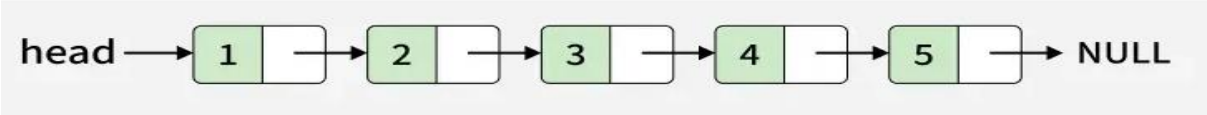
Singly Linked List

A **singly linked list** is a fundamental data structure, it consists of **nodes** where each node contains a **data** field and a **reference** to the next node in the linked list. The next of the last node is **null**, indicating the end of the list. Linked Lists support efficient insertion and deletion operations.

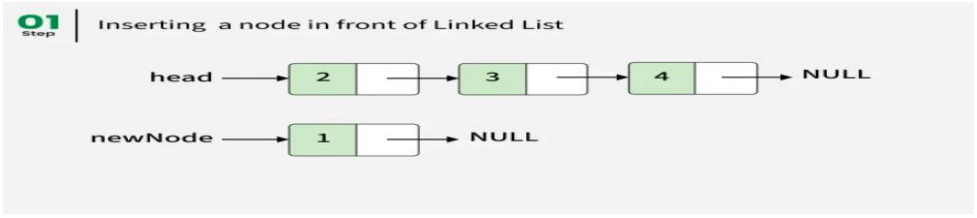


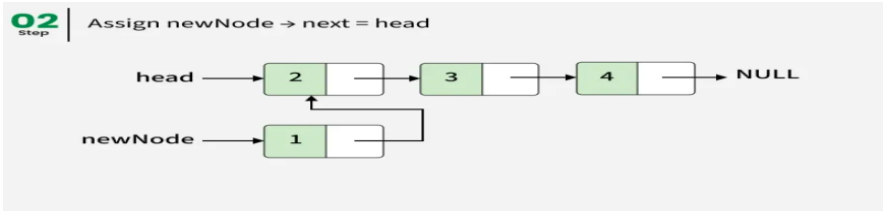
Insert a Node at Front of a Linked List

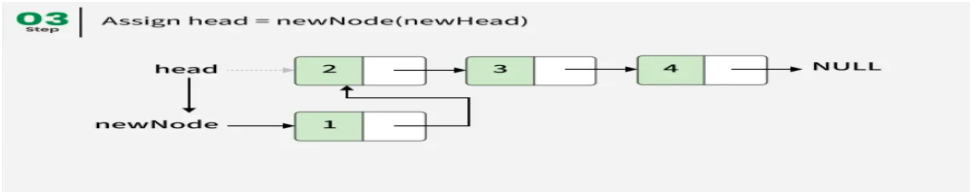
*To insert a new node at the front, we create a **new node** and point its **next** reference to the **current head** of the linked list. Then, we update the **head** to be this **new node**. This operation is efficient because it only requires adjusting a few pointers.*

Algorithm:

- Make the first node of Linked List linked to the new node
- Remove the head from the original first node of Linked List
- Make the new node as the Head of the Linked List.





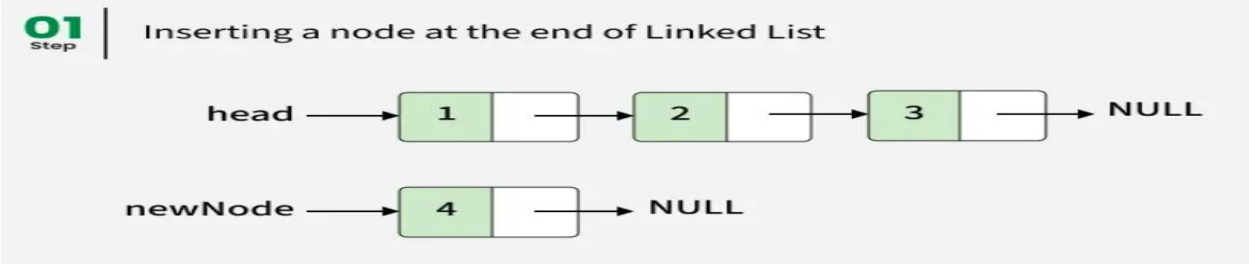


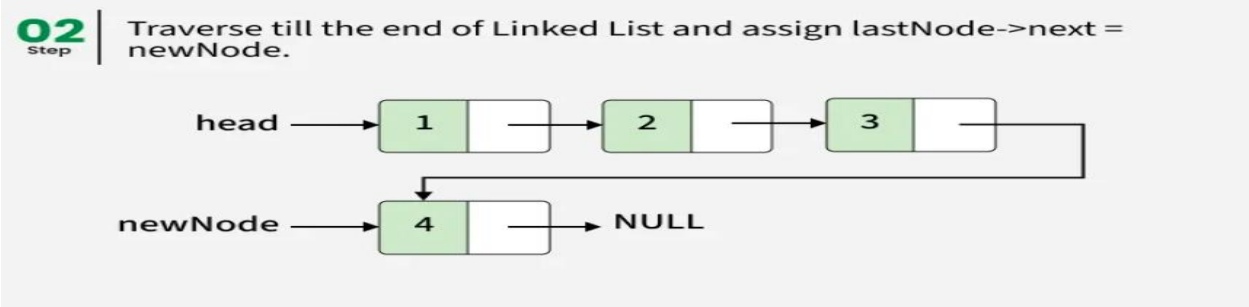
Insert Node at the End of a Linked List

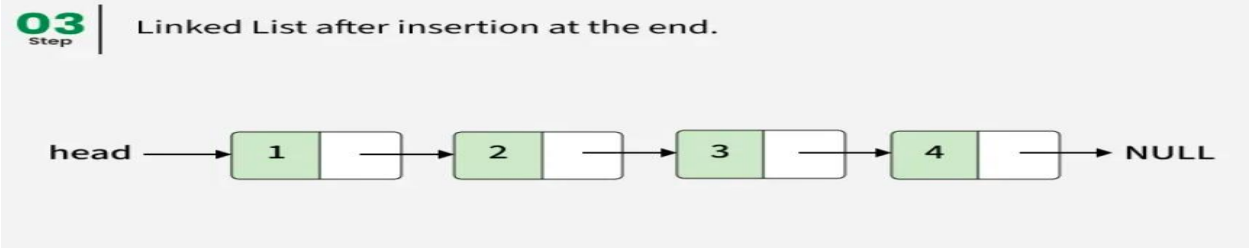
Inserting at the end involves traversing the entire list until we reach the last node. We then set the last node's next reference to point to the new node, making the new node the last element in the list.

Following is the approach to add a new node at the end of the linked list:

- Create a new node and set its next pointer as NULL since it will be the last node.
- Store the head reference in a temporary variable
- If the Linked List is empty, make the new node as the head and return
- Else traverse till the last node
- Change the next pointer of the last node to point to the new node







Insert a node at a specific position in a linked list

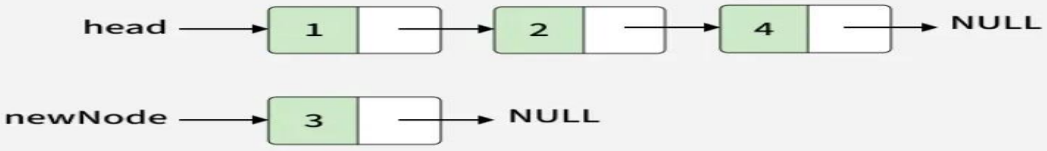
The idea is simple: create a new node, then find the spot where it should be placed. Walk through the list until you reach the node just before that position. Link the new node's next to the following node, and adjust the previous node's next to point to the new node.

Step By Step Implementations:

- Initialize a variable , say curr points to head and allocate the memory to the new node with the given val.
- Traverse the Linked list using curr pointer upto position-1 nodes.
- If curr's next is not null , then next pointer of the new node points to the next of curr node.
- The next pointer of current node points to the new node.
- return the head of linked list

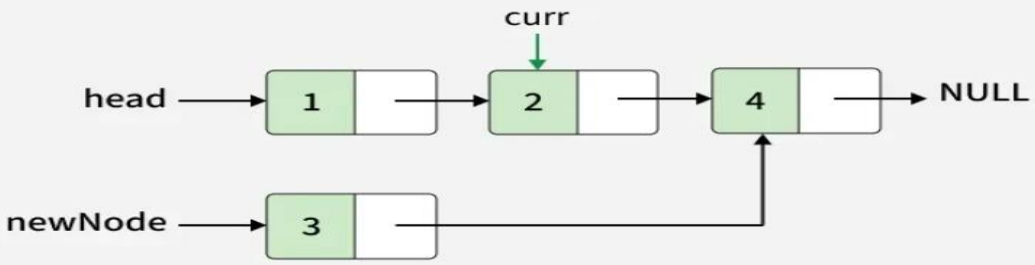
01 Step | Insert a node at a specific position in the linked list.

Pos = 3



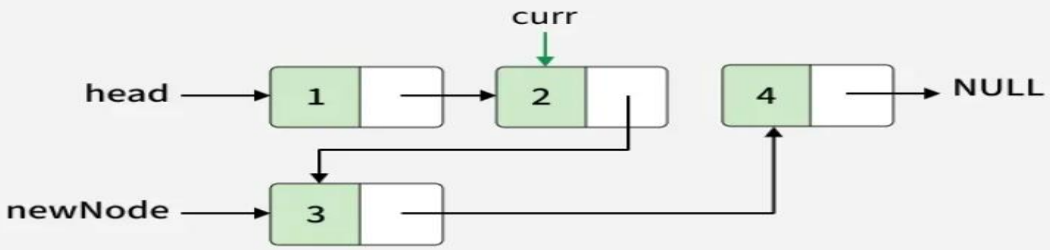
02 Step | Traverse till (position - 1) node using curr pointer and assign newNode→next = curr→next.

Pos = 3



03 Step | Assign curr→next = newNode.

Pos = 3



04 Step | Linked list after insertion at the given position.

Pos = 3