**Round Table Meeting**

There are *N* students in a round table meeting. Each student belongs to a university given in the array A[i], which denotes the university that the ith student belongs to.  
There are *Q* queries of the form xy, denoting two universities. The answer to each query is the minimum time taken by any one of the student from these universities to meet each other.

**Note:**

* Exactly *1* student of university *x* and *y* have to meet each other
* In this context, meeting means that the absolute distance between the positions is ≤1
* Time taken by the student to move from ith position to i+1th or i−1th position is exactly *1* second
* Both the students move together at the same time
* Both the students move optimally in the correct direction
* Two students can belong to the same university
* Round Table means that the nth position and 1st position are adjacent

**Input**

The first line of input contains *2* integers *N* and *Q*

The second line contains *N* space seperated integers denoting A[i]

*Q* lines follow . Each containing 2 integers *x* and *y*

**Output**

The output contains *q* lines each containing the answer to each query

**CONSTRAINTS**

Constraint: 1≤N≤50000 , 1≤Q≤100 The elements of the array are between *1* to 100000

*x* and *y* are guaranteed to be present in the array

**SAMPLE INPUT**

10 4

7 1 4 3 1 6 4 2 5 1

3 4

1 2

2 7

6 7

**SAMPLE OUTPUT**

0

1

1

2

**Explanation**

**Positions :** **1 2 3 4 5 6 7 8 9 10**

**Values :** **7 1 4 3 1 6 4 2 5 1**

Query 1: 3 and 4 are already adjacent , therefore answer is 0  
Query 2: 1 and 2, in this case there are three 1's present

Checking the answer for all the 1's

1(at posn 2) : Answer = 2  
1(at posn 5) : Answer= 1  
1(at posn 10) : Answer= 1

Therefore answer=1;

EDITORIAL

Store the positions for a particular university in a vector

For solving the query iterate the smaller vector(preferably) and do binary search on the larger vector(finding the next higher and lower value)

Expected Time Complexity : Q\*N log N

1. #include<bits/stdc++.h>
2. using namespace std;
3. #define ll long long
4. #define modu 1000000007
5. #define pb push\_back
6. vector<int> v[1000001];
7. int bs(int numElems,int b,int target)
8. {
9. int low = 0, high = numElems;
10. while (low != high)
11. {
12. int mid = (low + high) / 2;
13. if (v[b][mid] <= target)
14. {
15. low = mid + 1;
16. }
17. else
18. {
19. high = mid;
20. }
21. }
22. return high;
23. }
24. int mod(int a)
25. {
26. if(a<0)
27. return -a;
28. else
29. return a;
30. }
31. int main()
32. {
33. ios\_base::sync\_with\_stdio(false);
34. cin.tie(NULL);
35. int n,q;
36. cin>>n>>q;
37. int A[200005];
38. for(int i=0;i<n;i++)
39. {
40. cin>>A[i];
41. }
42. for(int i=n;i<2\*n;i++)
43. {
44. A[i]=A[i-n];
45. }
46. for(int i=0;i<2\*n;i++)
47. {
48. v[A[i]].pb(i);
49. }
50. while(q--)
51. {
52. int mindiff=200000;
53. int a,b;
54. cin>>a>>b;
55. for(int i=0;i<v[a].size();i++)
56. {
57. //cout<<v[a][i]<<" ";
58. int y=bs(v[b].size(),b,v[a][i]);
59. int val=(v[b][y]-v[a][i]);
60. mindiff=min(mindiff,mod(val));
61. if(y!=0)
62. mindiff=min(mindiff,mod(v[a][i]-v[b][y-1]));
63. }
64. cout<<mindiff/2<<endl;
65. }
66. }