Name : Udit UID : 22BCS16515 Section : 605-B Assignment -4

Ques 1. Longest Nice Subsrtring.

Code:

```
class Solution {
public:
    string longestNiceSubstring(string s) {
        int n = s.length();
        if (n < 2) return "";
        for (int i = 0; i < n; ++i) {
            char ch = s[i];
            if (s.find(tolower(ch)) == string::npos || s.find(toupper(ch)) == string::npos) {
                string left = longestNiceSubstring(s.substr(0, i));
                string right = longestNiceSubstring(s.substr(i + 1));
                return left.length() >= right.length() ? left : right;
            }
        }
        return s;
    }
};
```

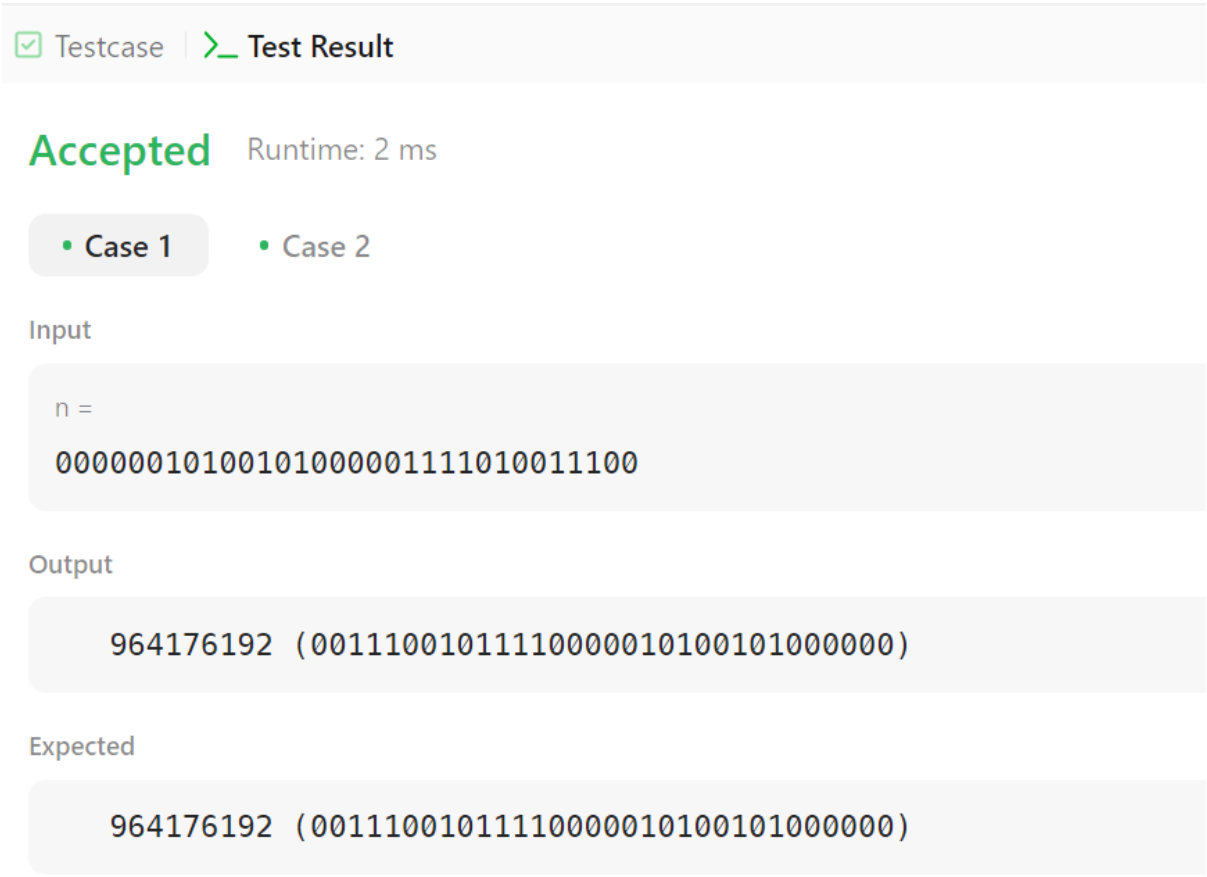
Output:

Ques 2. Reverse Bits.

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:

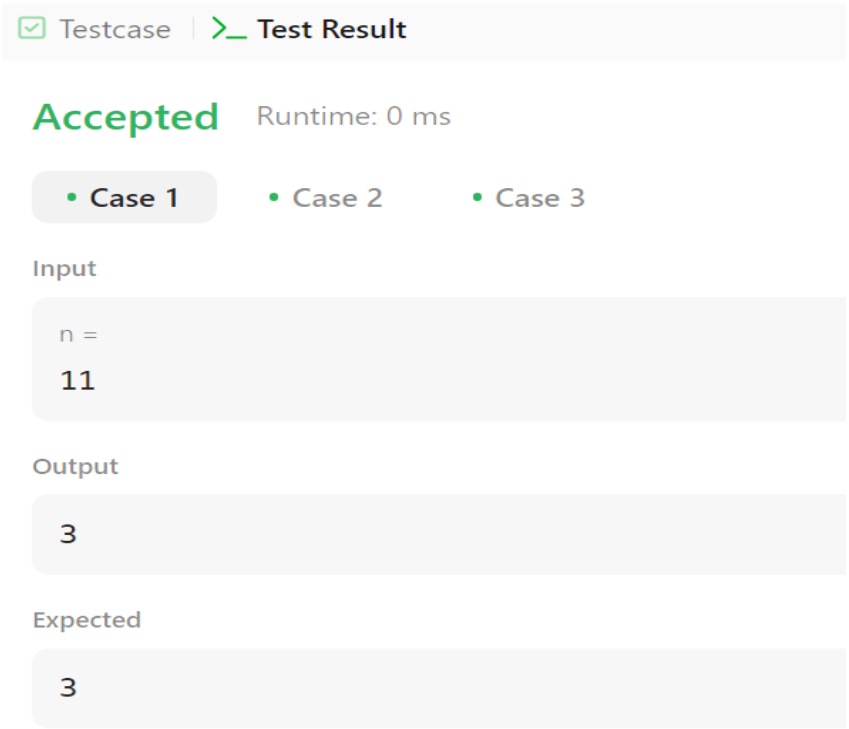


Ques 3. Number of 1 Bits.

Code:

```
class Solution {
public:
    int hammingWeight(int n) {
        int count = 0;
        while (n) {
            count += (n & 1); // Add 1 if the last bit is set
            n >>= 1; // Right shift n to check the next bit
        }
        return count;
    }
};
```

Output:

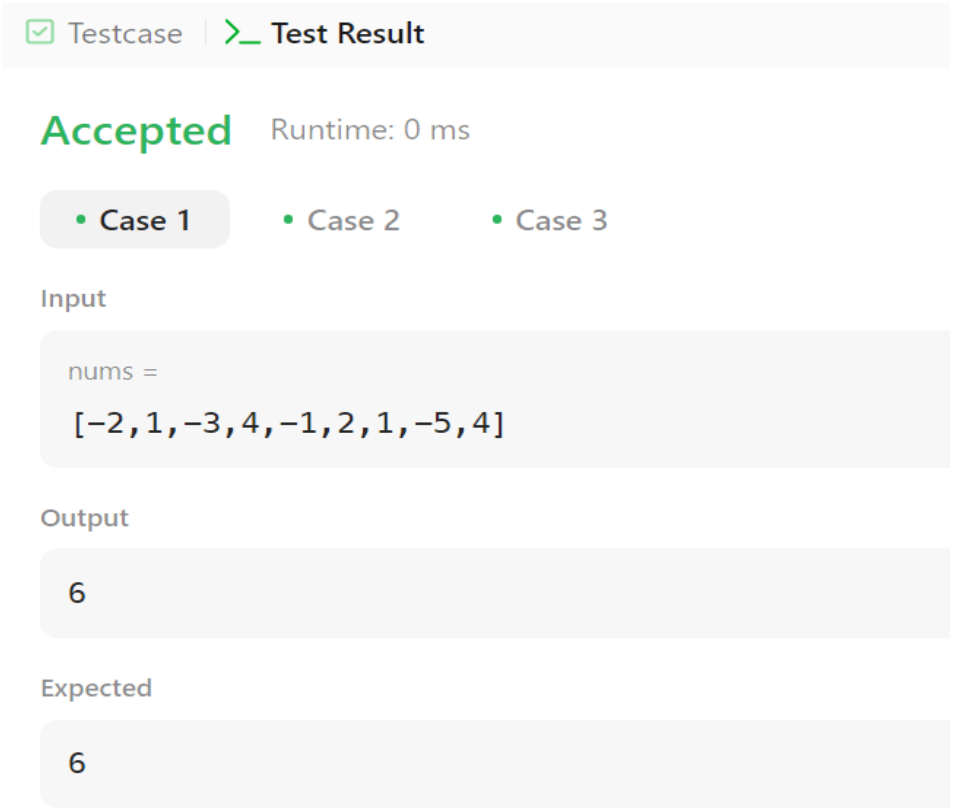


Ques 4. Maximum Subarray.

Code:

```
class Solution {
public:
    int maxSubArray(vector<int>& nums) {
        int maxSum = nums[0], currentSum = nums[0];
        for (int i = 1; i < nums.size(); i++) {
            currentSum = max(nums[i], currentSum + nums[i]);
            maxSum = max(maxSum, currentSum);
        }
        return maxSum;
    }
};
```

Output:



Ques 5. Search a 2-D matrix II.

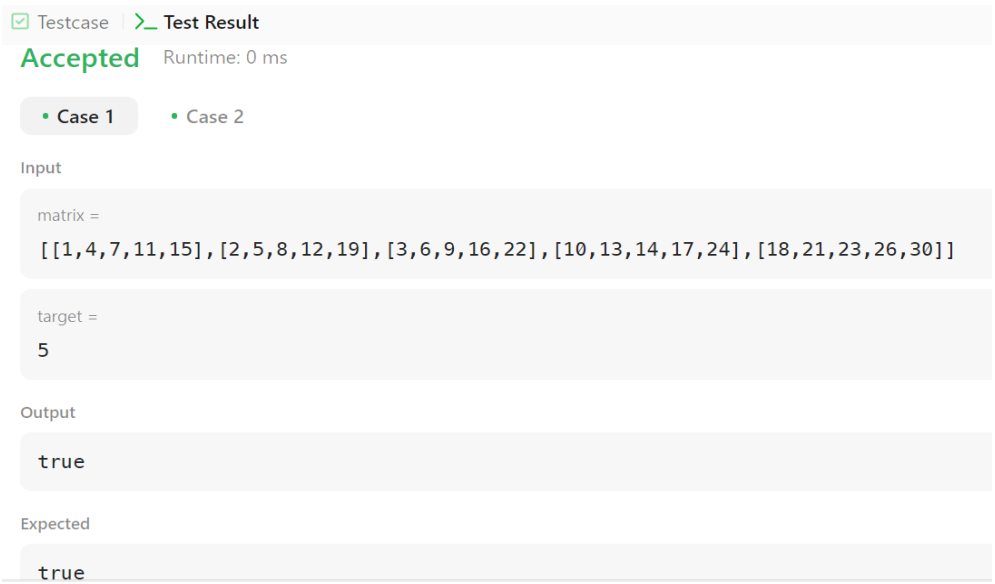
Code:

```
class Solution {
public:
    bool searchMatrix(vector<vector<int>>& matrix, int target) {
        int m = matrix.size(), n = matrix[0].size();
        int row = 0, col = n - 1; // Start from the top-right corner

        while (row < m && col >= 0) {
            if (matrix[row][col] == target) return true; // Found the target
            else if (matrix[row][col] > target) col--; // Move left
            else row++; // Move down
        }

        return false;
    }
};
```

Output:



Ques 6. Super Pow.

Code:

```
class Solution {
public:
    const int MOD = 1337;
    int modPow(int a, int b) {
        int result = 1;
        a %= MOD;
        while (b > 0) { a = (a * a) % MOD;
            b /= 2;}
        return result;
    }
    int superPow(int a, vector<int>& b) {
        a %= MOD;
        int result = 1;
        for (int digit : b) {
            result = (modPow(result, 10) * modPow(a, digit)) % MOD; }
        return result;
    }
};
```

Output: