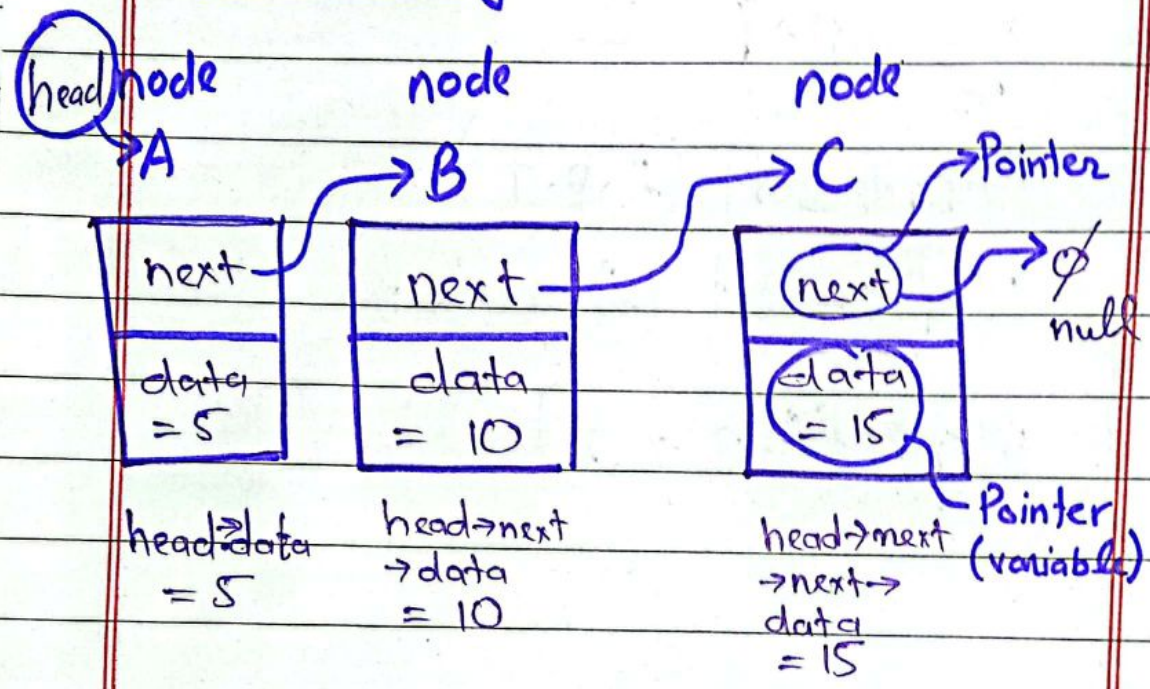
Date: _/__/20 Singly Day: _____

Linked List

Head Only:

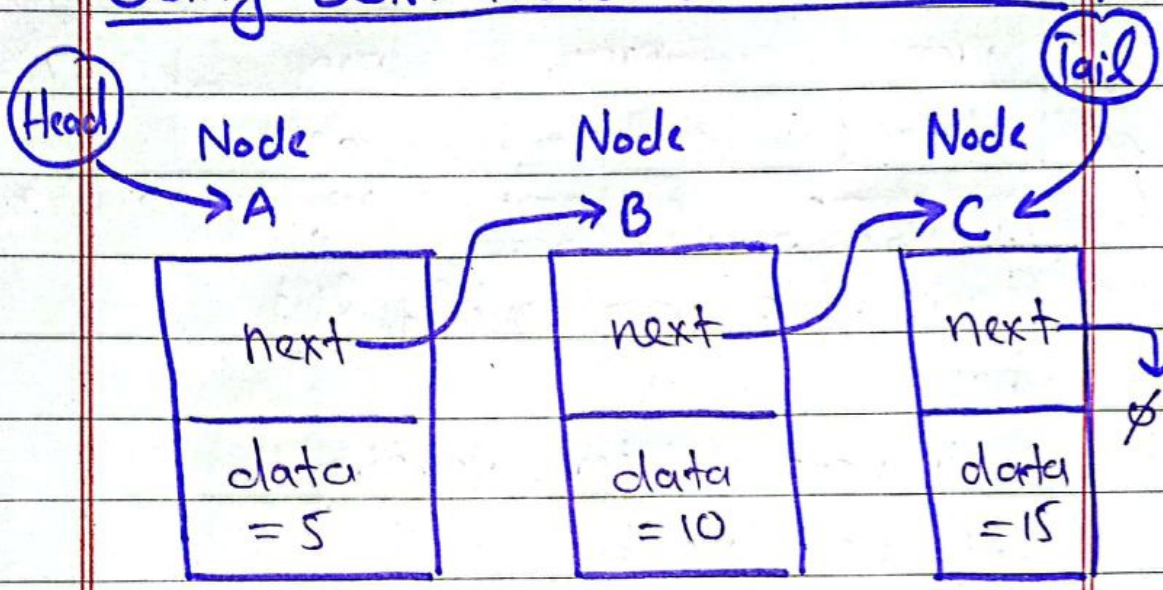


- Head points to the first node, entry point for traversal or adding a node to the begining.
- Next pointers in each node ~~has~~ points to the next node if there is no next node then it points to null

Date: __/__/20 Day: _____

- If there is no node in the list then Head points to null.

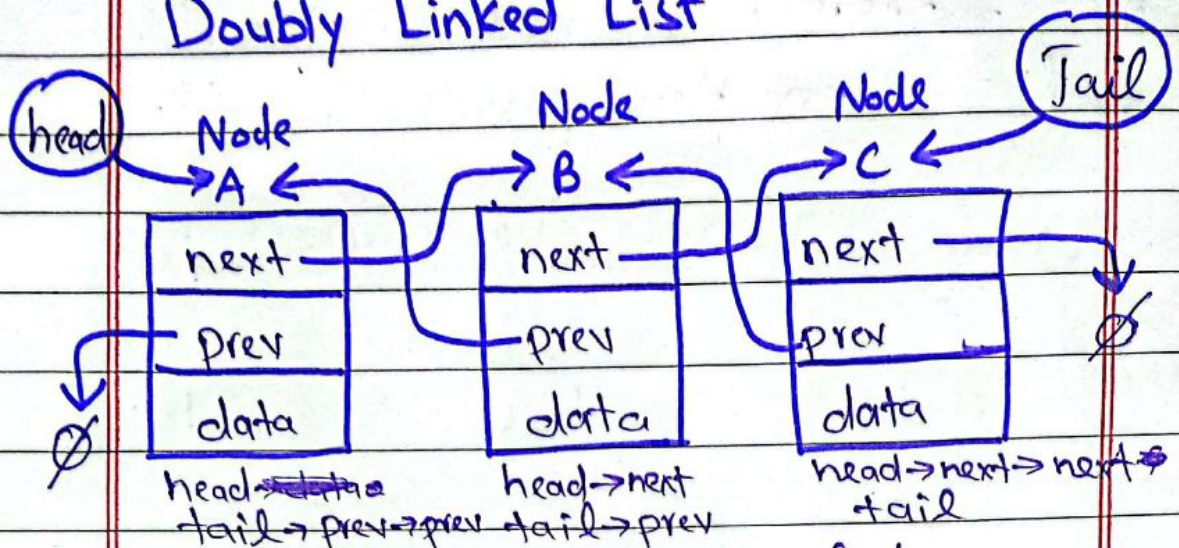
Using Both Head & Tail Pointers:



- Having a tail pointer makes it easier to add to the node at end of the list
- But to add a node anywhere in the middle of the list you will still have to traverse through the list. and would have to keep track of the previous node.

Date: __/__/20 Day: _____

Doubly Linked List

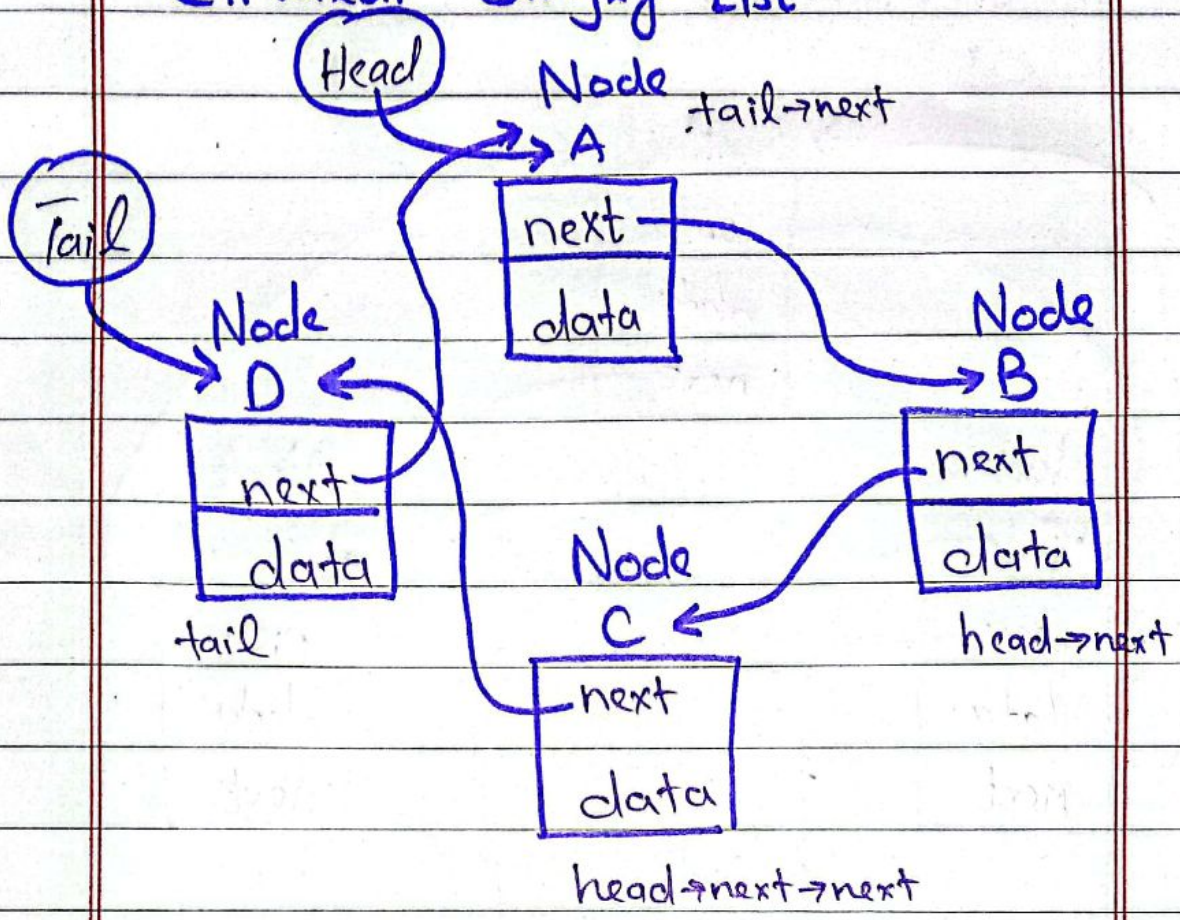


- Each node in a doubly list has
  - → a next pointer, pointing to the next node
  - → a previous pointer, pointing to the previous nod
  - → and a data variable.

- The previous node makes it easier to insert a node anywhere in the middle of list as you would not have to keep track of the previous node.

Date:__/__/20__ Day:_____

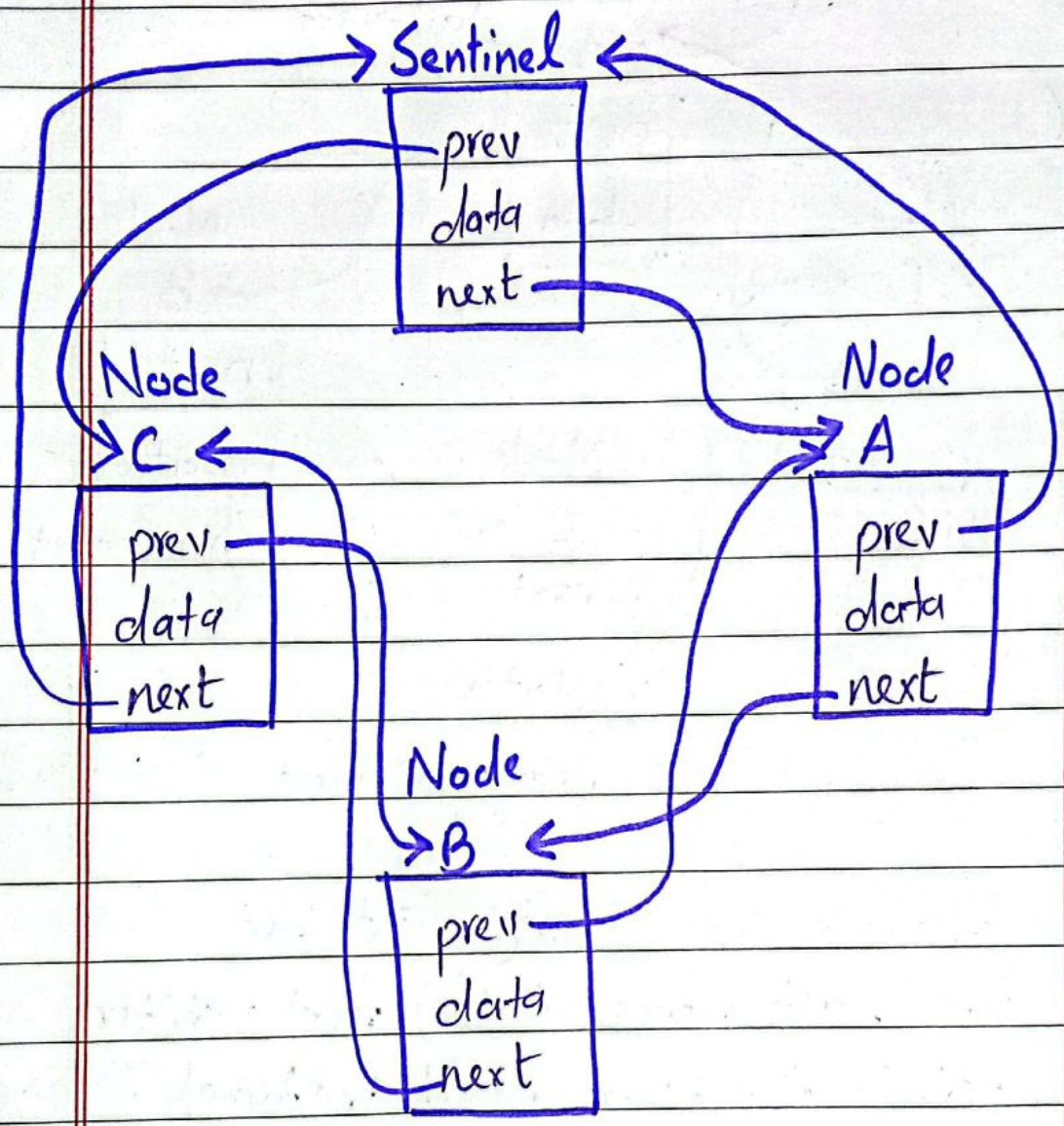
Circular Singly List



- Similar to singly list but the last node (tail) next pointer points to ~~null~~ head instead of null forming a Circular Relation.

Date: __/__/20__ Day: _____

Circular Doubly List



- In this list we use sentinel also called a dummy node instead of head and tail pointers.

- The sentinel will always

Date:__/__/20__ Day:_____

exist even when the list is empty.

- Sentinel→next points to the first node and sentinel→prev points to the last node.