F. One Occurrence

time limit per test 3 seconds
memory limit per test 768 megabytes
input standard input
output standard output

You are given an array a consisting of n integers, and q queries to it. i-th query is denoted by two integers l_i and r_i. For each query, you have to find **any** integer that occurs **exactly once**in the subarray of a from index l_i to index r_i (a subarray is a contiguous subsegment of an array). For example, if $a = [1, 1, 2, 3, 2, 4]$, then for query $(l_i = 2, r_i = 6)$ the subarray we are interested in is $[1, 2, 3, 2, 4]$, and possible answers are 1, 3 and 4; for query $(l_i = 1, r_i = 2)$ the subarray we are interested in is $[1, 1]$, and there is no such element that occurs exactly once.

Can you answer all of the queries?

Input

The first line contains one integer n ($1 \le n \le 5 \cdot 10^5$).

The second line contains n integers $a_1, a_2, \dots, a_n$ ($1 \le a_i \le 5 \cdot 10^5$).

The third line contains one integer q ($1 \le q \le 5 \cdot 10^5$).

Then q lines follow, i-th line containing two integers l_i and r_i representing i-th query ($1 \le l_i \le r_i \le n$).

Output

Answer the queries as follows:

If there is no integer such that it occurs in the subarray from index l_i to index r_i exactly once, print 0. Otherwise print any such integer.

Example

input Copy

```
6
1 1 2 3 2 4
2
2 6
1 2
```

output Copy

```
4
0
```

```cpp
/// Codeforces 1000F - One Occurrence
#include<bits/stdc++.h>
using namespace std;
const int MAXN = 500015;
const int BLOCK = 708;
struct dt{ int l,r,b,i; };
dt qr[MAXN];
int n,a[MAXN],cnt[MAXN],val[MAXN],ans[MAXN],bl[710],sum;

bool cmp(dt A, dt B){
  if (A.l / BLOCK != B.l / BLOCK) return A.l < B.l;
  return A.l / BLOCK % 2 ? A.r > B.r : A.r < B.r;
}
void Add(int x){
    cnt[a[x]]++;
    if(cnt[a[x]]==1){sum++,bl[a[x]/BLOCK]++;}
    if(cnt[a[x]]==2){sum--,bl[a[x]/BLOCK]--;}
}
void Remove(int x){
    cnt[a[x]]--;
    if(cnt[a[x]]==1){sum++,bl[a[x]/BLOCK]++;}
    if(cnt[a[x]]==0){sum--,bl[a[x]/BLOCK]--;}
}
int solve(){
    if(sum==0)return 0;

    for(int i=0; i<BLOCK; i++){
        if(bl[i]>0){
            for(int j=i*BLOCK; ; j++){
                if(cnt[j]==1){
                    return j;
                }
            }
        }
    }

    return 0;
}
int main(){
    scanf("%d",&n);
    for(int i=0; i<n; i++)scanf("%d",&a[i]);

    int q; scanf("%d",&q);
    for(int i=0; i<q; i++){
        int l,r; scanf("%d%d",&l,&r);
        l--; r--;
        qr[i].l = l;
        qr[i].r = r;
        qr[i].b = l/BLOCK;
        qr[i].i = i;
    }

    sort(qr,qr+q,cmp);

    int l=0,r=-1;
    for(int i=0; i<q; i++){
        while(r<qr[i].r) Add(++r);
        while(r>qr[i].r) Remove(r--);
        while(l>qr[i].l) Add(--l);
        while(l<qr[i].l) Remove(l++);
        ans[qr[i].i] = solve();
    }

    for(int i=0; i<q; i++){
        printf("%d\n",ans[i]);
    }

    return 0;
}
```