Hướng dẫn giải Code Challenge #4

Big-O & VNG - 11/09/2022

BÀI A: JERRY AND PASSWORDS

Tóm tắt đề bài:

Xem tại đây

Hướng dẫn giải

Đơn giản hoá đề bài ta sẽ có được một bài toán tìm số phần tử bên trái cần được loại bỏ để được một chuỗi các ký tự phân biệt.

Đầu tiên ta đi từ phải sang trái của chuỗi để đếm số lần xuất hiện của từng ký tự trong chuỗi. Ở đây ta có thể vận dụng cấu trúc dữ liệu "set" để chứa những ký tự phân biệt và dùng hàm count() để đếm chúng. Đồng thời trong quá trình di chuyển từ phải sang trái, ta kiểm tra xem ký tự s[i] đã được xuất hiện bên phải của chuỗi hay chưa. Nếu ký tự s[i] đã xuất hiện ở bên phải, ta chỉ cần loại bỏ những phần tử phía trước nó. Lúc này, số phần tử cần loại bỏ của ta sẽ là i + 1.

Ta cần có thêm một biến flag để kiểm tra trường hợp nếu mảng đã cho đã là một mảng chứa toàn các ký tự phân biệt. Khi đó ta sẽ trả về là 0 do ta không cần phải loại bỏ phần tử nào cả.

Mã nguồn tham khảo

- C++: https://ideone.com/J3Vw3v

BÀI B: ARRAY

Tóm tắt đề bài:

Xem tại đây

Hướng dẫn giải

Subtask 1:

Với subtask này chỉ cần lặp từng phần tử A_i, sau đó dùng một vòng lặp khác để đếm số phần tử có giá trị bằng A_i xuất hiện trong mảng

Độ phức tạp: $O(n^2)$

Subtask 2:

Với subtask này, việc sử dụng 2 vòng lặp lồng nhau sẽ dẫn đến bị quá thời gian. Chúng ta có 2 cách để giải quyết:

Cách 1: Chúng ta có thể sắp xếp lại mảng theo thứ tự tăng dần và dùng thuật toán Binary Search để tìm vị trí đầu tiên và cuối cùng mà phần tử đó xuất hiện trong mảng đã sắp xếp, và để đảm bảo đúng thứ tự output thì khi nhập chúng ta cần lưu lại thứ tự nhập tương ứng với mỗi số trong mảng.

Độ phức tạp: $O(nlog(n))$

Cách 2: Chúng ta có thể sử dụng cấu trúc dữ liệu HashMap để lưu lại số lần xuất hiện của từng giá trị, và xuất ra từng giá trị tương ứng.

Độ phức tạp: $O(nlog(n))$.

Mã nguồn tham khảo

- C++: https://ideone.com/IO2sDV

BÀI C: SUM

Tóm tắt đề bài:

Xem tại đây

Hướng dẫn giải

Bắt đầu từ ô u, v, ta nhận thấy các ô mà u, v đến được khi ghép lại sẽ tạo thành 1 hình thoi.

Vậy mốt chốt là ta tính tổng các phần tử $a[i][j]$ trên hình thoi với thời gian cho phép. Ta sử dụng phương pháp xoay ma trận, ta xoay 45 độ để đưa hình thoi về hình vuông.

Với x, y là tọa độ ban đầu và xp, yp là tọa độ sau khi xoay, ta có: $(xp, yp) = (x + y - 1, m - x + y)$

Ví dụ:

```
1 2 3
4 5 6
7 8 9
Sau khi xoay sẽ thành
0 0 1 0 0
0 4 0 2 0
7 0 5 0 3
0 8 0 6 0
0 0 9 0 0
```

Cuối cùng là sử dụng mảng cộng dồn để tính nhanh tổng các hình vuông trên từng truy vấn.

Mã nguồn tham khảo

- C++: https://ideone.com/jebJDr

BÀI D: ROX

Tóm tắt đề bài:

Xem tại đây

Hướng dẫn giải

- Gọi b_i là giá trị x của thao tác thứ i.
- Đặt $pre = b_1 \oplus b_2 \oplus ... \oplus b_q$.
- Dễ thấy, sau khi thực hiện q thao tác trên, phần tử thứ i của mảng a sẽ có giá trị là $a_i \oplus b_1 \oplus b_2 \oplus ... \oplus b_q = a_i \oplus pre$.
- Vậy, để giải quyết bài toán đã cho, ta sẽ duyệt qua q thao tác để tính pre. Tiếp đến, duyệt qua mảng a để cập nhật lại giá trị của từng phần tử và tính đáp án của bài toán.

Độ phức tạp: Việc tính pre sẽ có độ phức tạp $O(q)$, tính đáp án có độ phức tạp $O(n)$. Do đó, lời giải trên có độ phức tạp $O(n + q)$.

Mã nguồn tham khảo

- C++: https://ideone.com/zqagB7