Given a linked list, determine if there is a cycle in it.

Cycle → cycle if there is some node in the list that can be reached again by continously following the next pointer. internally pos is used to denote the index of the node that tail's next pointer is connected to.

e.g

$$\begin{tikzcd} 3 \arrow[r] & 2 \arrow[r] & 0 \arrow[r] & 4 \arrow[ll, bend left] \end{tikzcd}$$

true

Pos = 1

hasSeen = {

3 : true

2 : true

0 : true

4 : true

2

}

traverse through the list

return true

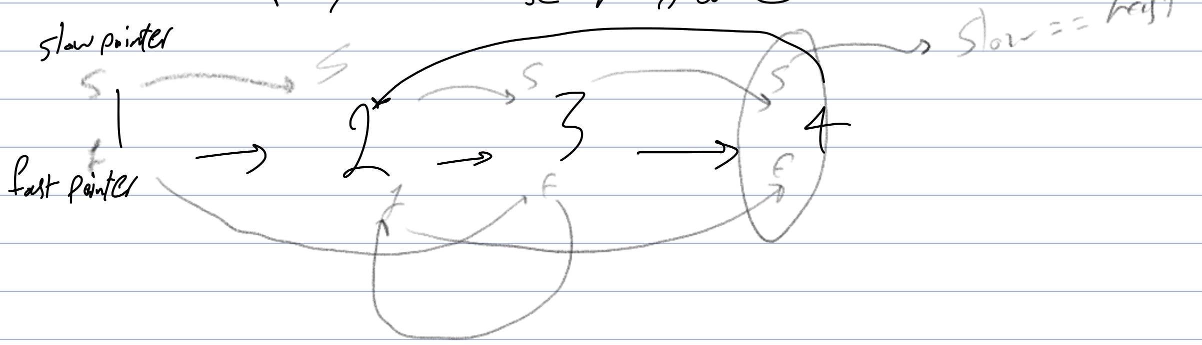
O(N) S

O(N) T

---

the O(1) space solution

Floyd's Tortoise & Hare



```
→ while fast and fast.next
      slow = slow.next
      fast = fast.next.next
      if slow == fast:
          return True
  return False
```