**Assignment 10**

**Name – Aditya Taragi Uid – 22BCS10422.**

**Problem 1: Pascals Triangle (**<https://leetcode.com/problems/pascals-triangle/> **)**

**Code:**class Solution {

public:

    vector<vector<int>> generate(int numRows) {

        vector<vector<int>> triangle(numRows);

        for (int i = 0; i < numRows; ++i) {

            triangle[i].resize(i + 1);

            triangle[i][0] = triangle[i][i] = 1;

            for (int j = 1; j < i; ++j) {

                triangle[i][j] = triangle[i - 1][j - 1] + triangle[i - 1][j];

            }

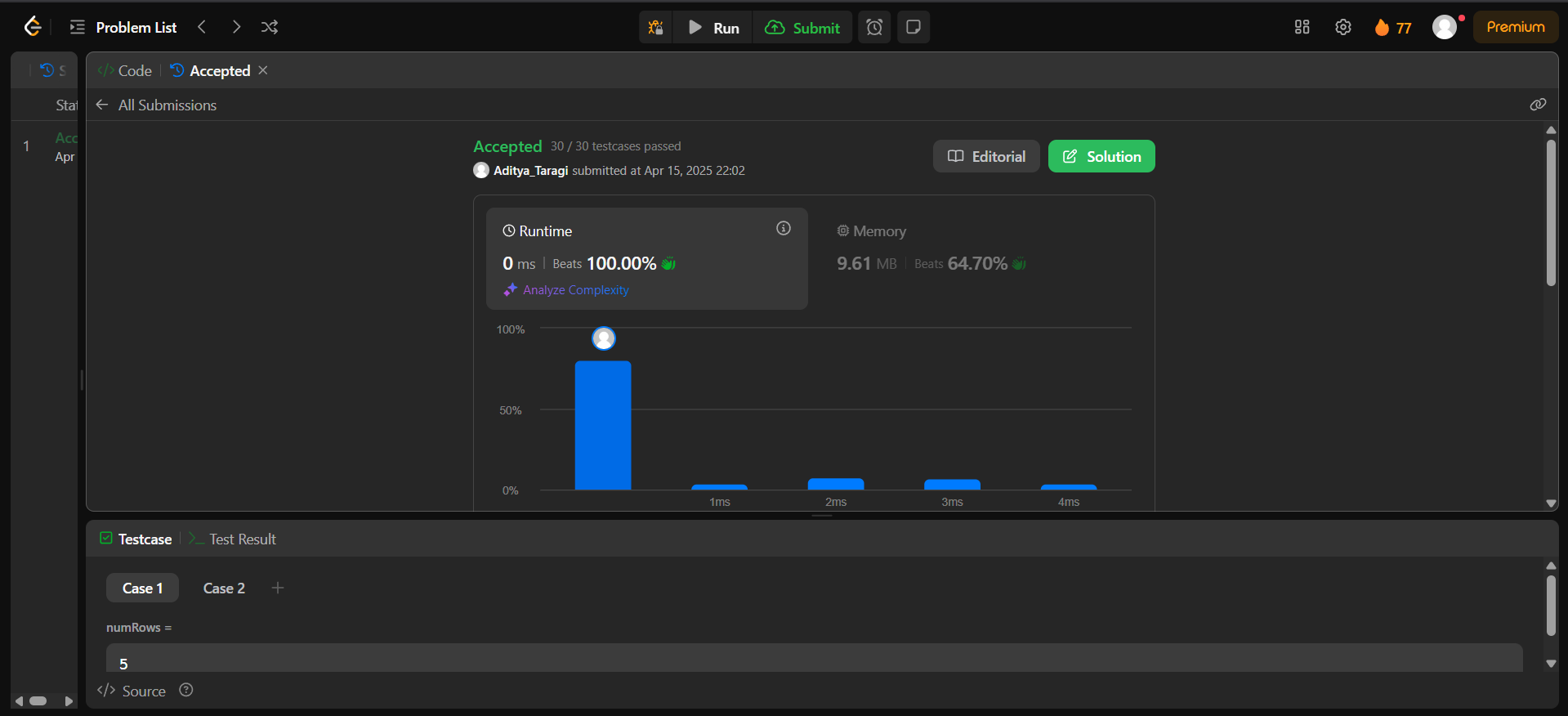
        }

        return triangle;

    }

};

**Screenshot:**



**Problem 2: Hamming Distance (**<https://leetcode.com/problems/hamming-distance/> **)**

**Code:**class Solution {

public:

    int hammingDistance(int x, int y) {

        int xorVal = x ^ y;

        int count = 0;

        while (xorVal > 0) {

            count += xorVal & 1;

            xorVal >>= 1;

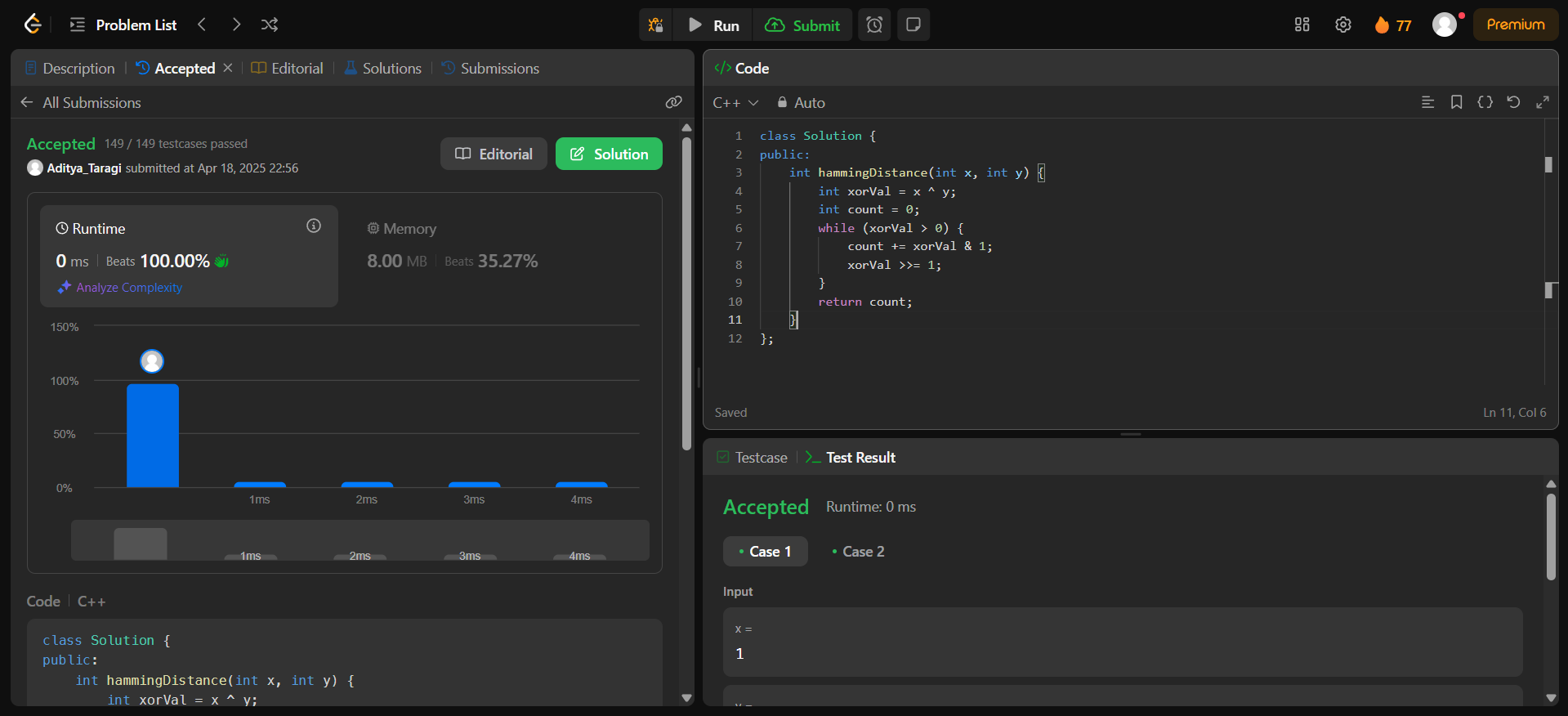
        }

        return count;

    }

};

**Screenshot:**



**Problem 3: Number of 1 bits (**<https://leetcode.com/problems/number-of-1-bits/> **)**

**Code:**class Solution {

public:

    int hammingWeight(int n) {

        int count = 0;

        while(n) {

            n = n & (n-1);

            count++;

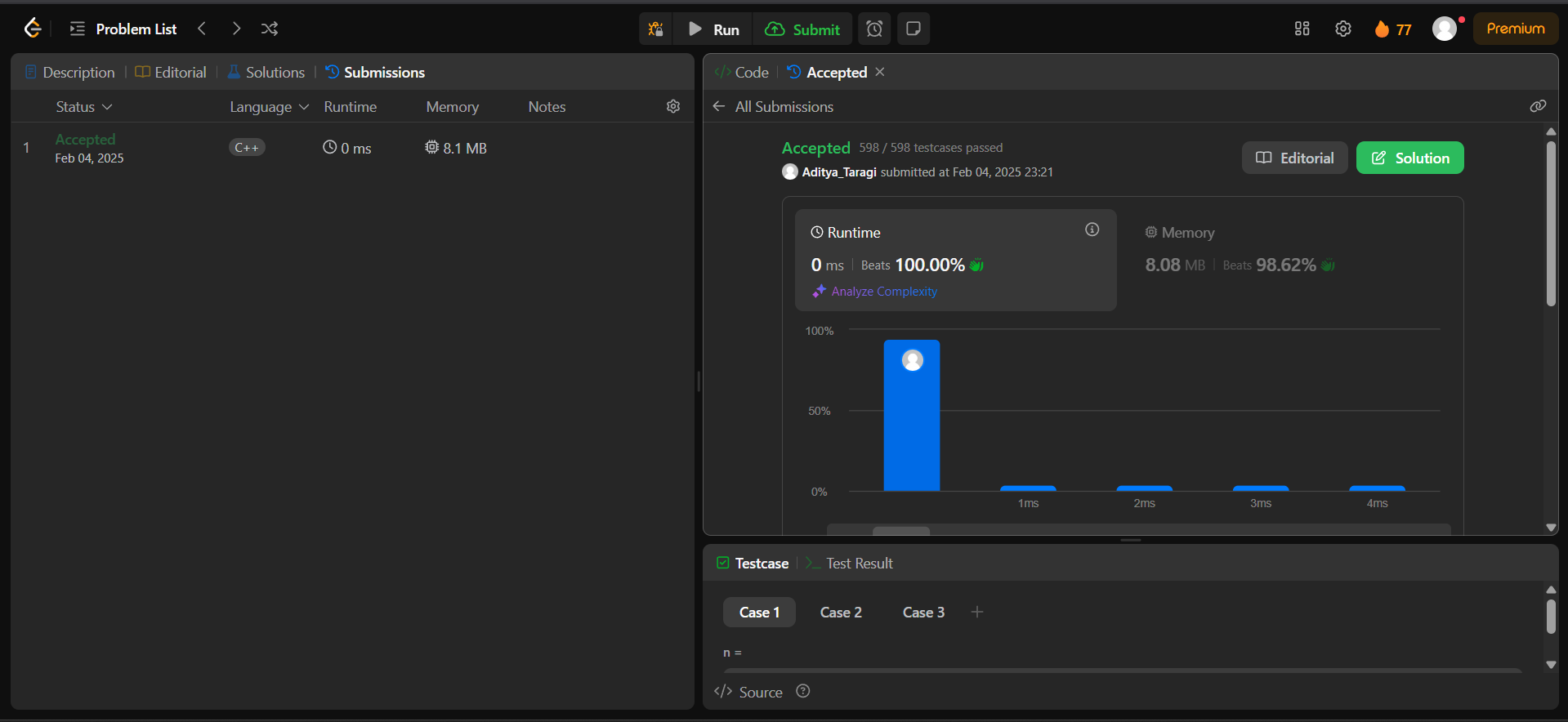
        }

        return count;

    }

};

**Screenshot:**



**Problem 4: Divide Two Integers (**<https://leetcode.com/problems/divide-two-integers/> **)**

**Code:**if (dividend == INT\_MIN && divisor == -1) return INT\_MAX;

        bool negative = (dividend < 0) ^ (divisor < 0);

        long long a = abs((long long)dividend);

        long long b = abs((long long)divisor);

        long long result = 0;

        while (a >= b) {

            long long temp = b, multiple = 1;

            while (a >= (temp << 1)) {

                temp <<= 1;

                multiple <<= 1;

            }

            a -= temp;

            result += multiple;

        }

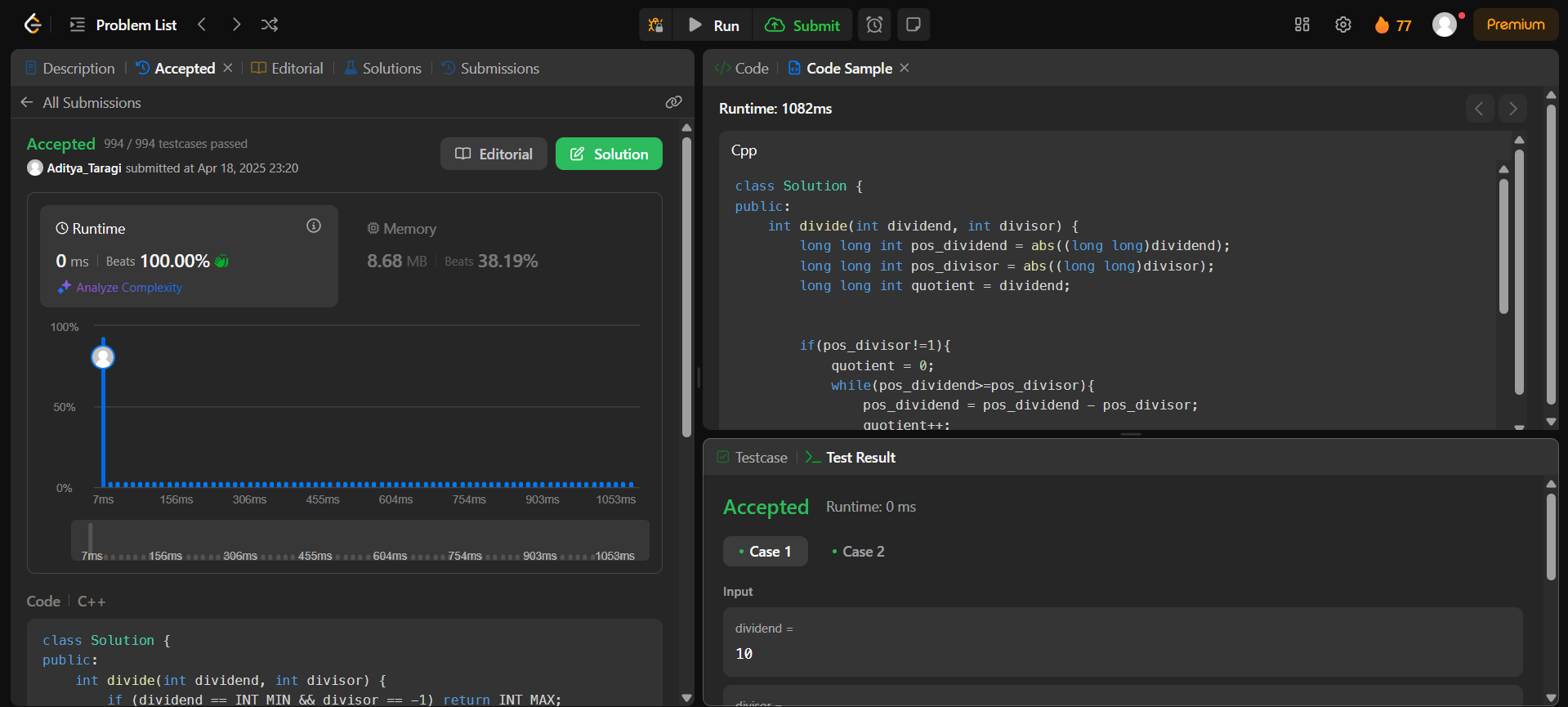
        if (negative) result = -result;

        return result;

}

};

**Screenshot:**



**Problem 5: Trapping Rain Water (**<https://leetcode.com/problems/trapping-rain-water/>**)**

**Code:**class Solution {

public:

    int trap(vector<int>& height) {

        int left = 0, right = height.size() - 1;

        int leftMax = 0, rightMax = 0, waterTrapped = 0;

        while (left < right) {

            if (height[left] < height[right]) {

                if (height[left] >= leftMax)

                    leftMax = height[left];

                else

                    waterTrapped += leftMax - height[left];

                left++;

            } else {

                if (height[right] >= rightMax)

                    rightMax = height[right];

                else

                    waterTrapped += rightMax - height[right];

                right--;

            }

        }

        return waterTrapped;

    }

};

**Screenshot:**

