Missing Number

Given an array containing *n* distinct numbers taken from `0, 1, 2, ..., n`, find the one that is missing from the array.

For example,
Given *nums* = `[0, 1, 3]` return `2`.

Note:
Your algorithm should run in linear runtime complexity. Could you implement it using only constant extra space complexity?

Credits:
Special thanks to @jianchao.li.fighter for adding this problem and creating all test cases.

Solution 1

```
class Solution {
public:
    int missingNumber(vector<int>& nums) {
        int result = nums.size();
        int i=0;

        for(int num:nums){
            result ^= num;
            result ^= i;
            i++;
        }

        return result;
    }
};
```

There are several similar problems in the problem list.

written by CodingKing original link here

Solution 2

The basic idea is to use XOR operation. We all know that a^b^b =a, which means two xor operations with the same number will eliminate the number and reveal the original number. In this solution, I apply XOR operation to both the index and value of the array. In a complete array with no missing numbers, the index and value should be perfectly corresponding(nums[index] = index), so in a missing array, what left finally is the missing number.

```
public int missingNumber(int[] nums) {

    int xor = 0, i = 0;
    for (i = 0; i < nums.length; i++) {
        xor = xor ^ i ^ nums[i];
    }

    return xor ^ i;
}
```

written by m010 original link here

Solution 3

1.XOR

```
public int missingNumber(int[] nums) { //xor
    int res = nums.length;
    for(int i=0; i<nums.length; i++){
        res ^= i;
        res ^= nums[i];
    }
    return res;
}
```

2.SUM

```
public int missingNumber(int[] nums) { //sum
    int len = nums.length;
    int sum = (0+len)*(len+1)/2;
    for(int i=0; i<len; i++)
        sum-=nums[i];
    return sum;
}
```

3.Binary Search

```
public int missingNumber(int[] nums) { //binary search
    Arrays.sort(nums);
    int left = 0, right = nums.length, mid= (left + right)/2;
    while(left<right){
        mid = (left + right)/2;
        if(nums[mid]>mid) right = mid;
        else left = mid+1;
    }
    return left;
}
```

Summary:

If the array is in order, I prefer `Binary Search` method. Otherwise, the `XOR` method is better.

written by mingjun original link here