Check for Majority Element in a sorted array

> 给定数组arr，指定元素ele，确认ele是否是arr中的主要元素——即ele在arr中是否出现超过 n / 2次

查找类题目，看到有序数组，自然而然想到二分法

要求查询指定的ele是否在数组中出现了超过n / 2次，其实只要找到ele在数组中第一次出现的位置，记为firstOccurrence，然后判断是否满足如下关系：

```
(firstOccurrence + N / 2) < N) && (arr[firstOccurrence + N / 2] == ele)
```

满足上式的判断条件，则ele是majority元素

使用二分法查找ele的首次出现位置时的判断条件如下：

如果mid == 0 || ele > arr[mid - 1]且arr[mid] == ele，此时mid为ele第一次出现的位置

```
#include <bits/stdc++.h>

using namespace std;

int binarySearch(vector<int> &arr, int low, int high, int &ele) {
    if (high >= low) {
        int mid = low + (high - low) / 2;
        if ((!mid || ele > arr[mid - 1]) && arr[mid] == ele) return mid;
        else if (ele > arr[mid]) return binarySearch(arr, mid + 1, high, ele);
        else return binarySearch(arr, low, mid - 1, ele);
    }
    return -1;
}

int checkMajority(vector<int> &arr, int &N, int &ele) {
    int firstOccurrence = binarySearch(arr, 0, N - 1, ele);
    if (firstOccurrence == -1) return false;
    return ((firstOccurrence + N / 2) < N) && (arr[firstOccurrence + N / 2] == ele);
}


int main() {
    int T;
    scanf("%d", &T);
    while (T--) {
        int N, ele;
        scanf("%d %d", &N, &ele);
        vector<int> arr(N);
        for (int i = 0; i < N; ++i) scanf("%d", &arr[i]);
        checkMajority(arr, N, ele) ? printf("TRUE\n") : printf("FALSE\n");

    }
}
```