The Playboy Chimp

Link submit:
https://uva.onlinejudge.org/index.php?option=com_onlinejudge&Itemid=8&category=316&page=show_problem&problem=1552

Solution:

C++	http://ideone.com/6pOLnz
Java	https://ideone.com/rd0Vjd
Python	https://ideone.com/0llqHJ

Tóm tắt đề:

Có một chú tinh tinh tên Luchu Bandor rất cao lớn, tuy nhiên bạn đời của anh ấy lại không cao như vậy nên cảm thấy rất khó chịu và quyết định tìm cho mình một người bạn mới phù hợp với mình hơn. Cậu ta quyết định tìm đến trường Lady Chimp's High School – nơi mà cậu có thể tìm thấy các cô tinh tinh khác. Bây giờ, Luchu quyết định chọn cho mình hai cô tinh tinh, một cô tinh tinh cao nhất nhưng thấp hơn Luchu, một cô khác thấp nhất nhưng cao hơn cậu ta.

Input:

Dòng đầu tiên chứa một số nguyên dương N là số lượng cô tinh tinh ($1 \le N \le 50000$).

Dòng thứ hai chứa N số nguyên dương có giới hạn trong đoạn $[1, 2^{31} - 1]$ là chiều cao của các cô tinh tinh, chiều cao được liệt kê theo thứ tự không giảm.

Dòng thứ ba chứa số nguyên dương Q ($1 \le Q \le 25000$) là số lượng các câu hỏi.

Dòng thứ tư chứa Q số nguyên dương có giới hạn trong đoạn $[1, 2^{31} - 1]$ là chiều cao của Luchu ở mỗi câu hỏi.

Output:

Với mỗi câu hỏi, xuất ra trên 1 dòng, gồm 2 giá trị, số thứ nhất là chiều cao của cô tinh tinh cao nhất mà thấp hơn Luchu và số thứ hai là chiều cao của con tinh tinh thấp nhất mà cao hơn Luchu. Trong trường hợp nếu không tìm được thì in ra kí tự "X".

Ví dụ:

4 1 4 5 7	1 5 5 7

4 4 6 8 10	7 X 7 X

Giải thích ví dụ:

Câu hỏi 1: chỉ có 1 tinh tinh có chiều cao thấp hơn 4 là 1 nên giá trị đầu tiên chắc chắn là 1. Có 2 tinh tinh cao hơn 4 là 5 và 7 nhưng ta lấy chiều cao nhỏ nhất, tức 5.

Câu hỏi 2: có 3 tinh tinh thấp hơn 6 là {1, 4, 5}, lấy chiều cao lớn nhất là 5. Có một tinh tinh cao hơn 6 là 7 nên lấy 7.

Câu hỏi 3 và 4: tất cả các con tinh tinh đều có chiều cao thấp hơn chiều cao của Luchu, nên chiều cao cao nhất mà nhỏ hơn Luchu được chọn là 7, vì không có con tinh tinh nào cao hơn Luchu nên xuất ra "X".

Hướng dẫn giải:

Đây là một bài khá đơn giản, tuy nhiên đòi hỏi ta vận dụng được cách xử lý các hàm tìm kiếm nhị phân phải chính xác. Từ bài toán ban đầu, ta sẽ chia thành 2 bài toán con để dễ giải quyết:

- Tìm phần tử lớn nhất nhỏ hơn giá trị X trong mảng.
- Tìm phần tử nhỏ nhất lớn hơn giá trị X trong mảng.

Ở bài toán con thứ nhất: ta thấy nó gần giống cách tìm lower bound, tuy nhiên lower bound là trả về phần tử đầu tiên có giá trị **không nhỏ hơn** giá trị X, nhìn lại bài này, thì ta thấy, giá trị ta cần tìm là nhỏ hơn X, tức là nằm ngay trước phần tử mà lower bound trả về (**nếu có**). Vậy ý tưởng để giải quyết lúc này là ta cứ tìm lower bound, sau đó lấy phần tử nằm liền trước nó, nếu không có (tức là phần tử trả về là phần tử đầu tiên của mảng) thì kết quả là "X".

Ở bài toán con thứ hai: có thể thấy nó chính là bài toán mà upper bound giải quyết, tức là ta chỉ việc tìm upper bound, sau đó kiểm tra nếu nó không là end (tức nằm ngoài mảng – mảng không có phần tử nào lớn hơn x) thì xuất phần tử đó ra, ngược lại xuất "X".

Độ phức tạp: **O(QlogN)** với Q là số truy vấn, với mỗi truy vấn, ta phải dùng 2 hàm tìm kiếm nhị phân với chi phí O(logN).