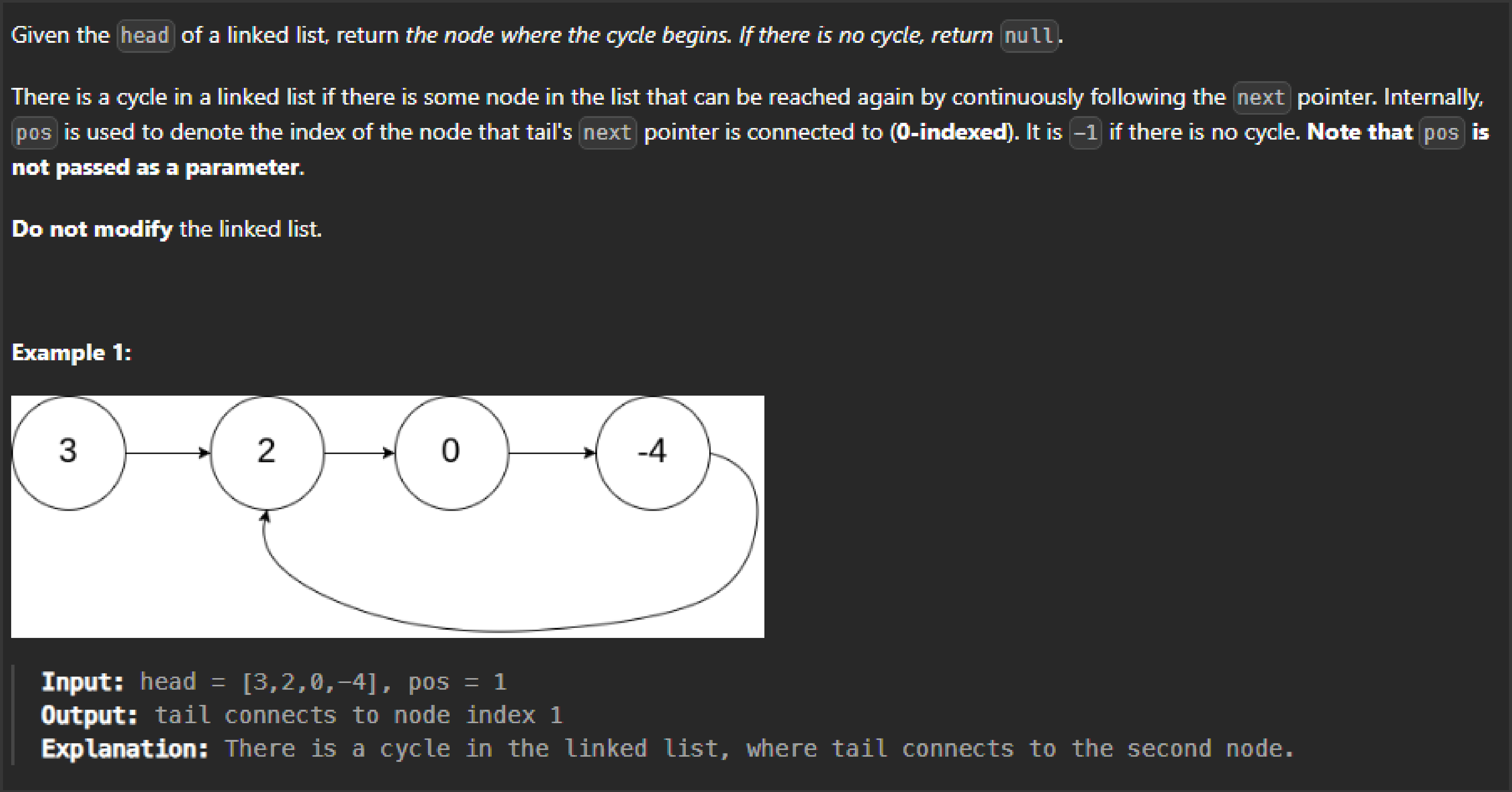
Starting point of loop in linked list



Given the `head` of a linked list, return *the node where the cycle begins. If there is no cycle, return* `null`.

There is a cycle in a linked list if there is some node in the list that can be reached again by continuously following the `next` pointer. Internally, `pos` is used to denote the index of the node that tail's `next` pointer is connected to (**0-indexed**). It is `-1` if there is no cycle. **Note that** `pos` **is not passed as a parameter**.

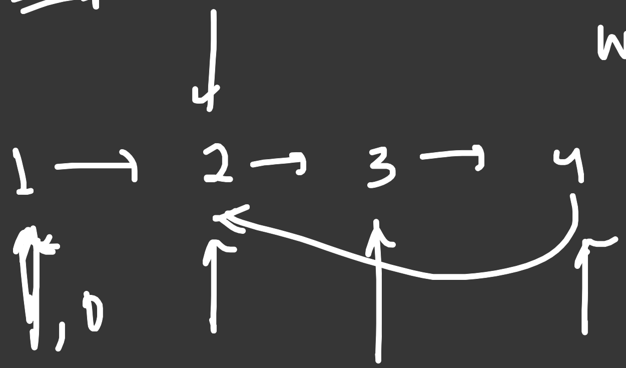
Do not modify the linked list.

Example 1:

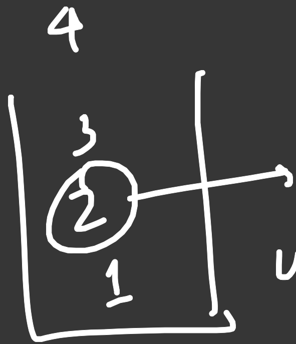
```
Input: head = [3,2,0,-4], pos = 1
Output: tail connects to node index 1
Explanation: There is a cycle in the linked list, where tail connects to the second node.
```

Brute force

Using hash ~~set~~ map to keeps track of the node .



we have return pos



return this

4, ~~3~~ 3

2

3, 1

2, ~~0~~ ⟶| return this

1, 0

while push

T.C = O(N)

S.C = O(N)

Optimal Approach:

Using slow and fast pointer

T.C = $O(N)$

S.C = $O(1)$