DSA PRACTICE – DAY 5

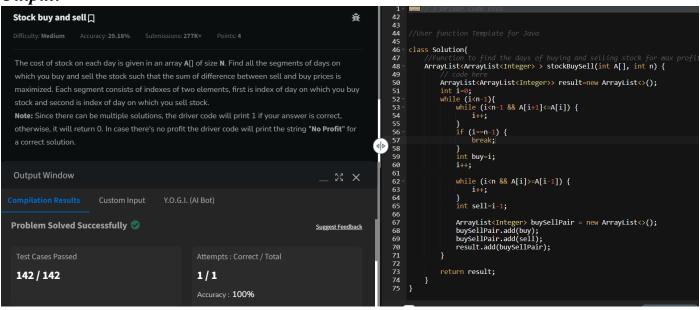
Name: Dhejan R **Reg No:** 22IT022 **Date:** 14/11/2024

1. Stock Buy and Sell

```
Code Solution:
```

```
class Solution {
  ArrayList<ArrayList<Integer>> stockBuySell(int A[], int n) {
    ArrayList<ArrayList<Integer>> result=new ArrayList<>();
    int i=0;
    while (i \le n-1)
       while (i < n-1 & A[i+1] < A[i])
         i++;
       if (i==n-1) break;
       int buy=i;
       i++;
       while (i<n && A[i]>=A[i-1]) {
         i++;
       int sell=i-1;
       ArrayList<Integer> buySellPair = new ArrayList<>();
       buySellPair.add(buy);
       buySellPair.add(sell);
       result.add(buySellPair);
    return result;
```

Output:



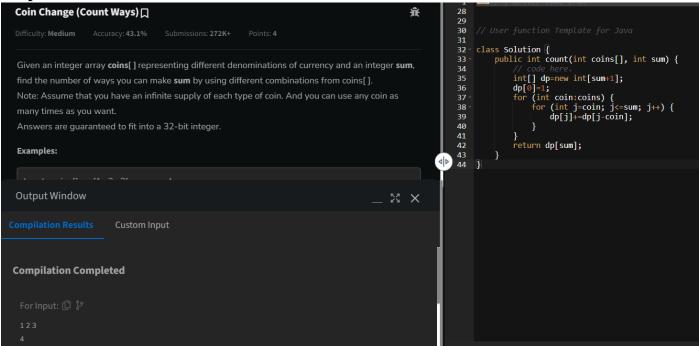
Time complexity: O (n)
Space Complexity: O (n)

2. Minimize heights II

Code Solution:

```
class Solution {
   public int count(int coins[], int sum) {
      // code here.
      int[] dp=new int[sum+1];
      dp[0]=1;
      for (int coin:coins) {
            for (int j=coin; j<=sum; j++) {
                 dp[j]+=dp[j-coin];
            }
        }
      return dp[sum];
   }
}</pre>
```

Output:



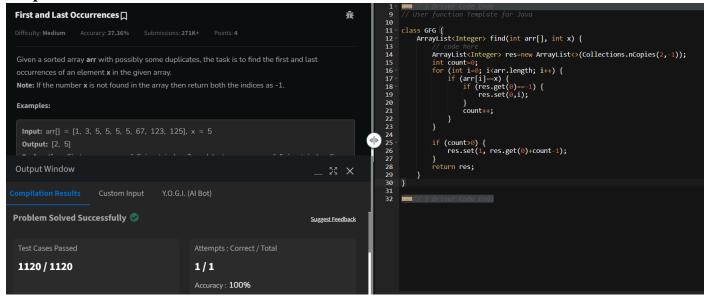
Time Complexity: O (m*n)
Space Complexity: O (m)

3. First and Last Occurences

Code Solution:

```
class GFG {
    ArrayList<Integer> find(int arr[], int x) {
        // code here
        ArrayList<Integer> res=new ArrayList<>(Collections.nCopies(2,-1));
        int count=0;
        for (int i=0; i<arr.length; i++) {
            if (arr[i]==x) {
                 res.set(0)==-1) {
                      res.set(0,i);
                 }
                 count++;
                 }
            if (count>0) {
                 res.set(1, res.get(0)+count-1);
            }
            return res;
            }
}
```

Output:



Time complexity: O (n)
Space Complexity: O (1)

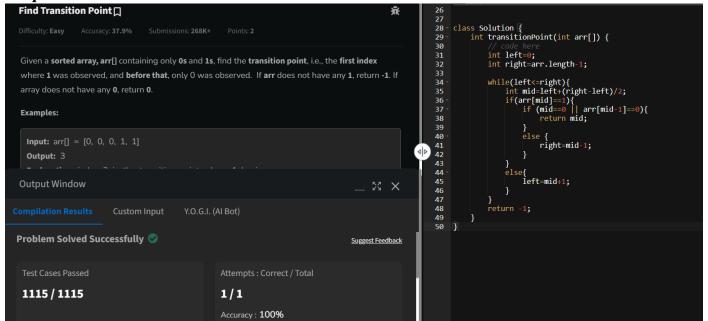
4. Fins Transition Point

Code Solution:

```
class Solution {
  int transitionPoint(int arr[]) {
    // code here
    int left=0;
  int right=arr.length-1;

  while(left<=right) {
    int mid=left+(right-left)/2;
    if(arr[mid]==1) {
      if (mid==0 || arr[mid-1]==0) {
         return mid;
      }
      else {
         right=mid-1;
      }
    }
    else {
        left=mid+1;
    }
  }
  return -1;
}</pre>
```

Output:



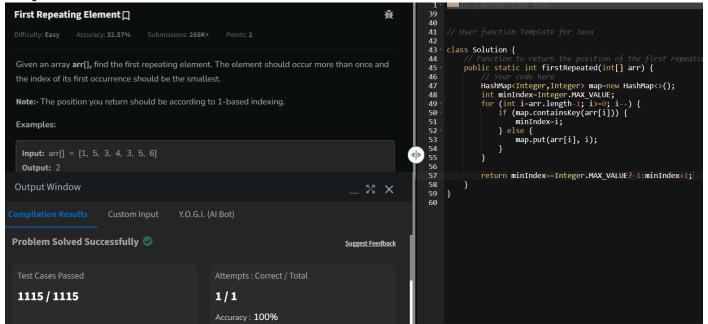
Time Complexity: O (logn) Space Complexity: O (1)

5. First Repeating Element

Code Solution:

```
class Solution {
    // Function to return the position of the first repeating element.
    public static int firstRepeated(int[] arr) {
        // Your code here
        HashMap<Integer,Integer> map=new HashMap<>();
        int minIndex=Integer.MAX_VALUE;
        for (int i=arr.length-1; i>=0; i--) {
            if (map.containsKey(arr[i])) {
                 minIndex=i;
            } else {
                 map.put(arr[i], i);
            }
        }
        return minIndex==Integer.MAX_VALUE?-1:minIndex+1;
        }
}
```

Output:



Time Complexity: O (n)
Space Complexity: O (n)

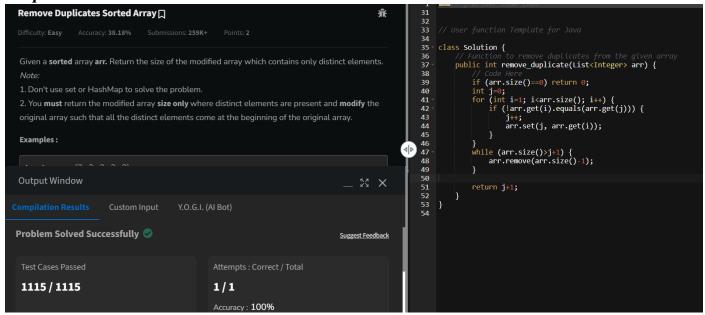
6. Remove Duplicates Sorted Array

Code Solution:

```
class Solution {
   // Function to remove duplicates from the given array
   public int remove duplicate(List<Integer> arr) {
```

```
// Code Here
if (arr.size()==0) return 0;
int j=0;
for (int i=1; i<arr.size(); i++) {
    if (!arr.get(i).equals(arr.get(j))) {
        j++;
        arr.set(j, arr.get(i));
    }
} while (arr.size()>j+1) {
    arr.remove(arr.size()-1);
}
return j+1;
}
```

Output:



Time Complexity: O (n)
Space Complexity: O (1)

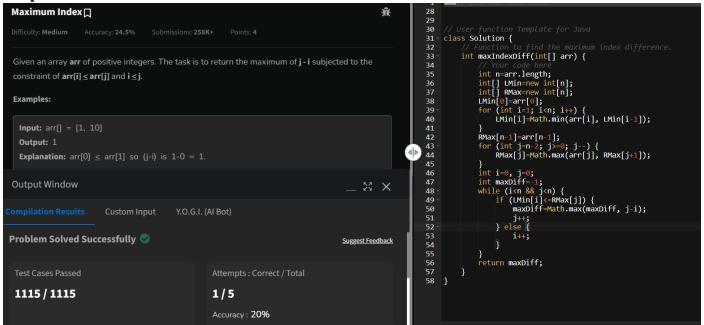
7. Maximum Index

Code Solution

```
class Solution {
   // Function to find the maximum index difference.
   int maxIndexDiff(int[] arr) {
      // Your code here
      int n=arr.length;
      int[] LMin=new int[n];
```

```
int[] RMax=new int[n];
LMin[0]=arr[0];
for (int i=1; i < n; i++) {
  LMin[i]=Math.min(arr[i], LMin[i-1]);
RMax[n-1]=arr[n-1];
for (int j=n-2; j>=0; j--) {
  RMax[j]=Math.max(arr[j], RMax[j+1]);
int i=0, j=0;
int maxDiff=-1;
while (i<n && j<n) {
  if (LMin[i]<=RMax[j]) {</pre>
    maxDiff=Math.max(maxDiff, j-i);
    j++;
  } else {
    i++;
return maxDiff;
```

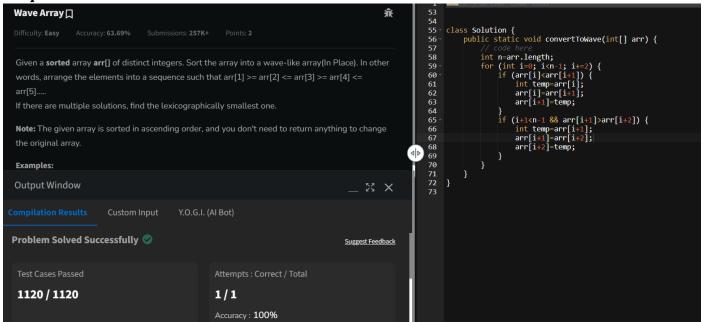
Output:



Time Complexity: O (n) Space Complexity: O (n)

8. Wave Array Code Solution class Solution { public static void convertToWave(int[] arr) { // code here int n=arr.length; for (int i=0; i<n-1; i+=2) { if (arr[i]<arr[i+1]) { int temp=arr[i]; arr[i]=arr[i+1]; arr[i+1]=temp; } if (i+1<n-1 && arr[i+1]>arr[i+2]) { int temp=arr[i+2]; arr[i+2]=temp; } } }

Output:



Time Complexity: O (n)
Space Complexity: O (1)