287. Find the Duplicate Number (Floyd Cycle Detection)

Problem: Given n+1 integers in range [1,n], find the duplicate without modifying array and using O(1) space. Approach: Treat array as linked list and apply Floyd’s Cycle Detection. Algorithm: 1. Use slow = nums[0], fast = nums[0]. 2. Move slow = nums[slow], fast = nums[nums[fast]] until they meet. 3. Reset slow = nums[0]. 4. Move both one step until they meet again. 5. That meeting point is the duplicate. Code (C++):

```
class Solution {
public:
    int findDuplicate(vector<int>& nums) {
        int slow = nums[0];
        int fast = nums[0];
        do {
            slow = nums[slow];
            fast = nums[nums[fast]];
        } while (slow != fast);

        slow = nums[0];
        while (slow != fast) {
            slow = nums[slow];
            fast = nums[fast];
        }
        return slow;
    }
};
```

Dry Run: nums = [1,3,4,2,2] slow = 1, fast = 1 slow = 3, fast = 2 slow = 2, fast = 2 (meet) Reset slow = 1 slow = 3, fast = 4 slow = 2, fast = 2 -> duplicate = 2

Time Complexity: O(n) Space Complexity: O(1)