1. Finding the Loop in a Circular Linked List

Floyd's cycle-finding algorithm, also known as ***tortoise*** *and* ***hare*** *algorithm*.

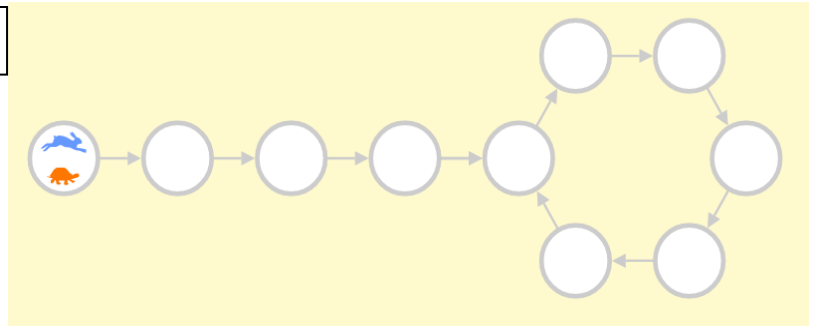
The idea is to have **two references** to the list and move them at **different speeds**.
Move the **first** forward by 1 node and the **second** by 2 nodes.

- If the linked list has a loop they will ***definitely*** meet.
  Proof :

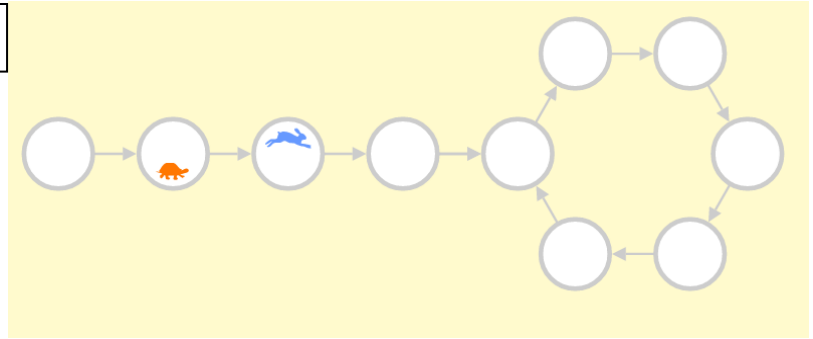
- Else either of the two references(or their next) will become **null**.

Linked List with Cycle (Loop)

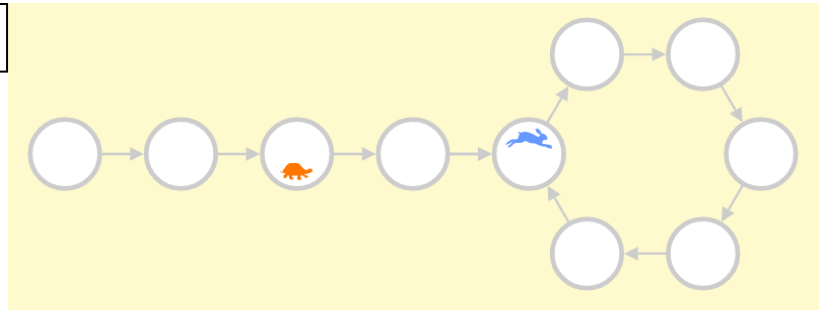
1



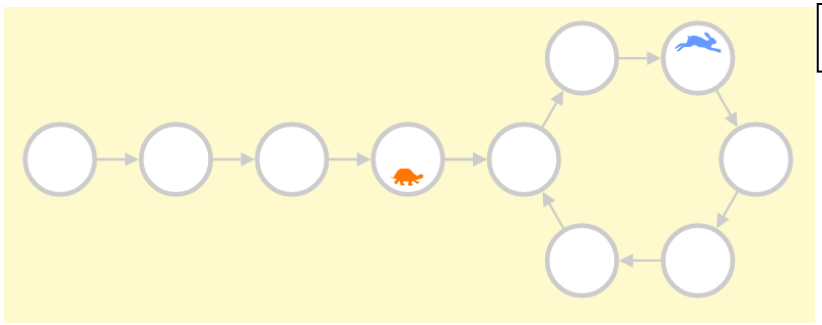
2



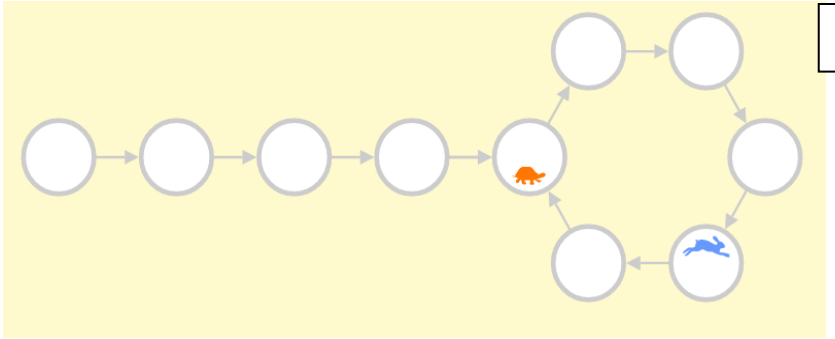
3



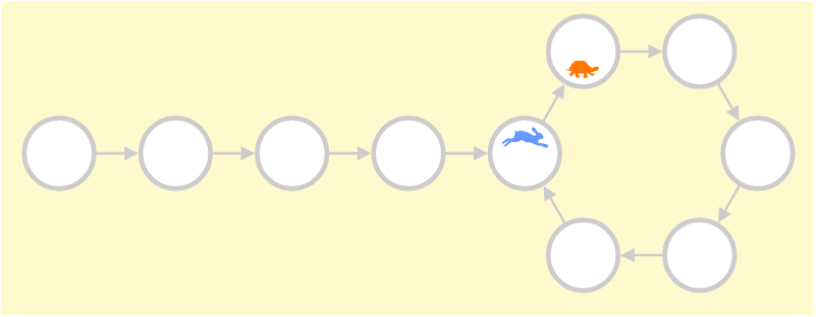
4



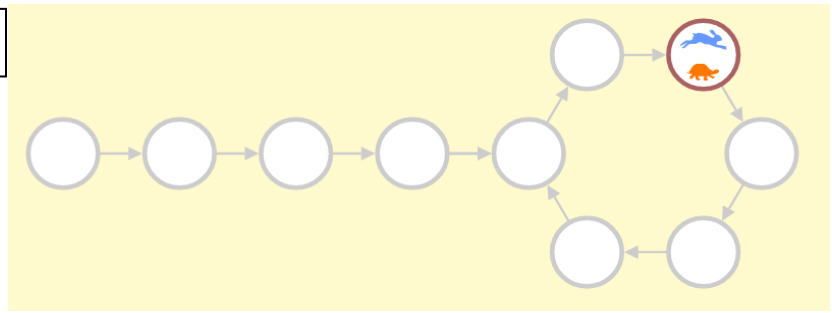
5



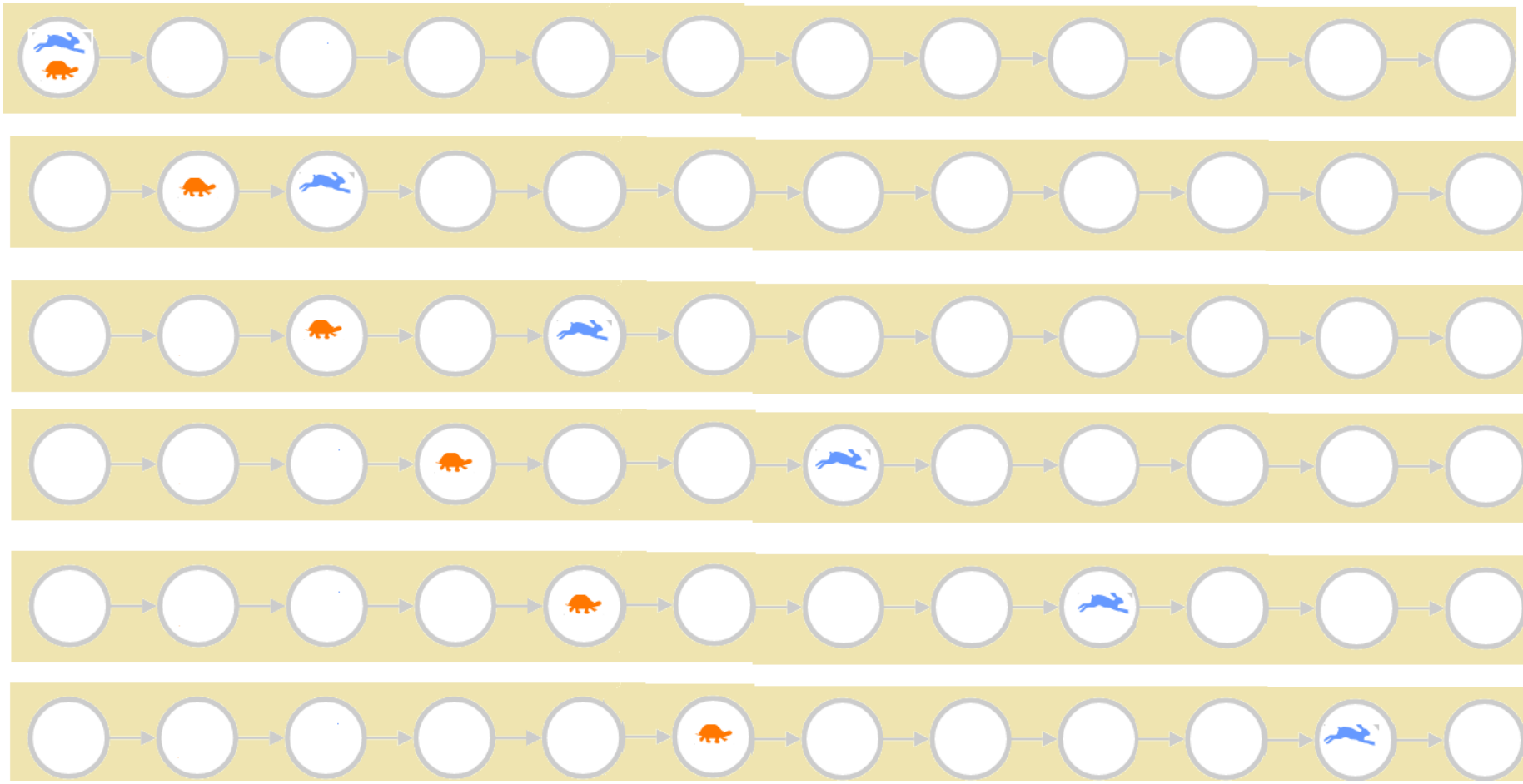
6



7

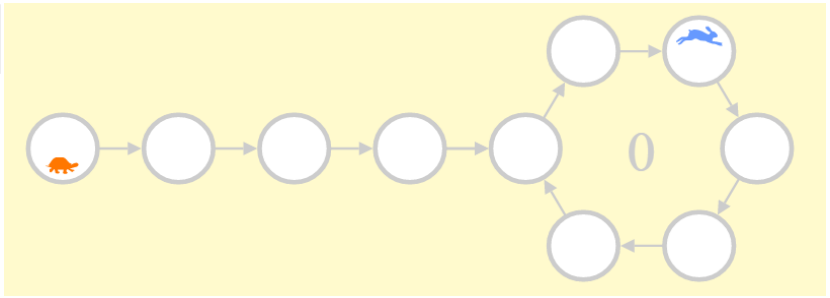


Linked List without Cycle (Loop)

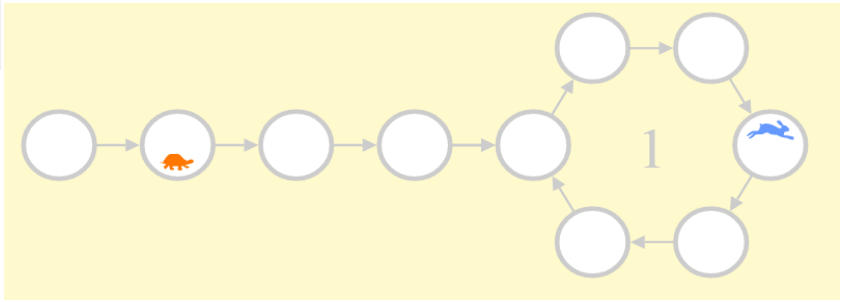


2. Finding the Length of a Loop in a Circular Linked List

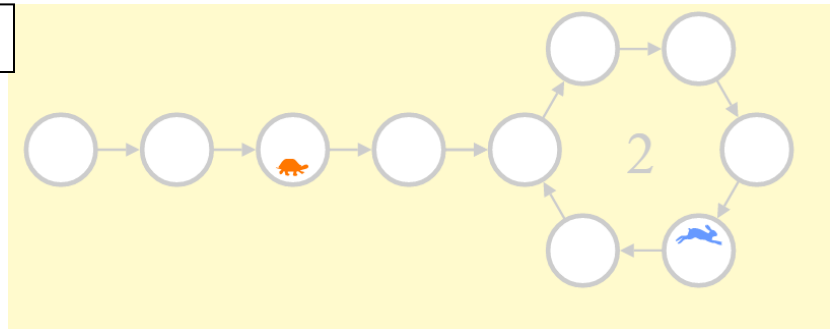
1



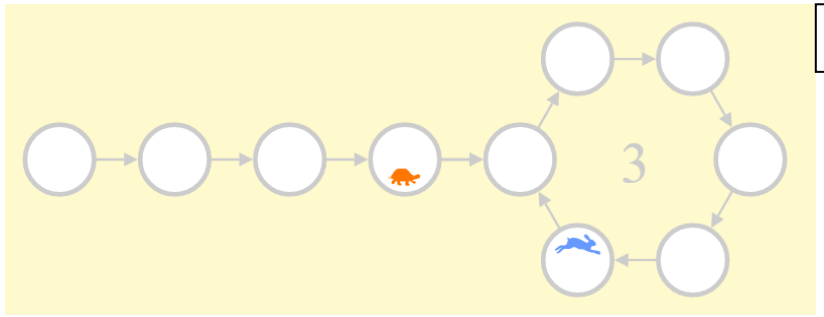
2



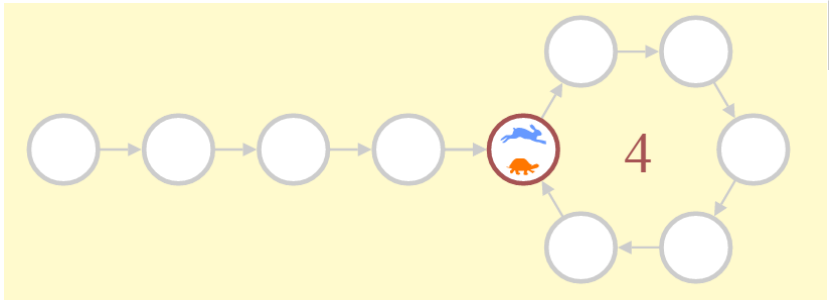
3



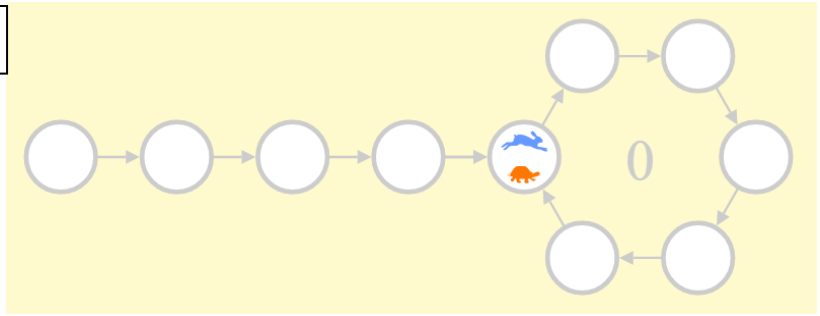
4



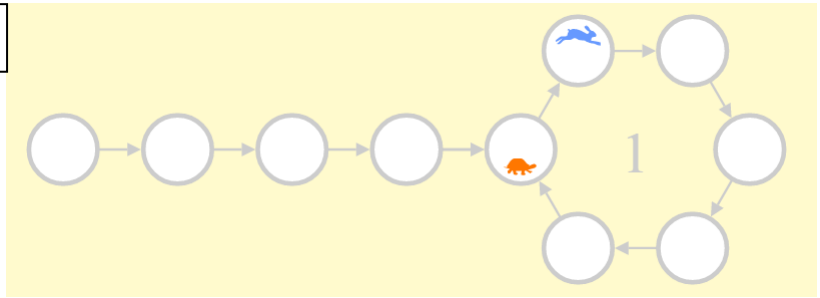
5



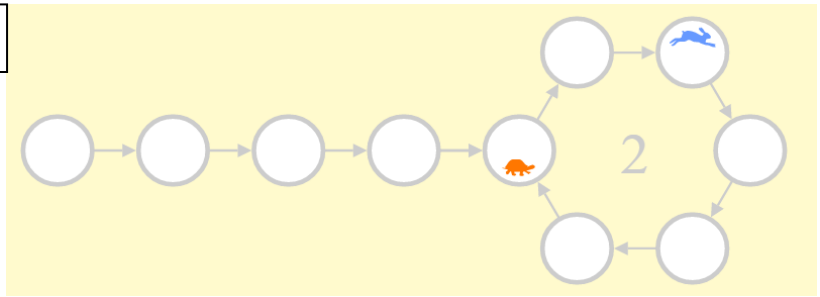
1



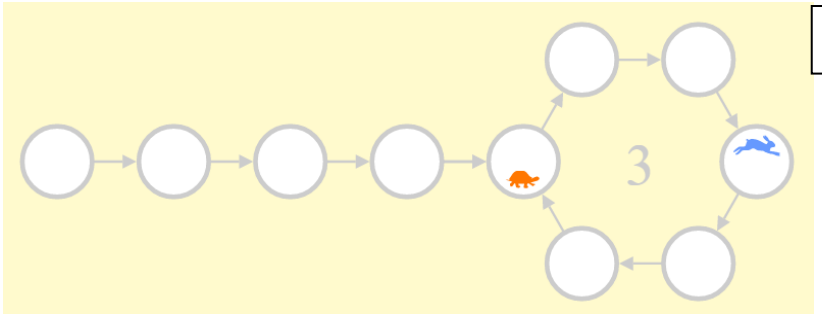
2



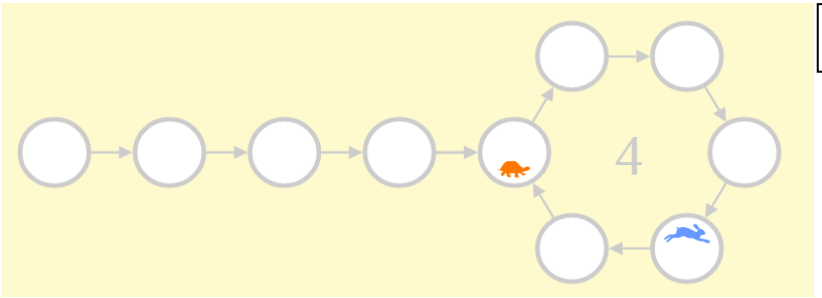
3



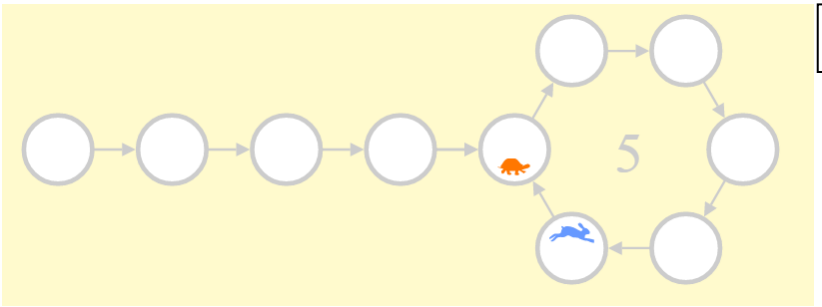
4



5



6



7