Method 3 :

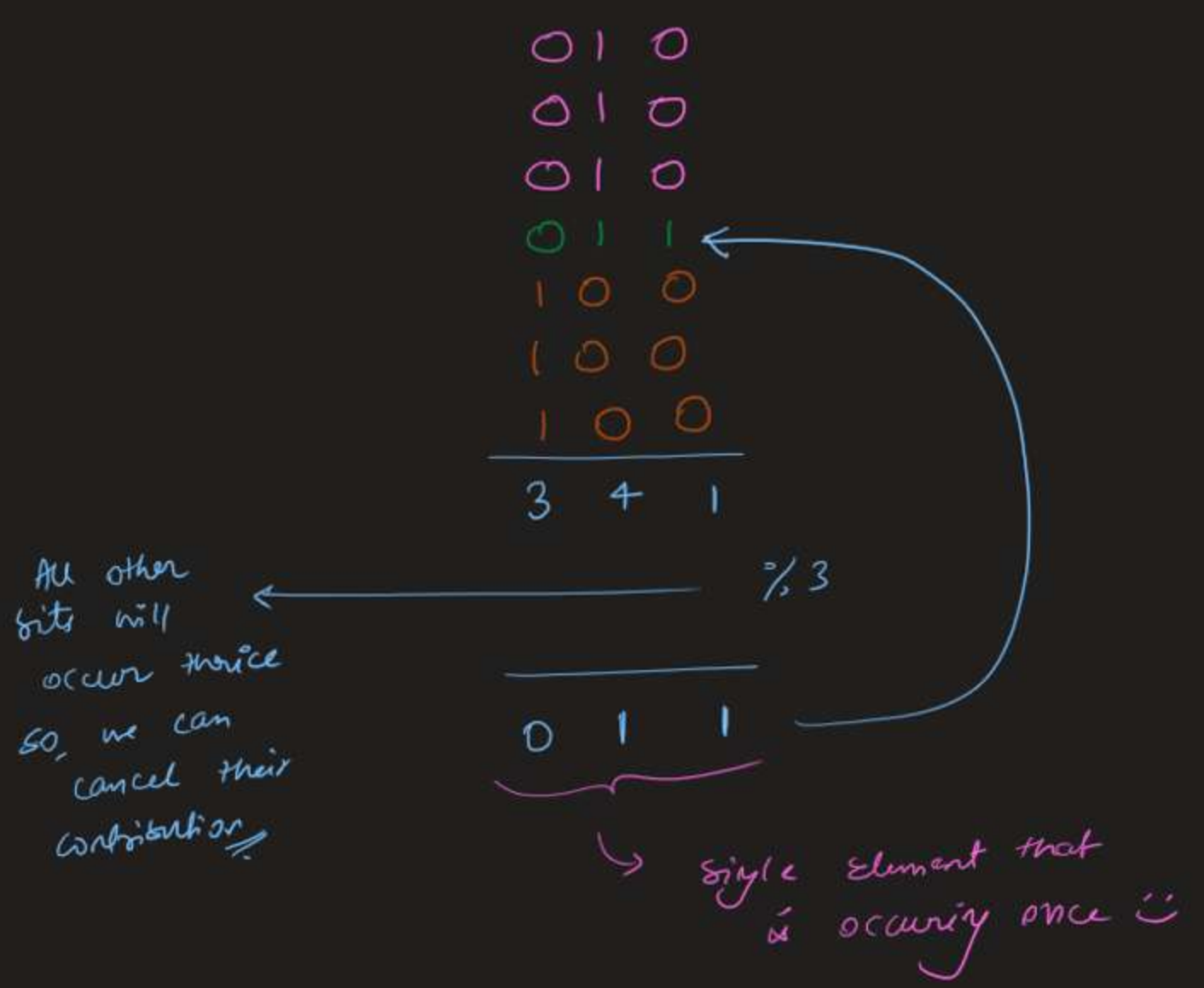
- $1 <= nums.length <= 3 * 10^4$
- $-2^{31} <= nums[i] <= 2^{31} - 1$

As we see the numbers are in INT_MAX range → i.e. Numbers will at max be 32 bit Integers

↓

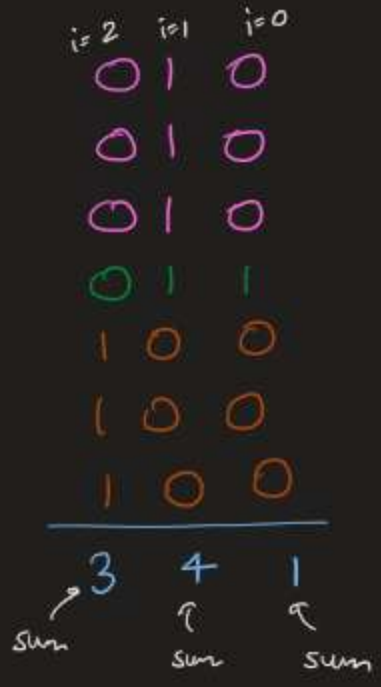
then why can't we think in bits

[2 2 2 3 4 4 4]


0 1 0
0 1 0
0 1 0
0 1 1
1 0 0
1 0 0
1 0 0
3 4 1
%3
0 1 1
All other bits will occur thrice so, we can cancel their contribution
Single Element that is occuring once

```cpp
class Solution {
public:
    int singleNumber(vector<int>& nums) {
        int ans = 0;
        for(int i = 0; i < 32; i++) {
            int sum = 0;
            for(int j = 0; j < nums.size(); j++) {
                if(((nums[j] >> i) & 1) == 1) {
                    sum++;
                    sum %= 3;
                }
            }
            if(sum != 0) {
                ans |= sum << i;
            }
        }
        return ans;
    }
};
```

Truncating last bit i times to get to ith bit



Making number back from bits :> ans = 0 1 1

time: O(32 * N)

space: O(1)