Problem Statement **Suggest Edit**

You are given an array/list of integers of length ‘N’, you are supposed to find all the elements that occur strictly more than floor(N/3) times in the given array/list.

Input Format :

```
The first line contains a single integer ‘T’ denoting the number of test cases.
The test cases follow.

The first line of each test case contains a single integer ‘N’ denoting the number
of elements in the array.

The second line contains ‘N’ single space-separated integers denoting the elements
of the array/list.
```

Output Format :

```
For each test case, print all the majority elements separated by a single space.

The output of every test case will be printed in a separate line.

You may return the majority elements in any order.
```

Note :

```
You don’t need to print anything; It has already been taken care of.
```

Constraints :

```
1 <= T <= 100
3 <= N <= 5000
1 <= ARR[i] <= 10^5

Where ‘T’ denotes the number of test cases, ‘N’ denotes the number of elements in
the array/list and ARR[i] denotes the i-th element of the array/list.

Time Limit: 1 sec
```

Sample Input 1 :

```
2
7
3 2 2 1 5 2 3
```

```
5
7 4 4 9 7
```

Sample Output 1:

```
2
4 7
```

Explanation Of Sample Input 1:

```
In the first test case, floor(N/3)=floor(7/3) is equal to 2, and 2 occurs 3 times
which is strictly more than N/3. No other element occurs more than 2 times.

In the second test case, floor(N/3)=floor(5/3) is equal to 1, and 4 and 7 both
occur 2 times. No other element occurs more than once.
```

Sample Input 2:

```
2
6
1 2 4 4 3 4
4
6 6 6 7
```

Sample Output 2:

```
4
6
```

Explanation Of Sample Input 2:

```
In the first test case,  floor(N/3)=floor(6/3) is equal to 2, and 4 occurs 3 times
which is strictly more than N/3. No other element occurs more than 2 times.
In the second test case,  floor(N/3)=floor(4/3) is equal to 1, and 6 occurs 3
times. No other element occurs more than once.
```