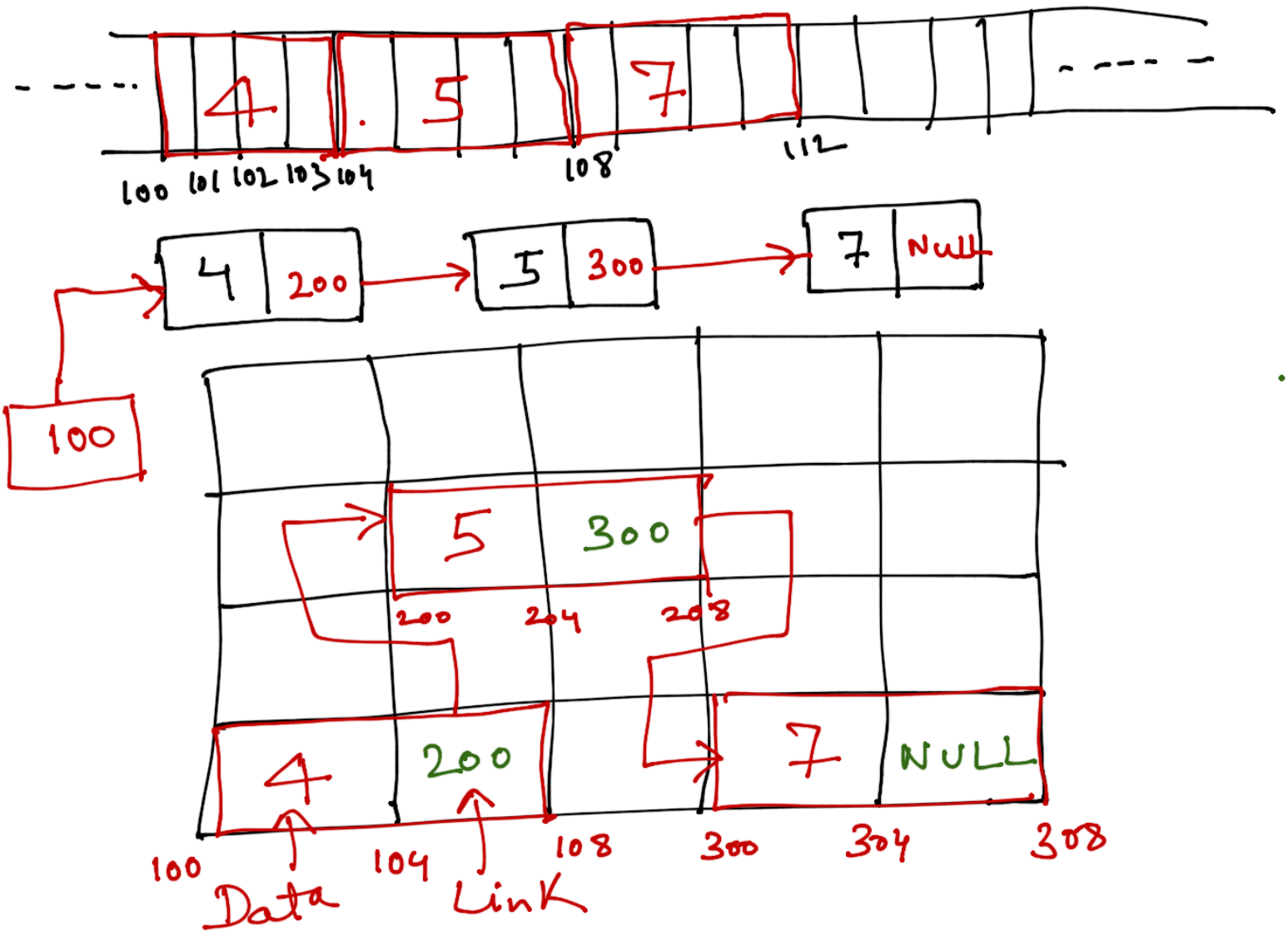
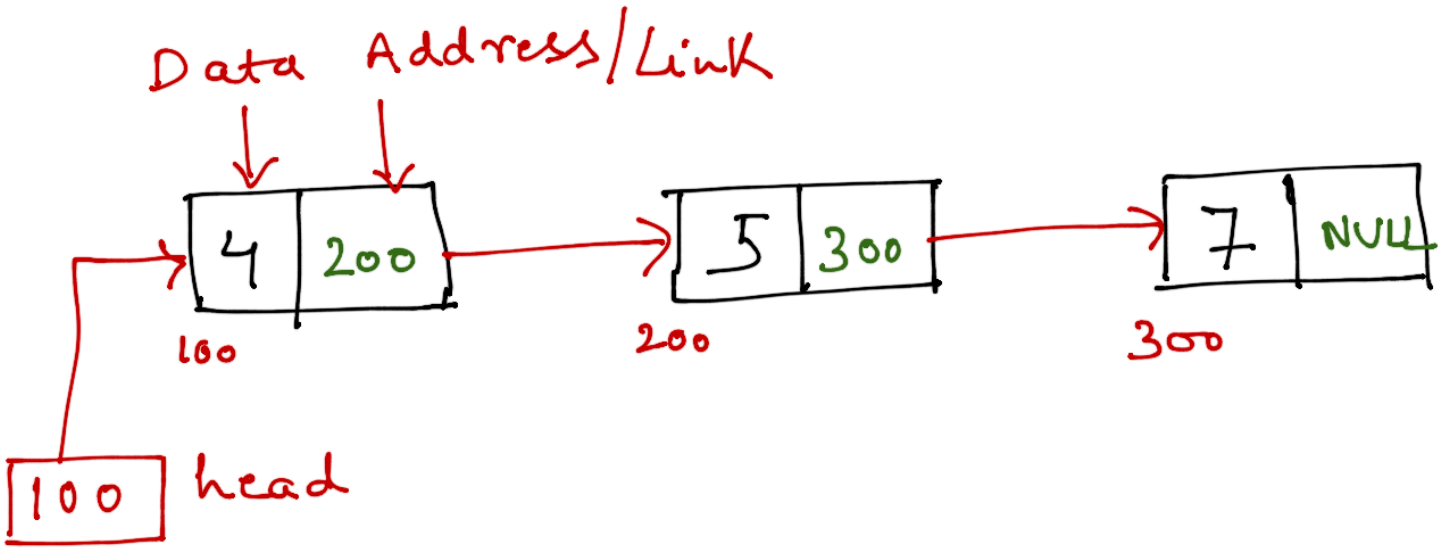
Linked Lists

* Contiguous memory chunk NOT required.



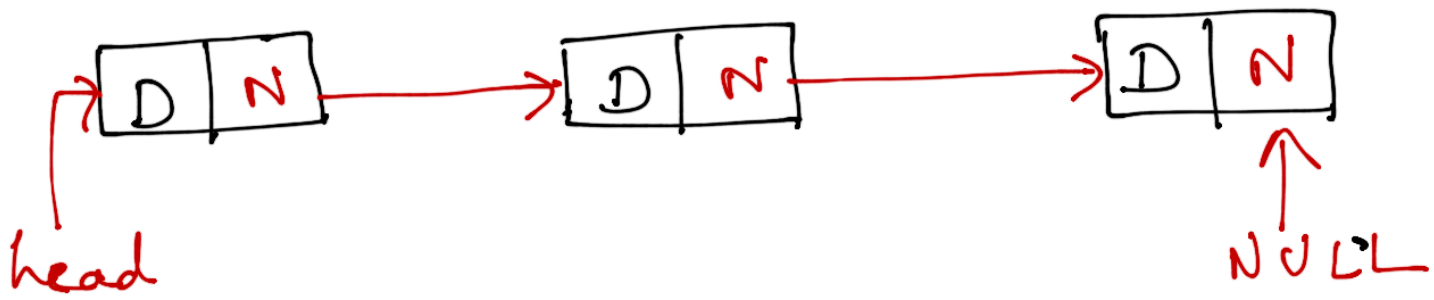


(Address of first node)

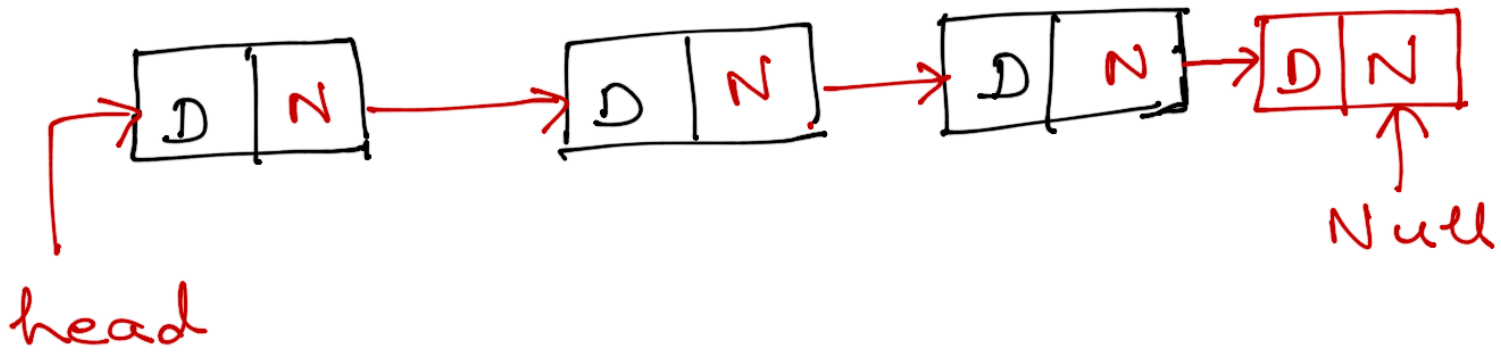
Operations on Linked List

1. Traversing
2. Append a new node (to the end)
3. Prepend a new node (to the start)
4. Insert at a specific position
5. Deleting a node from the list
6. Updating a node in the list

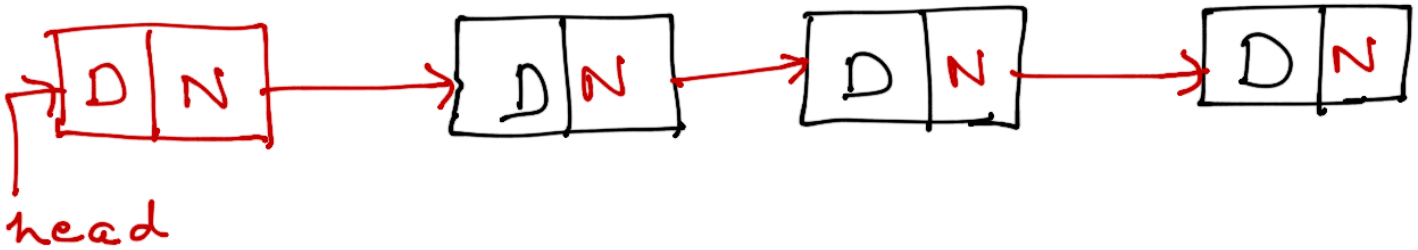
1. Traversing a linked list



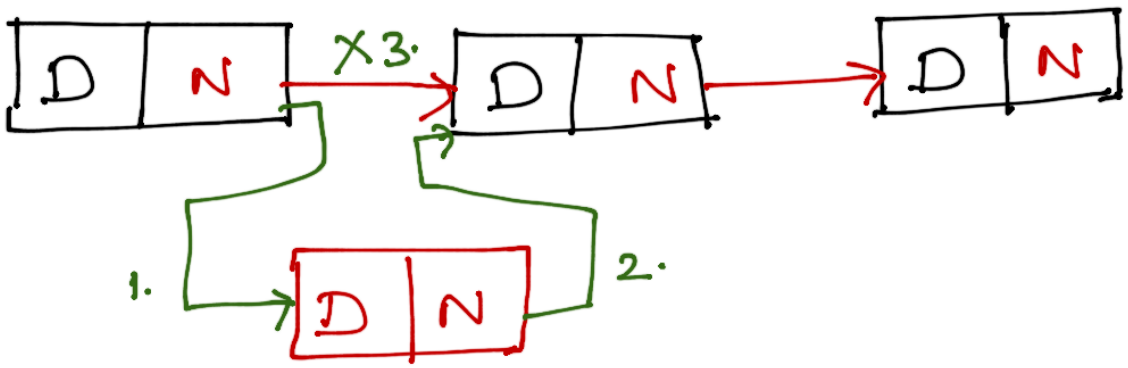
2. Append a new node



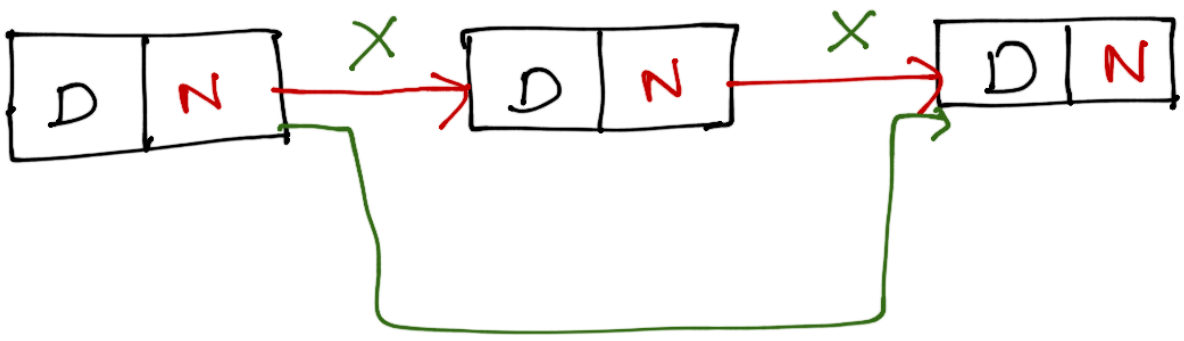
3. Prepend a new node



4. Insert a new node to a Specific position



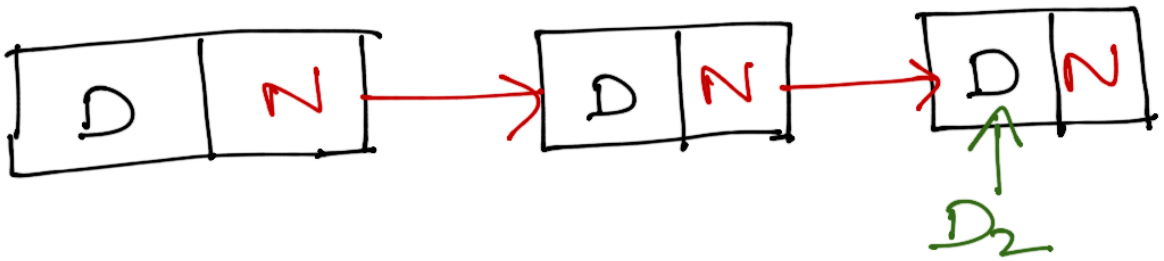
5. Deleting a node



a. If Last node ?

b. If first node ?

6. Updating a node in the list

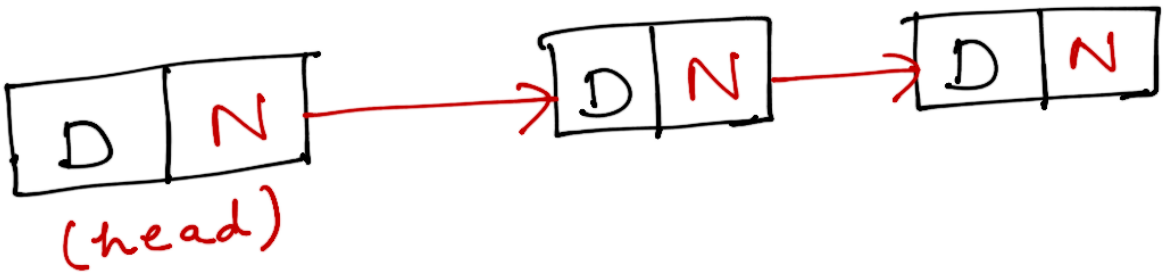


a. Traverse & get to the node

b. Then replace the value

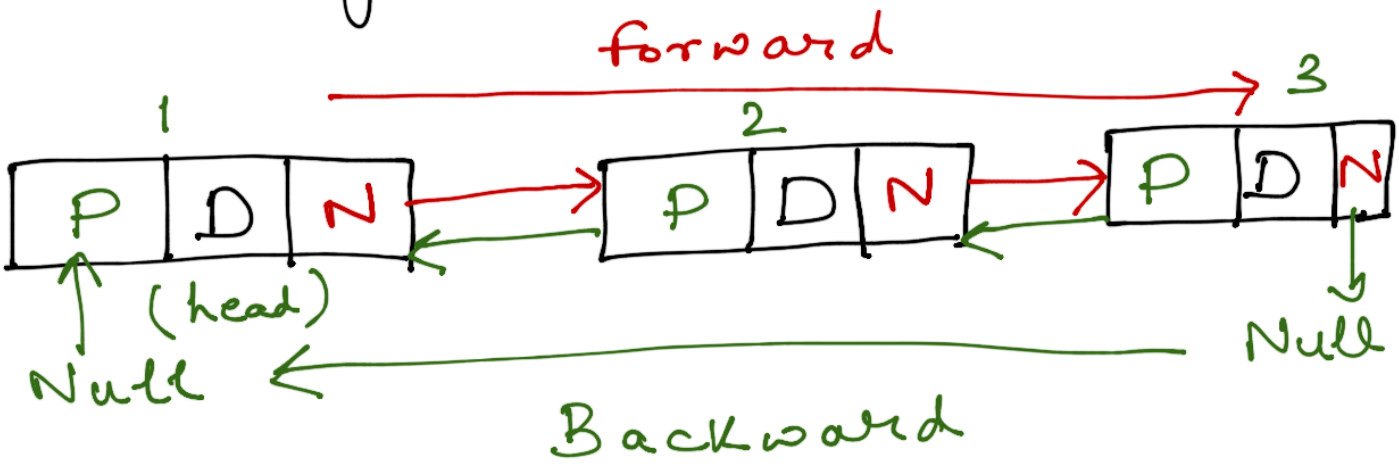
Types of Linked List

1. Singly Linked List

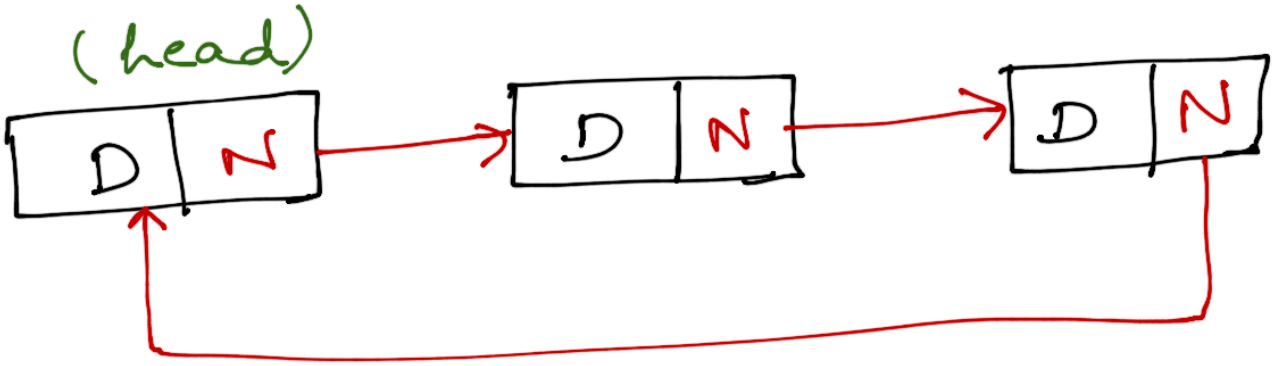


one way to traverse

2. Doubly Linked List



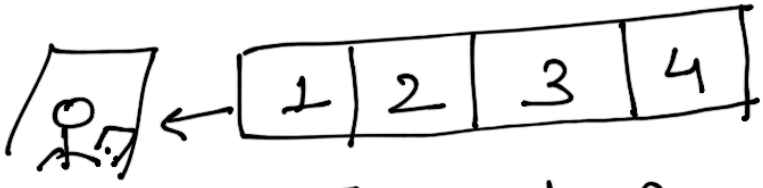
3. Circular Linked List



Applications of Linked List

1. Stack Implementation (LIFO)

2. Queue Implementation (FIFO)



Ticket counter

3. Graphs (Adjacency list representation)

4. Hash Tables → Each Bucket itself is a linked list

5. Undo functionality in Word / Photoshop

6. LRU / MRU

7. Symbol table management in compiler design