Find duplicate

arr = [3, 1, 3, 4, 2]

n+1 = size

1 to 4

3 is duplicate

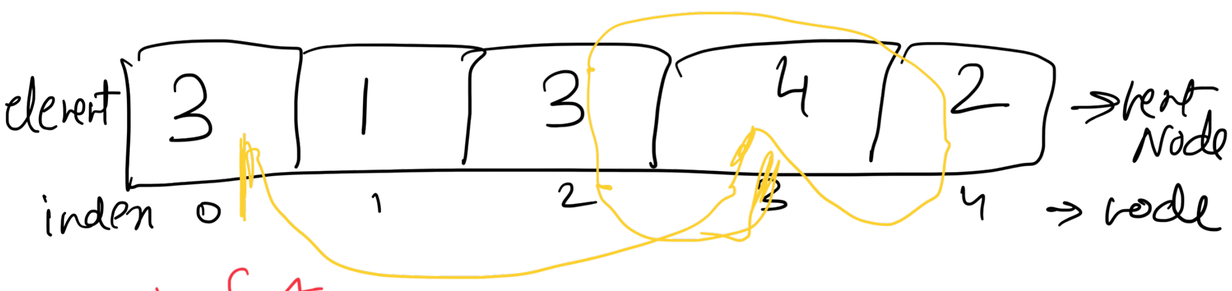
only 1 to n range be values

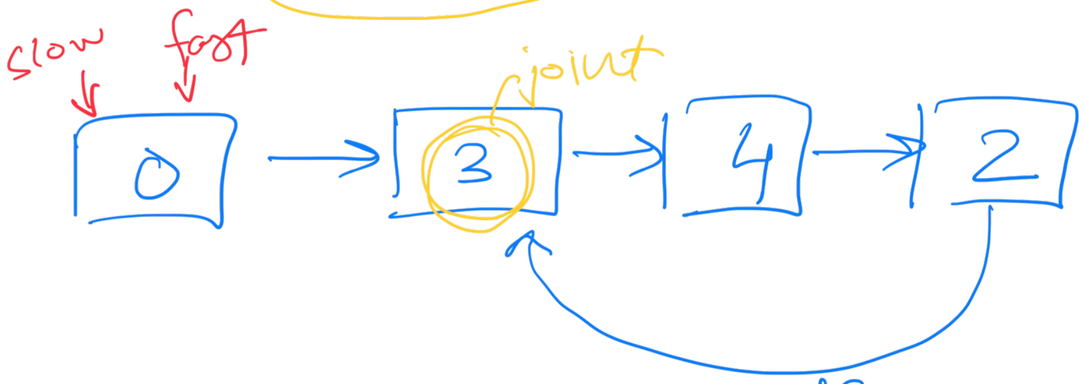
arr = [3, 3, 3, 3, 3] is also valid

not necessary that all 1 to n values need be present

we can use unordered set but we want SC O(1) not O(n)

we will use slow-fast pointer concept





cycle exists because duplicate value exists

slow = arr[0]
fast = arr[0]

(1) slow → +1
fast → +2
till slow = fast

(2) slow = arr[0]
slow → +1
fast → +1
till slow = fast

Pseudo Code

slow = arr[0], fast = arr[0]

we are using do while loop to check s=f but in the start its same only so if we used while it wouldn't enter loop any how that's why do while

```
do {
    slow = arr[slow]      // +1
    fast = arr[arr[fast]] // +2
} while (slow != fast)
slow = arr[0]
while (slow != fast) {
    slow = arr[slow]      // +1
    fast = arr[fast]      // +1
}
return slow
```

TC: O(n) SC: O(1)