

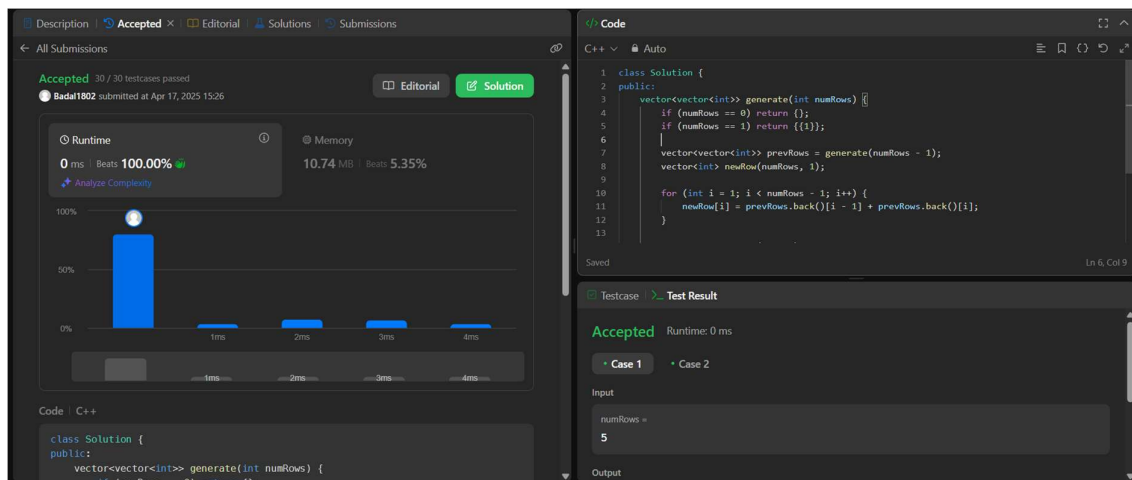
Experiment 10

Pascal's Triangle

Code:

```
class Solution {
public:
    vector<vector<int>> generate(int numRows) {
        if (numRows == 0) return {};
        if (numRows == 1) return {{1}};
        vector<vector<int>> prevRows = generate(numRows - 1);
        vector<int> newRow(numRows, 1);
        for (int i = 1; i < numRows - 1; i++) {
            newRow[i] = prevRows.back()[i - 1] + prevRows.back()[i];
        }
        prevRows.push_back(newRow);
        return prevRows;
    }
};
```

OUTPUT:



Hamming Distance

Code:

```
class Solution {  
public:  
    int hammingDistance(int x, int y) {  
        int ans = x ^ y;  
        int count = 0;  
        while(ans){  
            count += (ans & 1);  
            ans >>= 1;  
        }  
        return count;  
    }  
};
```

OUTPUT:

The screenshot displays a code editor interface with the following components:

- Accepted** status: Submitted at Apr 17, 2025 15:51.
- Runtime**: 0 ms | Beats 100.00%.
- Memory**: 8.00 MB | Beats 35.37%.
- Code Editor**: Shows the C++ code for the Hamming Distance function.
- Testcase**: Shows the input for Case 1: x = 1, y = 1.
- Test Result**: Shows the output for Case 1: 0.

The code in the editor is as follows:

```
1 class Solution {  
2 public:  
3     int hammingDistance(int x, int y) {  
4         int ans = x ^ y;  
5         int count = 0;  
6         while(ans){  
7             count += (ans & 1);  
8             ans >>= 1;  
9         }  
10        return count;  
11    }  
12 };
```

The test result shows the following input and output:

```
Input:  
x =  
1  
y =  
1  
Output:  
0
```

Task Scheduler

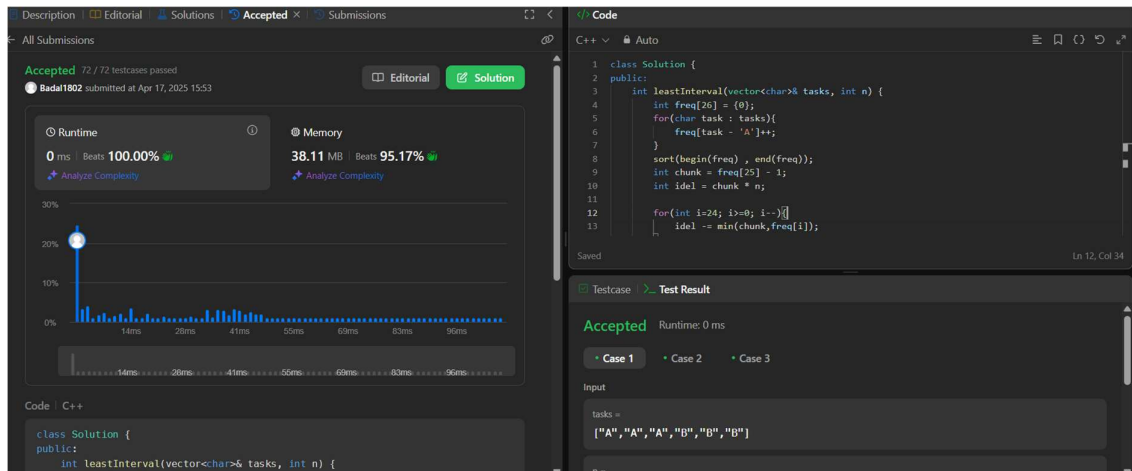
CODE:

```
class Solution {
public:
    int leastInterval(vector<char>& tasks, int n) {
        int freq[26] = {0};
        for(char task : tasks){
            freq[task - 'A']++;
        }
        sort(begin(freq) , end(freq));
        int chunk = freq[25] - 1;
        int idel = chunk * n;

        for(int i=24; i>=0; i--){
            idel -= min(chunk,freq[i]);
        }

        return idel < 0 ? tasks.size() : tasks.size() + idel;
    }
};
```

OUTPUT:

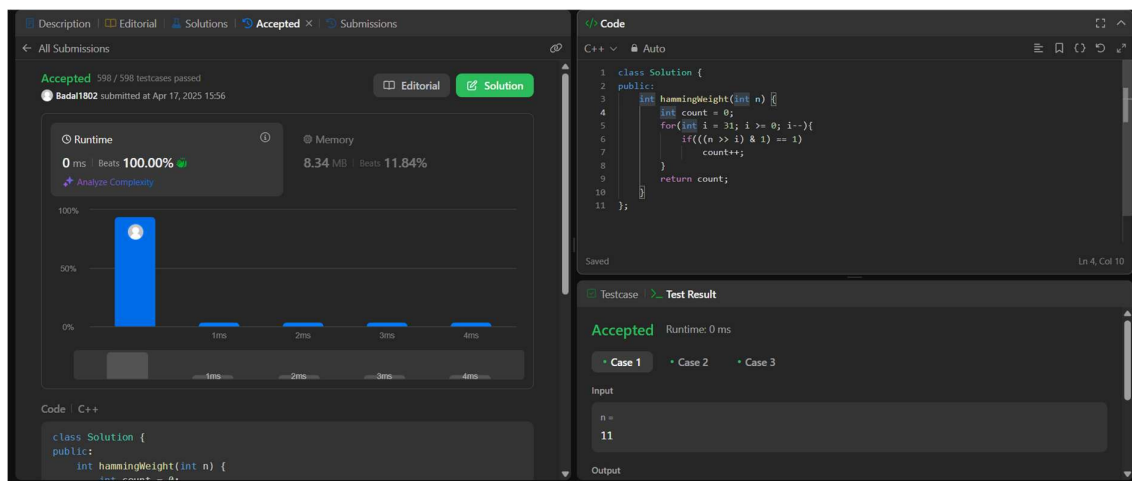


Number of 1 Bits

CODE:

```
class Solution {
public:
    int hammingWeight(int n) {
        int count = 0;
        for(int i = 31; i >= 0; i--){
            if(((n >> i) & 1) == 1)
                count++;
        }
        return count;
    }
};
```

OUTPUT:



Divide Two Integers

CODE:

```
class Solution {
public:
    int divide(int dividend, int divisor) {
        if (dividend == divisor) return 1;
        if (dividend == INT_MIN && divisor == -1) return INT_MAX;
        if (divisor == 1) return dividend;

        int sign = (dividend < 0) ^ (divisor < 0) ? -1 : 1;

        long long n = abs((long long)dividend);
        long long d = abs((long long)divisor);
        int ans = 0;

        while (n >= d) {
            int p = 0;
            while (n >= (d << p)) p++;
            p--;
            n -= (d << p);
            ans += (1 << p);
        }

        return sign * ans;
    }
};
```

OUTPUT:

DescriptionEditorialSolutionsAcceptedSubmissions

All Submissions

Accepted994 / 994 testcases passed

Badal1802 submitted at Apr 17, 2025 16:00

EditorialSolution

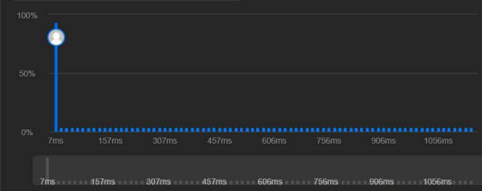
Runtime

0 ms | Beats 100.00%

Analyze Complexity

Memory

8.62 MB | Beats 38.28%



7ms157ms307ms457ms606ms756ms906ms1056ms

CodeC++

```
class Solution {
public:
    int divide(int dividend, int divisor) {
        if (dividend == divisor) return 1;
        if (dividend == INT_MIN && divisor == -1) return INT_MAX;
        if (divisor == 1) return dividend;

        int sign = (dividend < 0) ^ (divisor < 0) ? -1 : 1;

        long long n = abs((long long)dividend);
        long long d = abs((long long)divisor);
        int ans = 0;
    }
};
```

Code

C++Auto

```
1 class Solution {
2 public:
3     int divide(int dividend, int divisor) {
4         if (dividend == divisor) return 1;
5         if (dividend == INT_MIN && divisor == -1) return INT_MAX;
6         if (divisor == 1) return dividend;
7
8         int sign = (dividend < 0) ^ (divisor < 0) ? -1 : 1;
9
10        long long n = abs((long long)dividend);
11        long long d = abs((long long)divisor);
12        int ans = 0;
13    }
14};
```

SavedLn 3, Col 27

TestcaseTest Result

AcceptedRuntime: 0 ms

Case 1Case 2

Input

dividend =

10

divisor =