**[Find the Duplicate Number](https://leetcode.com/problems/find-the-duplicate-number/)**

Given an array of integers nums containing n + 1 integers where each integer is in the range [1, n] inclusive.

There is only **one repeated number** in nums, return *this repeated number*.

You must solve the problem **without** modifying the array nums and uses only constant extra space.

**Example 1:**

**Input:** nums = [1,3,4,2,2]

**Output:** 2

**Example 2:**

**Input:** nums = [3,1,3,4,2]

**Output:** 3

**Example 3:**

**Input:** nums = [3,3,3,3,3]

**Output:** 3

**Constraints:**

* 1 <= n <= 105
* nums.length == n + 1
* 1 <= nums[i] <= n
* All the integers in nums appear only **once** except for **precisely one integer** which appears **two or more** times.

**Follow up:**

* How can we prove that at least one duplicate number must exist in nums?
* Can you solve the problem in linear runtime complexity?

class Solution {

public:

int findDuplicate(std::vector<int>& nums) {

int left = 1;

int right = nums.size() - 1;

while (left < right) {

int mid = left + (right - left) / 2;

int count = 0;

// Count the numbers less than or equal to mid

for (int num : nums) {

if (num <= mid) {

count++;

}

}

// If count is greater than mid, the duplicate lies in the left half

if (count > mid) {

right = mid;

} else { // Otherwise, it lies in the right half

left = mid + 1;

}

}

return left;

}

};

Link : <https://leetcode.com/problems/find-the-duplicate-number/submissions/1212809874/?envType=daily-question&envId=2024-03-24>