**Maximum repeating number**

[array](http://www.practice.geeksforgeeks.org/tag-page.php?tag=array&isCmp=0)[Amazon](http://www.practice.geeksforgeeks.org/tag-page.php?tag=Amazon&isCmp=1)

Given an array of size *n*, the array contains numbers in range from 0 to *k-1* where *k* is a positive integer and *k <= n.* Find the maximum repeating number in this array. For example, let *k* be 10 the given array be *arr[]*= {1, 2, 2, 2, 0, 2, 0, 2, 3, 8, 0, 9, 2, 3}, the maximum repeating number would be 2 if there are two or more maximum repeating numbers print the element having least value (*See*last sample case for the *more clarifications*). Expected time complexity is*O(n)*and extra space allowed is *O(1)*.

**Input:**

The first line of input contains an integer T, denoting the number of test cases. First line of each test case contain n and k .

Next line contains n integers of array.

**Output:**

Print the output for each test case in a seprare line.

**Constraints:**

1<=T<=100

1<=N<=100

0<=k<=N

0<=a[i]<=k-1  
  
**Example:**

**Input:**

2  
4 3  
2 2 1 2  
6 3  
2 2 1 0 0 1

**Output:**

2  
0

\*\*For More Examples Use Expected Output\*\*

<http://www.practice.geeksforgeeks.org/problem-page.php?pid=593>

#include <iostream>

#include <stdio.h>

#include <map>

using namespace std;

int main() {

    int t;

    scanf("%d", &t);

    while(t--) {

       int n,k;

       scanf("%d %d", &n, &k);

       int arr[n];

       for(int i =0; i<n; i++) {

          scanf("%d", &arr[i]);

       }

       std::map<int,int> m;

       for(int i =0; i<n; i++) {

          m[arr[i]]++;

       }

       int max\_veces=0;

       int min\_num=INT\_MAX;

       for(std::map<int,int>::iterator it = m.begin(); it != m.end(); it++) {

           if(it->second > max\_veces) {

              max\_veces = it->second;

              min\_num = it->first;

           }

           if(it->second == max\_veces) {

               if(it->first < min\_num) {

                  min\_num = it->first;

                }

           }

       }

       cout << min\_num << endl;

    }

  // system("pause");

 return 0;

}