Assignment 4

Student Name: Nishant Kumar
UID: 22BCS15009
Branch: CSE
Section/Group: 22BCS_FL_IOT-602 A
Semester: 6th
Date of Performance: 21/02/2025
Subject Name: Advanced Programming Lab - 2
Subject Code: 22CSP-351

Problem 1763. Longest Nice Substring

- **Implementation/Code:**

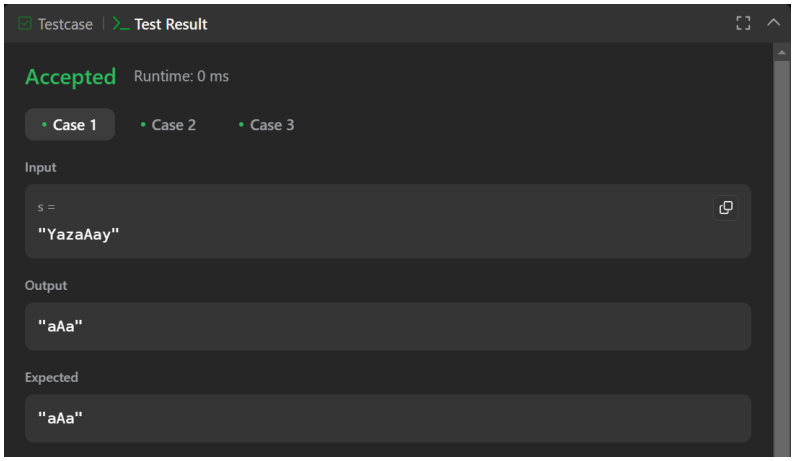
```
class Solution {
public:
    string longestNiceSubstring(string s) {
        if (s.length() < 2) return "";

        unordered_set<char> charSet(s.begin(), s.end());

        for (int i = 0; i < s.length(); i++) {
            if (charSet.count(tolower(s[i])) && charSet.count(toupper(s[i]))) {
                continue;
            }
            string left = longestNiceSubstring(s.substr(0, i));
            string right = longestNiceSubstring(s.substr(i + 1));

            return left.length() >= right.length() ? left : right;
        }
        return s;
    }
};
```

- **Output:**

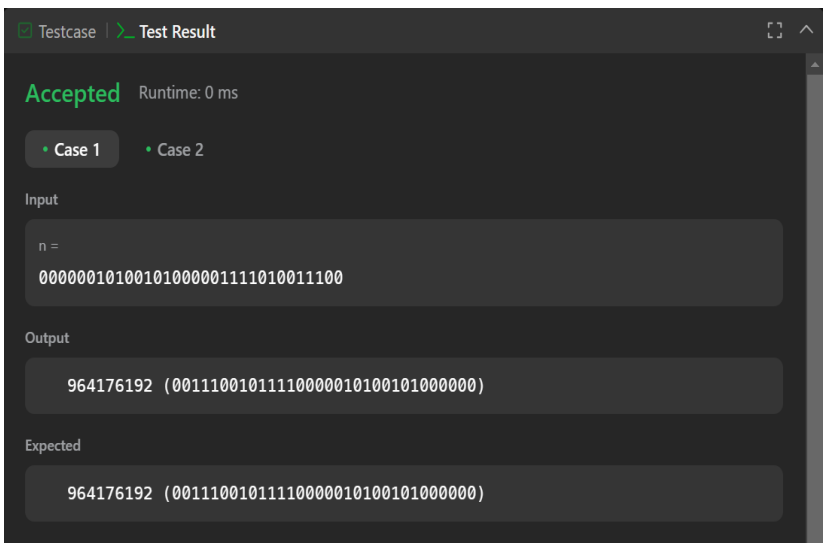


Problem 190. Reverse Bits

- **Implementation/Code:**

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

- **Output:**

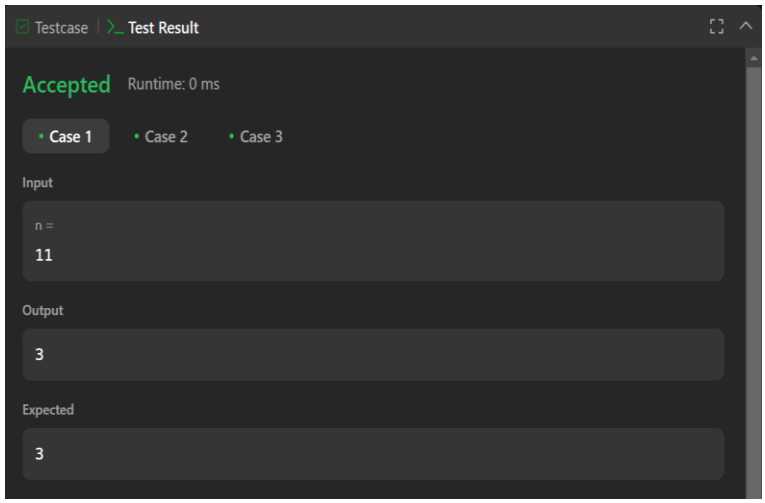


Problem 191. Number of 1 Bits

- **Implementation/Code:**

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

- **Output:**

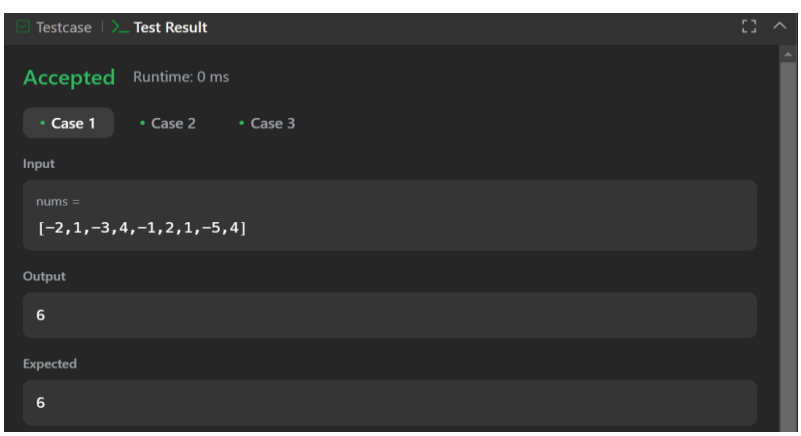


Problem 53. Maximum Subarray

- **Implementation/Code:**

```
class Solution {
public:
    int maxSubArray(vector<int>& nums) {
        int sum = 0;
        int n = nums.size();
        int maximum = nums[0];
        for (int i = 0; i < n; i++) {
            sum += nums[i];
            maximum = max(maximum, sum);
            if (sum < 0) sum = 0;
        }
        return maximum;
    }
};
```

- **Output:**



Problem 240. Search a 2D Matrix II

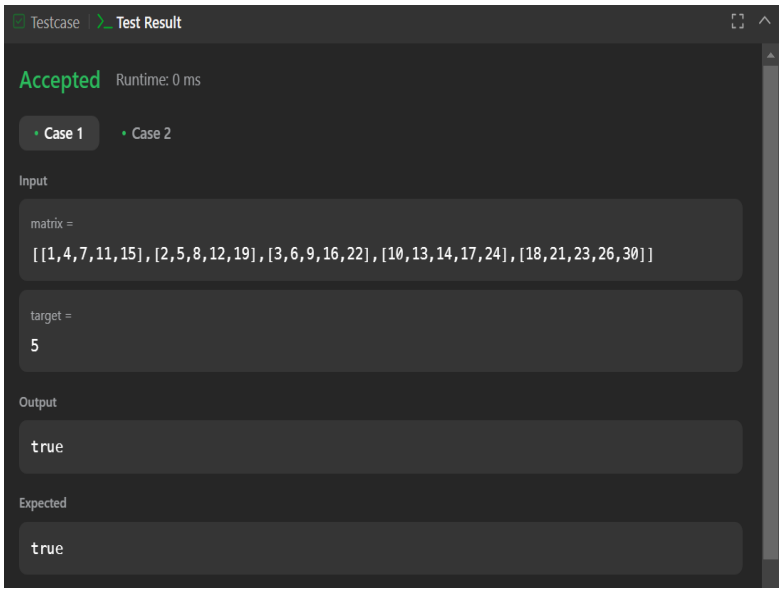
- **Implementation/Code:**

```
class Solution {
public:
    bool searchMatrix(vector<vector<int>>& matrix, int target) {
        int rows = matrix.size(), cols = matrix[0].size();
        int low = 0, high = rows * cols - 1;

        while (low <= high) {
            int mid = low + (high - low) / 2;
            int row = mid / cols, col = mid % cols;
            int midVal = matrix[row][col];

            if (midVal == target) return true;
            else if (midVal < target) low = mid + 1;
            else high = mid - 1;
        }
        return false;
    }
};
```

- **Output:**



Problem 372. Super Pow

- **Implementation/Code:**

```
class Solution {
public:
    const int MOD = 1337;
```

```
    int powerMod(int a, int k) {
        a %= MOD;
        int res = 1;
        while (k > 0) {
            if (k % 2 == 1) {
                res = (res * a) % MOD;
            }
            a = (a * a) % MOD;
            k /= 2;
        }
        return res;
    }

    int superPow(int a, vector<int>& b) {
        int result = 1;
        for (int digit : b) {
            result = powerMod(result, 10) * powerMod(a, digit) % MOD;
        }
        return result;
    }
};
```

- **Output:**