# Unique Art

locked

by anuj_95



Some art students decide to visit a museum, which has framed paintings placed on the wall in a long row. Each painting of a certain type, represented by an integer. Each student decides to only look at a *contiguous set of paintings*. In addition, each student appreciates uniqueness, so each student only appreciates paintings with a unique type *among all paintings he/she looked at.*

Find the number of paintings that each student appreciates.

Complete the functions `initialize` and `student` to solve the challenge. `initialize` takes an array denoting the types of the paintings. `student` takes in two integers denoting the leftmost and rightmost index that the next student will look at, and returns an integer denoting the number of paintings that that student will appreciate.

`initialize` will be called once at the beginning, and then `student` will be called as many times as number of students.

**Input Format**

The first line of the input contains a single integer $n$, denoting the number of framed paintings.

The second line consists of $n$ space-separated integers $t_1, t_2, \ldots, t_n$, each denoting the type of painting.

The third line consists of an integer $q$, denoting the number of students.

The next $q$ lines each contains two space-separated integers $i$ and $j$ denoting the leftmost and rightmost index that the next student will look at.

We index the paintings $1$ to $n$.

**Constraints**

- $1 \leq n \leq 10^6$
- $-10^9 \leq t_i \leq 10^9$
- $1 \leq i \leq j \leq n$
- $1 \leq q \leq 10^6$

**Output Format**

For each student in input order, print a single line containing a single integer denoting the number of paintings that that student will appreciate.

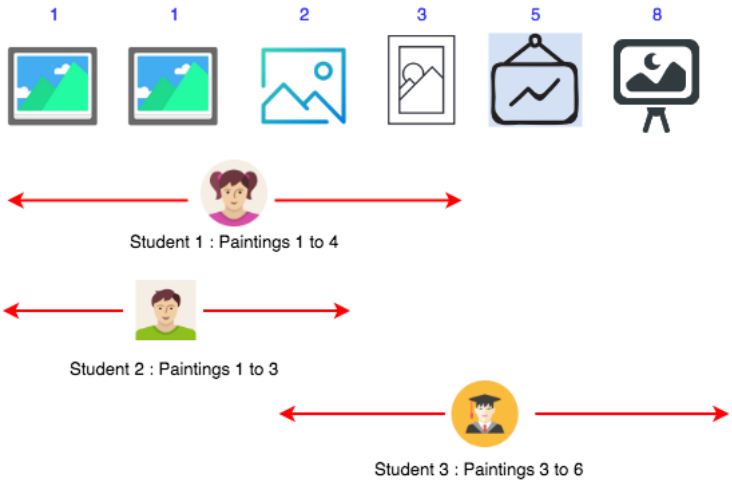
**Sample Input 0**

```
6
1 1 2 3 5 8
3
1 4
1 3
3 6
```

**Sample Output 0**

```
2
1
4
```

**Explanation 0**




The first student looks at paintings $1$ to $4$. They have types $[1, 1, 2, 3]$. Only $2$ and $3$ are unique among them, so the first student will only appreciate two paintings. Similarly, the second student appreciates only one painting and the third student appreciates all four paintings in sight.

Submissions: 1420
Max Score: 70
Difficulty: Hard

Rate This Challenge:

More

Current Buffer (saved locally, editable) C++14

```cpp
#include <cmath>
#include <unordered_map>
#include <vector>
#include <iostream>
#include <algorithm>
using namespace std;

struct Query {  int L, R, i;    };

constexpr int MAX = 1'000'000;
int ans = 0;
int cnt[MAX + 1], t[MAX + 1], res[MAX + 1];
Query q[MAX + 1];

void add(int i)
{
    ++cnt[t[i]];
    ans += cnt[t[i]] == 1;
    ans -= cnt[t[i]] == 2;
}

void remove(int i)
{
    --cnt[t[i]];
    ans += cnt[t[i]] == 1;
    ans -= cnt[t[i]] == 0;
}

int main()
{
    ios::sync_with_stdio(false);
    cin.tie(nullptr);

    int n;  cin >> n;
    for (int i = 0; i < n; ++i)
        cin >> t[i];

    int id = -1;
    unordered_map<int, int> m;
    for (int i = 0; i < n; ++i)
    {
```

```
        if (m.count(t[i]) == 0)
            m[t[i]] = ++id;

        t[i] = m[t[i]];
    }

    int sqrt_n = sqrt(n);

    int Q;
    cin >> Q;
    for (int i = 0; i < Q; ++i)
    {
        cin >> q[i].L >> q[i].R;
        --q[i].L;
        --q[i].R;
        q[i].i = i;
    }

    sort(q, q + Q, [sqrt_n](const Query &lhs, const Query &rhs)
    {
        if (lhs.L / sqrt_n == rhs.L / sqrt_n)  //same block
            return lhs.R < rhs.R;

        return lhs.L / sqrt_n < rhs.L / sqrt_n; //sort by block
    });

    int curL = 0;
    int curR = 0;
    for (int i = 0; i < Q ; ++i)
    {
        int L = q[i].L;
        int R = q[i].R;
        while (curL < L)
        {
            remove(curL);
            ++curL;
        }

        while (curL > L)
        {
            add(curL - 1);
            --curL;
        }

        while (curR <= R)
        {
            add(curR);
            ++curR;
        }

        while (curR > R + 1)
        {
            remove(curR - 1);
            --curR;
        }
        res[q[i].i] = ans;
    }

    for (int i = 0; i < Q; ++i)
        cout << res[i] << "\n";
    return 0;
}

```

Line: 1 Col: 1

Upload Code as File  Test against custom input  Run Code  Submit Code