Page :
Date : / /

Find the duplicate number (Leetcode)

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

∴ There is ~~atleast~~ one repeated number.
↓
This is the ~~no of~~ no. to be returned

1st approach

Use a hashmap to store ~~all the~~ the data of every value & return the value which is repeated.

Here, time complexity = O(n)
Space complexity = O(n)

But, we cannot use this approach as in the question it is specified that we cannot modify the array & we can use ~~only to~~ only constant extra space.

So

~~Core~~ Correct Approach

First, we have to observe 2 things,

1. We have to observe that it is a linked-list problem
2. We need to know Floyd's Cycle Detection algorithm.

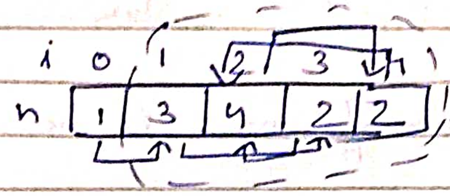
Given

Example 1

len = n+1
nums[i] in [1, n]

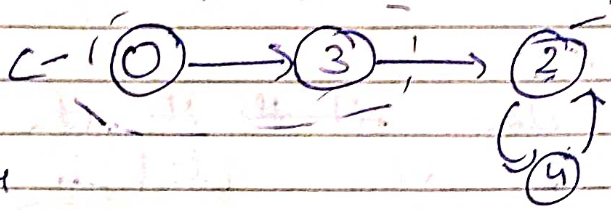
Input : [1, 3, 4, 2, 2]
Output : 2

1—4



this portion forms cycle linked list

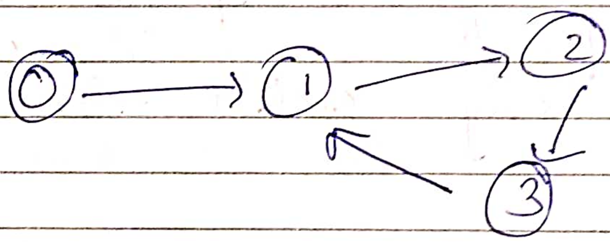
this part is before the cycle



this is the element that forms the circle that needs to be returned.

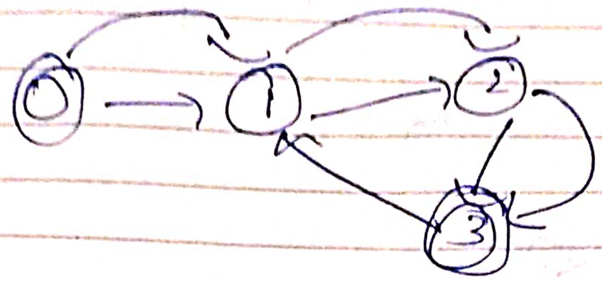
Thus, to find where the cycle starts, we need to use Floyd's Cycle Detection Algorithm.

We take an example.

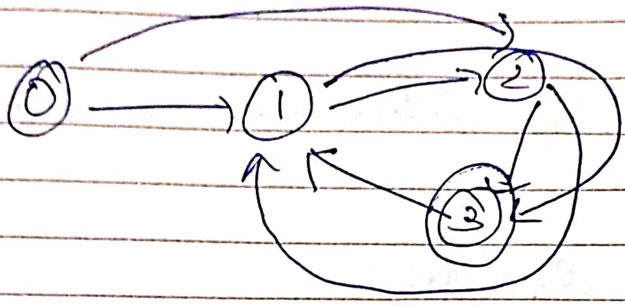


Here, we take 2 pointers. One slow pointer & one fast pointer. Slow pointer is going to make 1 hop at a time & fast pointer is going to make two hops at a time.

First slow pointer.

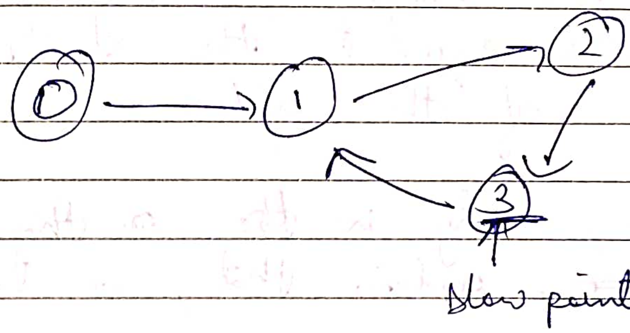


Now fast pointer

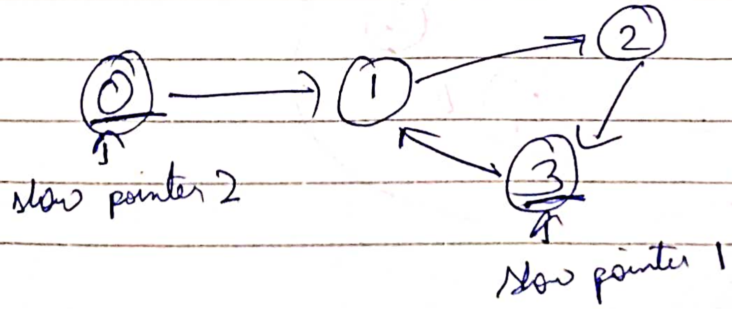


We find the point where both of them first intersect. In this case it is 3.

We leave the slow pointer at this intersection point. We are now ~~at~~ done with the fast pointer.



We ~~to~~ now take a second slow pointer ~~at~~ & put it in the beginning.

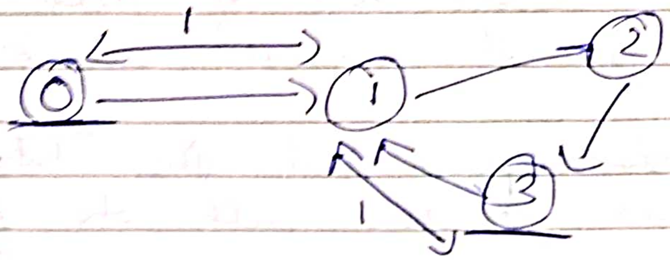


We are now going to shift each one of these slow pointers one by one until they intersect

This is the point of intersection

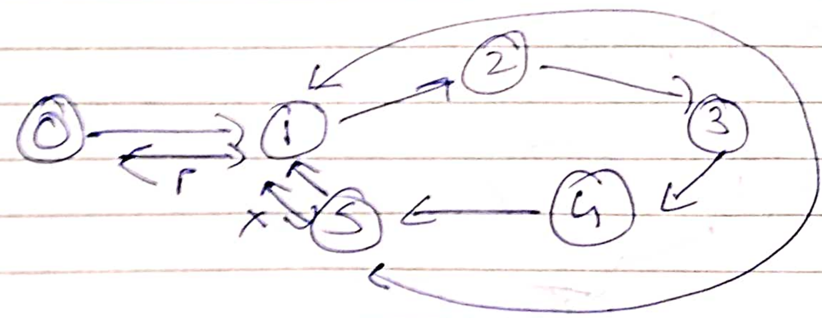
This second point of intersection is the result.

Explanation of the algorithm:



Distance b/w the first intersection & beginning of the cycle is always equal to the starting point to the beginning of the cycle.

Now, the question is, why is ~~the~~ ~~a~~ this the case. To ~~know more~~, explain that, we take a bigger example.



C-x

We know,

$$2 \times \text{slow} = \text{fast}$$
$$= p + C - x + C$$
$$2(p + C - x) = p + 2C - x$$
$$\Rightarrow \cancel{2}p \cancel{+ 2C} - \cancel{2}x = \cancel{p} + \cancel{2C} - \cancel{x}$$
$$\Rightarrow p - x = 1$$
$$\Rightarrow p = x$$

Thus, it is proved.