Majority Element (Leetcode 169)

Trivial solution is using unordered map.

Code.

```
int majorityElement (vector <int> & nums)
    unordered_map <int, int> umap;
    for ( int x : nums){
        umap[x]++;
        if (umap[x] >  nums.size()/2 ){
            return x;
        }
    return 0;
```

Other solution is using Moore voting algorithm.

Traverse the array with a counter which increases if the element is same. ~~Turn to 0 if counter~~ Decreases till count is 0.

last remaining element will be majority element.

This will work because we are given that majority element definitely exists.

Code

```
int majorityElement (vector <int> nums) {
        int count = 1;
        int major = nums [0];

        for ( int i = 1; i < nums.size() ; i++) {
            if (major == nums[i]) {
                count++;
            }
            else if (count == 0) {
                major = nums[i];
            }
            else {
                count--
            }
        }
        return major;
}.
```

Time complexity - O(n)

space complexity - O(1).