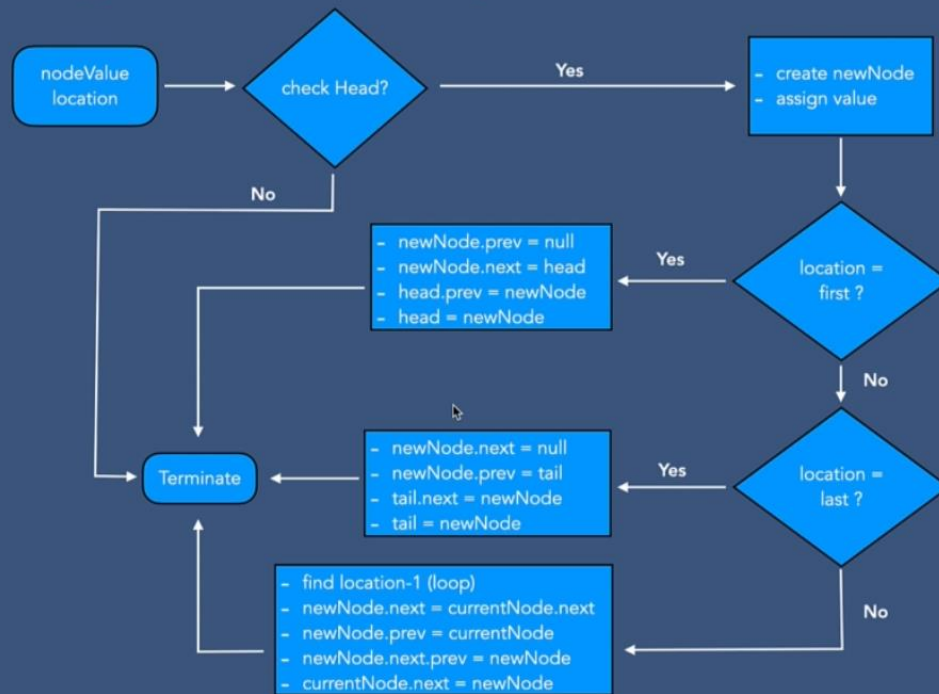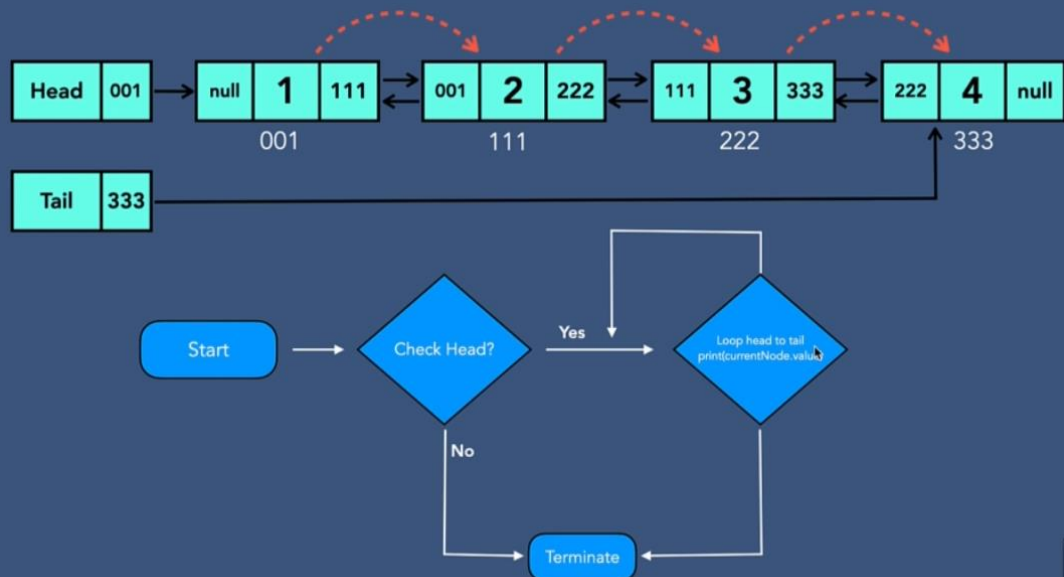# Insertion Algorithm - Doubly Linked List
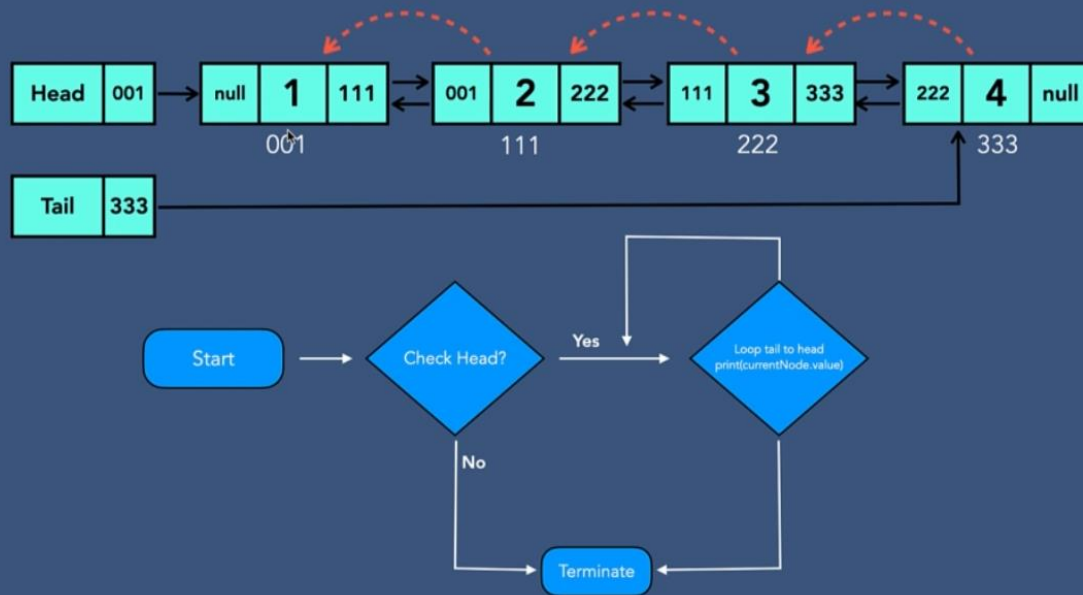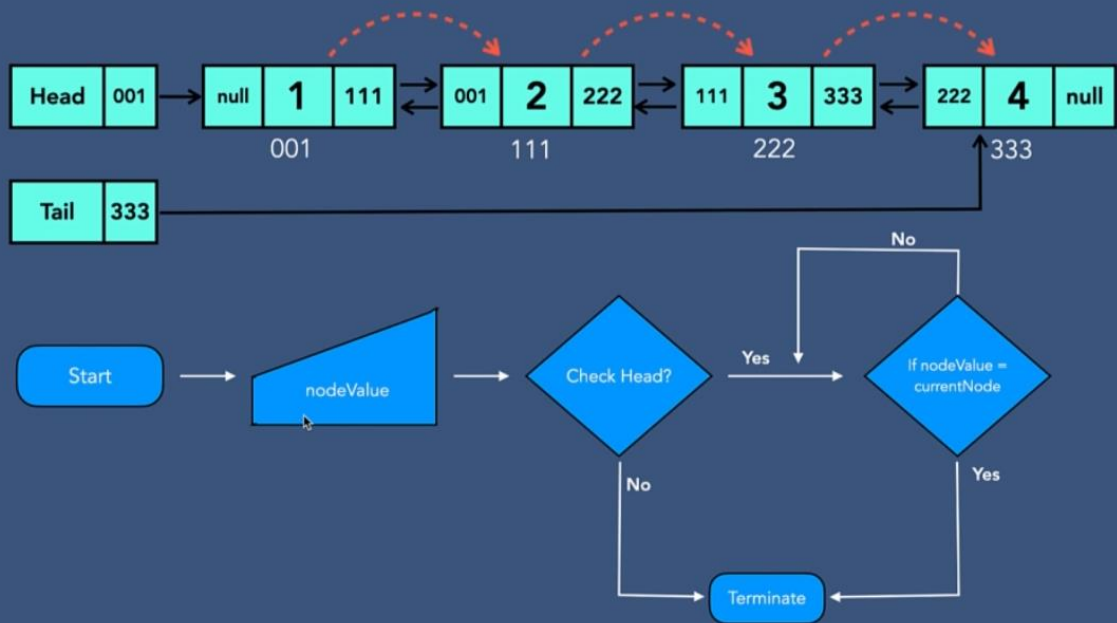

Traversal - Doubly Linked List

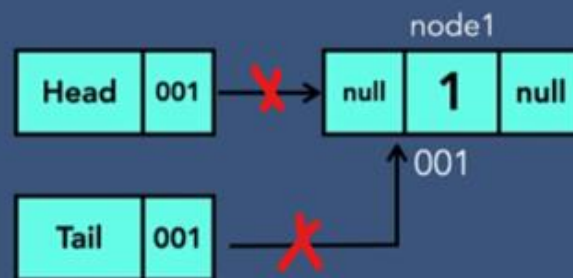# Reverse Traversal - Doubly Linked List

Searching- Doubly Linked List

Deletion - Doubly Linked List
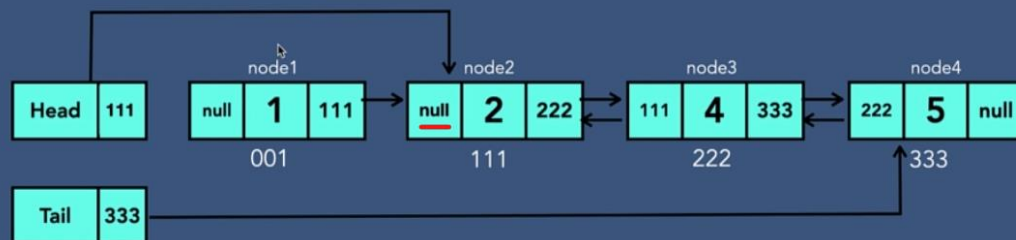
Deleting the first node
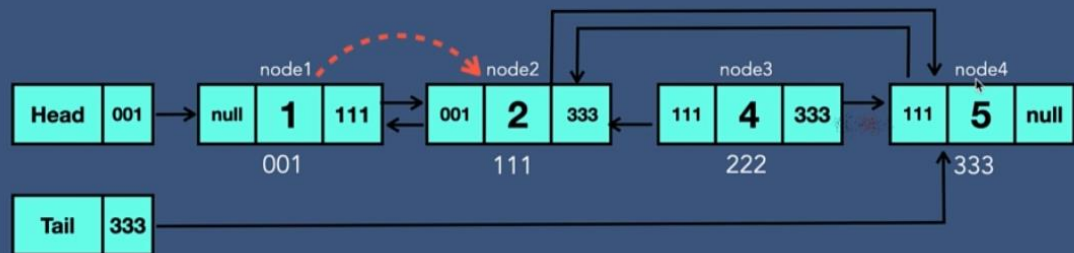
Case 1 - one node

Deletion - Doubly Linked List

Deleting the first node

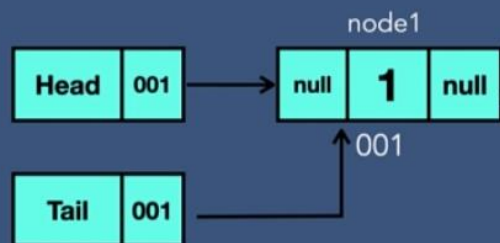Case 2 - more than one node

Deletion - Doubly Linked List

Deleting any given node

node1   node2   node3   node4

| Head | 001 | → | null | **1** | 111 | ⇄ | 001 | **2** | 333 | ← | 111 | **4** | 333 | → | 111 | **5** | null |

001   111   222   333

| Tail | 333 |

Deletion - Doubly Linked List

Deleting the last node

Case 1 - one node

node1

| Head | 001 | → | null | **1** | null |

001

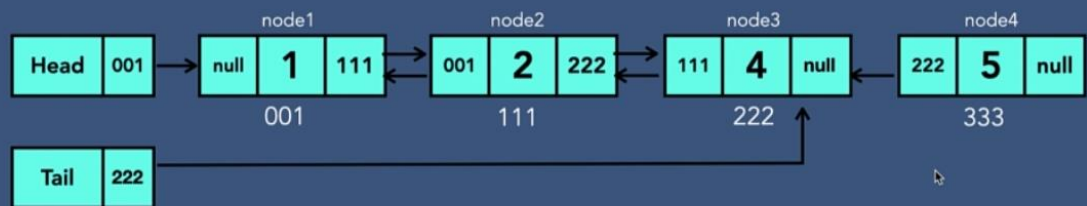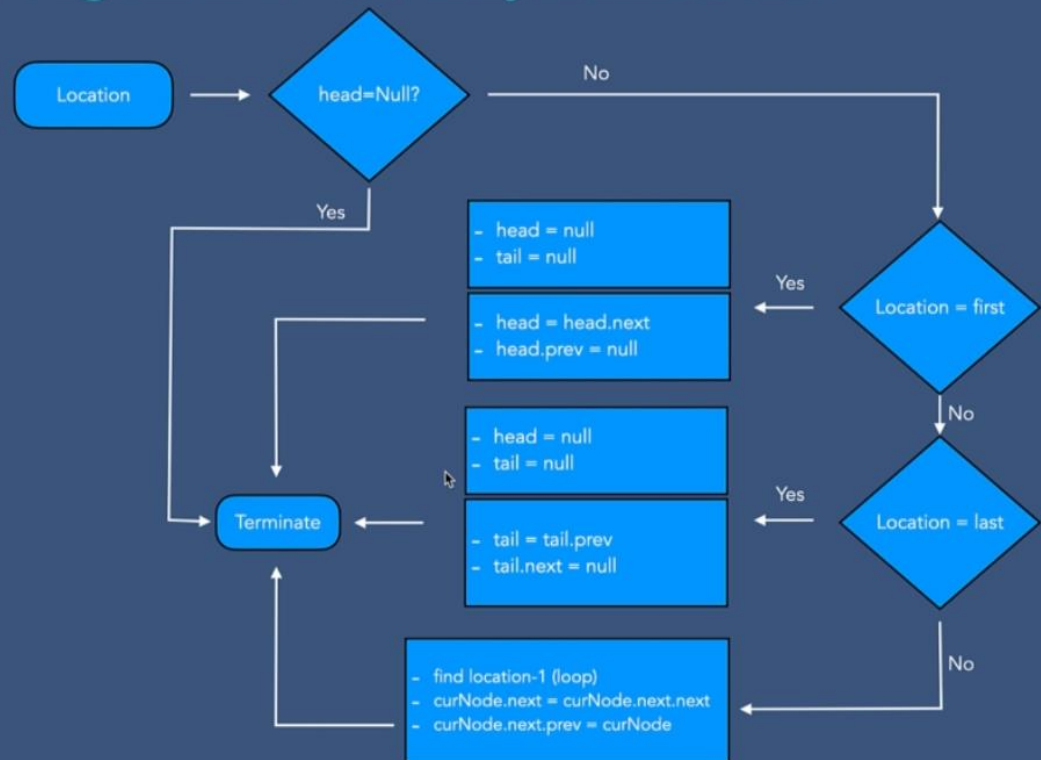| Tail | 001 |

Deletion - Doubly Linked List
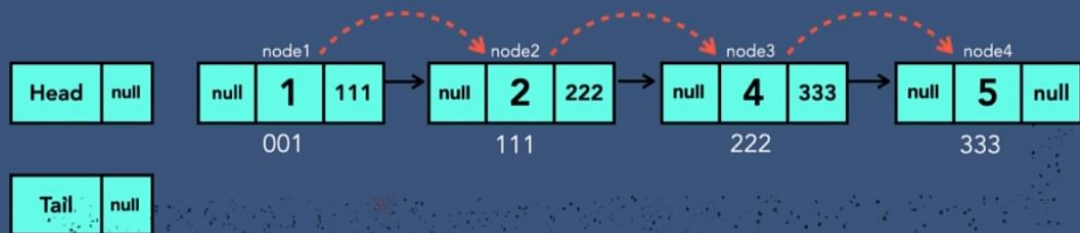
Deleting the last node

Case 2 - more than one node

Deletion Algorithm - Doubly Linked List

Delete Entire Doubly Linked List

Time and Space Complexity of Doubly Linked List

Doubly Linked List	Time complexity	Space complexity
Creation	O(1)	O(1)
Insertion	O(n)	O(1)
Searching	O(n)	O(1)
Traversing (forward ,backward)	O(n)	O(1)
Deletion of a node	O(n)	O(1)
Deletion of DLL	O(n)	O(1)