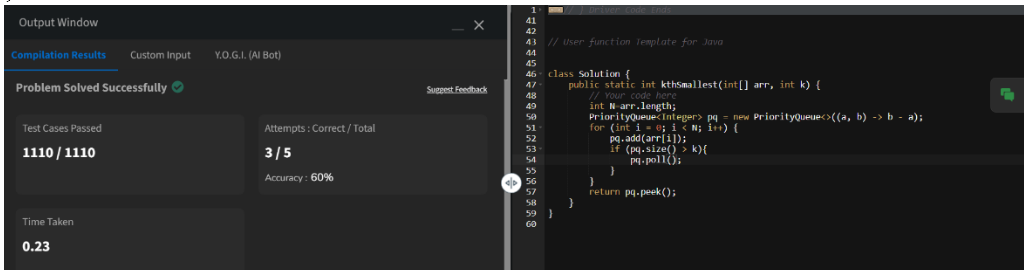
Date: 13/11/2024

DSA Practice Problems

1. Kth Smallest Element

```
class Solution {
    public static int kthSmallest(int[] arr, int k) {
        int N=arr.length;
        PriorityQueue<Integer> pq = new PriorityQueue<>((a, b) -> b - a);
        for (int i = 0; i < N; i++) {
            pq.add(arr[i]);
            if (pq.size() > k){
                pq.poll();
            }
        }
        return pq.peek();
    }
}
```



Problem Solved Successfully

Test Cases Passed
1110 / 1110

Attempts : Correct / Total
3 / 5
Accuracy : 60%

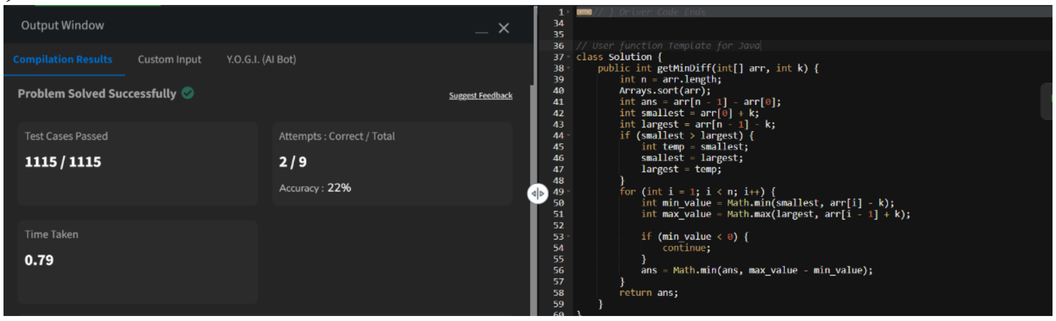
Time Taken
0.23

```
// } Driver Code Ends


// User function Template for Java


class Solution {
    public static int kthSmallest(int[] arr, int k) {
        // Your code here
        int N=arr.length;
        PriorityQueue<Integer> pq = new PriorityQueue<>((a, b) -> b - a);
        for (int i = 0; i < N; i++) {
            pq.add(arr[i]);
            if (pq.size() > k){
                pq.poll();
            }
        }
        return pq.peek();
    }
}
```

Time Complexity: O (N log N)

2. Minimize the Height II

```
class Solution {
    public int getMinDiff(int[] arr, int k) {
        int n = arr.length;
        Arrays.sort(arr);
        int ans = arr[n - 1] - arr[0];
        int smallest = arr[0] + k;
        int largest = arr[n - 1] - k;
        if (smallest > largest) {
            int temp = smallest;
            smallest = largest;
            largest = temp;
        }
        for (int i = 1; i < n; i++) {
            int min_value = Math.min(smallest, arr[i] - k);
            int max_value = Math.max(largest, arr[i - 1] + k);
            if (min_value < 0) {
                continue;
            }
            ans = Math.min(ans, max_value - min_value);
        }
        return ans;
    }
}
```



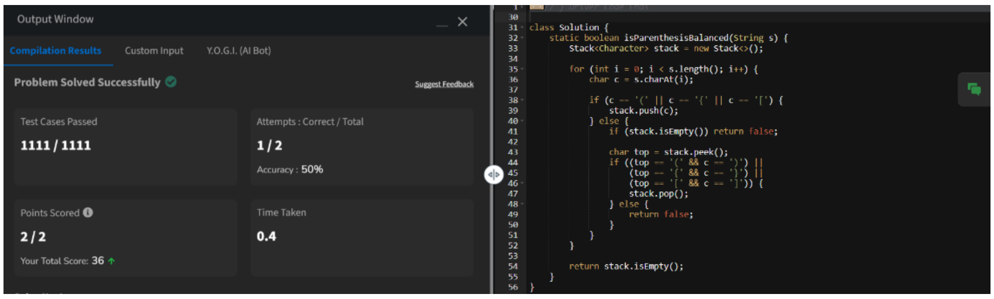
```
Output Window

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

Problem Solved Successfully                              Suggest Feedback

Test Cases Passed          Attempts : Correct / Total
1115 / 1115                2 / 9
                           Accuracy : 22%

Time Taken
0.79
```

```java
// } Driver Code Ends


// User function Template for Java
class Solution {
    public int getMinDiff(int[] arr, int k) {
        int n = arr.length;
        Arrays.sort(arr);
        int ans = arr[n - 1] - arr[0];
        int smallest = arr[0] + k;
        int largest = arr[n - 1] - k;
        if (smallest > largest) {
            int temp = smallest;
            smallest = largest;
            largest = temp;
        }
        for (int i = 1; i < n; i++) {
            int min_value = Math.min(smallest, arr[i] - k);
            int max_value = Math.max(largest, arr[i - 1] + k);

            if (min_value < 0) {
                continue;
            }
            ans = Math.min(ans, max_value - min_value);
        }
        return ans;
    }
}
```

Time Complexity: O(n log n)

3. Paranthesis checker

```java
class Solution {
    static boolean isParenthesisBalanced(String s) {
        Stack<Character> stack = new Stack<>();

        for (int i = 0; i < s.length(); i++) {
            char c = s.charAt(i);

            if (c == '(' || c == '{' || c == '[') {
                stack.push(c);
            } else {
                if (stack.isEmpty()) return false;

                char top = stack.peek();
                if ((top == '(' && c == ')') ||
                    (top == '{' && c == '}') ||
                    (top == '[' && c == ']')) {
                    stack.pop();
                } else {
                    return false;
                }
            }
        }

        return stack.isEmpty();
    }
}
```



Problem Solved Successfully

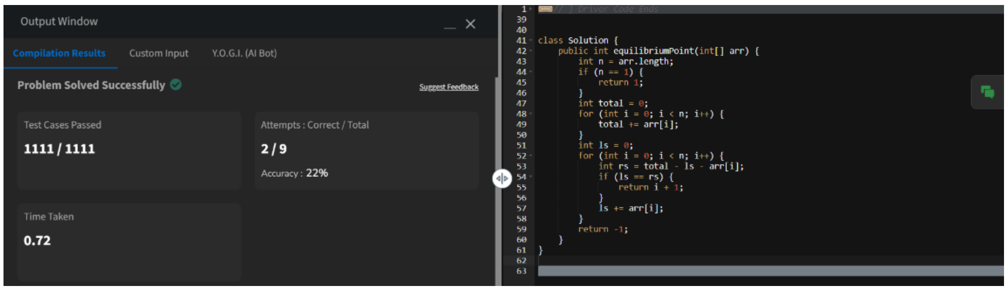
```
Test Cases Passed        Attempts : Correct / Total
1111 / 1111              1 / 2
                         Accuracy : 50%

Points Scored            Time Taken
2 / 2                    0.4
Your Total Score: 36
```

```java
class Solution {
    static boolean isParenthesisBalanced(String s) {
        Stack<Character> stack = new Stack<>();

        for (int i = 0; i < s.length(); i++) {
            char c = s.charAt(i);

            if (c == '(' || c == '{' || c == '[') {
                stack.push(c);
            } else {
                if (stack.isEmpty()) return false;

                char top = stack.peek();
                if ((top == '(' && c == ')') ||
                    (top == '{' && c == '}') ||
                    (top == '[' && c == ']')) {
                    stack.pop();
                } else {
                    return false;
                }
            }
        }

        return stack.isEmpty();
    }
}
```

Time Complexity: O(n)

4. **Equilibrium Point**

```
class Solution {
    public int equilibriumPoint(int[] arr) {
        int n = arr.length;
        if (n == 1) {
            return 1;
        }
        int total = 0;
        for (int i = 0; i < n; i++) {
            total += arr[i];
        }
        int ls = 0;
        for (int i = 0; i < n; i++) {
            int rs = total - ls - arr[i];
            if (ls == rs) {
                return i + 1;
            }
            ls += arr[i];
        }
        return -1;
    }
}
```


Output Window

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

Problem Solved Successfully

Suggest Feedback

Test Cases Passed
1111 / 1111

Attempts : Correct / Total
2 / 9
Accuracy : 22%

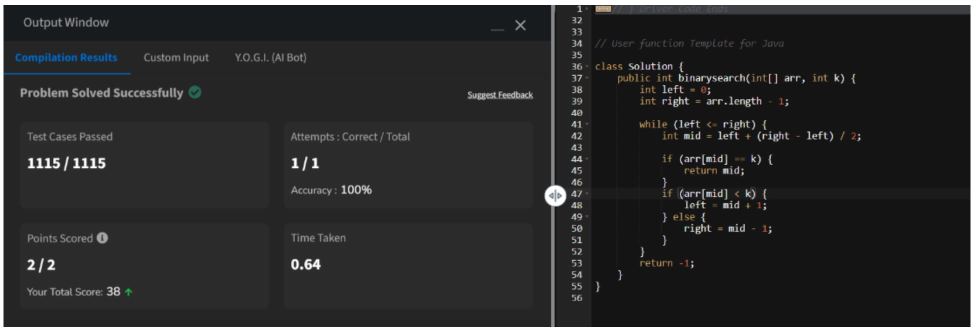
Time Taken
0.72

```
// } Driver Code Ends


class Solution {
    public int equilibriumPoint(int[] arr) {
        int n = arr.length;
        if (n == 1) {
            return 1;
        }
        int total = 0;
        for (int i = 0; i < n; i++) {
            total += arr[i];
        }
        int ls = 0;
        for (int i = 0; i < n; i++) {
            int rs = total - ls - arr[i];
            if (ls == rs) {
                return i + 1;
            }
            ls += arr[i];
        }
        return -1;
    }
}
```

Time Complexity: O(n)

5. Binary Search

```
class Solution {
    public int binarysearch(int[] arr, int k) {
        int left = 0;
        int right = arr.length - 1;

        while (left <= right) {
            int mid = left + (right - left) / 2;

            if (arr[mid] == k) {
                return mid;
            }
            if (arr[mid] < k) {
                left = mid + 1;
            } else {
                right = mid - 1;
            }
        }
        return -1;
    }
}
```



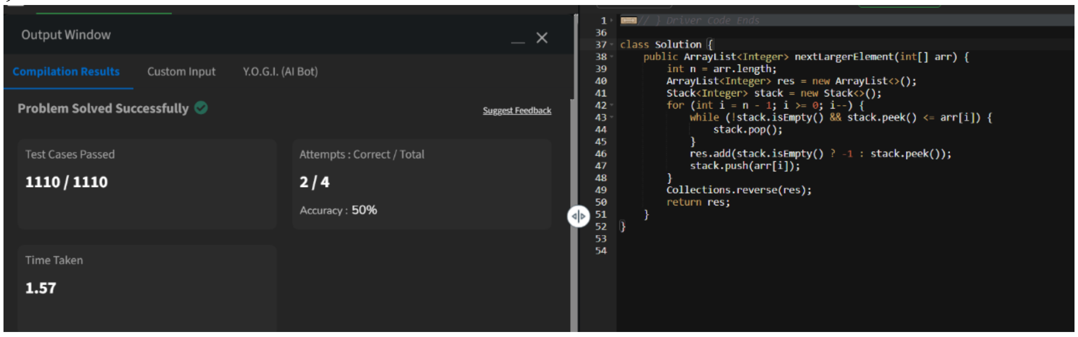
```
Output Window
Compilation Results    Custom Input    Y.O.G.I. (AI Bot)
Problem Solved Successfully                         Suggest Feedback

Test Cases Passed            Attempts : Correct / Total
1115 / 1115                  1 / 1
                             Accuracy : 100%

Points Scored                Time Taken
2 / 2                        0.64
Your Total Score: 38
```

```java
// } Driver Code Ends


// User function Template for Java

class Solution {
    public int binarysearch(int[] arr, int k) {
        int left = 0;
        int right = arr.length - 1;

        while (left <= right) {
            int mid = left + (right - left) / 2;

            if (arr[mid] == k) {
                return mid;
            }
            if (arr[mid] < k) {
                left = mid + 1;
            } else {
                right = mid - 1;
            }
        }
        return -1;
    }
}
```

Time Complexity: O(log n)

6. Next Greater Element

```java
class Solution {
    public ArrayList<Integer> nextLargerElement(int[] arr) {
        int n = arr.length;
        ArrayList<Integer> res = new ArrayList<>();
        Stack<Integer> stack = new Stack<>();

        for (int i = n - 1; i >= 0; i--) {
            while (!stack.isEmpty() && stack.peek() <= arr[i]) {
                stack.pop();
            }
            if (stack.isEmpty()) {
                res.add(-1);
            } else {
                res.add(stack.peek());
            }
            stack.push(arr[i]);
        }

        Collections.reverse(res);
        return res;
    }
}
```



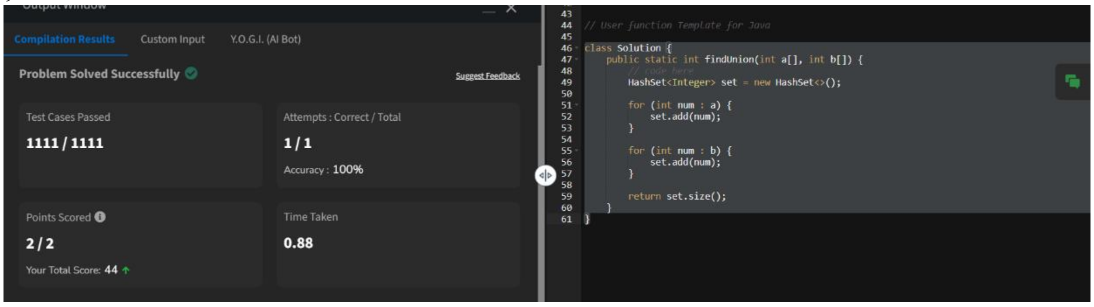
```
Output Window
Compilation Results    Custom Input    Y.O.G.I. (AI Bot)
Problem Solved Successfully                         Suggest Feedback

Test Cases Passed            Attempts : Correct / Total
1110 / 1110                  2 / 4
                             Accuracy : 50%

Time Taken
1.57
```

```java
// } Driver Code Ends

class Solution {
    public ArrayList<Integer> nextLargerElement(int[] arr) {
        int n = arr.length;
        ArrayList<Integer> res = new ArrayList<>();
        Stack<Integer> stack = new Stack<>();
        for (int i = n - 1; i >= 0; i--) {
            while (!stack.isEmpty() && stack.peek() <= arr[i]) {
                stack.pop();
            }
            res.add(stack.isEmpty() ? -1 : stack.peek());
            stack.push(arr[i]);
        }
        Collections.reverse(res);
        return res;
    }
}
```

Time Complexity: O(n)

7. **Union of two arrays**

```
class Solution {
    public static int findUnion(int a[], int b[]) {
        // code here
        HashSet<Integer> set = new HashSet<>();

        for (int num : a) {
            set.add(num);
        }

        for (int num : b) {
            set.add(num);
        }

        return set.size();
    }
}
```



Problem Solved Successfully

Test Cases Passed
1111 / 1111

Attempts : Correct / Total
1 / 1
Accuracy : 100%

Points Scored
2 / 2
Your Total Score: 44

Time Taken
0.88

```
// User function Template for Java

class Solution {
    public static int findUnion(int a[], int b[]) {
        // code here
        HashSet<Integer> set = new HashSet<>();

        for (int num : a) {
            set.add(num);
        }

        for (int num : b) {
            set.add(num);
        }

        return set.size();
    }
}
```

Time Complexity: O(n+m)