ASSIGNMENT 4

Student Name: Mehak Chauhan **UID: 22BCS12675**
Branch: CSE **Section: 22BCS_IOT_605 B**
Semester: 6th **DOP:05-03-2025**
Subject: Advanced Programming Lab-II **Subject Code: 22CSP-351**

Question 1

1763. Longest Nice Substring

Solved

Easy | Topics | Companies | Hint

A string `s` is **nice** if, for every letter of the alphabet that `s` contains, it appears **both** in uppercase and lowercase. For example, `"abABB"` is nice because `'A'` and `'a'` appear, and `'B'` and `'b'` appear. However, `"abA"` is not because `'b'` appears, but `'B'` does not.

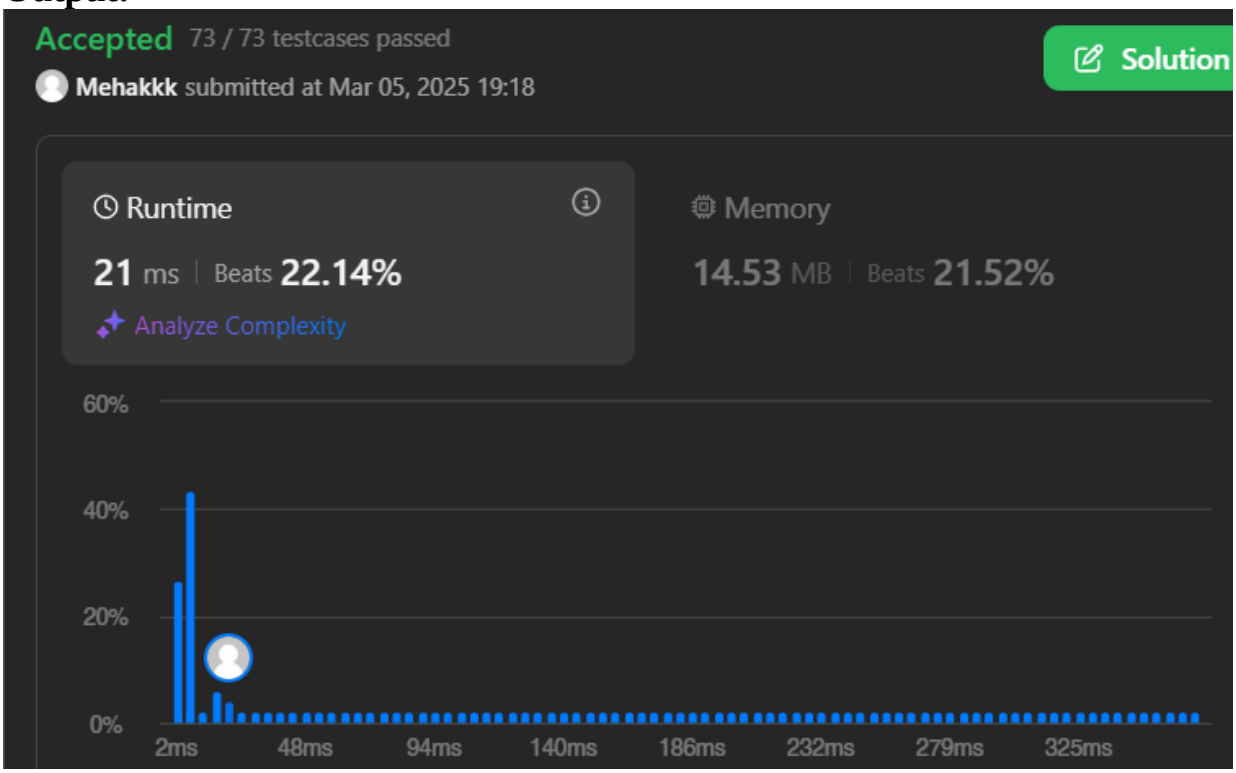
Given a string `s`, return *the longest* ***substring*** *of* `s` *that is* ***nice****. If there are multiple, return the substring of the* ***earliest*** *occurrence. If there are none, return an empty string.*

Code:

```
class Solution {
public:

   bool isNice(string h){

        for(int i=0; i<h.length(); i++){

            char c = h[i];
            if(c >= 65 && c <=90){
                c = c + 32;
            }
            else{
                c = c - 32;
            }
            if(h.find(c) == string ::npos){
                return false;
            }
        }
        return true;
    }


    string longestNiceSubstring(string s) {

        string res= "";
        int n = s.size();
```

```
        for(int i=0; i<n; i++)
        {
            string t="";
            t += s[i];

            for(int j = i+1; j<n; j++)
            {
                t += s[j];
                if(isNice(t)  && t.length()> res.length())
                    res = t;
            }
        }
        return res;
    }
};
```

Output:



Question 2

190. Reverse Bits

Solved

Easy | Topics | Companies

Reverse bits of a given 32 bits unsigned integer.

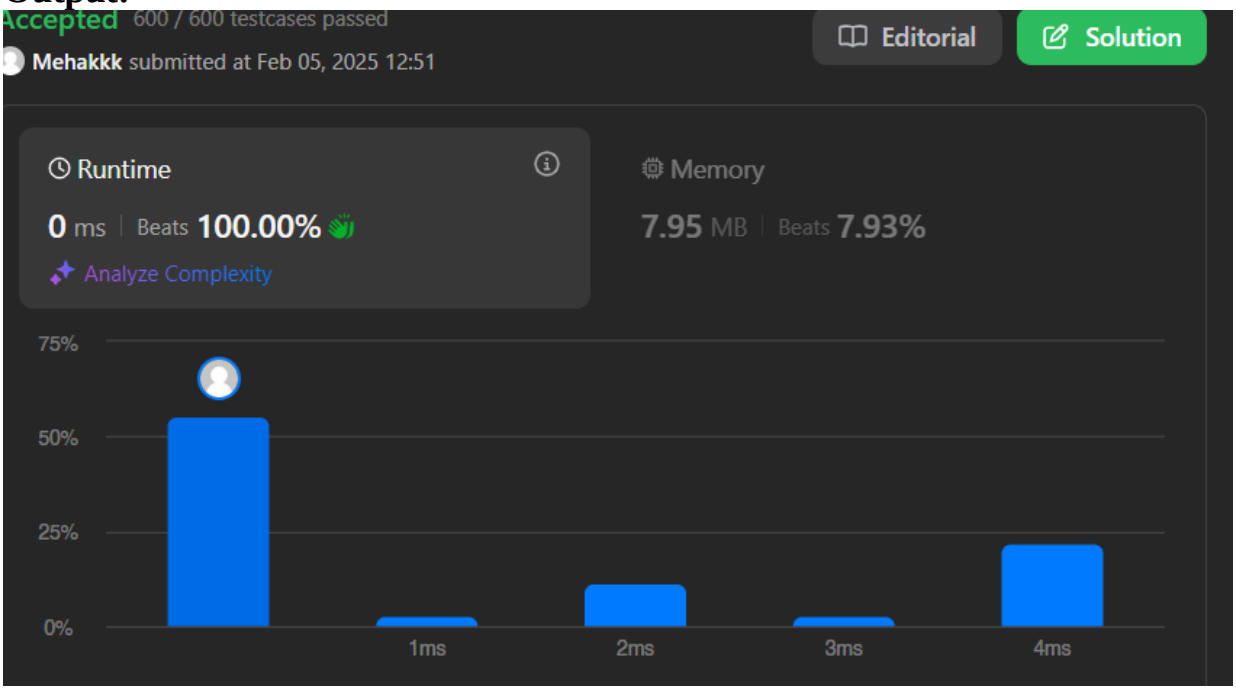
Note:

- Note that in some languages, such as Java, there is no unsigned integer type. In this case, both input and output will be given as a signed integer type. They should not affect your implementation, as the integer's internal binary representation is the same, whether it is signed or unsigned.

Code:

```
class Solution {
public:
    uint32_t reverseBits(uint32_t n) {
        uint32_t result = 0;
        for (int i = 0; i < 32; i++) {
            int bit = n & 1;
            result = (result << 1) | bit;
            n = n >> 1;
        }
        return result;
    }
};
```

Output:



Question 3

191. Number of 1 Bits

Solved

Easy | Topics | Companies

Given a positive integer `n`, write a function that returns the number of set bits in its binary representation (also known as the Hamming weight).

Example 1:

> **Input:** n = 11
>
> **Output:** 3

Code:

```
class Solution {
public:
    int hammingWeight(uint32_t n) {
        int res = 0;
        for (int i = 0; i < 32; i++) {
            if (((n >> i) & 1) ==1) {
                res += 1;
            }
        }
        return res;
    }
};
```

Output:

Accepted 598 / 598 testcases passed

Mehakkk submitted at Mar 05, 2025 11:05

Editorial | Solution

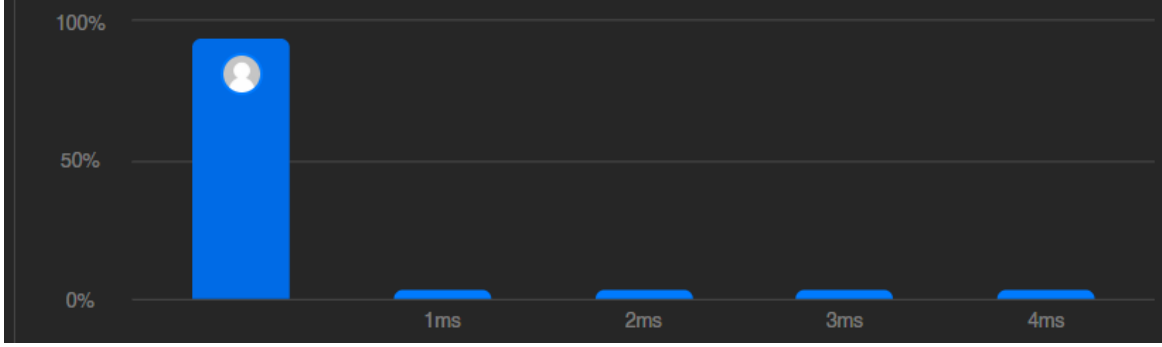
Runtime

0 ms | Beats 100.00%

Analyze Complexity

Memory

8.19 MB | Beats 80.49%



Question 4

53. Maximum Subarray

Solved

Medium | Topics | Companies

Given an integer array `nums`, find the subarray with the largest sum, and return *its sum*.

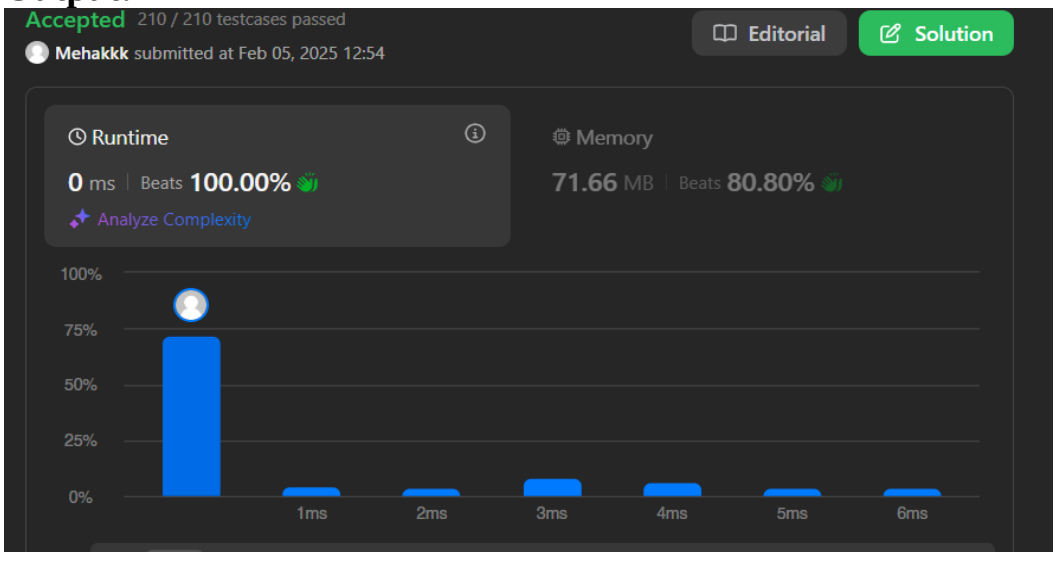
Example 1:

```
Input: nums = [-2,1,-3,4,-1,2,1,-5,4]
Output: 6
Explanation: The subarray [4,-1,2,1] has the largest sum 6.
```

Code:

```cpp
class Solution {
public:
    int maxSubArray(vector<int>& nums) {
        int result = nums[0];
        int curr_sum = nums[0];


        for(int i=1;i<nums.size();i++){
            if(nums.size()==1){
                return nums[i];
            }
            curr_sum= max(nums[i],curr_sum+nums[i]);
            result= max(result,curr_sum);
        }

        return result;
    }
};
```

Output:



Question 5

240. Search a 2D Matrix II

Solved

Medium | Topics | Companies

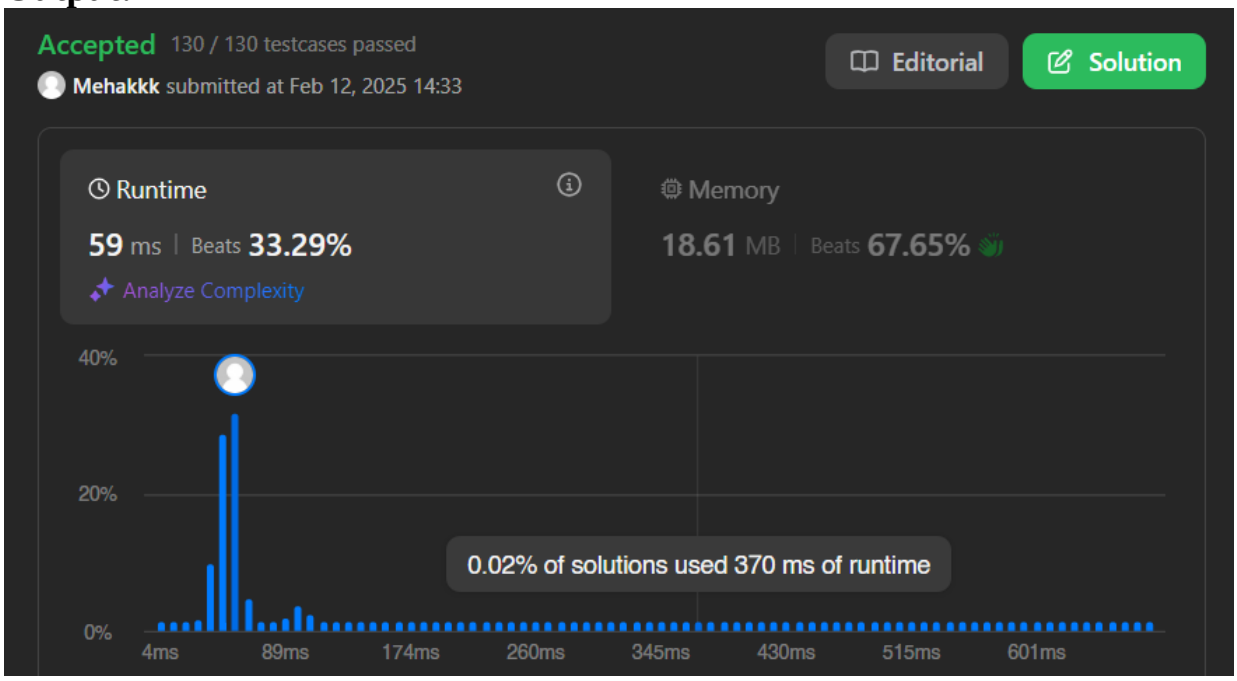
Write an efficient algorithm that searches for a value `target` in an `m x n` integer matrix `matrix`. This matrix has the following properties:

- Integers in each row are sorted in ascending from left to right.
- Integers in each column are sorted in ascending from top to bottom.

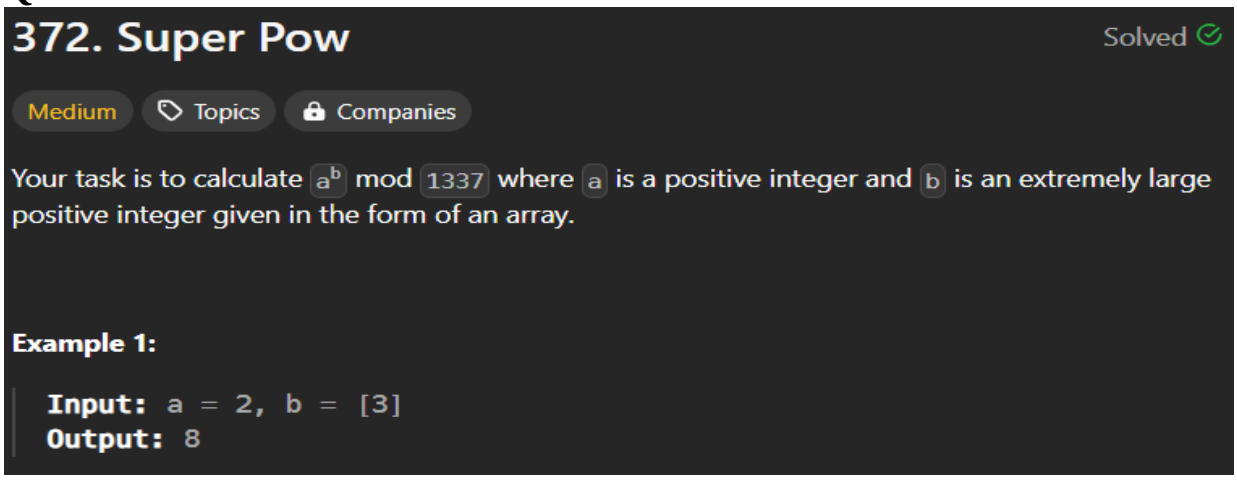
Code:

```
class Solution {
public:
    bool searchMatrix(vector<vector<int>>& matrix, int target) {
        int row_size= matrix.size();
        int col_size = matrix[0].size();

        int row=0;
        int col= col_size-1;

        while( row<row_size && col>=0){
            if(matrix[row][col]==target){
                return true;
            }
            else if(matrix[row][col]<target){
                row++;
            }
            else{
                col--;
            }
        }
        return false;
    }
};
```

Output:



Question 6



Code

```
int n=1337;
int phi=1140;
class Solution {
public:
    int Chinese_Remainder(int a,  int x, vector<int>& b){
        if (x==n) return 0;
        int p=n/x;
        int M;//modInverse i.e. x*M==1%p
        if (x==7) M=82;//can be computed by extended euclidean algorithm
        else M=4;

        int s=b.size();
        int exp=0;
        for(int i=0; i<s; i++)
            exp=(b[i]+10*exp)%(p-1);
        bitset<12> e(exp);
```

```
        int y=1;
        a%=n;
        for(int i=11; i>=0; i--){
            y=y*y%n;
            if (e[i]==1) y=y*a%n;
        }
        int ans=y*M*x%n;
        while( ans<0)
            ans+=n;
    //    cout<<ans<<endl;
        return ans;//Chinese Remainder Theorem
    }
    int superPow(int a, vector<int>& b) {
        int g=gcd(a, n);
    //    cout<<"gcd="<<g<<endl;
        if (g!=1) return Chinese_Remainder(a, g, b);
        int s=b.size();
        int exp=0;
        for(int i=0; i<s; i++)
            exp=(b[i]+10*exp)%phi;
        bitset<12> e(exp);
        int y=1;
        a%=n;
        for(int&& i=11; i>=0; i--){
            y=y*y%n;
            if (e[i]==1) y=y*a%n;
        }
        return y;
    }
};
```

Output