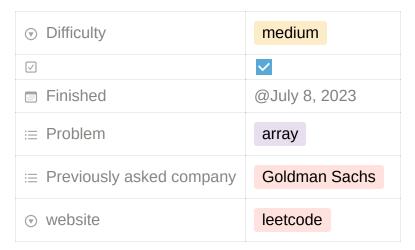
287. Find the Duplicate Number



Question:

Given an array of integers [n] 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
```

Optimal solution:

We cannot use hash map because we have to solve the problem **with** only constant extra space.

We cannot use sorting because we have to solve the problem **without** modifying the array nums

Algorithm used: Floyd's Algorithm to find starting point of cycle in linked list cycle.

Time complexity: O(n)

where n is the number of nodes in the linked list

Space complexity: O(1)

```
class Solution(object):
    def findDuplicate(self, nums):
        slow, fast = 0, 0
        while True:
        slow = nums[slow]
        fast = nums[nums[fast]]
        if slow == fast:
```

break slow2 = 0 while True: slow = nums[slow] slow2 = nums[slow2] if slow == slow2: return slow