MAJORITY ELEMENT - II

☆ Majority element where number occuring more than N|2 times was to be returned used Moore's Voting Algorithm. Here we will try to return elements which occur more than N|3 times.

Brute force solution is to traverse through the array and check for each element occurs more than N|3 times. Time complexity is $O(N^2)$ in worst case

Better solution is to use a hashmap. In first iteration we store each element's no. of occurrences. While doing this we check if the count in the hashmap is exceeding 3 and if it does, we add it to the answer array. Time complexity is O(N) but we have space complexity as well now : O(N).

Optimal Solution is using Moore's Voting Algorithm. We just change it to such that instead of 1 counter, we will work with two counters.

Pseudocode :

```
majorityElement (arr , N)  {
   count1 = 0 , count2 = 0
   el1 = -1 , el2 = -1
   for (i = 0 → N)  {
       if (count1 == 0  && arr[i] != el2) {
           count1 = 1
           el1 = arr[i]
       } else if (count2 == 0  &&
                                          arr[i] != el1) {
           count2 = 1
           el2 = arr[i]
       } else if (arr[i] == el1) {
           count1 += 1
       } else if (arr[i] == el2) {
           count2 += 1
       } else {
           count1 -= 1
           count2 -= 1
       }
   }
}
```