> 给一个长度为 n 的数组，其索引应该在[0,n)，但是现在你要装进去 n + 1 个元素[0,n]，那么肯定有一个元素装不下嘛，请你找出这个缺失的元素。

算法： 利用异或的性质，一个数异或其自身结果为0, 异或0为其自身。

```
class Solution {
public:
  int missingNumber(std::vector<int>& nums) {
    int len = nums.size();
    int res = 0;
    res ^= len;

    for (int i = 0; i < len; i++) {
      res ^= i ^ nums[i];
    }

    return res;
  }
};
```

进阶

> 给一个长度为N的数组nums，其中本来装着[1..N]这N个元素，无序。但是现在出现了一些错误，nums中的一个元素出现了重复，也就同时导致了另一个元素的缺失。请你写一个算法，找到nums中的重复元素和缺失元素的值。

```
class Solution {
public:
  std::vector<int> findErrorNums(std::vector<int>& nums) {
    int n   = nums.size();
    int dup = -1;
    for (int i = 0; i < n; i++) {
      int index = abs(nums[i]) - 1;
      if (nums[index] < 0) {
        // 重复
        dup = abs(nums[i]);
      } else {
        nums[index] *= -1;
      }
    }

    int missing = -1;
    for (int i = 0; i < n; i++) {
      if (nums[i] > 0) {
        missing = i + 1;
```

```
        }
      }

      return std::vector<int>{dup, missing};
    }
};
```

对这种数组问题，关键点在于索引和元素成对出现，常用处理为排序、异或和映射。