**ADVANCED PROGRAMMING LAB-2 ASSIGNMENT**

**Submitted By:**

**Name- Anshika Yadav**

**UID- 22BCS10051**

**Section-22BCS\_IOT\_605-B**

1. **Longest Nice Substring**

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.size() >= right.size() ? left : right;

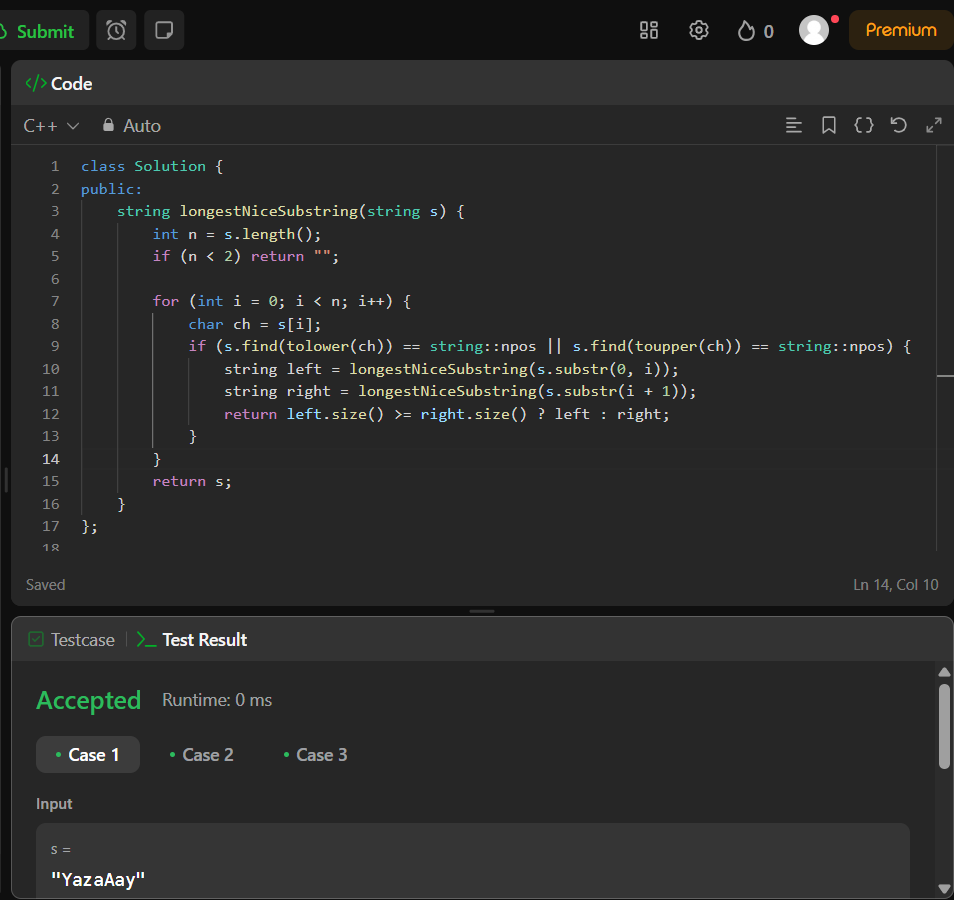
}

}

return s;

}

};



1. **Reverse Bits**

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);

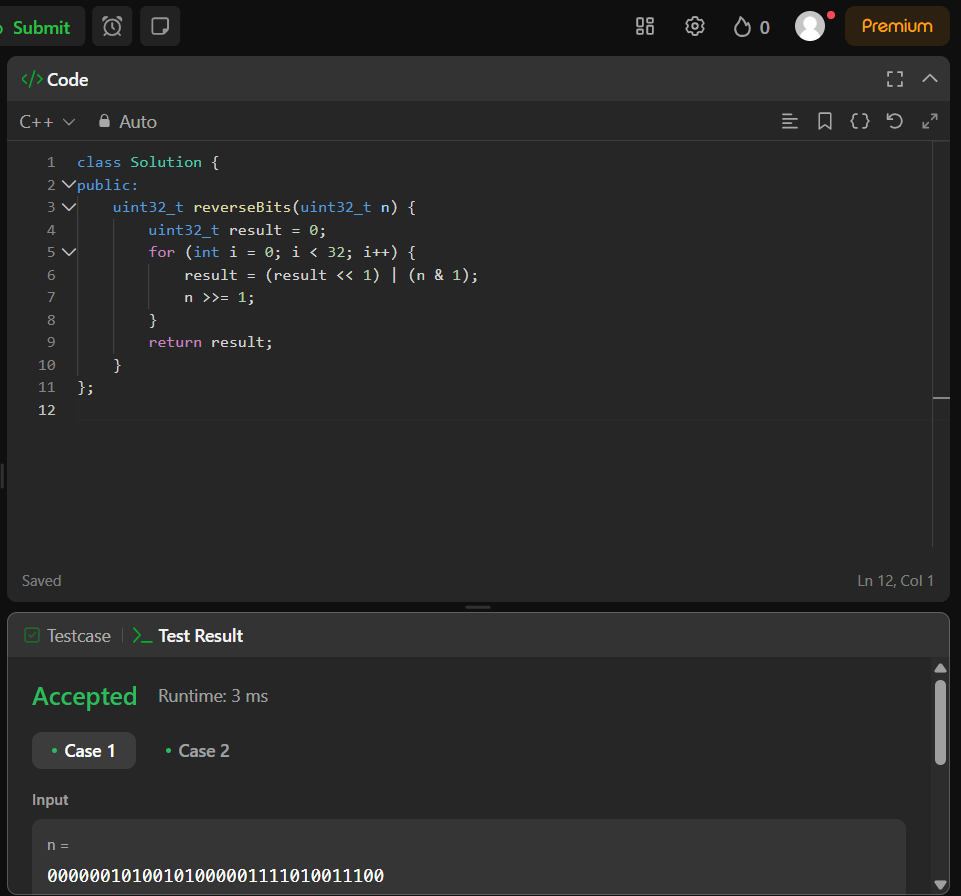
n >>= 1;

}

return result;

}

};



1. **Number of 1 bits**

class Solution {

public:

int hammingWeight(int n) {

int count = 0;

while (n) {

count += (n & 1);

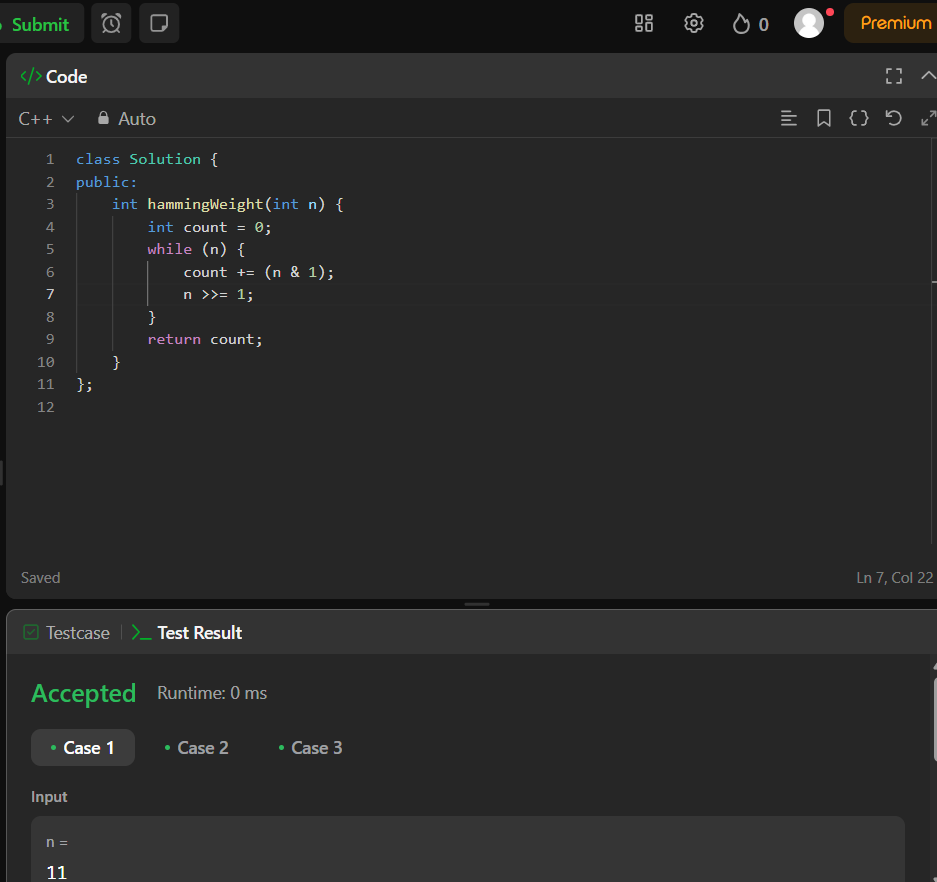
n >>= 1;

}

return count;

}

};



1. **Maximum Subarray**

class Solution {

public:

int maxSubArray(vector<int>& nums) {

int currentSum = nums[0];

int maxSum = nums[0];

for (int i = 1; i < nums.size(); i++) {

currentSum = max(nums[i], currentSum + nums[i]);

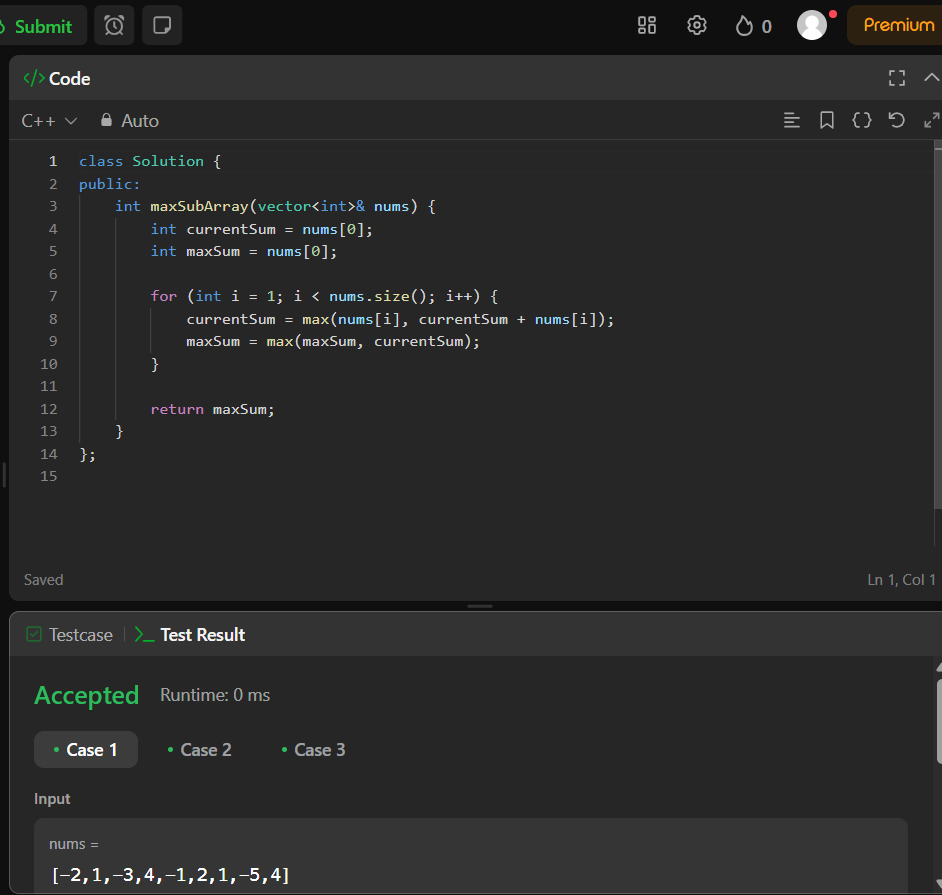
maxSum = max(maxSum, currentSum);

}

return maxSum;

}

};

****

1. **Search a 2D Matrix II**

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;

while (row < m && col >= 0) {

if (matrix[row][col] == target) return true;

else if (matrix[row][col] > target) col--;

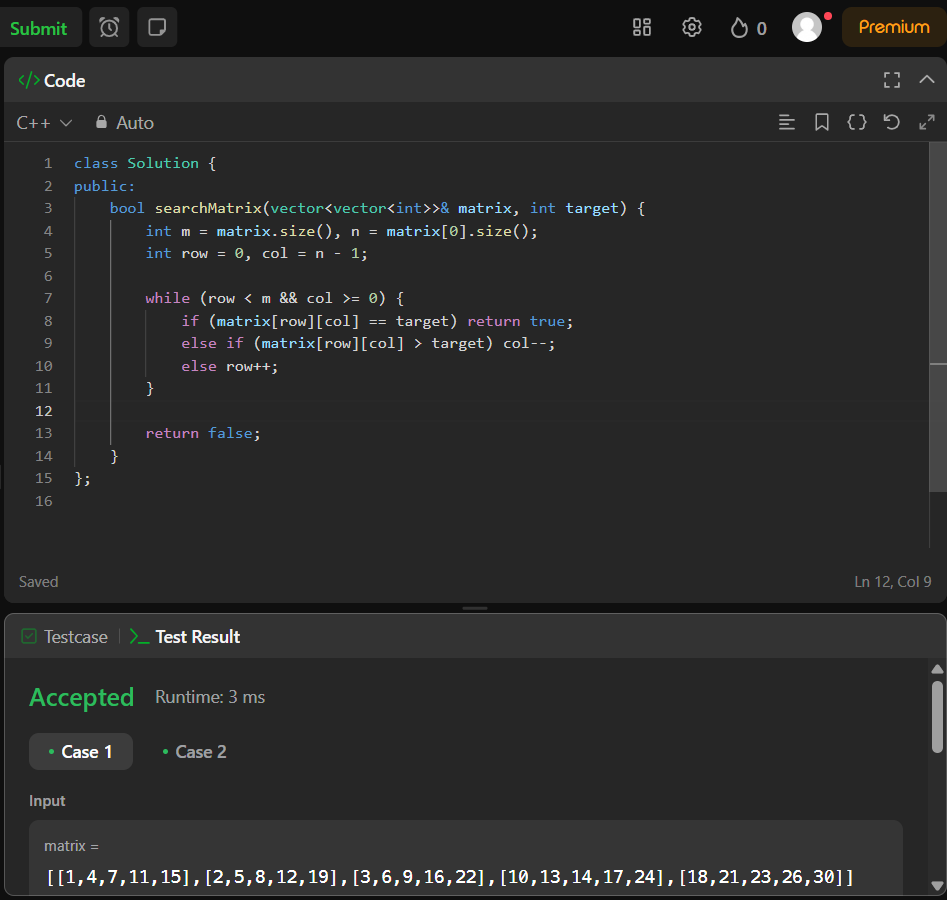
else row++;

}

return false;

}

};



1. **Super Pow**

class Solution {

public:

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

int mod = 1337;

a = a % mod;

int result = 1;

for (int i = b.size() - 1; i >= 0; --i) {

result = (result \* modPow(a, b[i], mod)) % mod;

a = modPow(a, 10, mod);

}

return result;

}

int modPow(int x, int y, int mod) {

int res = 1;

x = x % mod;

while (y > 0) {

if (y % 2 == 1) {

res = (res \* x) % mod;

}

x = (x \* x) % mod;

y = y / 2;

}

return res;

}

};

