



# DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

Discover. Learn. Empower.

## Assignment-10

**Student Name:**Ujjwal kumar

**Branch:** CSE

**Semester:** 6<sup>th</sup>

**Subject:** AP

**UID:** 22BCS14185

**Section:** IOT\_640(B)

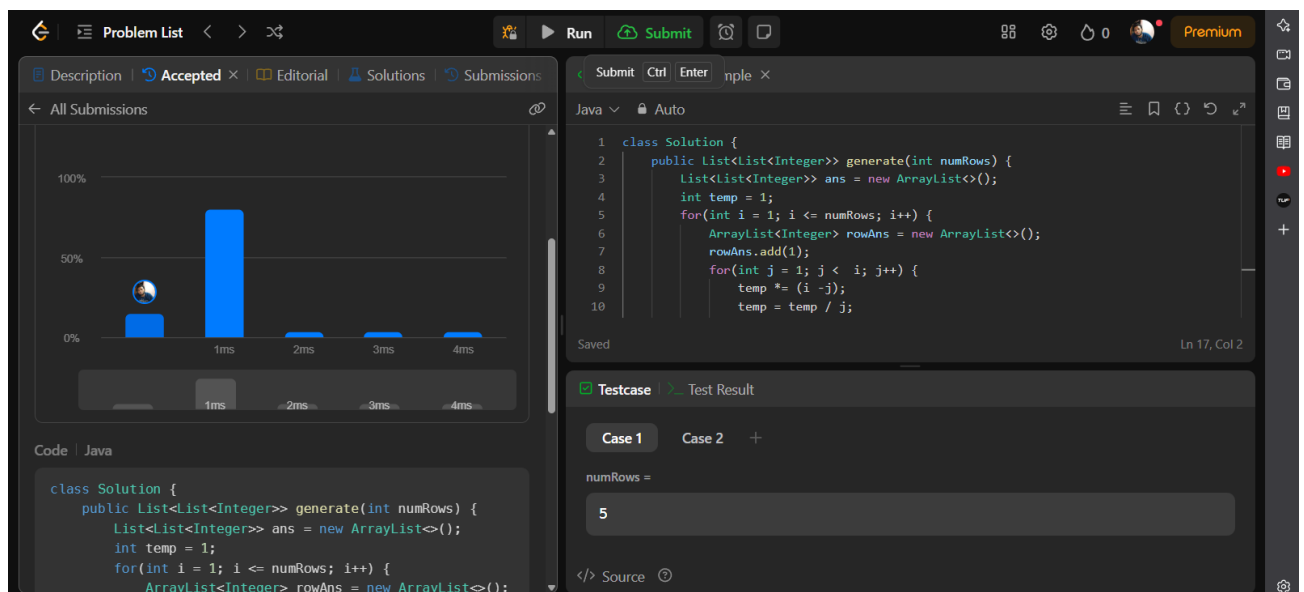
**Date:** 20/04/2025

**Subject Code:** 22CSP-351

### Problem 1:

```
class Solution {  
    public List<List<Integer>> generate(int numRows) {  
        List<List<Integer>> ans = new ArrayList<>();  
        int temp = 1;  
        for(int i = 1; i <= numRows; i++) {  
            ArrayList<Integer> rowAns = new ArrayList<>();  
            rowAns.add(1);  
            for(int j = 1; j < i; j++) {  
                temp *= (i - j);  
                temp = temp / j;  
                rowAns.add(temp);  
            }  
            ans.add(rowAns);  
        }  
        return ans;  
    }  
}
```

### Screenshot:





# DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

Discover. Learn. Empower.

## Problem 2:

```
class Solution {
    public int divide(int dividend, int divisor) {

        if (dividend == Integer.MIN_VALUE && divisor == -1) {
            return Integer.MAX_VALUE;
        }

        long ldividend = Math.abs((long) dividend);
        long ldivisor = Math.abs((long) divisor);
        int result = 0;

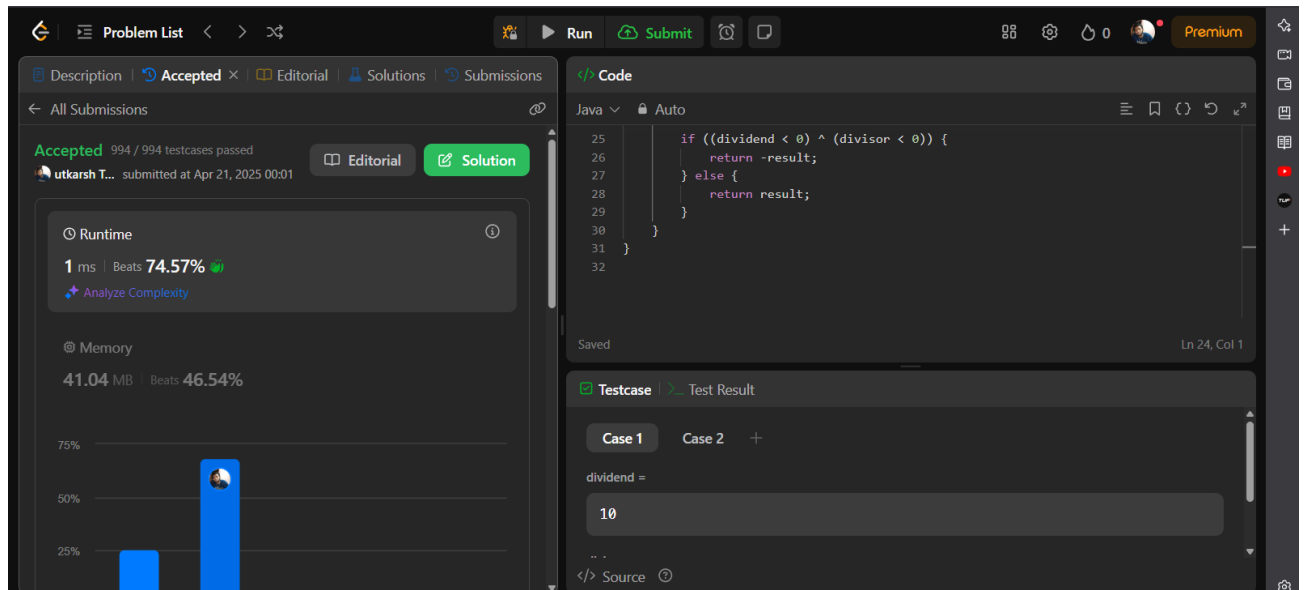
        while (ldividend >= ldivisor) {
            long temp = ldivisor, multiple = 1;

            while (ldividend >= (temp << 1)) {
                temp <<= 1;
                multiple <<= 1;
            }

            ldividend -= temp;
            result += multiple;
        }

        if ((dividend < 0) ^ (divisor < 0)) {
            return -result;
        } else {
            return result;
        }
    }
}
```

## Screenshot:



## Problem 3:

```
class Solution {
    public int trap(int[] height) {
        int left = 0, right = height.length - 1;
        int leftMax = 0, rightMax = 0;
        int water = 0;

        while (left < right) {
            if (height[left] < height[right]) {
                if (height[left] >= leftMax) {
                    leftMax = height[left];
                } else {
                    water += leftMax - height[left];
                }
                left++;
            } else {
                if (height[right] >= rightMax) {
                    rightMax = height[right];
                } else {
                    water += rightMax - height[right];
                }
                right--;
            }
        }

        return water;
    }
}
```



# DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

Discover. Learn. Empower.

## Screenshot:

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

- Top Bar:** Includes navigation icons, a 'Problem List' dropdown, and buttons for 'Run', 'Submit', and 'Enter'.
- Left Sidebar:**
  - Runtime:** Shows '1 ms' and 'Beats 63.34%' with a green checkmark. A link to 'Analyze Complexity' is present.
  - Memory:** Shows '46.54 MB' and 'Beats 39.23%'.
  - Bar Chart:** A chart showing performance across different time intervals (1ms to 5ms). The 1ms bar is the highest, reaching approximately 50%.
- Main Editor:** Displays Java code for a 'trap' function. The code is as follows:

```
1 class Solution {
2     public int trap(int[] height) {
3         int left = 0, right = height.length - 1;
4         int leftMax = 0, rightMax = 0;
5         int water = 0;
6
7         while (left < right) {
8             if (height[left] < height[right]) {
9                 if (height[left] >= leftMax) {
10                     leftMax = height[left];
11                 }
12             }
13             // Similar logic for right side
14         }
15     }
16 }
```
- Bottom Right:** Shows a 'Testcase' section with 'Case 1' selected. The input is 'height = [0, 1, 0, 2, 1, 0, 1, 3, 2, 1, 2, 1]'.