**1763. Longest Nice Substring**

class Solution {

public:

bool isnice(const unordered\_set<char>& hs){

for(char c:hs){

if(islower(c)&& hs.find(toupper(c))==hs.end()){

return false;

}

if(isupper(c)&& hs.find(tolower(c))==hs.end()){

return false;

}

}

return true;

}

string longestNiceSubstring(string s) {

int n=s.length();

if (n<2) return "";

string ans="";

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

unordered\_set<char> hs;

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

hs.insert(s[j]);

if (isnice(hs)) {

if (j-i+1 > ans.length()) {

ans = s.substr(i, j - i + 1);

}

}

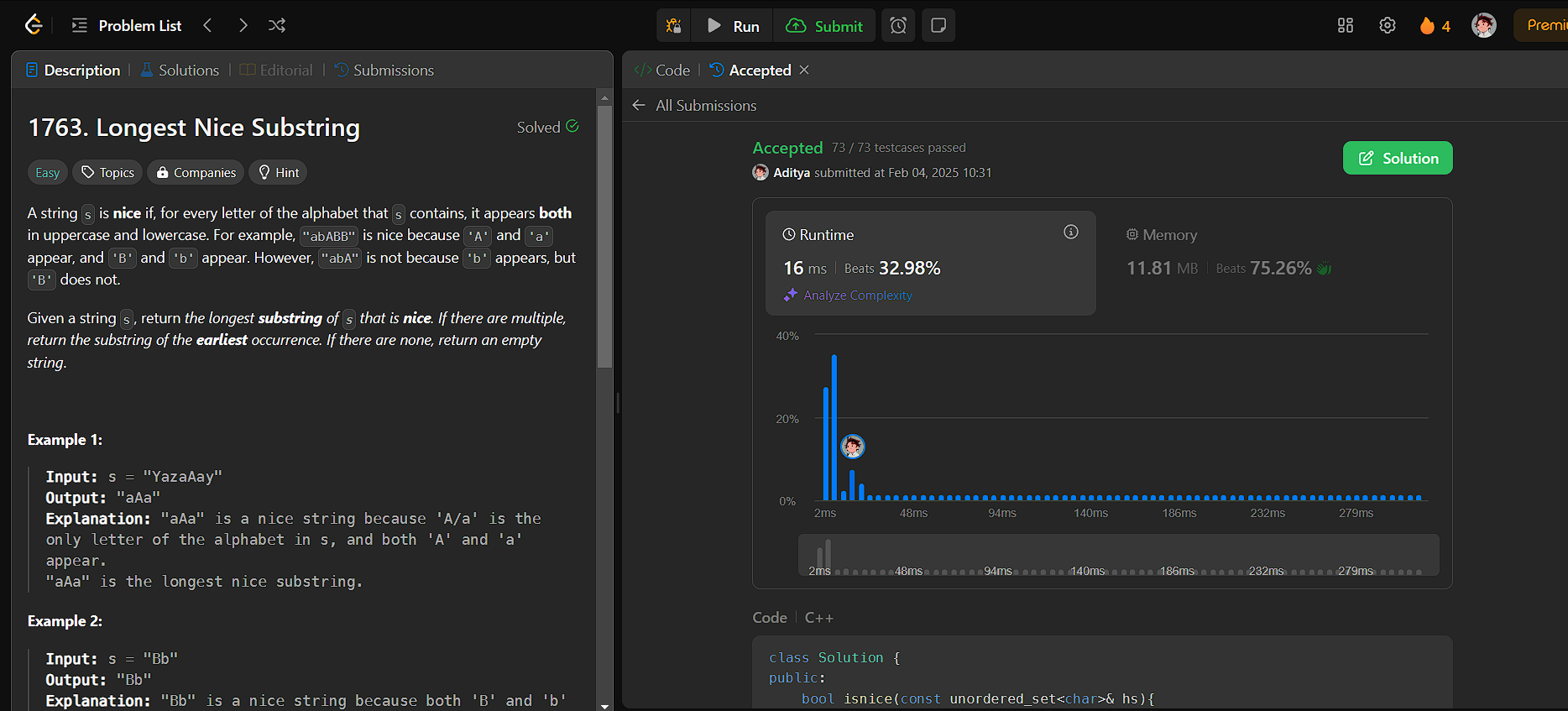
}

}

return ans;

}

};



**190. Reverse Bits**

class Solution {

public:

    uint32\_t reverseBits(uint32\_t n) {

        uint32\_t ans=0;

        for(int i=0; i<32; i++){

            ans<<=1;

            ans |=(n&1);

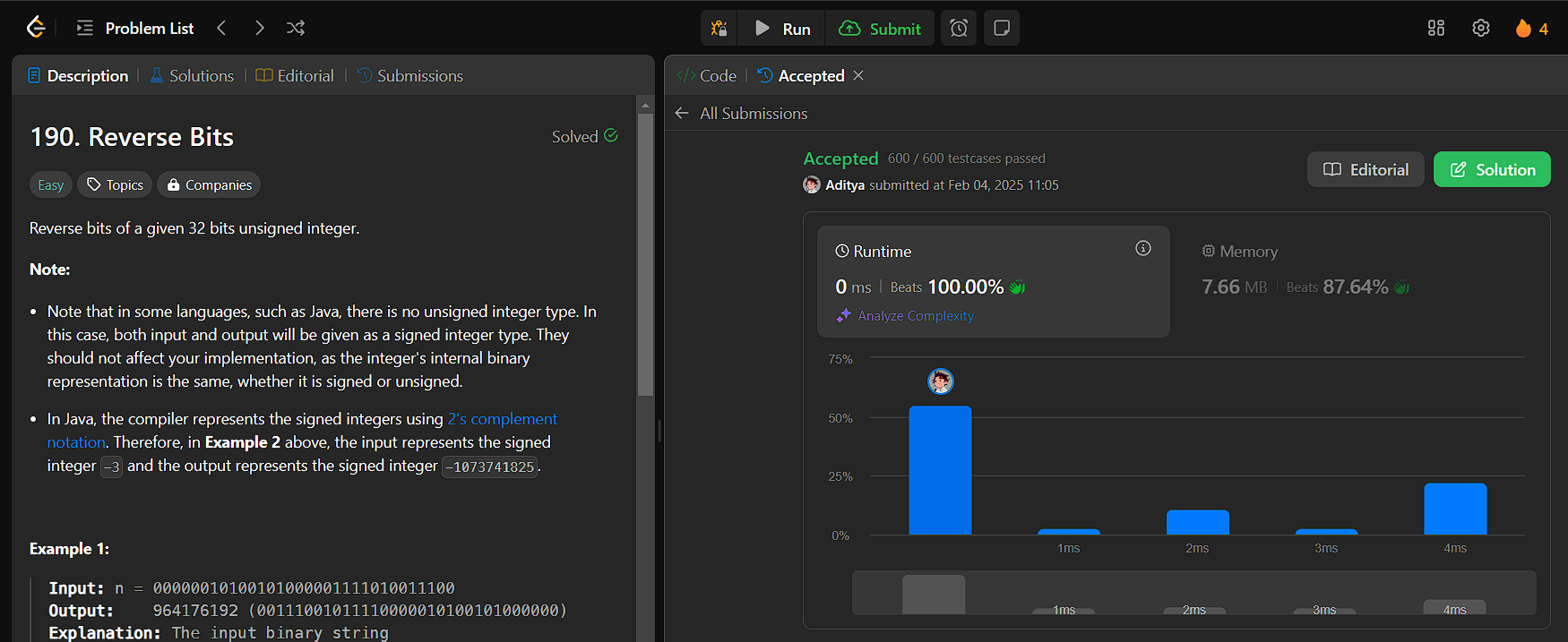
            n>>=1;

        }

        return ans;

    }

};



**191. Number of 1 Bits**

class Solution {

public:

    int hammingWeight(int n) {

        int ans=0;

        for(int i=31; i>=0; i--){

            if(((n>>i)&1)==1){

                ans++;

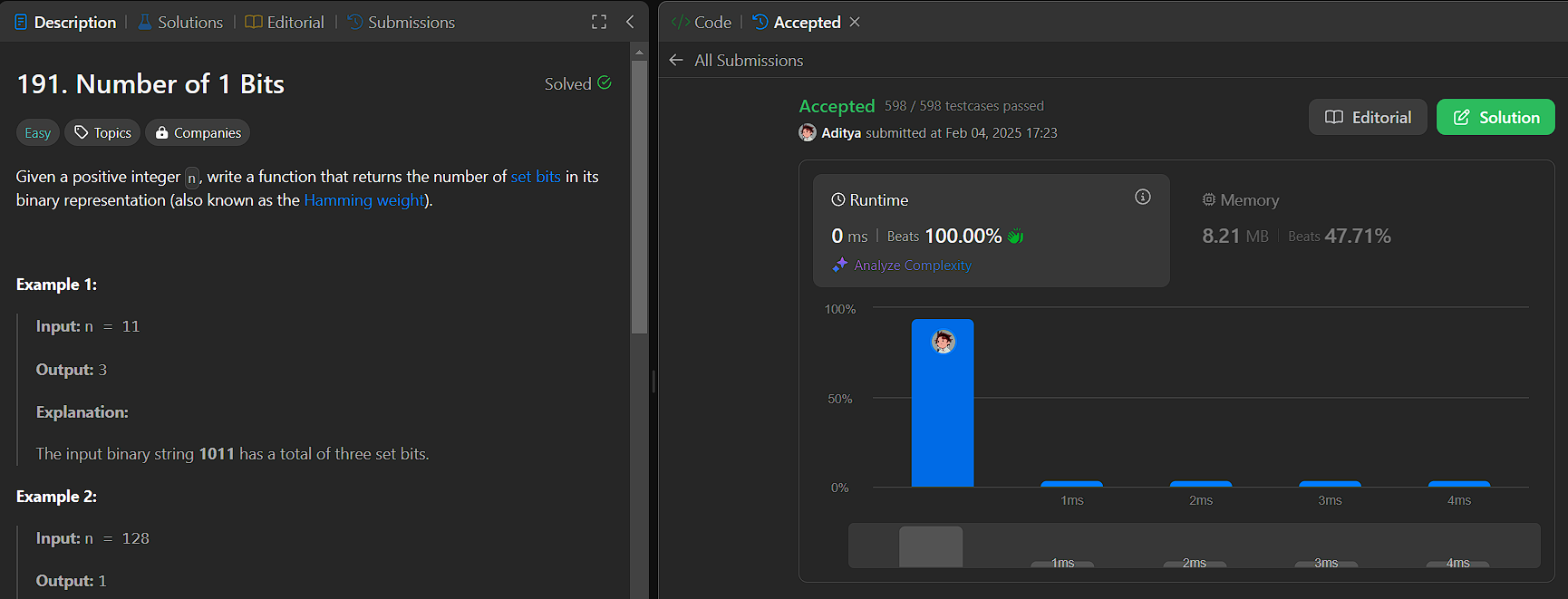
            }

        }

        return ans;

    }

};



**53. Maximum Subarray**

class Solution {

public:

    int maxSubArray(vector<int>& nums) {

        int ans=INT\_MIN;

        int sum=0;

        for(int i=0; i<nums.size(); i++){

            sum+=nums[i];

            ans=max(ans,sum);

            if(sum<0){

                sum=0;

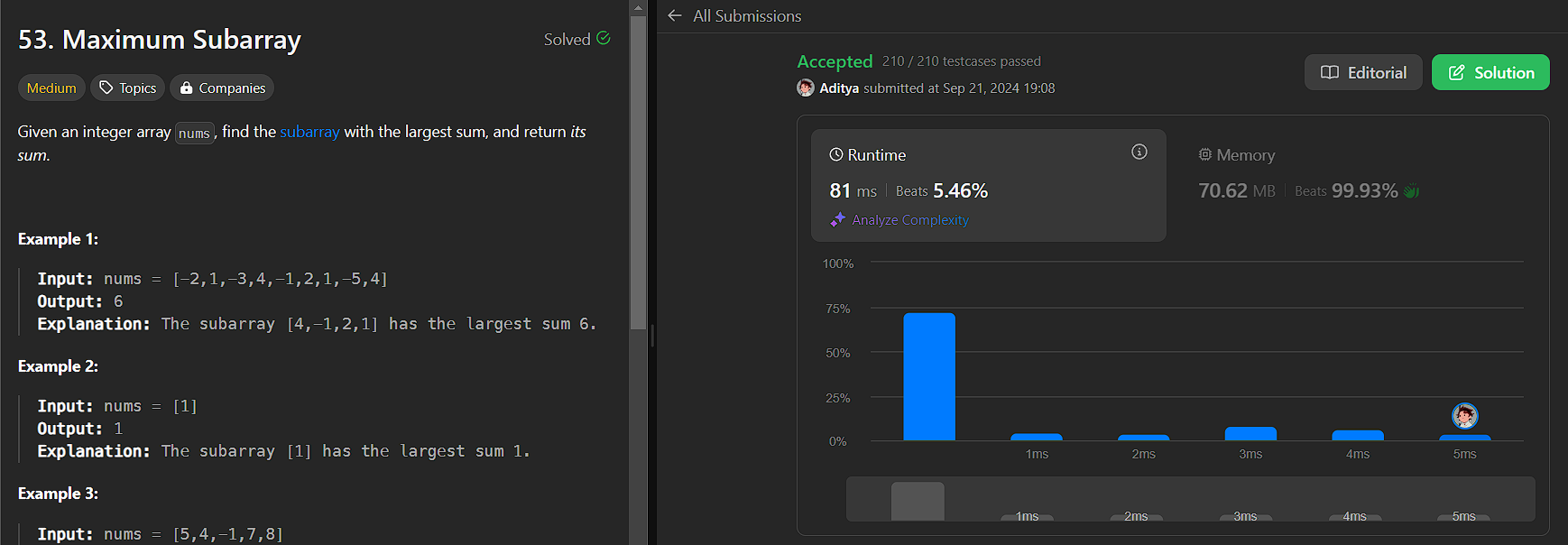
            }

        }

        return ans;

    }

};



**240. Search a 2D Matrix II**

class Solution {

public:

    bool searchMatrix(vector<vector<int>>& matrix, int target) {

        int cols=matrix[0].size()-1;

        int n=matrix.size()-1;

        int rows=0;

        while(rows<=n && cols>=0){

            int toCompare=matrix[rows][cols];

            if(toCompare>target){

                cols--;

            }else if(toCompare<target){

                rows++;

            }else{

                return true;

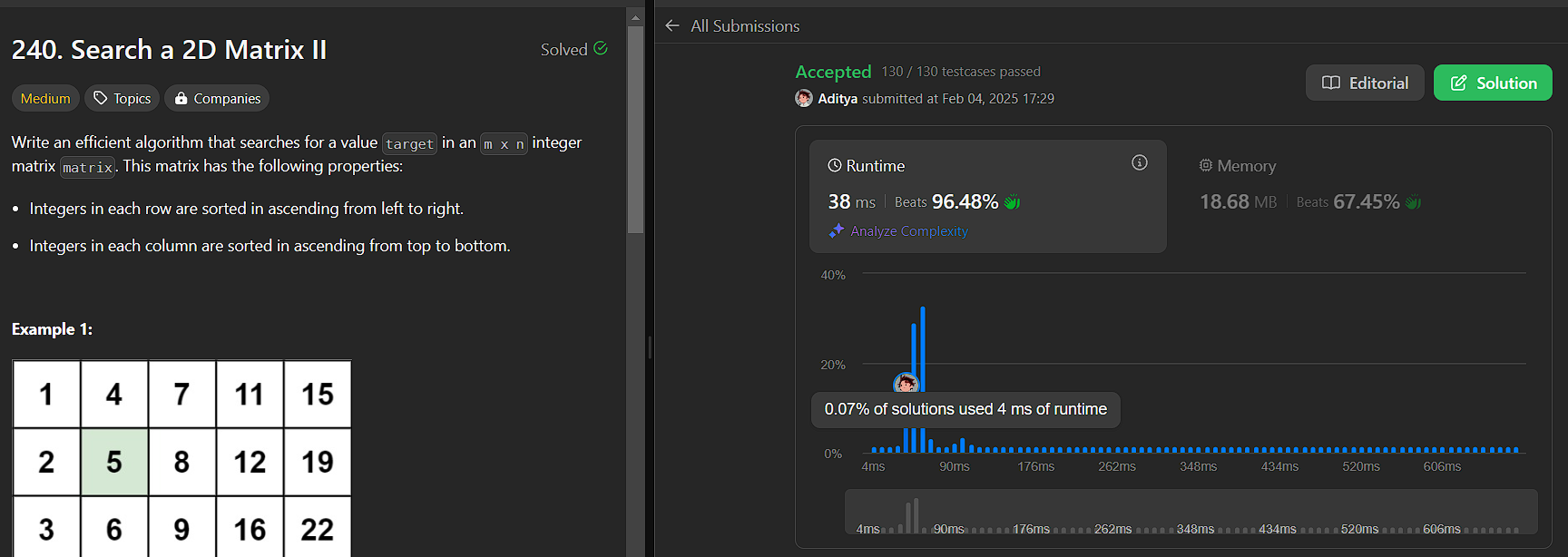
            }

        }

        return false;

    }

};



**372. Super Pow**

class Solution {

    const int base = 1337;

    int powmod(int a, int k)

    {

        a %= base;

        int result = 1;

        for (int i = 0; i < k; ++i)

            result = (result \* a) % base;

        return result;

    }

    public:

    int superPow(int a, vector<int>& b) {

        if (b.empty()) return 1;

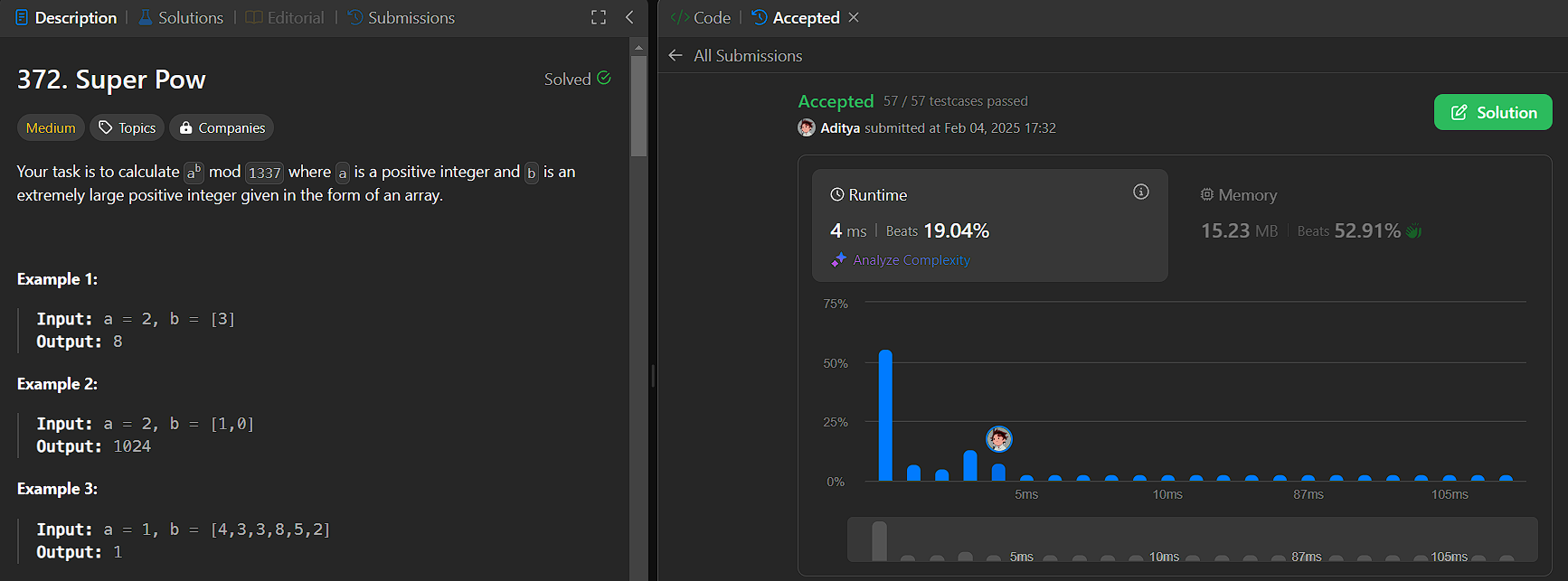
        int last\_digit = b.back();

        b.pop\_back();

        return powmod(superPow(a, b), 10) \* powmod(a, last\_digit) % base;

    }

};



**932. Beautiful Array**

class Solution {

public:

    vector<int> fun(vector<int> v){

        if(v.size()<3)

            return v;

        vector<int> t1;

        vector<int> t2;

        for(int i=0;i<v.size();i++){

            if(i%2==0)

                t1.push\_back(v[i]);

            else

                t2.push\_back(v[i]);

        }

        t1=fun(t1);

        t2=fun(t2);

        vector<int> t3;

        for(int i=0;i<t1.size();i++)

            t3.push\_back(t1[i]);

        for(int i=0;i<t2.size();i++)

            t3.push\_back(t2[i]);

        return t3;

    }

    vector<int> beautifulArray(int n) {

        vector<int> t(n);

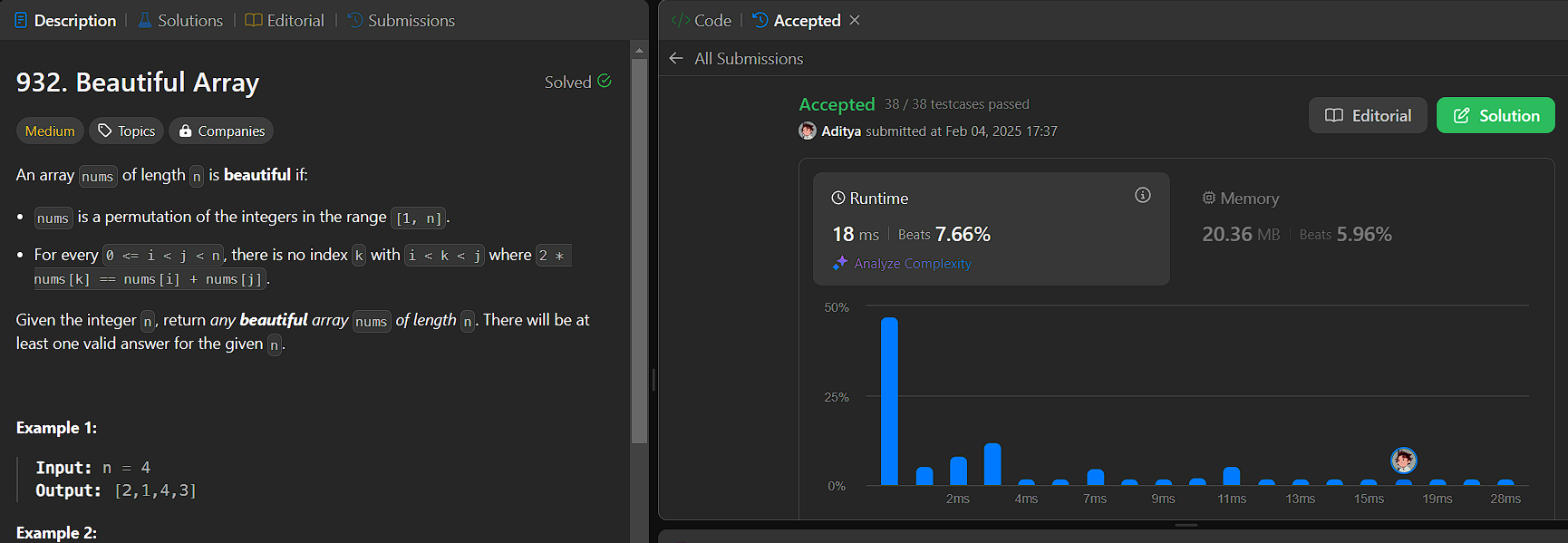
        for(int i=1;i<=n;i++)

            t[i-1]=i;

        return fun(t);

    }

};



**372. Super Pow**

class Solution {