

1763. [Longest Nice Substring](#)

Code:

```
class Solution {
public:
    string longestNiceSubstring(string s) {

        string output = "";
        int count = 0;
        for(int i = 0; i < s.length(); i++){
            int smallMask = 0;
            int largeMask = 0;
            char ch = s[i];
            int chint = 0;
            if(ch >= 65 && ch <= 90){
                chint = ch - 'A';
                largeMask = 1 << chint;
            }
            else{
                chint = ch - 'a';
                smallMask = 1 << chint;
            }
            for(int j = i + 1; j < s.length(); j++){
                ch = s[j];
                if(ch >= 65 && ch <= 90){
                    chint = ch - 'A';
                    largeMask |= 1 << chint;
                }
                else{
                    chint = ch - 'a';
                    smallMask |= 1 << chint;
                }
            }
        }
    }
};
```

```

        //checking for nice
        if((smallMask^largeMask) == 0){
            if(count<j-i+1){
                count = j-i+1;
                string temp(s.begin()+i,s.begin()+j+1);
                output = temp;
            }
        }
    }
}
return output;

}

};

```

Output:



190. [Reverse Bits](#)

Code:

```
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;
    }
};
```

Output:

☒ Testcase | [Test Result](#)

Accepted Runtime: 3 ms

• Case 1

• Case 2

Input

n =
00000010100101000001111010011100

Output

964176192 (00111001011110000010100101000000)

191. [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:

☒ Testcase | [Test Result](#)

Accepted Runtime: 0 ms

- Case 1
- Case 2
- Case 3

Input

n =
11

Output

3

53. [Maximum Subarray](#)

Code:

```
class Solution {
public:
    int maxSubArray(vector<int>& nums) {
        int maxSum = INT_MIN;
        int currentSum = 0;
        for (int i = 0; i < nums.size(); i++) {
            currentSum += nums[i];
            if (currentSum > maxSum) {
                maxSum = currentSum;
            }

            if (currentSum < 0) {
                currentSum = 0;
            }
        }
        return maxSum;
    }
};
```

Output:

☒ Testcase ☒ Test Result

Accepted Runtime: 0 ms

• Case 1

• Case 2

• Case 3

Input

nums =
[-2, 1, -3, 4, -1, 2, 1, -5, 4]

Output

6

240. [Search a 2D Matrix II](#)

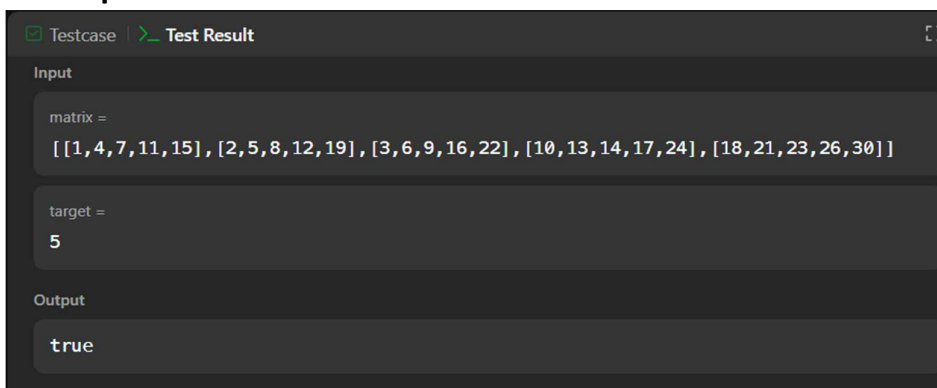
Code:

```
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;
    }
};
```

Output:



The screenshot shows a test result window with a dark theme. At the top, there are tabs for 'Testcase' and 'Test Result'. Below the tabs, the 'Input' section contains two fields: 'matrix =' with the value `[[1,4,7,11,15],[2,5,8,12,19],[3,6,9,16,22],[10,13,14,17,24],[18,21,23,26,30]]` and 'target =' with the value `5`. The 'Output' section shows the result `true`.

372.[Super Pow](#)

Code:

```
class Solution {
private:
    int solve(int base, int power, int mod) {
        int ans = 1;
        while (power > 0) {
            if (power & 1) {
                ans = (ans * base) % mod;
            }
            base = (base * base) % mod;
            power >>= 1;
        }
        return ans;
    }

public:
    int superPow(int a, vector<int>& b) {
        a%=1337;
        int n = b.size();
        int m = 1140;
        int expi = 0;
        for(int i : b){
            expi = (expi*10+i)%m;
        }
        if (expi == 0) {
            expi = m;
        }
        return solve(a,expi,1337);
    }
};
```

Output:

☒ Testcase | > Test Result

Case 1 Case 2 Case 3

Input

a =
2

b =
[3]

Output

8