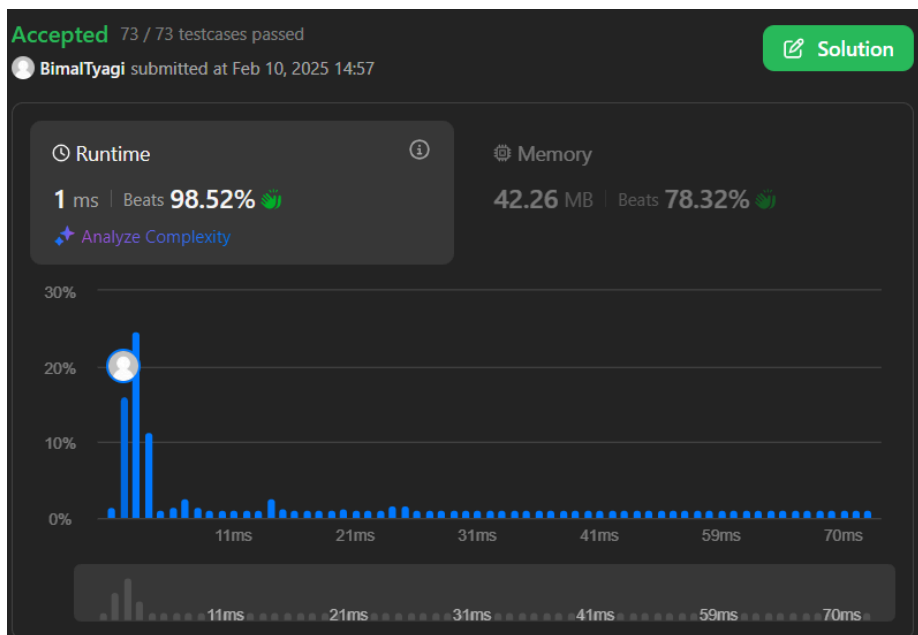


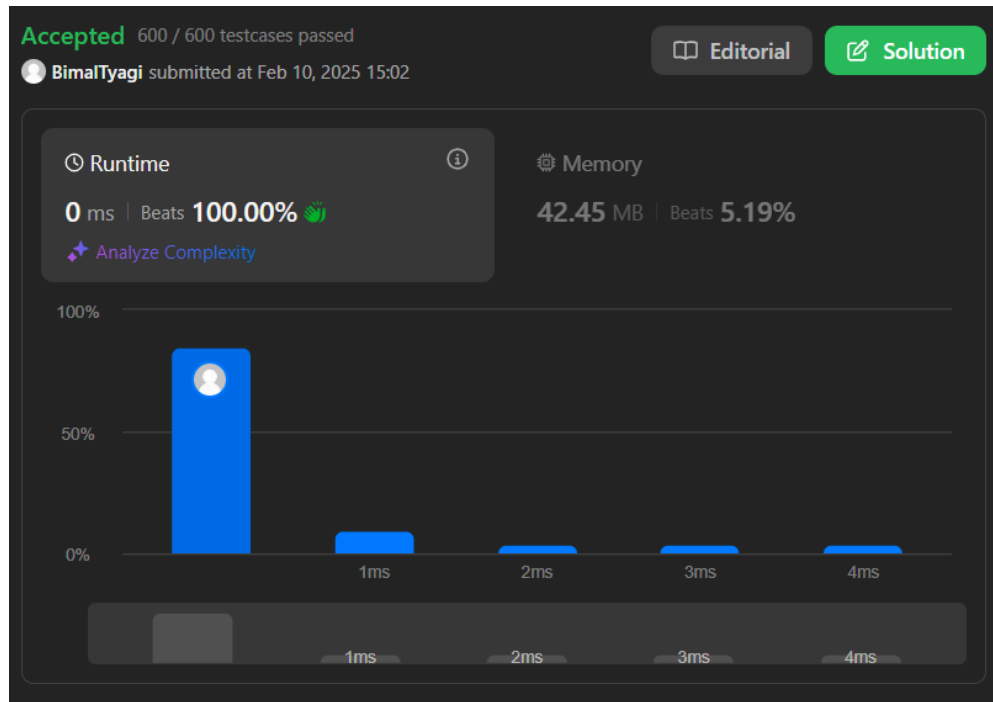
1. Longest Nice Substring

```
class Solution {
    public String longestNiceSubstring(String s) {
        int n = s.length(), m = 0, L = 0;
        for (int i = 0; i < n; i++)
            for (int j = i; j < n; j++) {
                int x = 0, y = 0;
                for (int k = i; k <= j; k++) {
                    x |= 1 << (s.charAt(k) - 'A');
                    y |= 1 << (s.charAt(k) - 'a');
                }
                if ((x | y) == (x ^ y) && j - i + 1 > m) {
                    m = j - i + 1;
                    L = i;
                }
            }
        return s.substring(L, L + m);
    }
}
```



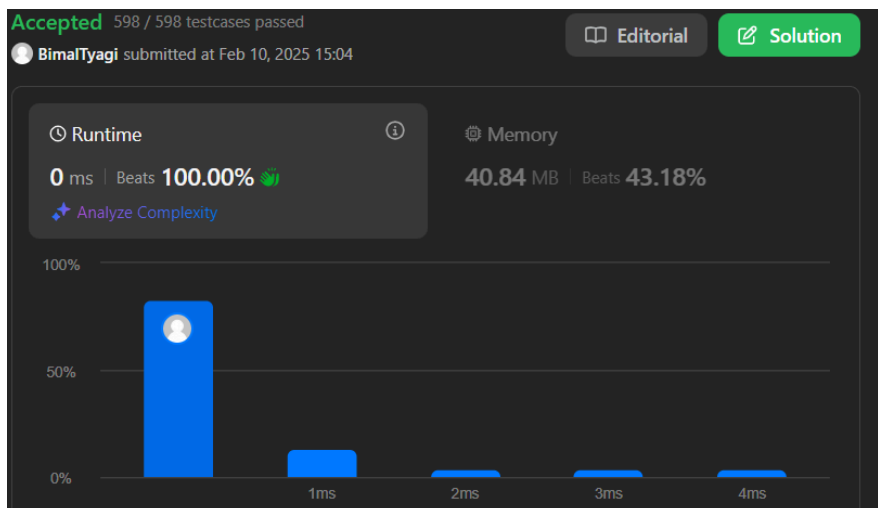
2. Number of 1 Bits

```
class Solution {
    public int hammingWeight(int n) {
        int c = 0;
        while (n != 0) {
            c += n & 1;
            n >>= 1;
        }
        return c;
    }
}
```



3. Maximum Subarray

```
class Solution {
    public int maxSubArray(int[] N) {
        int S = N[0], C = 0;
        for (int i : N) {
            C = Math.max(i, C + i);
            S = Math.max(S, C);
        }
        return S;
    }
}
```



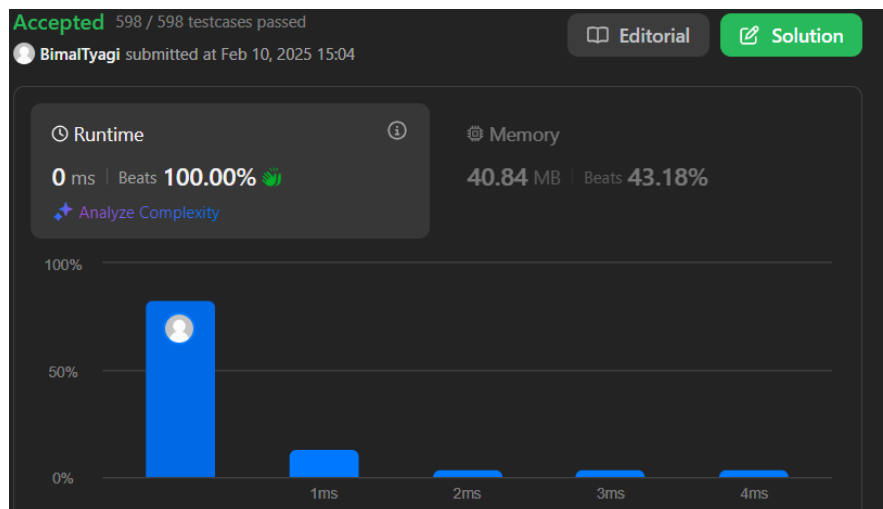
4. Search a 2D Matrix II

```
class Solution {
    public boolean searchMatrix(int[][] M, int T) {
        int r = 0, c = M[0].length - 1;
```

```

while (r < M.length && c >= 0) {
    if (M[r][c] == T) return true;
    if (M[r][c] > T) c--;
    else r++;
}
return false;
}
}

```

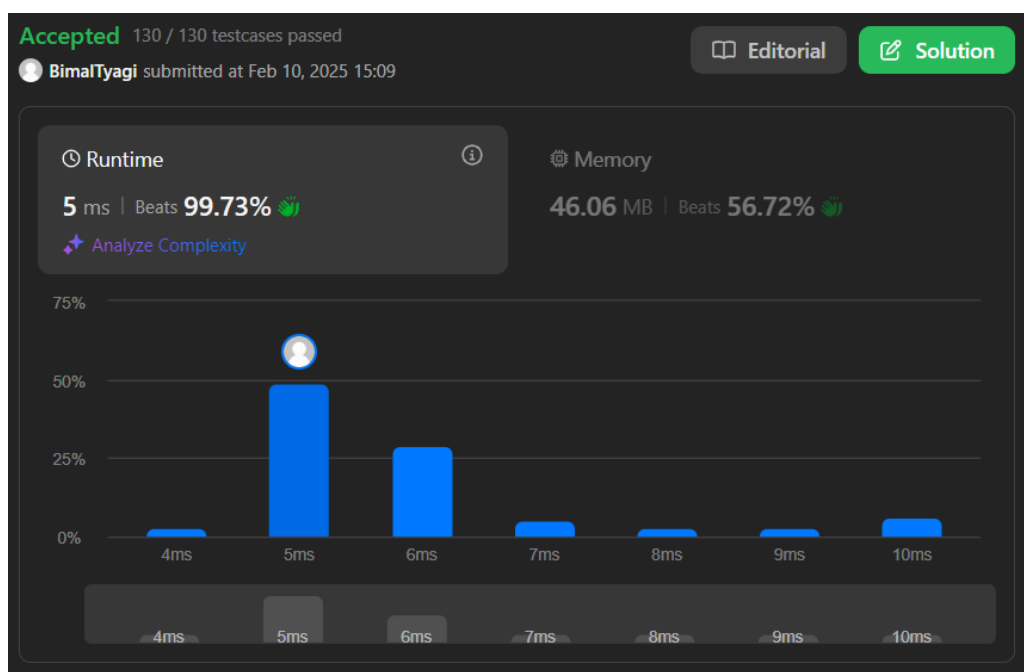


5. Super Pow

```

class Solution {
    int mod = 1337;
    public int superPow(int a, int[] b) {
        return f(a, b, b.length - 1);
    }
    int f(int a, int[] b, int i) {
        if (i < 0) return 1;
        return pow(f(a, b, i - 1), 10) * pow(a, b[i]) % mod;
    }
    int pow(int a, int b) {
        int r = 1;
        for (a %= mod; b > 0; b /= 2, a = a * a % mod)
            if (b % 2 == 1) r = r * a % mod;
        return r;
    }
}


```






6. Reverse Bits

```
class Solution {  
    public int reverseBits(int n) {  
        int result = 0;  
        for (int i = 0; i < 32; i++) {  
            result <<= 1;  
            result |= (n & 1);  
            n >>= 1;  
        }  
        return result;  
    }  
}
```


Accepted 57 / 57 testcases passed


 **Solution**

 **BimalTyagi** submitted at Feb 10, 2025 15:16

 **Runtime** 

1 ms | Beats **100.00%** 

 [Analyze Complexity](#)

 **Memory**

44.72 MB | Beats **21.62%**

