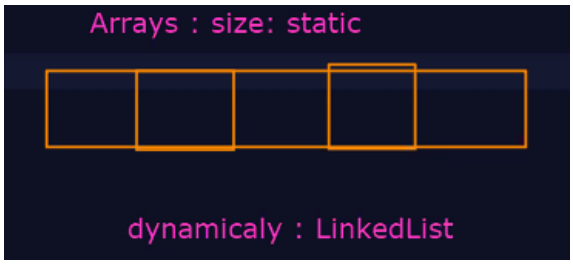
Day3_ADSA_Sanket_Shalukar

Thursday, September 18, 2025 10:23 AM

Topics that are in the day 2

- LinkedList



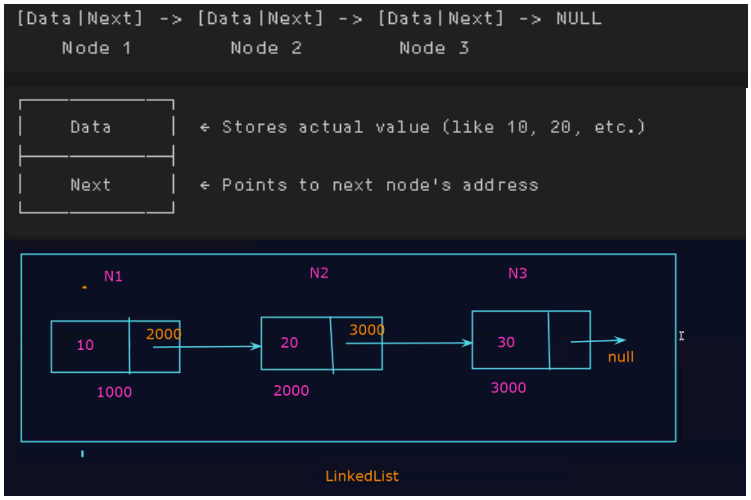
LinkedList

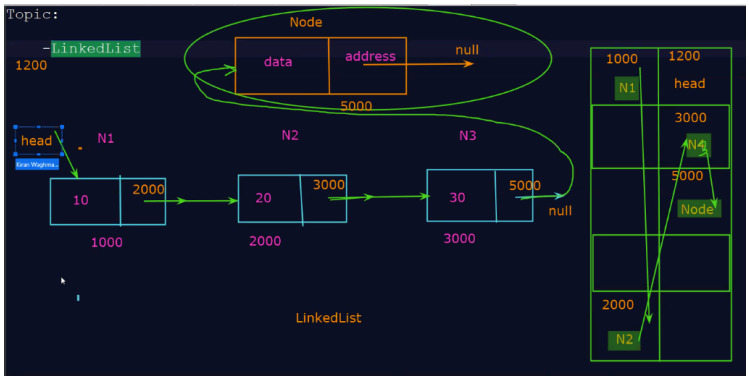
A LinkedList is a chain of connected boxes (called nodes) where each box contains:

1. Data (the actual information)
2. Pointer/Reference (address of the next box)

Think of it like a treasure hunt - each clue tells you where to find the next clue!

Basic LinkedList Structure





1. data : holds the data value

2. address: pointer to next node

Head Pointer: Points to the first node of the linkedlist

Basic terms:

Head: Pointer to the first node of linkedlist

Node: Consist of data and pointer to next node

Data: Value/Information stored in the node

Next Pointer: (Address): Points to next node address

Importance of LinkedList :

1

2. Ease of Insertion & deletion

3. Efficient utilization of memeory

4. Implement advanced datastrucures based on LL

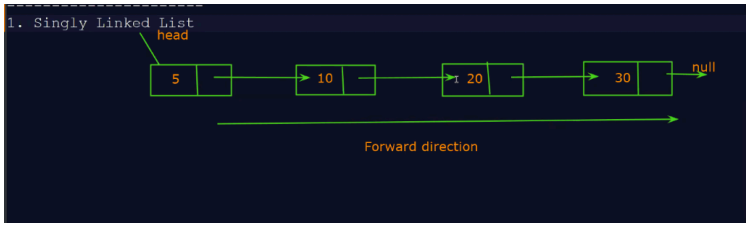
Types of Linked List:

---

1. Singly Linked List

Operations:

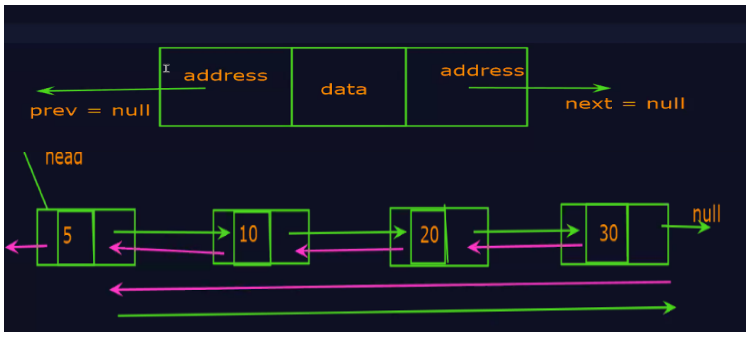
1. Insertion
2. Deletion
3. Traverse
4. Search



2. Doubly Linked List :

Operations:

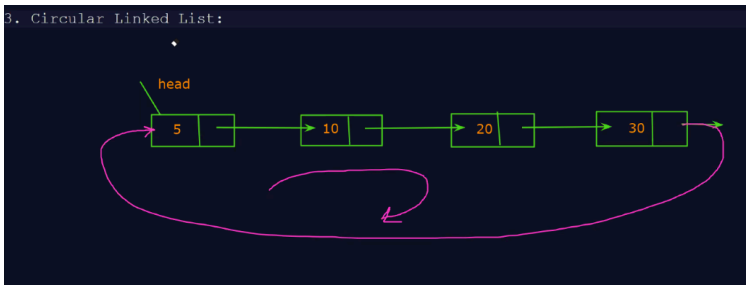
1. Insertion
2. Deletion
3. Traverse



3. Circular Linked List:

Operations:

1. Insertion
2. Deletion
3. Traverse



Singly Linked List :

```java
class LL1 {
    Node head;
    static class Node {
        int data;
        Node next;
        Node(int d) {
            data = d;
            next = null; // null: no previous data
        }
    }
    public static void main(String[] args) {
        LL1 l1 = new LL1();
        l1.head = new Node(11);
        Node second = new Node(22);
        Node third = new Node(33);
        l1.head.next = second;
        second.next = third;
        // Print the linked list values
        Node current = l1.head;
        while (current != null) {
            System.out.println(current.data);
            current = current.next;
        }
    }
}
```

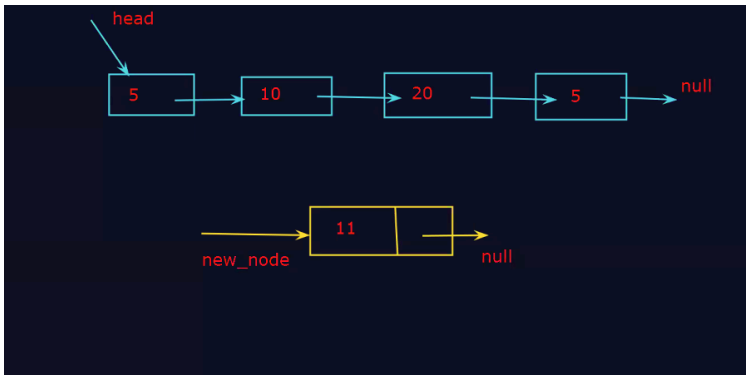
Output:

```text
11
22
33
```

Insertion () : Insertion at the beginning

```
    void insert(int new_data){
    Node new_node = new Node(new_data);
    new_node.next = head;
    head = new_node;


    }
```



Reverse String

```
class StringRevDemo{
    static void reverse(String str){
        if((str == null) || (str.length() <= 1))
            System.out.println(str);
        else{
            System.out.println(str.charAt(str.length()-1));
            reverse(str.substring(0, str.length()-1);

        }

    }

    public static void main(String[] args) {

        String input = "CDACMumbai";
        reverse(input);

    }

}
```