Problem 41: First Missing Positive

Problem Information

Difficulty: Hard
Acceptance Rate: 41.92%
Paid Only: No
Tags: Array, Hash Table

Problem Description

Given an unsorted integer array `nums`. Return the _smallest positive integer_ that is _not present_ in `nums`.

You must implement an algorithm that runs in `O(n)` time and uses `O(1)` auxiliary space.

Example 1:

Input: nums = [1,2,0] **Output:** 3 **Explanation:** The numbers in the range [1,2] are all in the array.

Example 2:

Input: nums = [3,4,-1,1] **Output:** 2 **Explanation:** 1 is in the array but 2 is missing.

Example 3:

Input: nums = [7,8,9,11,12] **Output:** 1 **Explanation:** The smallest positive integer 1 is missing.

Constraints:

* `1 <= nums.length <= 105` * `-231 <= nums[i] <= 231 - 1`

Code Snippets

C++:

```
class Solution {
public:
int firstMissingPositive(vector<int>& nums) {

}
};
```

Java:

```
class Solution {
public int firstMissingPositive(int[] nums) {

}
}
```

Python3:

```
class Solution:
def firstMissingPositive(self, nums: List[int]) -> int:
```