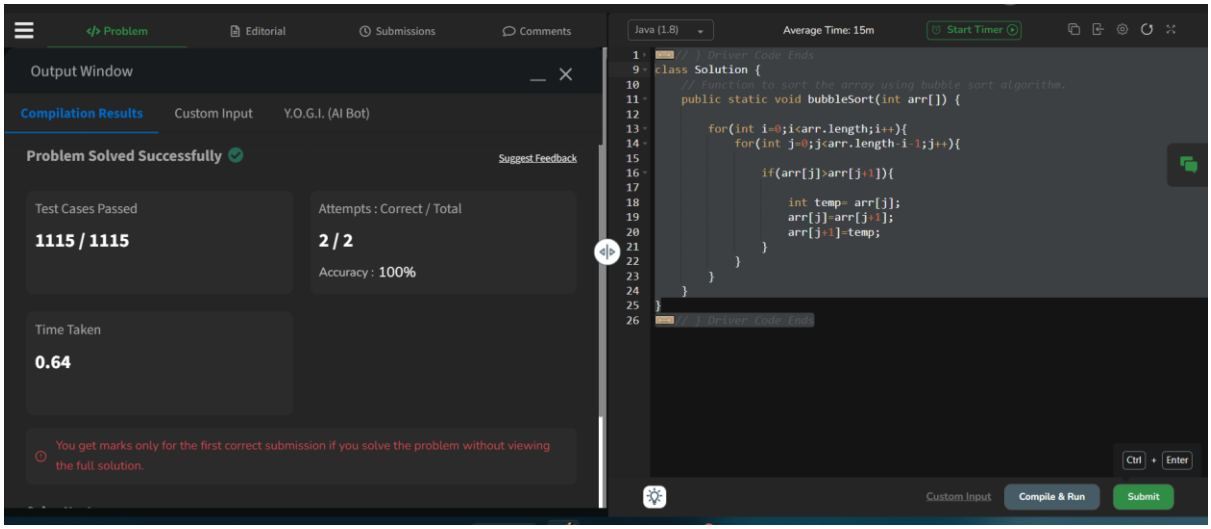


Bubble Sort

Code:

```
class Solution {
   // Function to sort the array using bubble sort algorithm.
   public static void bubbleSort(int arr[]) {

      for(int i=0;i<arr.length;i++){
         for(int j=0;j<arr.length-i-1;j++){

            if(arr[j]>arr[j+1]){

               int temp= arr[j];
               arr[j]=arr[j+1];
               arr[j+1]=temp;

            }

         }
   }
}
```

Output:



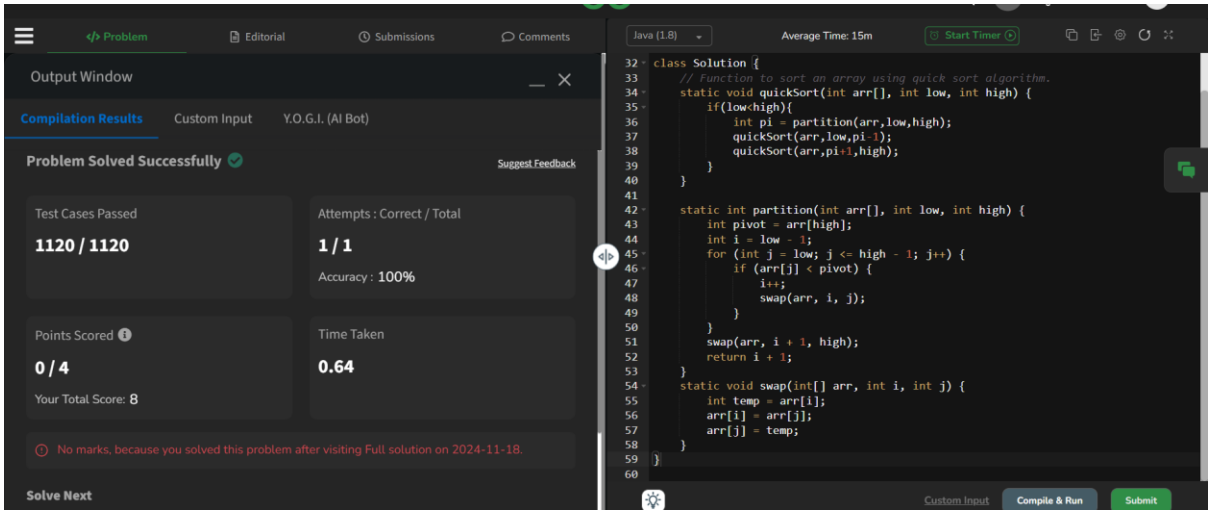
QUITE SORT

Code:

```
class Solution {
   // Function to sort an array using quick sort algorithm.
   static void quickSort(int arr[], int low, int high) {
      if(low<high){
         int pi = partition(arr,low,high);
         quickSort(arr,low,pi-1);
         quickSort(arr,pi+1,high);
      }
   }

   static int partition(int arr[], int low, int high) {
      int pivot = arr[high];
      int i = low - 1;
      for (int j = low; j <= high - 1; j++) {
         if (arr[j] < pivot) {
            i++;
            swap(arr, i, j);
         }
      }
      swap(arr, i + 1, high);
      return i + 1;
   }
   static void swap(int[] arr, int i, int j) {
      int temp = arr[i];
      arr[i] = arr[j];
      arr[j] = temp;
```

```
    }
}
```

Output:



Edit Distance

CODE:

```
class Solution {
    public int editDistance(String s1, String s2) {
        // Code here
        int m=s1.length();
        int n=s2.length();
        int[][] dp=new int[m+1][n+1];

        for(int i=0;i<=m;i++)
        dp[i][0]=i;

        for(int j=0;j<=n;j++)
        dp[0][j]=j;

        for(int i=1;i<=m;i++){
            for(int j=1;j<=n;j++){
                if(s1.charAt(i-1)==s2.charAt(j-1))
                dp[i][j]=dp[i-1][j-1];

                else
                    dp[i][j]=Math.min(dp[i-1][j],Math.min(dp[i][j-1],dp[i-1][j-1]))+1;
            }
```

```
        }
        return dp[m][n];
    }
}
```

Output: