

## DSA CODING PRACTICE – 4

### 1. Stock Buy and Sell (Medium level)

```
45 class Interval {
46     int buy;
47     int sell;
48 }
49
50 class Solution {
51     // Function to find the days of buying and selling stock for max profit.
52     ArrayList<ArrayList<Integer>> stockBuySell(int A[], int n) {
53         ArrayList<ArrayList<Integer>> res = new ArrayList<>();
54         if (n == 1) {
55             return res;
56         }
57
58         ArrayList<Interval> sol = new ArrayList<>();
59         int i = 0, c = 0;
60
61         while (i < n - 1) {
62             // Finding local minima
63             while (i < n - 1 && A[i + 1] <= A[i]) {
64                 i++;
65             }
66
67             if (i == n - 1) {
68                 break;
69             }
70
71             Interval e = new Interval();
72             e.buy = i++;
73             while (i < n && A[i] >= A[i - 1]) {
74                 i++;
75             }
76
77             e.sell = i - 1;
78             sol.add(e);
79             c++;
80         }
81
82         if (c == 0) {
83             return res;
84         } else {
85             for (int j = 0; j < sol.size(); j++) {
86                 res.add(new ArrayList<Integer>());
87                 res.get(j).add(0, sol.get(j).buy);
88                 res.get(j).add(1, sol.get(j).sell);
89             }
90         }
91
92         return res;
93     }
94 }
```

## OUTPUT :

Courses ▾ Tutorials ▾ Jobs ▾ Practice ▾ Contests ▾

Java (1.8) Average Time: 20m Start Timer

Output Window

Compilation Results Custom Input Y.O.G.I. (AI Bot)

Problem Solved Successfully ✓ Suggest Feedback

Test Cases Passed: 142 / 142

Attempts: Correct / Total: 1 / 1

Accuracy: 100%

Points Scored: 0 / 4

Your Total Score: 26

Time Taken: 0.09

Solve Next

You and your books

```
45 class Interval {
46     int buy;
47     int sell;
48 }
49
50 class Solution {
51     // Function to find the days of buying and selling stock for max profit.
52     ArrayList<ArrayList<Integer>> stockBuySell(int A[], int n) {
53         ArrayList<ArrayList<Integer>> res = new ArrayList<>();
54         if (n == 1) {
55             return res;
56         }
57         ArrayList<Interval> sol = new ArrayList<>();
58         int i = 0, c = 0;
59
60         while (i < n - 1) {
61             // Finding local minima
62             while (i < n - 1 && A[i + 1] <= A[i]) {
63                 i++;
64             }
65
66             if (i == n - 1) {
67                 break;
68             }
69
70             Interval e = new Interval();
71             e.buy = i++;
72             while (i < n && A[i] >= A[i - 1]) {
73                 i++;
74             }
75             e.sell = i - 1;
76             sol.add(e);
77         }
```

## 2. Remove Duplicates Sorted Array: (Easy level)

Courses ▾ Tutorials ▾ Jobs ▾ Practice ▾ Contests ▾

Java (1.8) Average Time: 20m Start Timer

Output Window

Compilation Results Custom Input Y.O.G.I. (AI Bot)

1115 / 1115

2 / 4

Accuracy: 50%

Time Taken: 1.39

You get marks only for the first correct submission if you solve the problem without viewing the full solution.

Solve Next

Sorted subsequence of size 3 Sort 0s, 1s and 2s Sort Unsorted Subarray

```
1 // } Driver Code Ends
2
3 // User function Template for Java
4
5 class Solution {
6     public int remove_duplicate(List<Integer> arr) {
7         if (arr.size() == 0 || arr.size() == 1) {
8             return arr.size();
9         }
10
11         int j = 0;
12         for (int i = 1; i < arr.size(); i++) {
13             if (!arr.get(j).equals(arr.get(i))) {
14                 j++;
15                 arr.set(j, arr.get(i));
16             }
17         }
18         return j + 1;
19     }
20 }
```

### 3. First Repeating Element : (Easy level)

**Output Window**

**Compilation Results** Custom Input Y.O.G.I. (AI Bot)

**Problem Solved Successfully** ✓

Test Cases Passed: **1115 / 1115**

Attempts : Correct / Total: **1 / 3**

Accuracy : 33%

Points Scored: **2 / 2**

Your Total Score: 30 ↑

Time Taken: **1.78**

**Solve Next**

Sorted subsequence of size 3 Array Duplicates

```
1 // Driver Code Ends
2
3 // User function Template for Java
4
5 class Solution {
6     // Function to return the position of the first repeating element.
7     public static int firstRepeated(int[] arr) {
8         // Your code here
9         HashMap<Integer, Integer> map = new HashMap<>();
10        for (int i = 0; i < arr.length; i++) {
11            map.put(arr[i], map.getOrDefault(arr[i], 0) + 1);
12        }
13        for (int i = 0; i < arr.length; i++) {
14            if (map.get(arr[i]) > 1) {
15                return i + 1;
16            }
17        }
18        return -1;
19    }
20 }
```

### 4. Find Transition Point : (Easy Level)

**Output Window**

**Compilation Results** Custom Input Y.O.G.I. (AI Bot)

**Problem Solved Successfully** ✓

Test Cases Passed: **1115 / 1115**

Attempts : Correct / Total: **1 / 2**

Accuracy : 50%

Points Scored: **2 / 2**

Your Total Score: 32 ↑

Time Taken: **0.53**

**Solve Next**

Index of an Extra Element Missing in Array Left most and right most index

```
17
18 Solution obj = new Solution();
19 System.out.println(obj.transitionPoint(arr));
20 System.out.println("~");
21 t--;
22 }
23 }
24 // } Driver Code Ends
25
26
27
28 class Solution {
29     int transitionPoint(int arr[]) {
30         // code here
31         int n = arr.length;
32         int index = -1;
33
34         for(int i=0; i < n; i++) {
35             if(arr[i] != 1){
36                 continue;
37             }
38             else{
39                 index = i;
40                 break;
41             }
42         }
43         return index;
44     }
45 }
46
47
48 }
```

## 5. Wave Array : (Easy Level)

The screenshot shows the GeeksforGeeks IDE interface. The top navigation bar includes links for Courses, Tutorials, Jobs, Practice, and Contests. The main header displays the GeeksforGeeks logo and a search bar. The left sidebar contains tabs for Problem, Editorial, Submissions, and Comments. The central area is divided into two main sections: the 'Output Window' on the left and the 'Code Editor' on the right.

The 'Output Window' displays the following information:

- Problem Solved Successfully** (with a green checkmark icon)
- Test Cases Passed:** 1120 / 1120
- Attempts:** Correct / Total: 1 / 1
- Accuracy:** 100%
- Points Scored:** 2 / 2
- Your Total Score:** 34 (with an upward arrow icon)
- Time Taken:** 0.88
- Solve Next:** Three way partitioning, Sort by Absolute Difference

The 'Code Editor' shows the following Java code:

```
1 // } Driver Code Ends
2
3 // User function Template for Java
4
5 class Solution {
6     public static void convertToWave(int[] arr) {
7         // code here
8         int i = 0;
9         while(i < arr.length-1){
10             int temp = arr[i];
11             arr[i] = arr[i+1];
12             arr[i+1] = temp;
13             i+=2;
14         }
15     }
16 }
```

## 6. First and Last Occurrences

The screenshot shows the GeeksforGeeks IDE interface. The top navigation bar includes links for Courses, Tutorials, Jobs, Practice, and Contests. The main header displays the GeeksforGeeks logo and a search bar. The left sidebar contains tabs for Problem, Editorial, Submissions, and Comments. The central area is divided into two main sections: the 'Output Window' on the left and the 'Code Editor' on the right.

The 'Output Window' displays the following information:

- Problem Solved Successfully** (with a green checkmark icon)
- Test Cases Passed:** 1120 / 1120
- Attempts:** Correct / Total: 1 / 5
- Accuracy:** 20%
- Points Scored:** 4 / 4
- Your Total Score:** 38 (with an upward arrow icon)
- Time Taken:** 1.15
- Solve Next:** Magical Number, Find the closest number, First and last occurrences of x

The 'Code Editor' shows the following Java code:

```
1 // } Driver Code Ends
2 // User function Template for Java
3
4 class GFG {
5     ArrayList<Integer> find(int arr[], int x) {
6         ArrayList<Integer> list = new ArrayList<>();
7         int n = arr.length;
8         int l = 0, r = n - 1;
9
10        list.add(-1);
11        list.add(-1);
12
13        while (l <= r) {
14            if (arr[l] == x && arr[r] == x) {
15                list.set(0, l);
16                list.set(1, r);
17                break;
18            }
19            if (arr[l] != x) {
20                l++;
21            }
22            if (arr[r] != x) {
23                r--;
24            }
25        }
26        return list;
27    }
28 }
```

## 7. Coin Change : ( Medium Level )

The screenshot displays a coding platform interface with the following components:

- Navigation Bar:** Includes links for Courses, Tutorials, Jobs, Practice, and Contests.
- Problem Header:** Shows 'Output Window', 'Compilation Results', 'Custom Input', and 'Y.O.G.I. (AI Bot)'.
- Problem Status:** 'Problem Solved Successfully' with a green checkmark and a 'Suggest Feedback' link.
- Test Results:**
  - Test Cases Passed: 1111 / 1111
  - Attempts: Correct / Total: 1 / 1
  - Accuracy: 100%
  - Points Scored: 4 / 4
  - Time Taken: 0.17
  - Your Total Score: 42 (with an upward arrow)
- Solve Next:** A section with buttons for 'Rod Cutting', 'Coin Change (Minimum Coins)', and 'Minimum number of Coins'.
- Code Editor:** Displays Java code for a dynamic programming solution. The code includes a driver code and a user function template. The solution class 'Solution' contains a 'count' method that uses a 1D DP array to calculate the minimum number of coins for a given sum.
- Bottom Bar:** Includes a 'Custom Input' field and buttons for 'Compile & Run' and 'Submit'.

```
1 // Driver Code Ends
28
29
30 // User function Template for Java
31
32 class Solution {
33     public int count(int coins[], int sum) {
34         int n = coins.length;
35         int[] dp = new int[sum + 1];
36         dp[0] = 1;
37         for (int coin : coins) {
38             for (int i = coin; i <= sum; i++) {
39                 dp[i] += dp[i - coin];
40             }
41         }
42         return dp[sum];
43     }
44 }
45
```