Date
Page

☆ Majority element

☆ Given an array, return an a majority element.
→ Majority element appears >= n/2

☆ Hashing

→ use map to store occurence of all elements
→ check which element's occurence is >= n/2, return that element.

T.C. = O(n)
S.C = O(n)

☆ Sorting

T.C. = O(n log n)
S.C = O(1)

☆ Moore's voting algorithm

☆ Approach

→ let's understand approach through example.

example

arr-1 4 4 3 4 2 4 4 3 2 4

count = ~~0~~ ~~1~~ ~~2~~ ~~1~~ ~~2~~ ~~1~~ ~~2~~ ~~3~~ ~~2~~ ~~1~~ 2
element = ~~0~~ 4

```
    if(count = =0)
        element = arr[i]

    if (element = = arr[i])
        count++;
    else
        count--;
```

After the traversal, element is our answer.

$\boxed{\text{Answer} = 4}$

code :

```
    int cnt =0, ele =0;

    for(int i=0; i<size; i++) {

        if (cnt = =0)
            ele = arr[i];

        if (ele= =arr[i]) cnt++;
        else cnt--;

    }
```

☆ Using Bit Manipulation (Bit-mask)

array = 3 3 2 2 1 1 3 3 3

representation in binary

```
0 1 1 —
0 1 1 —
0 1 0
0 1 0
0 0 1
0 0 1
0 1 1
0 1 1
--------->
8 6 6 > 8/2
— — —
0 1 1 = 3 → majority
             element
```

count set bits
if (> n/2)
then majority element
pos set bit

$Tc = O(N \log_2 N)$
$Sc = O(1)$