题目:

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.

Subscribe to see which companies asked this question.

1.时间：O（N）; 空间：O（N）　->使用额外空间

```
class Solution {
public:
    int missingNumber(vector<int>& nums) {
        if (nums.empty()) return 0;

        std::unordered_map<int, int> hashTable;
```

```
        /* 每个元素最多出现一次 */
        for (int i = 0; i < nums.size(); ++i)
            hashTable[nums[i]] = 1;
        /* 检测缺失的元素 */
        for (int i = 0; i < nums.size() + 1; ++i){
            if (hashTable.find(i) == hashTable.end())
                return i;
        }

        return 0;
    }
};
```

2.时间：O（N）；空间：O（1）　->0~n 相加-数组中的数，缺点：可能会溢出

3.时间：O（N）；空间：O（1）

```
class Solution {
    /* 利用 0^a = a, a ^ b ^ a = b */
    /* 1.从所有 0~n 的数异或一遍；
    2.将异或后的数把数组中的数异或一遍，这样出现过两次的数都被置 0，剩下的就是
0^missing num = missing num;
    */
public:
    int missingNumber(vector<int>& nums) {
```

```
        if (nums.empty()) return 0;


        int result = nums.size();
        for (int i = 0; i < nums.size(); ++i){
            result ^= (i ^ nums[i]);
        }


        return result;
    }
};
```

4.时间：O（N）；空间：O（1）

```
class Solution {
    /* 将数字 x 放在位置 i 上，最后扫描一次，如果 num[i] != i，那么 i 即是所求，
        注意，例如数据：[1,2,3,4,5]，这里 5 超出数组下标上界，需要额外处理*/
public:
    int missingNumber(vector<int>& nums) {
        if (nums.empty()) return 0;


        for (int i = 0; i < nums.size();){
            int& num = nums[i];
            if (num >= nums.size()){
                ++i;
```

```
                continue;
            } else if (num != i){
                std::swap(num, nums[num]);
            } else{
                i++;
            }
        }

        for (int i = 0; i < nums.size(); ++i){
            if (nums[i] != i) return i;
        }

        /* 在[0,1,2,3,4]这种情况时，返回 nums.size() */
        return nums.size();
    }
};
```