找到所有数组中消失的数字

https://leetcode.cn/problems/find-all-numbers-disappeared-in-an-array/solutions/601946/zhao-dao-suo-you-shu-zu-zhong-xiao-shi-d-mabl

方法一：原地修改
思路及解法

我们可以用一个哈希表记录数组 nums 中的数字，由于数字范围均在 [1,n] 中，记录数字后我们再利用哈希表检查 [1,n] 中的每一个数是否出现，从而找到缺失的数字。

由于数字范围均在 [1,n] 中，我们也可以用一个长度为 n 的数组来代替哈希表。这一做法的空间复杂度是 O(n) 的。我们的目标是优化空间复杂度到 O(1)。

注意到 nums 的长度恰好也为 n，能否让 nums 充当哈希表呢?

由于 nums 的数字范围均在 [1,n] 中，我们可以利用这一范围之外的数字，来表达「是否存在」的含义。

具体来说，遍历 nums，每遇到一个数 x，就让 nums[x−1] 增加 n。由于 nums 中所有数均在 [1,n] 中，增加以后，这些数必然大于 n。最后我们遍历 nums，若 nums[i] 未大于 n，就说明没有遇到过数 i+1。这样我们就找到了缺失的数字。

注意，当我们遍历到某个位置时，其中的数可能已经被增加过，因此需要对 n 取模来还原出它本来的值。

代码

```
class Solution {
    public List<Integer> findDisappearedNumbers(int[] nums) {
        int n = nums.length;
        for (int num : nums) {
            int x = (num - 1) % n;
            if (nums[x] <= n) {
                nums[x] += n;
            }
        }
        List<Integer> ret = new ArrayList<Integer>();
        for (int i = 0; i < n; i++) {
            if (nums[i] <= n) {
                ret.add(i + 1);
            }
        }
        return ret;
    }
}
```

复杂度分析

时间复杂度：O(n)。其中 n 是数组 nums 的长度。

空间复杂度：O(1)。返回值不计入空间复杂度。