Trạng thái	Đã xong
Bắt đầu vào lúc	Chủ Nhật, 30 tháng 3 2025, 10:52 PM
Kết thúc lúc	Chủ Nhật, 30 tháng 3 2025, 10:54 PM
Thời gian thực hiện	1 phút 56 giây

Câu hỏi 1

Đúng

Mô tả tiếng Việt:

Cho mảng 2 chiều chứa các số nguyên, kích thước M x N.

Hiện thực hàm:

int ascendingRows(int arr[][1000], int row, int col);

Trong đó; `arr`, `row` và `col` lần lượt là mảng 2 chiều, số hàng và số cột của mảng. Một hàng trong mảng được gọi là **HN1** nếu trong hàng đó, mỗi phần tử đều có giá trị không lớn hơn các phần tử đứng sau nó. Tìm số hàng **HN1** có trong mảng.

Ghi chú: (Các) thư viện `iostream`, và `string` đã được khai báo, và `namespace std` đã được sử dụng.

English version:

Given a two-dimensional array whose each element is integer, its size is M x N.

Implement the following function:

int ascendingRows(int arr[][1000], int row, int col);

Where `arr`, `row` and `col` are the given two-dimensional array, its number of rows and its number of columns. A row of the given array is called as **HN1** if on this row, each element's value is not higher than all elements after it. Find the number of **HN1** rows of the given array.

Note: Libraries `iostream`, and `string` have been imported, and `namespace std` has been used.

For example:

Test	Result
`int arr[][1000] = {{32,4,9},{-80,37,71},{-91,-79,-55}};` `cout << ascendingRows(arr,3, 3);`	2
`int arr[][1000] = {{-28,-8,-60,18},{-100,44,-1,24},{-94,92,-70,75}};` `cout << ascendingRows(arr,3,4);`	0

Answer: (penalty regime: 0 %)

Reset answer

```
int ascendingRows(int arr[][1000], int row, int col) {
    int count = 0;  // Biến đếm số hàng là HN1

    // Duyệt qua từng hàng trong mảng
    for (int i = 0; i < row; i++) {
        bool isAscending = true;  // Giả định hàng hiện tại là HN1

        // Kiểm tra xem hàng hiện tại có phải HN1 không
        for (int j = 0; j < col - 1; j++) {
            // Nếu phần tử hiện tại lớn hơn phần tử tiếp theo
            // thì hàng này không phải là HN1
            if (arr[i][j] > arr[i][j + 1]) {
                isAscending = false;
                break;  // Thoát vòng lặp bên trong
            }
        }

        // Nếu hàng là HN1, tăng biến đếm
        if (isAscending) {
            count++;
        }
    }

    return count;
}
```

	Test	Expected	Got	
✓	`int arr[][1000] = {{32,4,9},{-80,37,71},{-91,-79,-55}};` `cout << ascendingRows(arr,3, 3);`	2	2	✓

	Test	Expected	Got	
✓	`int arr[][1000] = {{-28,-8,-60,18},{-100,44,-1,24},{-94,92,-70,75}};` `cout << ascendingRows(arr,3,4);`	`0`	`0`	✓

Passed all tests! ✓

Câu hỏi **2**

Đúng

Mô tả tiếng Việt:

Cho mảng 2 chiều chứa các số nguyên, kích thước M x N.

Hiện thực hàm:

int primeColumns(int arr[][1000], int row, int col);

Trong đó; `arr`, `row` và `col` lần lượt là mảng 2 chiều, số hàng và số cột của mảng. Một cột của mảng được gọi là **HN2** nếu tổng tất cả các phần tử trong cột đó là số nguyên tố. Tìm số cột **HN2** có trong mảng.

Ghi chú: (Các) thư viện `iostream`, `vector` và `string` đã được khai báo, và `namespace std` đã được sử dụng.

English version:

Given a two-dimensional array whose each element is integer, its size is M x N.

Implement the following function:

int primeColumns(int arr[][1000], int row, int col);

Where `arr`, `row` and `col` are the given two-dimensional array, its number of rows and its number of columns. A column of the given array is called as **HN2** if the sum of all elements on it is a prime number. Find the number of **HN2** columns in the given array.

Note: Libraries `iostream`, `vector`, and `string` have been imported, and `namespace std` has been used.

For example:

Test	Result
`int arr[][1000] = {{-64,-28,-3,64},{-56,90,57,-31}};` `cout << primeColumns(arr,2,4);`	`0`
`int arr[][1000] = {{34,-15,11,-70,-23,24},{-39,-90,63,-45,-52,48},{-42,92,55,92,82,81}};` `cout << primeColumns(arr,3,6);`	`1`

Answer: (penalty regime: 0 %)

Reset answer

```
// Hàm kiểm tra số nguyên tố
bool isPrime(int num) {
    // Trường hợp đặc biệt
    if (num <= 1) {
        return false;
    }
    if (num <= 3) {
        return true;
    }
    if (num % 2 == 0 || num % 3 == 0) {
        return false;
    }

    // Kiểm tra các số từ 5 đến căn bậc hai của num
    for (int i = 5; i * i <= num; i += 6) {
        if (num % i == 0 || num % (i + 2) == 0) {
            return false;
        }
    }

    return true;
}

// Hàm đếm số cột có tổng là số nguyên tố
int primeColumns(int arr[][1000], int row, int col) {
    int count = 0;  // Biến đếm số cột HN2

    // Duyệt qua từng cột trong mảng
    for (int j = 0; j < col; j++) {
        int sum = 0;  // Biến lưu tổng các phần tử trong cột

        // Tính tổng các phần tử trong cột j
        for (int i = 0; i < row; i++) {
            sum += arr[i][j];
        }

```

```
        // Kiểm tra xem tổng có phải số nguyên tố không
        if (isPrime(sum)) {
            count++;  // Nếu đúng, tăng biến đếm
        }
    }

    return count;
}
```

	Test	Expected	Got	
✓	`int arr[][1000] = {{-64,-28,-3,64},{-56,90,57,-31}};` `cout << primeColumns(arr,2,4);`	`0`	`0`	✓
✓	`int arr[][1000] = {{34,-15,11,-70,-23,24},{-39,-90,63,-45,-52,48},{-42,92,55,92,82,81}};` `cout << primeColumns(arr,3,6);`	`1`	`1`	✓

Passed all tests! ✓