Two-Pointer Technique in a Linked List

Last Updated : 23 Jul, 2025

The **two-pointer technique** is a common technique used in linked list problems to solve a variety of problems efficiently. It involves using two pointers that traverse the linked list at different speeds or in different directions.

Use Cases of Two-Pointer Technique in a Linked List:

1. Finding the Middle of a Linked List:

One common application of the two-pointer technique is finding the middle of a linked list. Here's how to do it:

- Initialize both pointers to the head of the linked list.
- Move the fast pointer two nodes at a time, while moving the slow pointer one node at a time.
- When the fast pointer reaches the end of the linked list, the slow pointer will be at the middle node.

2. Finding a Loop in a Linked List:

Another application of the two-pointer technique is detecting loops in a linked list. Here's how to do it:

- Initialize both pointers to the head of the linked list.
- Move the fast pointer two nodes at a time, while moving the slow pointer one node at a time.
- If the fast pointer ever catches up to the slow pointer, there is a loop in the linked list.

3. Reversing a Linked List:

The two-pointer technique can also be used to reverse a linked list. Here's how to do it:

- Initialize three pointers:
  - **Current pointer (curr):** Points to the current node.
  - **Previous pointer (prev):** Points to the previous node.
  - **Next pointer (next):** Points to the next node.
- Set **curr** to the head of the linked list.
- While **curr** is not **null**:
  - Set **next** to the next node of **curr**.
  - Set curr's next pointer to prev**.**
  - Set **prev** to **curr**.
  - Set **curr** to **next**.
- The head of the reversed linked list is now **prev**.

4. Finding the nth Node from the End of a Linked List:

To find the nth node from the end of a linked list using the two-pointer technique, follow these steps:

- Initialize two pointers, **p** and **q**, to the head of the linked list.
- Advance the pointer **q, n** nodes ahead of the pointer **p**.
- Now, advance both pointers **p** and **q** one node at a time.
- When the pointer **q** reaches the end of the linked list, the pointer **p** will be at the nth node from the end.

5. Finding the Intersection Point of Two Linked Lists:

To find the intersection point of two linked lists using the two-pointer technique, follow these steps:

- Initialize two pointers, **p** and **q,** to the heads of the two linked lists, respectively.
- Traverse the first linked list using the pointer **p** until you reach the end.
- Then, set the pointer **p** to the head of the second linked list.
- Similarly, traverse the second linked list using the pointer **q** until you reach the end.
- Then, set the pointer **q** to the head of the first linked list.
- Now, advance both pointers **p** and **q** one node at a time.
- If the two pointers ever meet, then the point of intersection is the node where they meet.
- If the two pointers reach the end of both linked lists without meeting, then there is no intersection point.