Find non duplicate number in an array

Posted on April 21, 2020 | by Prashant Yadav

Posted in Algorithms, Arrays | Tagged Easy

Learn how to find the non duplicate number from the given array.

Given an array of n integers, every number in it appears twice expect for one. we have to find the one with single occurrence.

Example

```
Input:
[2, 1, 2]
[3, 5, 5, 6, 6]

Output:
1
3
```

There are two ways of solving this problem

1. By using nested loops to check the frequency of array elements, but it will take O(n ^ 2) which is not so efficient.

2. We iterate all the elements of the array and keep track of their count using an hash map and then return the element with single occurrence. This is a more optimized way because it will O(n) time but we will also need O(n) extra space to keep the elements counts.

Find non duplicate number in an array

If you read the problem statement carefully then you will find out that it is not mentioned that the count of the number will not repeated, for example

```
Input:
[2, 2, 1, 1, 1]

Output:
1
```

In this case `1` was repeated thrice so actually as per the question it is twice plus once. So we have to return the `1`.

Implementation

- Iterate all the elements of the array and keep the track of the count using a hashmap.
- Find the element with odd count in the hashmap and return it.

We will be using javascript objects as a hash map.

```
const singleNumber = (nums) => {
    //Hashmap
    const track = {};

    //Count the frequency
    for(let i = 0; i < nums.length; i++){
        if(track[nums[i]]){
            track[nums[i]]++;
        }else{
            track[nums[i]] = 1;
        }
    }

    //Return the element with odd frequency
    for(let key in track){
        if(track[key] % 2 !== 0){
            return key;
        }
    }
};
```

Time complexity: O(n + n) => O(n).

Space complexity: O(n).

We can highly reduce the lines of code by using the new ES6 features.

```
const singleNumber = (nums) => {
    //Hashmap
    const track = nums.reduce((a, b) => {
      a[b] ? a[b]++ : a[b] = 1;
      return a;
    }, {});

    //Return the element with odd count
    return Object.keys(track).filter(e => track[e] % 2 !== 0)[0];
};
```

Copy

We are using array reduce method to aggregate the elements count in the array and then returning the element with odd frequency by filtering the hash map elements.

Recommended Posts:

Reverse a string using stack

Count number of sub-string occurrence in a string

Iterative heap sort in Javascript

Next permutation problem

Program to print the pyramid pattern

Difference between square of sum of numbers and sum of square of numbers.

Program to check palindrome linked list

Linear search algorithm in javascript

Flatten binary tree to linked list

Maximum Collatz sequence under 1000000

Prev Next

Handcrafted with ♥somewhere in **Mumbai**