**[Set Map]. Bài 1. Phần tử phân biệt**

Cho dãy số A[] gồm có N phần tử, bạn hãy đếm xem trong mảng có bao nhiêu phần tử phân biệt? Chú ý giải bài này với 3 cách : Sử dụng set, map, sắp xếp.

**Input Format**

Dòng đầu tiên là số nguyên N. Dòng tiếp theo gồm N số nguyên A[i]

**Constraints**

1≤ N ≤ 10^5; -10^9 ≤ A[i] ≤ 10^9

**Output Format**

In ra số lượng phần tử khác nhau trong mảng.

**Sample Input 0**

5

2 1 2 1 3

**Sample Output 0**

3

**[Set Map]. Bài 2. Truy vấn phần tử trong mảng.**

Cho dãy số A[] gồm có N phần tử, bạn hãy trả lời các truy vấn để xác định xem phần tử X nào đó có nằm trong mảng hay không? Bạn hãy thử giải bài này bằng 3 cách : Set, Map, Binary Search.

**Input Format**

* Dòng đầu tiên là số nguyên N.
* Dòng thứ 2 gồm N số nguyên A[i]
* Dòng thứ 3 là số lượng truy vấn Q.
* Q dòng tiếp theo mỗi dòng là một số nguyên X.

**Constraints**

1≤ N ≤ 10^5; 0 ≤ A[i], X ≤ 10^9; 1≤ Q ≤ 1000;

**Output Format**

Đối với mỗi truy vấn in ra YES nếu phần tử X xuất hiện trong mảng, ngược lại in ra NO.

**Sample Input 0**

5

1 2 1 3 4

2

1

6

**Sample Output 0**

YES

NO

**[Set Map]. Bài 3. Phần tử riêng biệt.**

Cho dãy số A[] gồm có N phần tử, bạn hãy liệt kê các giá trị xuất hiện trong dãy theo thứ tự xuất hiện, mỗi giá trị chỉ liệt kê một lần.

Gợi ý : Dùng set để lưu các phần tử đã được in ra trước đó, gặp phần tử a[i] thì kiểm tra nhanh trong set có giá trị a[i] chưa để in ra

**Input Format**

Dòng đầu tiên là số nguyên N. Dòng thứ 2 gồm N số nguyên A[i]

**Constraints**

1≤ N ≤ 10^5; 0 ≤ A[i], X ≤ 10^9;

**Output Format**

In ra các giá trị xuất hiện trong mảng theo thứ tự xuất hiện.

**Sample Input 0**

8

1 2 3 2 1 3 4 8

**Sample Output 0**

1 2 3 4 8

**[Set Map]. Bài 4. Truy vấn trên mảng**

Cho dãy số A[] gồm có N phần tử, có 3 thao tác như sau :

* Thao tác 1 : Thêm 1 phần tử X vào mảng.
* Thao tác 2 : Xóa 1 phần tử X khỏi mảng. Trong trường hợp phần tử X không xuất hiện trong mảng, sẽ không thực hiện xóa, nếu trong mảng có nhiều phần tử X thì chỉ xóa đi 1 phần tử X trong mảng.
* Thao tác 3 : Truy vấn xem phần tử X có xuất hiện trong mảng hay không?

Ban đầu mảng A có N phần tử, với các truy vấn phần tử X trong mảng, in ra YES nếu X xuất hiện trong mảng, ngược lại in ra NO.

**Input Format**

* Dòng đầu tiên là số nguyên N.
* Dòng thứ 2 gồm N số nguyên A[i].
* Dòng thứ 3 là số lượng thao tác Q.
* Q dòng tiếp theo mỗi dòng là thông tin của một thao tác, gồm 2 số, số đầu tiên là loại thao tác, số tiếp theo là phần tử X trong thao tác.

**Constraints**

1≤ N ≤ 10^4; 0 ≤ A[i], X ≤ 10^9; 1≤ Q ≤ 1000; 0 ≤ X ≤ 10^9;

**Output Format**

Đối với mỗi thao tác loại 3, in ra YES nếu X xuất hiện trong mảng, ngược lại in ra NO.

**Sample Input 0**

6

1 2 3 4 5 6

4

1 1

2 6

3 1

3 6

**Sample Output 0**

YES

NO

**Explanation 0**

Ban đầu mảng có 6 phần tử 1, 2, 3, 4, 5, 6. Sau thao tác đầu tiên mảng có thêm 1 phần tử 1 và trở thành mảng (1, 1, 2, 3, 4, 5, 6). Sau thao tác thứ 2, mảng xóa đi 1 phần tử 6 và trở thành mảng (1, 1, 2, 3, 4, 5). Thao tác thứ 3 truy vấn phần tử 1, câu trả lời là YES vì 1 xuất hiện trong mảng. Thao tác thứ 4 truy vấn phần tử 6, câu trả lời là NO vì 6 không xuất hiện trong mảng.

**[Set Map]. Bài 9. Union**

Cho mảng A[] và B[] lần lượt gồm N và M số nguyên, nhiệm vụ của bạn là tìm hợp của 2 mảng này và in theo thứ tự từ lớn về nhỏ.

Gợi ý : Để tìm hợp của 2 mảng thì chỉ cần dùng 1 set và lưu mọi phần tử của 2 mảng, set sẽ tự loại bỏ các phần tử trùng nhau trong cả 2 mảng.

Cú pháp duyệt ngược của set như sau :

for(auto it = se.rbegin(); it != se.rend(); ++it){

cout << \*it << ' ';

}

**Input Format**

* Dòng 1 gồm N và M
* Dòng 2 gồm N số trong mảng A[]
* Dòng 3 gồm M số trong mảng B[]

**Constraints**

* 1<=N,M<=10^5
* -10^9<=A[i],B[i]<=10^9

**Output Format**

In ra tập hợp của 2 mảng theo thứ tự giảm dần

**Sample Input 0**

9 6

7 9 2 0 0 5 6 4 7

9 7 3 4 5 7

**Sample Output 0**

9 7 6 5 4 3 2 0

**[Set Map]. Bài 10. Intersection**

Cho mảng A[] và B[] lần lượt gồm N và M số nguyên, nhiệm vụ của bạn là tìm giao của 2 mảng này và in theo thứ tự xuất hiện trong mảng A[].

Gợi ý :

Cách 1 : Dùng 2 set để lưu 2 mảng A, B sau đó duyệt 1 trong 2 mảng và tìm kiếm sự xuất hiện của phần tử đang duyệt trong mảng còn lại.

Cách 2 : Dùng 1 map để đánh dấu, duyệt qua các phần tử trong mảng A[] và cho map của từng phần tử trong mảng A[] = 1, duyệt qua mảng B[] thì kiểm tra nếu map của phần tử trong mảng B đã bằng 1 thì chuyển thành 2. Cuối cùng duyệt map và in ra những phần tử được đánh dấu là 2.

**Input Format**

* Dòng 1 gồm N và M
* Dòng 2 gồm N số trong mảng A[]
* Dòng 3 gồm M số trong mảng B[]

**Constraints**

* 1<=N,M<=10^5
* -10^9<=A[i],B[i]<=10^9

**Output Format**

In ra tập hợp của 2 mảng theo thứ tự xuất hiện trong mảng A[]

**Sample Input 0**

9 8

7 5 0 0 1 6 2 7 3

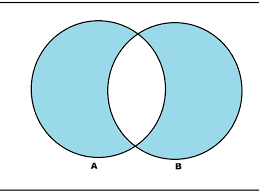
6 4 9 4 2 8 3 5

**Sample Output 0**

5 6 2 3

**[Set Map]. Bài 12. Symmetric difference**

Cho mảng A[] và B[] lần lượt gồm N và M số nguyên, nhiệm vụ của bạn là tìm những phần tử xuất hiện ở 1 trong 2 mảng nhưng không được xuất hiện ở cả 2 mảng.



**Input Format**

* Dòng 1 gồm N và M
* Dòng 2 gồm N số trong mảng A[]
* Dòng 3 gồm M số trong mảng B[]

**Constraints**

* 1<=N,M<=10^5
* -10^9<=A[i],B[i]<=10^9

**Output Format**

In ra đáp án của bài toán theo thứ tự từ bé đến lớn

**Sample Input 0**

9 8

9 0 2 3 9 7 8 1 4

1 0 2 1 1 2 0 5

**Sample Output 0**

3 4 5 7 8 9

**[Set Map]. Bài 13. Loại trừ**

Cho mảng A[] và B[] có N và M phần tử là các số nguyên, hãy tìm các phần tử xuất hiện trong mảng 1 mà không xuất hiện trong mảng 2.

**Input Format**

Dòng đầu tiên là N và M; Dòng thứ 2 là N số của mảng A[]; Dòng thứ 3 là M số của mảng B[];

**Constraints**

1<=N, M<=10^6; -10^9<=A[i], B[i] <= 10^9;

**Output Format**

In ra các số xuất hiện trong mảng 1 mà không xuất hiện trong mảng 2 theo thứ tự từ bé đến lớn, mỗi giá trị thỏa mãn chỉ liệt kê 1 lần.

**Sample Input 0**

3 4

1 2 3

1 2 4 5

**Sample Output 0**

3

**[Set Map]. Bài 14. Suffix And Query**

Cho mảng A[] gồm N phần tử. Bạn được yêu cầu trả lời cho các truy vấn, mỗi truy vấn yêu cầu bạn đếm số các giá trị khác nhau từ chỉ số L tới chỉ số N - 1.

Gợi ý : Gọi F[i] là số lượng các giá trị khác nhau tính từ chỉ số i tới chỉ số n - 1 trong mảng, xây dựng trước mảng F[i], sau đó mỗi truy vấn chỉ cần truy cập vào F[l] là có thể tìm được số lượng các giá trị khác nhau từ l tới n - 1. Cách xây dựng mảng F[] là duyệt từ cuối về và dùng 1 set để lưu các giá trị khác nhau tính từ cuối, gán F[i] = set.size()

**Input Format**

* Dòng đầu tiên là N - số lượng phần tử của mảng
* Dòng thứ 2 gồm N phần tử A[i] của mảng
* Dòng thứ 3 là số truy vấn Q
* Q dòng tiếp theo mỗi dòng là một số L.

**Constraints**

* 1<=N<=1000000
* 1<=A[i]<=10^9
* 1<=Q<=10000
* 0<=L<=N-1

**Output Format**

Với mỗi truy vấn in kết quả trên 1 dòng.

**Sample Input 0**

9

3 3 3 4 2 0 3 1 2

4

8

6

1

7

**Sample Output 0**

1

3

5

2