

## AP Assignment-4

Name: Shivangi Gupta

UID: 22BCS15008

Class: IoT\_601-A

1763. <https://leetcode.com/problems/longest-nice-substring/description/>

```
class Solution {
public:
    string longestNiceSubstring(string s) {
        int n = s.size();
        for (int len = n; len > 0; len--) {
            for (int i = 0; i + len <= n; i++) {
                unordered_set<char> lower, upper;
                bool valid = true;
                for (int j = i; j < i + len; j++) {
                    if (islower(s[j])) lower.insert(s[j]);
                    if (isupper(s[j])) upper.insert(s[j]);
                }
                for (char c : lower) {
                    if (upper.find(toupper(c)) == upper.end()) {
                        valid = false;
                        break;
                    }
                }
                for (char c : upper) {
                    if (lower.find(tolower(c)) == lower.end()) {
                        valid = false;
                        break;
                    }
                }
                if (valid) return s.substr(i, len);
            }
        }
    }
}
```

```

return "";
}
};

```

Code:

Accepted 73 / 73 testcases passed  
Shivangi Gupta submitted at Feb 23, 2025 18:55

**Runtime** 314 ms | Beats 12.72%  
**Memory** 100.90 MB | Beats 13.81%

Code: C++

```

class Solution {
public:
    string longestNiceSubstring(string s) {
        int n = s.size();
        for (int len = n; len > 0; len--) {
            unordered_set<char> lower, upper;
            bool valid = true;
            for (int j = 0; j < len; j++) {
                if (islower(s[j])) lower.insert(s[j]);
                if (isupper(s[j])) upper.insert(s[j]);
            }
            for (char c : lower) {
                if (upper.find(tolower(c)) == upper.end()) {
                    valid = false;
                    break;
                }
            }
            for (char c : upper) {
                if (lower.find(toupper(c)) == lower.end()) {
                    valid = false;
                    break;
                }
            }
            if (valid) return s.substr(0, len);
        }
        return "";
    }
};

```

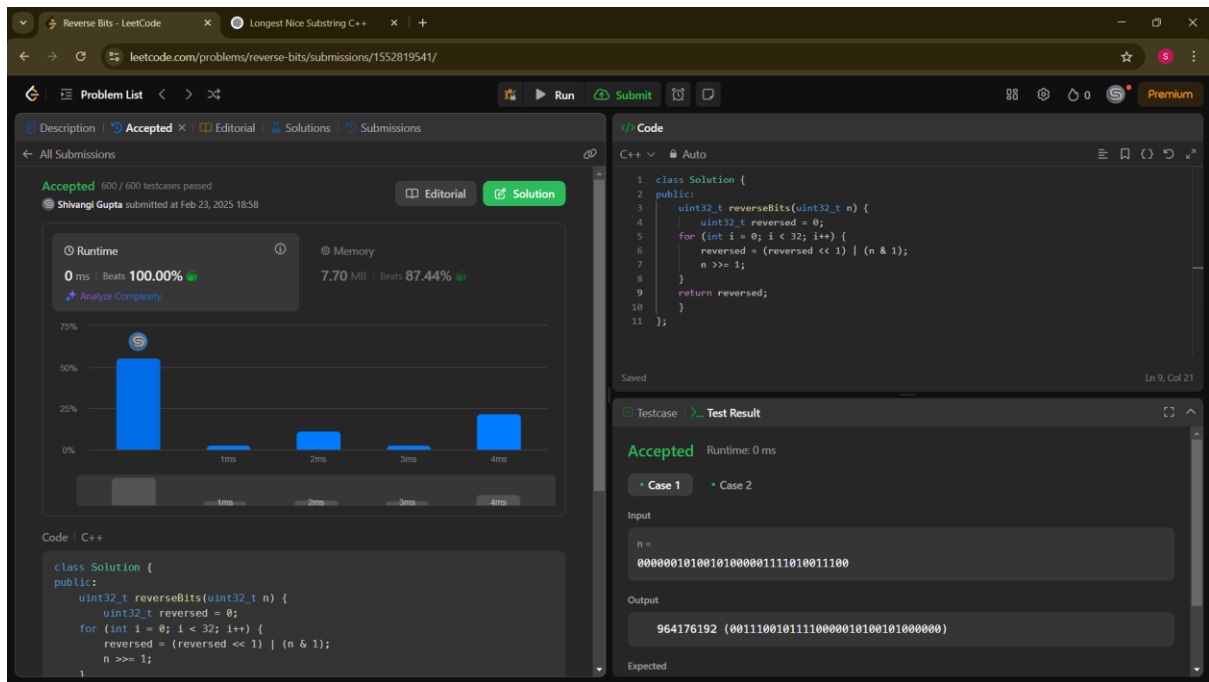
190. <https://leetcode.com/problems/reverse-bits/>

Code:

```

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

```



191. Number of 1-bits <https://leetcode.com/problems/number-of-1-bits/description/>

Code:

```
class Solution {
```

```
public:
```

```
    int hammingWeight(int n) {
```

```
        int count = 0;
```

```
        while (n) {
```

```
            count += n & 1;
```

```
            n >>= 1;
```

```
        }
```

```
        return count;
```

```
    }
```

```
};
```

The screenshot shows a LeetCode submission for the problem "Number of 1 Bits". The submission is accepted, with a runtime of 0 ms and memory usage of 8.34 MB. The code is in C++ and implements a function to count the number of 1 bits in an integer n.

```

class Solution {
public:
    int hammingWeight(int n) {
        int count = 0;
        while (n) {
            count += n & 1;
            n >>= 1;
        }
        return count;
    }
};

```

53. <https://leetcode.com/problems/maximum-subarray/description/>

Code:

```
class Solution {
```

```
public:
```

```
    int maxSubArray(vector<int>& nums) {
```

```
        int max_sum = INT_MIN;
```

```
        int curr_sum = 0;
```

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

```
            curr_sum += nums[i];
```

```
            if(curr_sum > max_sum){
```

```
                max_sum = curr_sum;
```

```
            }
```

```
            if(curr_sum < 0){
```

```
                curr_sum = 0;
```

```
            }
```

```
        }
```

```
        return max_sum;
```

```
    }
```

```
};
```

Maximum Subarray - LeetCode

Accepted 210 / 210 testcases passed

Shivangi Gupta submitted at Aug 10, 2024 07:59

Runtime: 74 ms | Beats: 5.21% | Memory: 70.62 MB | Beats: 99.93%

```

1 //Kadane's Algo
2 class Solution {
3 public:
4     int maxSubArray(vector<int>& nums) {
5         int max_sum = INT_MIN;
6         int curr_sum = 0;
7         for(int i=0; i<nums.size(); i++){
8             curr_sum += nums[i];
9             if(curr_sum > max_sum){
10                 max_sum = curr_sum;
11             }
12             if(curr_sum < 0){
13                 curr_sum = 0;
14             }
15         }
16         return max_sum;
17     }
18 };

```

Status	Language	Runtime	Memory	Notes
Accepted	C++	74 ms	70.6 MB	

240. Search a 2D Matrix-II <https://leetcode.com/problems/search-a-2d-matrix-ii/description/>

Code:

```
class Solution {
```

```
public:
```

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

```
        int rows=matrix.size();
```

```
        int cols=matrix[0].size();
```

```
        if (rows == 0) return false;
```

```
        int s=0,e=rows*cols-1;
```

```
        while(s<=e){
```

```
            int mid=s+(e-s)/2;
```

```
            int midValue=matrix[mid/cols][mid%cols];
```

```
            if(target==midValue) return true;
```

```
            else if(midValue<target) s=mid+1;
```

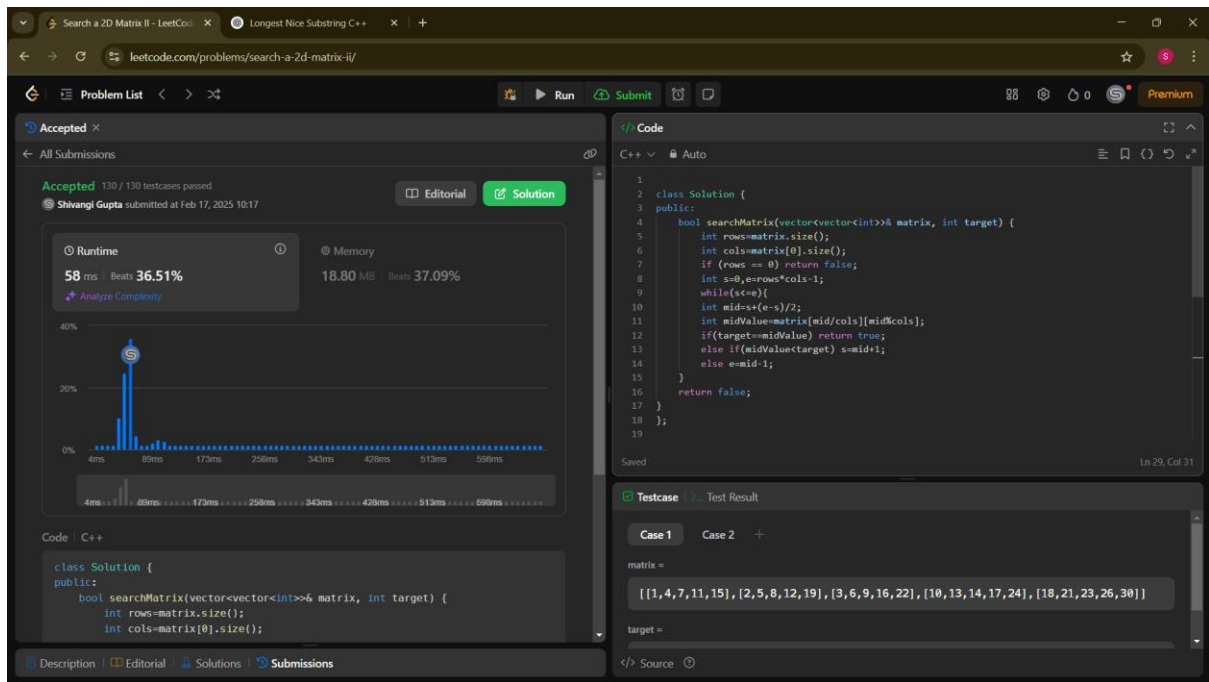
```
            else e=mid-1;
```

```
        }
```

```
        return false;
```

```
    }
```

```
};
```



372. Super Pow <https://leetcode.com/problems/super-pow/description/>

Code:

```
class Solution {
```

```
public:
```

```
int mod = 1337;
```

```
int quickPow(int a, long long b) {
```

```
    int result = 1;
```

```
    a = a % mod;
```

```
    while (b > 0) {
```

```
        if (b % 2 == 1) {
```

```
            result = (result * a) % mod;
```

```
        }
```

```
        a = (a * a) % mod;
```

```
        b /= 2;
```

```
    }
```

```
    return result;
```

```
}
```

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

```

    long long exponent = 0;

    for (int i = 0; i < b.size(); i++) {
        exponent = (exponent * 10 + b[i]) % (mod - 1);
    }

    return quickPow(a, exponent);
}

};

```

**372. Super Pow** Attempted

Medium Topics Companies

Your task is to calculate  $a^b \bmod 1337$  where  $a$  is a positive integer and  $b$  is an extremely large positive integer given in the form of an array.

**Example 1:**  
Input:  $a = 2, b = [3]$   
Output: 8

**Example 2:**  
Input:  $a = 2, b = [1,0]$   
Output: 1024

**Example 3:**  
Input:  $a = 1, b = [4,3,3,8,5,2]$   
Output: 1

**Constraints:**

- $1 \leq a \leq 2^{31} - 1$
- $1 \leq b.length \leq 2000$
- $0 \leq b[i] \leq 9$

**Code:**

```

7   a = a % mod;
8   while (b > 0) {
9       if (b % 2 == 1) {
10          result = (result * a) % mod;
11      }
12      a = (a * a) % mod;
13      b /= 2;
14  }
15  return result;
16  }
17  int superPow(int a, vector<int>& b) {
18      long long exponent = 0;
19
20      // Step 1: Compute b % (mod - 1) efficiently
21      for (int i = 0; i < b.size(); i++) {
22          exponent = (exponent * 10 + b[i]) % (mod - 1); // (mod-1) is 1336
23      }
24
25      // Step 2: Compute the result using modular exponentiation
26      return quickPow(a, exponent);

```

**Testcase:** Test Result

Case 1	Case 2	Case 3	Case 4
a =			78267
b =			

Source Reset Testcases