1. **specifications**

An array A has a majority element if more than half of its values are the same. Determine whether A has a majority element and, if so, report its value.

Note: your algorithm should *not* assume that the elements can be ordered (so sorting is not an option). However, comparison of two elements to determine whether they are the same is possible in constant time.

2. **examples**

3. **size**

4. **targets**

brute force – Th(n^2)

use a Map to store counts – Th(n), assuming hash table O(1) access counts

5. **tactics**

Note: your algorithm should *not* assume that the elements can be ordered (so sorting is not an option). However, comparison of two elements to determine whether they are the same is possible in constant time.

6. **approaches**

process input – split the array in half, friends find majority elt if there is one in the first half and the second half, we figure out if there's a majority elt overall (and what it is)

produce output – n/a

narrow the search space – n/a

7. **generalize / define subproblems**

original problem: majority(A) – majority element in A, if it exists

subproblem: majority(A,low,high) – majority element in A[low..high] (inclusive) if it exists

8. **base case(s)**

n=0 → no majority

n = 1 → majority is A[low] (low=high)

9. **main case**

majority(A,low,high)

split A: mid ← (low+high)/2

majority1 ← majority(A,low,mid)

majority2 ← majority(A,mid+1,high)

report ?? as the majority

- neither majority1 nor majority 2 exist – no majority in low..high
- either majority1 or majority2 exists, but not both – if there is a maj, it's the one reported – but it depends on how many times it occurs in the other half
- majority1 and majority2 exist, same –
- majority1 and majority2 exist, different –

10. **top level**
    a) **initial subproblem**
    b) **setup**
    c) **wrapup**

11. **special cases**

12. **algorithm**

13. **termination**
    a) **making progress**
    b) **reaching the end**

14. **correctness**
    a) **establish the base case(s)**
    b) **show the main case**
    c) **final answer**

15. **implementation**

16. **time and space**