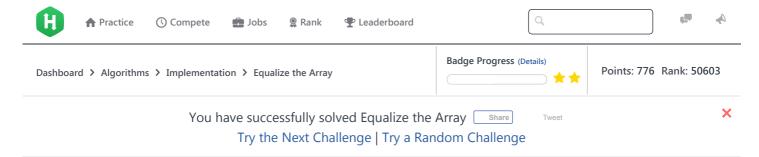
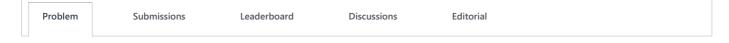
1/14/2018 HackerRank



# Equalize the Array





Karl has an array of n integers defined as  $A = a_0, a_1, \ldots, a_{n-1}$ . In one operation, he can delete any element from the array.

Karl wants all the elements of the array to be equal to one another. To do this, he must delete zero or more elements from the array. Find and print the *minimum* number of deletion operations Karl must perform so that all the array's elements are equal.

#### **Input Format**

The first line contains an integer, n, denoting the number of elements in array A.

The next line contains n space-separated integers where element i corresponds to array element  $a_i$  (0 < i < n).

#### **Constraints**

- $1 \le n \le 100$
- $1 \le a_i \le 100$

## **Output Format**

Print a single integer denoting the minimum number of elements Karl must delete for all elements in the array to be equal.

# **Sample Input**

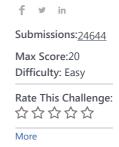
5 3 3 2 1 3

# **Sample Output**

2

### **Explanation**

Array A = [3, 3, 2, 1, 3]. If we delete  $a_2 = 2$  and  $a_3 = 1$ , all of the elements in the resulting array, A' = [3, 3, 3], will be equal. Deleting these 2 elements is minimal because our only other options would be to delete 4 elements to get an array of either [1] or [2]. Thus, we print 2 on a new line, as that is the minimum number of deletions resulting in an array where all elements are equal.



1/14/2018 HackerRank

```
C++14
 1 ▼ #include <bits/stdc++.h>
 2 using namespace std;
 4 int main()
 5 ▼ {
 6
       int size; cin >> size;
       multiset<int> arr;
 7
 8
 9
       for(int i=0; i<size; ++i)</pre>
10
       {
            int temp=0; cin>>temp;
11
12
            arr.insert(temp);
       }
13
14
15
       int count=1, Max_Fre=0;
       // Here itr through out the multi-set, but skip the first position
16
17
       for(auto itr=arr.begin(); itr!=arr.end(); ++itr)
18
           if(itr!=arr.begin())
19 ▼
              int a=*(--itr); // itr, step back one position.
20
              int b=*(++itr); // itr, move forward to current position(forward).
21
22
              (a==b) ? ++count : count=1;
23
             Max_Fre=max(count, Max_Fre);
24
             // find the frequency of the element which is maximum.
25
          }
26
       // print the no. of elements which = frequency of maximum occurred element
27
       cout<<size-Max_Fre<<endl;</pre>
       return 0;
28
29
    }
30
                                                                                                                Line: 30 Col: 1
                      Test against custom input
                                                                                                     Run Code
                                                                                                                  Submit Code
1 Upload Code as File
                                        Congrats, you solved this challenge!
                                               Challenge your friends: f y in

✓ Test Case #0

                                                       ✓ Test Case #1
                                                                                                ✓ Test Case #2
               ✓ Test Case #3
                                                        ✓ Test Case #4
                                                                                                ✓ Test Case #5

✓ Test Case #6

✓ Test Case #7

                                                                                                ✓ Test Case #8

✓ Test Case #9

                                                       ✓ Test Case #10
                                                                                                ✓ Test Case #11

✓ Test Case #14

              ✓ Test Case #12
                                                       ✓ Test Case #13
              ✓ Test Case #15
                                                       ✓ Test Case #16
                                                                                                           Next Challenge
```

Contest Calendar | Blog | Scoring | Environment | FAQ | About Us | Support | Careers | Terms Of Service | Privacy Policy | Request a Feature