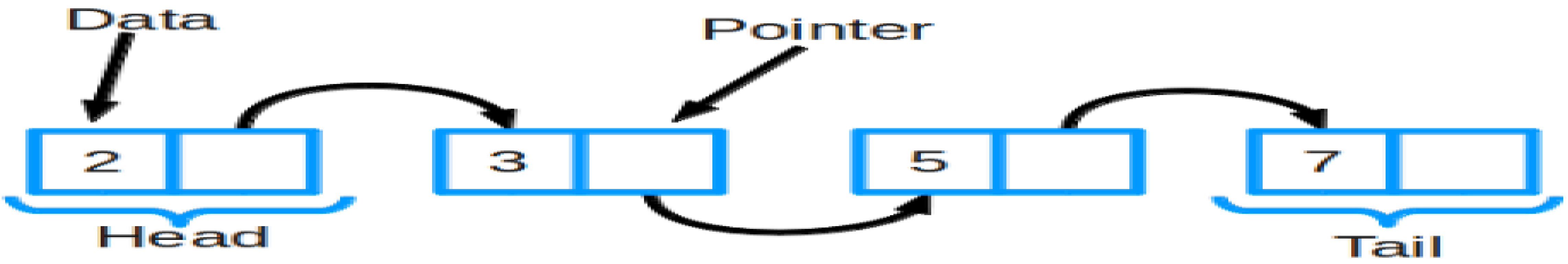
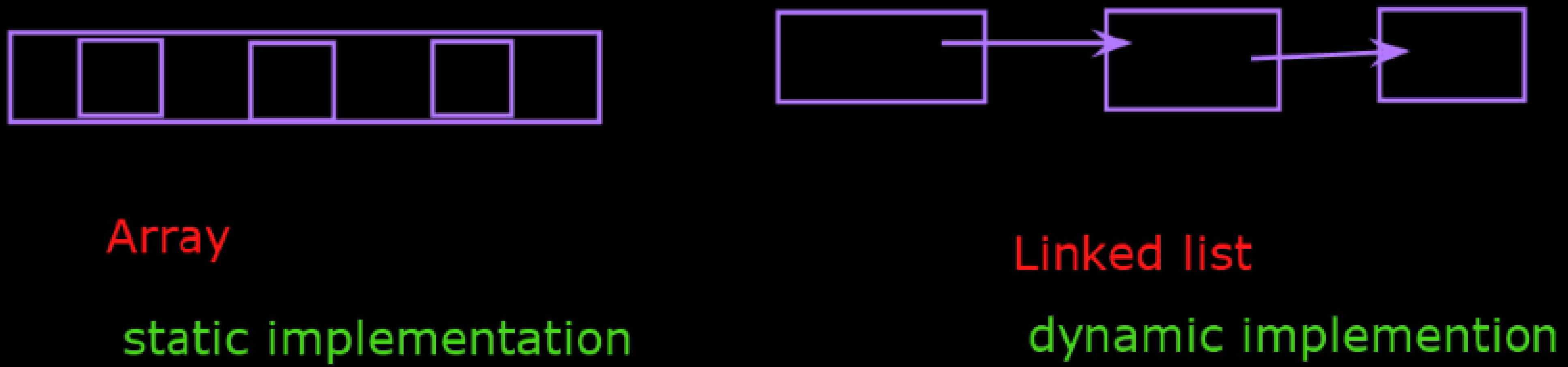
Linked list



```
Linked List:
--------------
-Sorting implemention can be done with
1. Arrays
2. Linked list
```



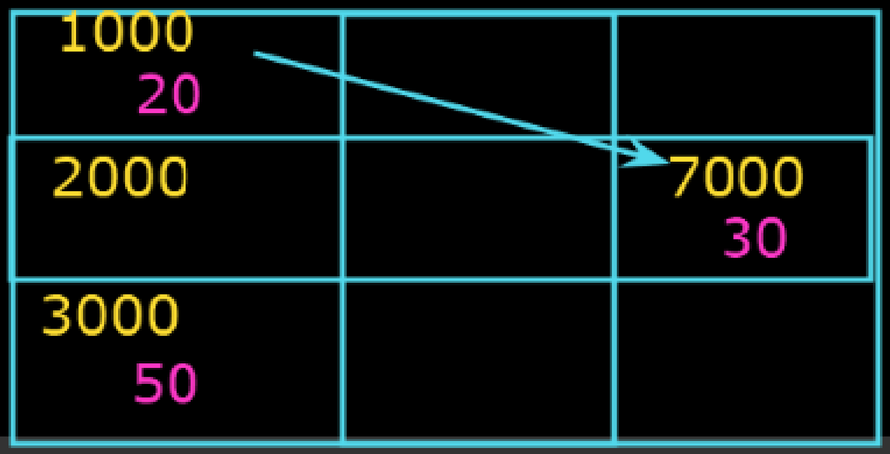
Linked List

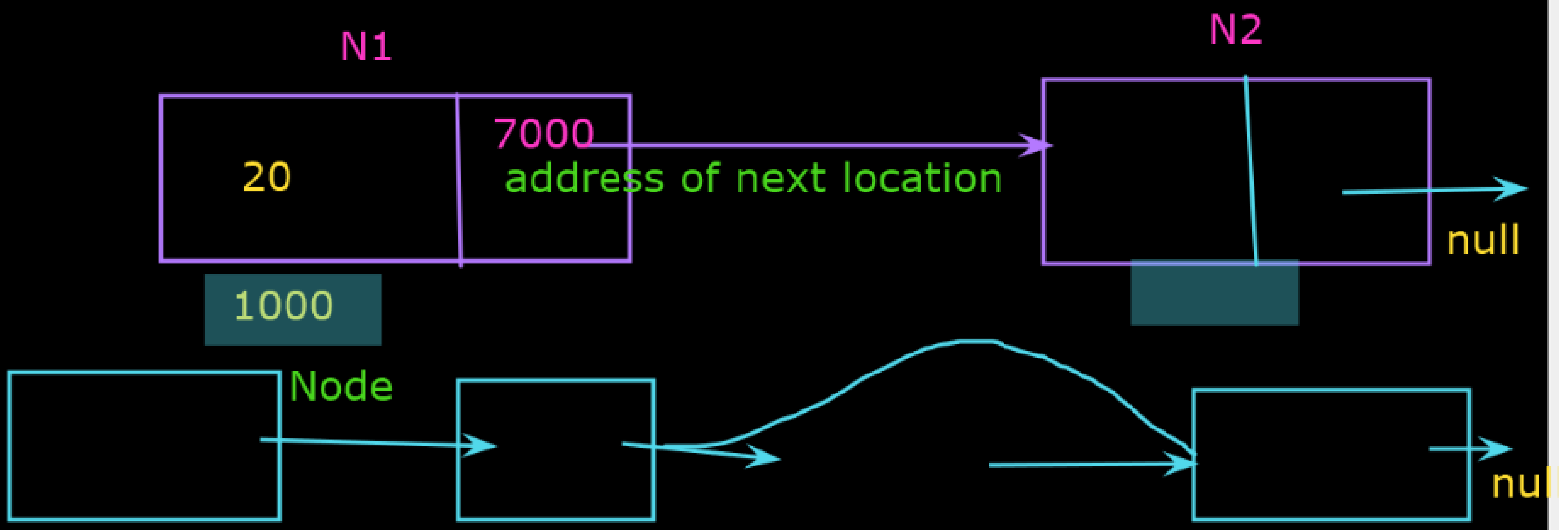
- A linked list is a sequence of data structures, which are connected together via links.

- Linked List is a sequence of links which contains items.
- Each link contains a connection to another link.
- Linked list is the second most-used data structure after array.
- Following are the important terms to understand the concept of Linked List.

1. **Link** – Each link of a linked list can store a data called an **element**.
2. **Next** – Each link of a linked list contains a link to the next link called **Next.**
3. **LinkedList** – A Linked List contains the **connection link** to the first link called **First**.

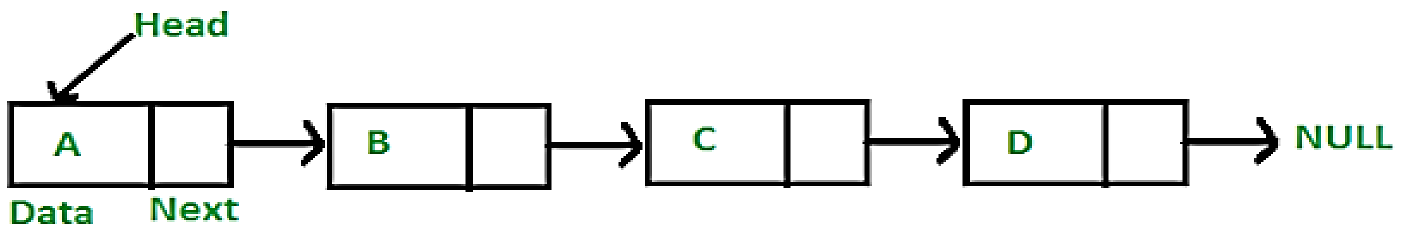
```
     }
    ---------------------------

Linked List:
---------------
-Sorting implemention can be done with
1. Arrays
2. Linked list
```





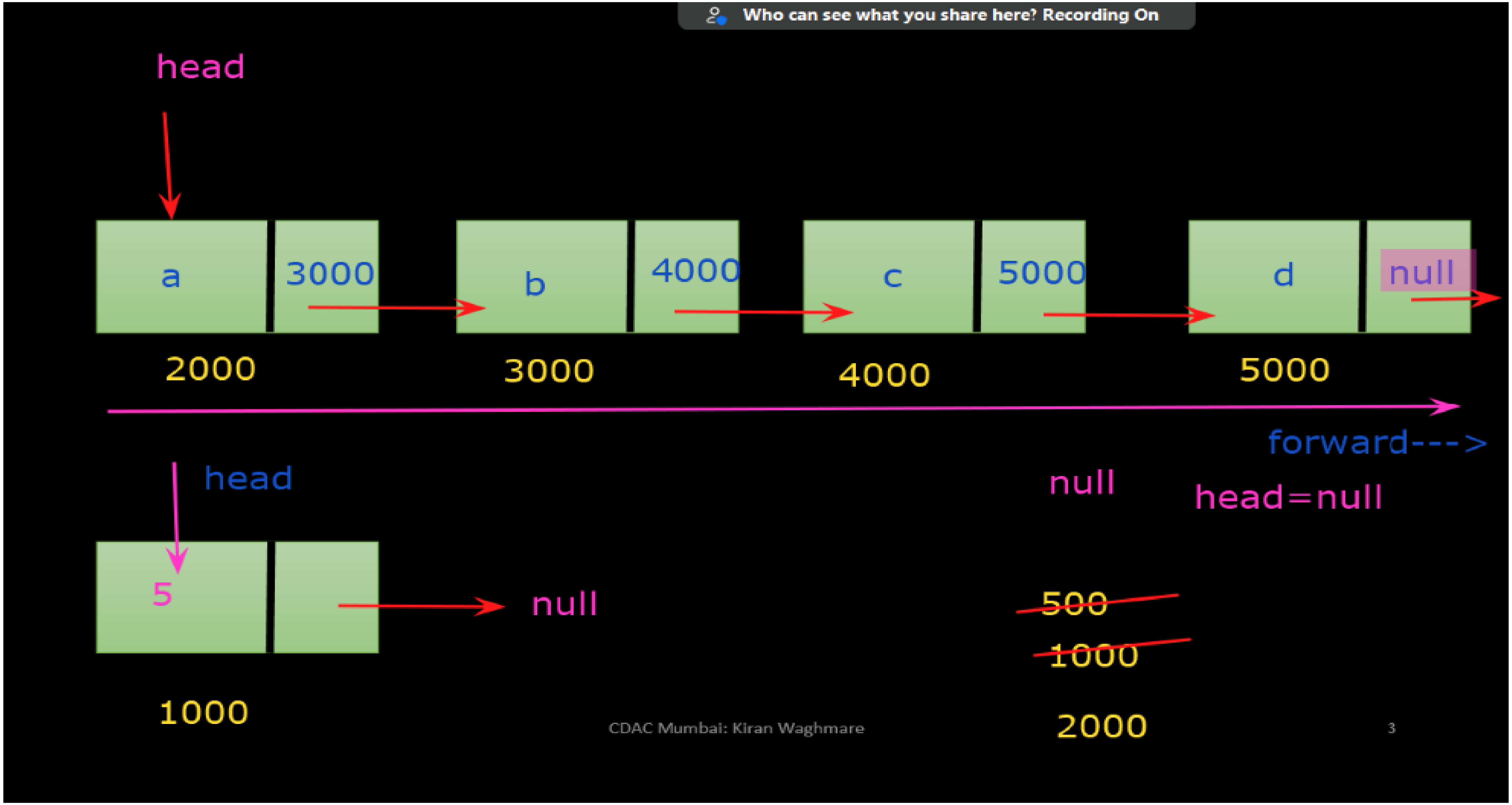
Linked List Representation

- **Linked list can be visualized as a chain of nodes, where every node points to the next node.**



- **As per the above illustration, following are the important points to be considered.**

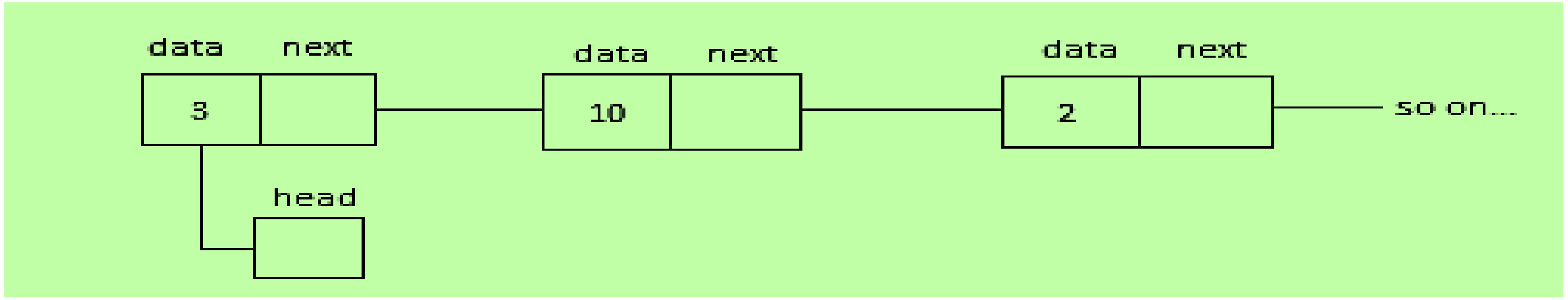
1. Linked List contains a **link element** called **first.**
2. Each link carries a **data field(s)** and a **link field** called **next.**
3. Each link is **linked with its next link** using its **next link.**
4. **Last link carries a link as null** to mark the end of the list.


Who can see what you share here? Recording On
head
a
3000
b
4000
c
5000
d
null
2000
3000
4000
5000
forward--->
head
null
head=null
5
null
~~500~~
~~1000~~
1000
2000
CDAC Mumbai: Kiran Waghmare
3

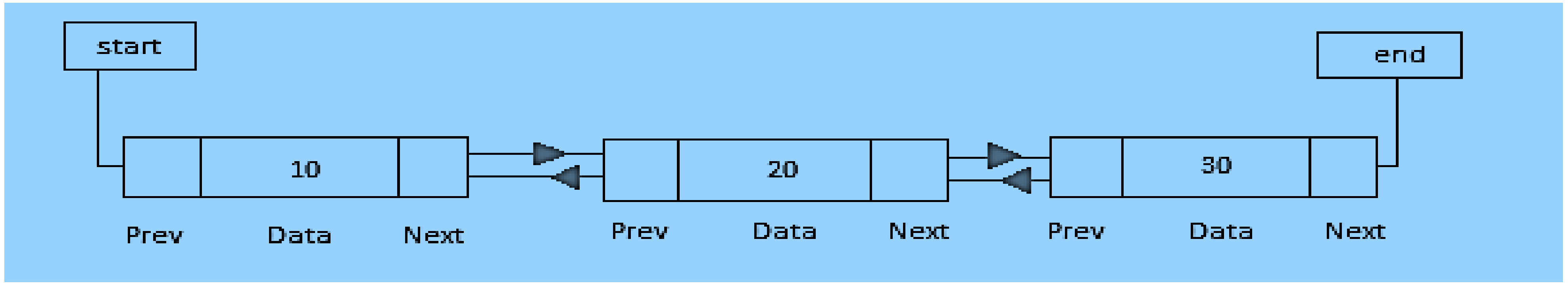
Types of Linked List

- **Following are the various types of linked list.**
  1. **Simple Linked List** – Item navigation is forward only.
  2. **Doubly Linked List** – Items can be navigated forward and backward.
  3. **Circular Linked List** – Last item contains link of the first element as next and the first element has a link to the last element as previous.

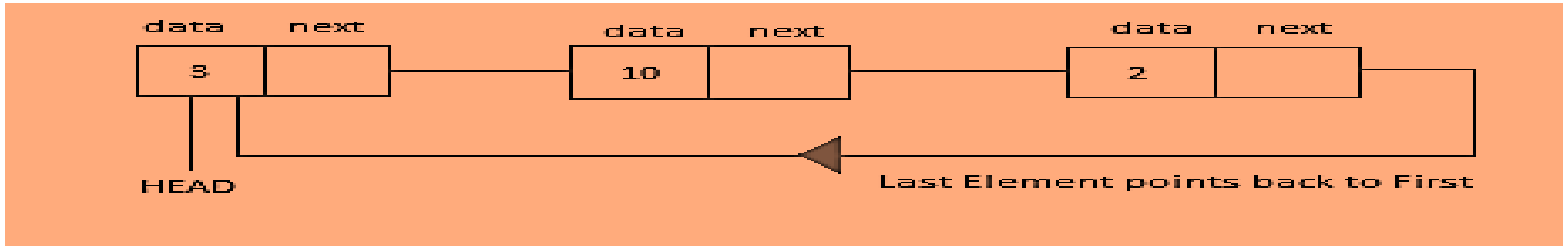
- **Simple Linked List**

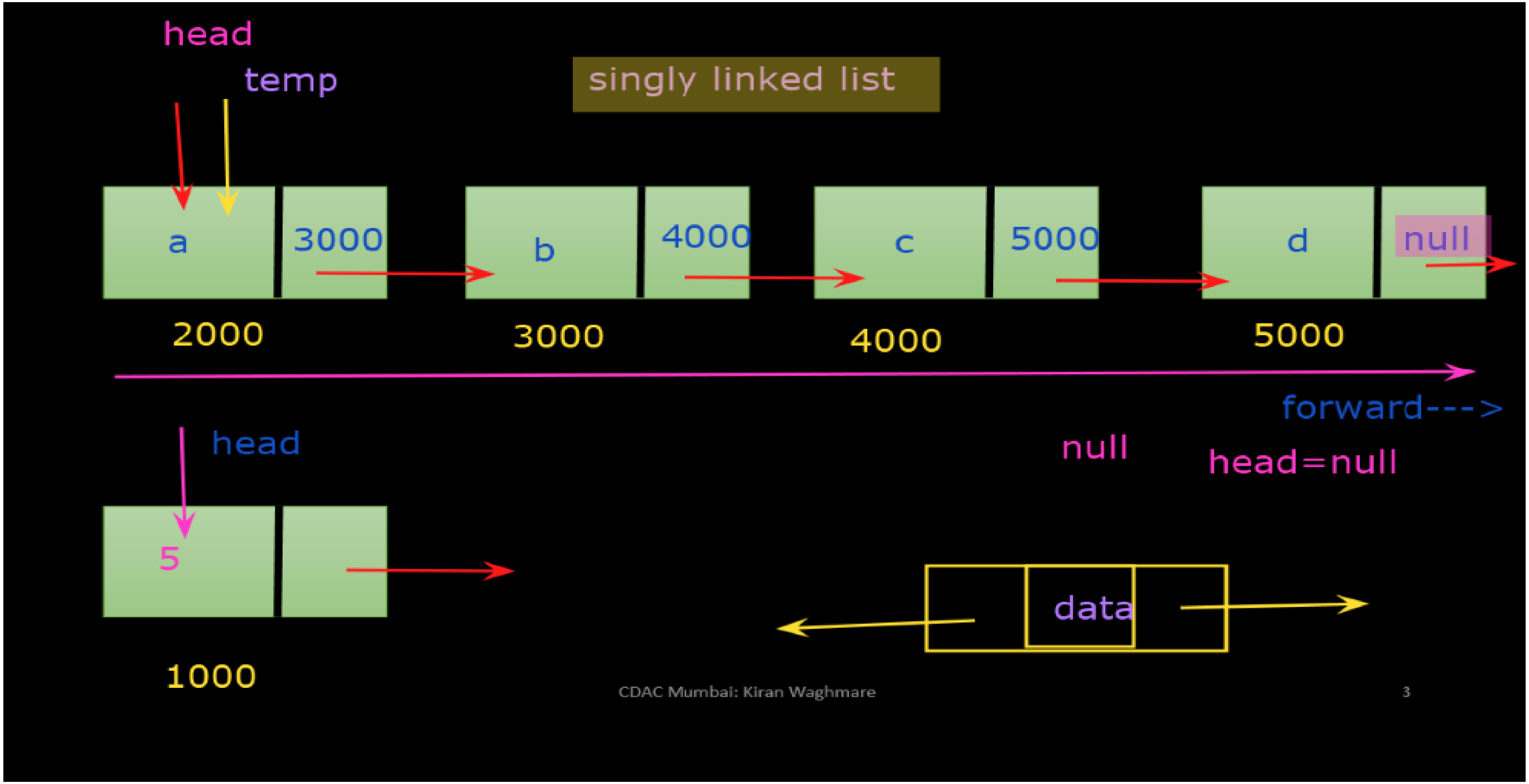


- **Doubly Linked List**

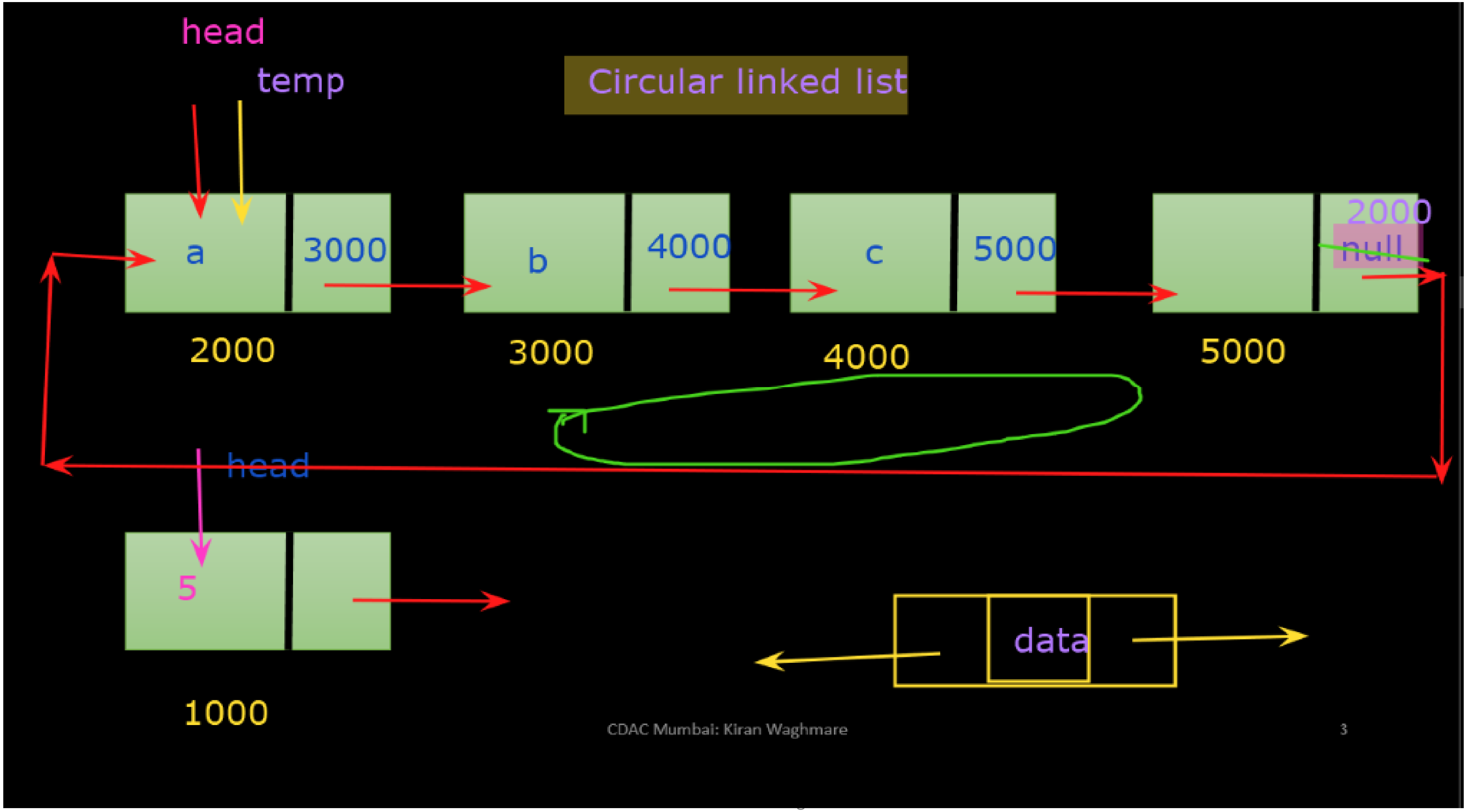


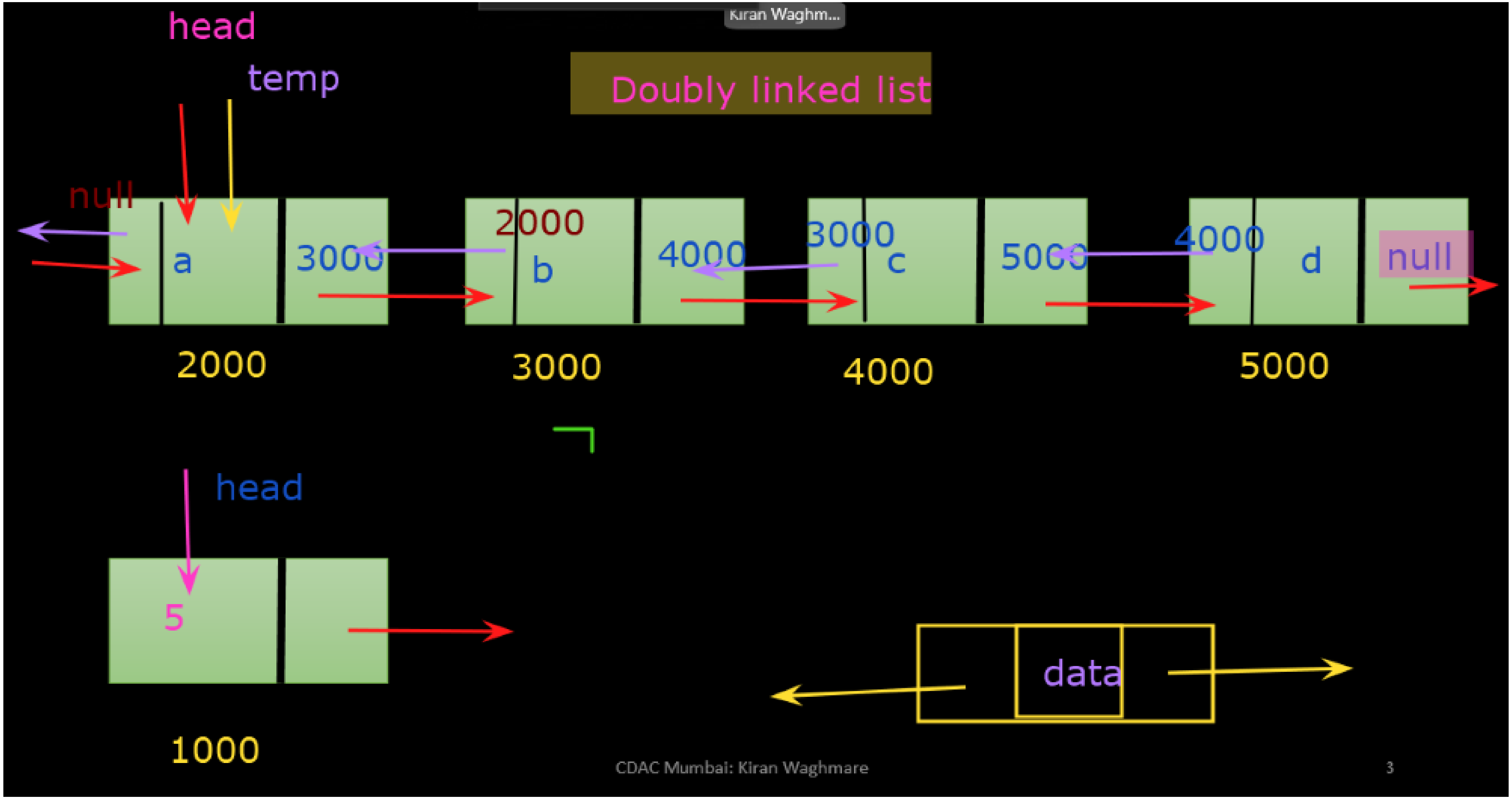
- **Circular Linked List**




head
temp
singly linked list
a
3000
b
4000
c
5000
d
null
2000
3000
4000
5000
forward--->
head
null
head=null
5
data
1000
CDAC Mumbai: Kiran Waghmare
3

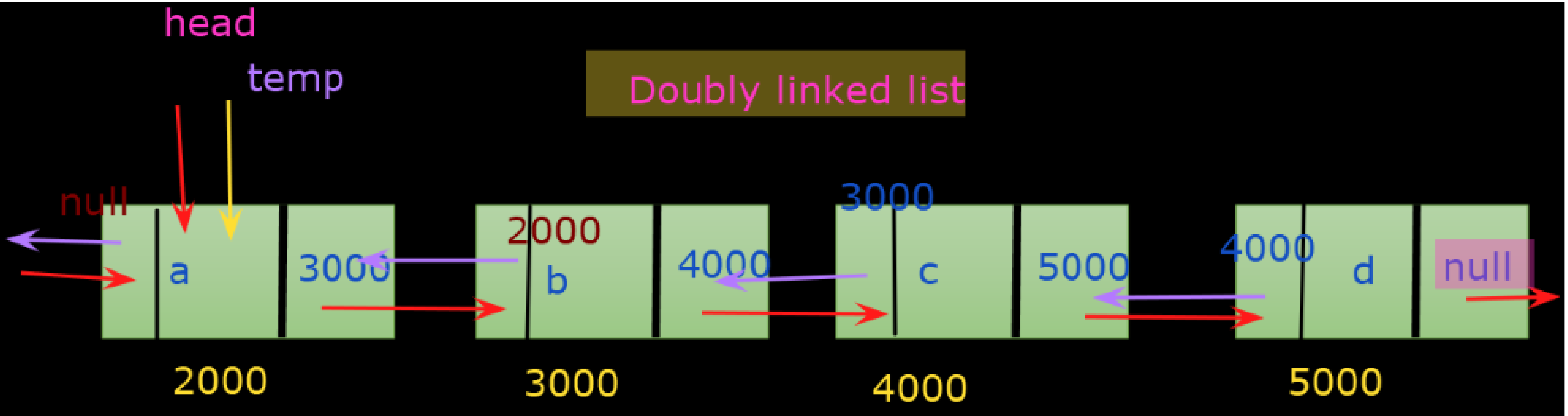
Circular linked list

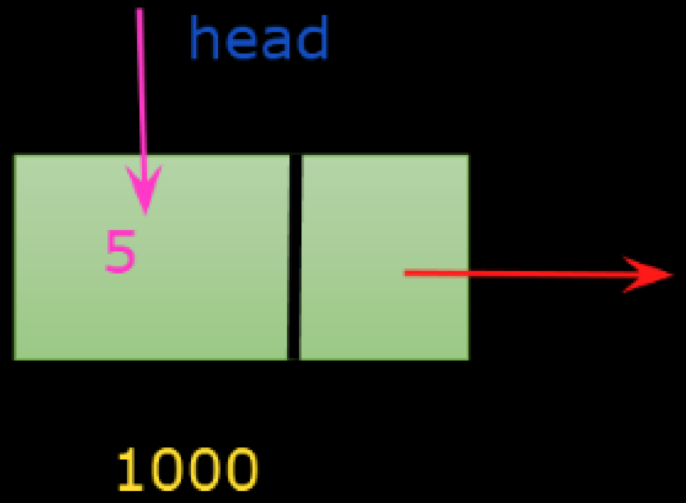


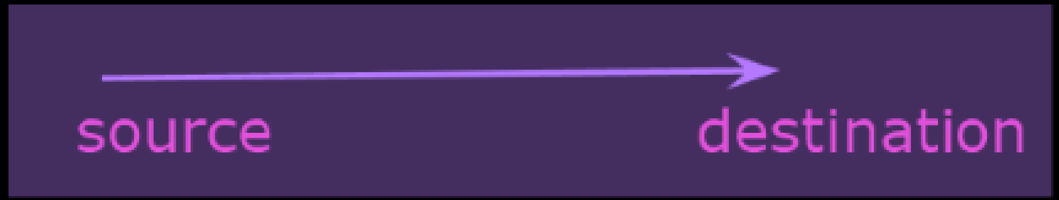


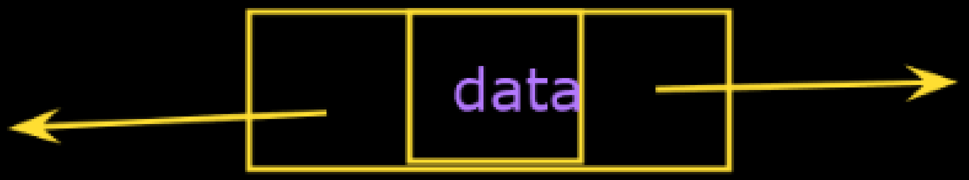
Kiran Waghm...
Doubly linked list
head
temp
null
a
3000
2000
b
4000
3000
c
5000
4000
d
null
2000
3000
4000
5000
head
5
1000
data
CDAC Mumbai: Kiran Waghmare
3

Doubly linked list



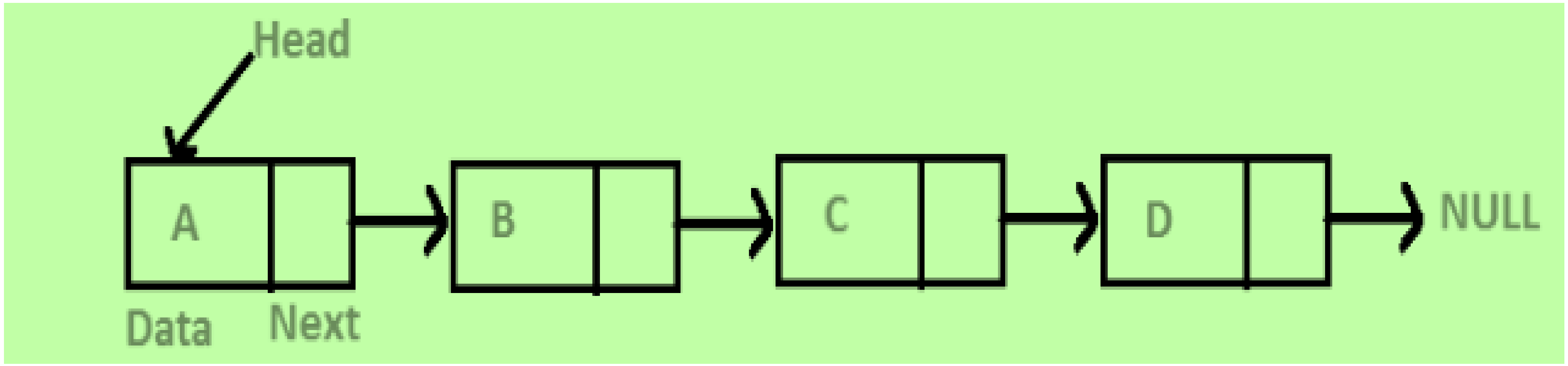






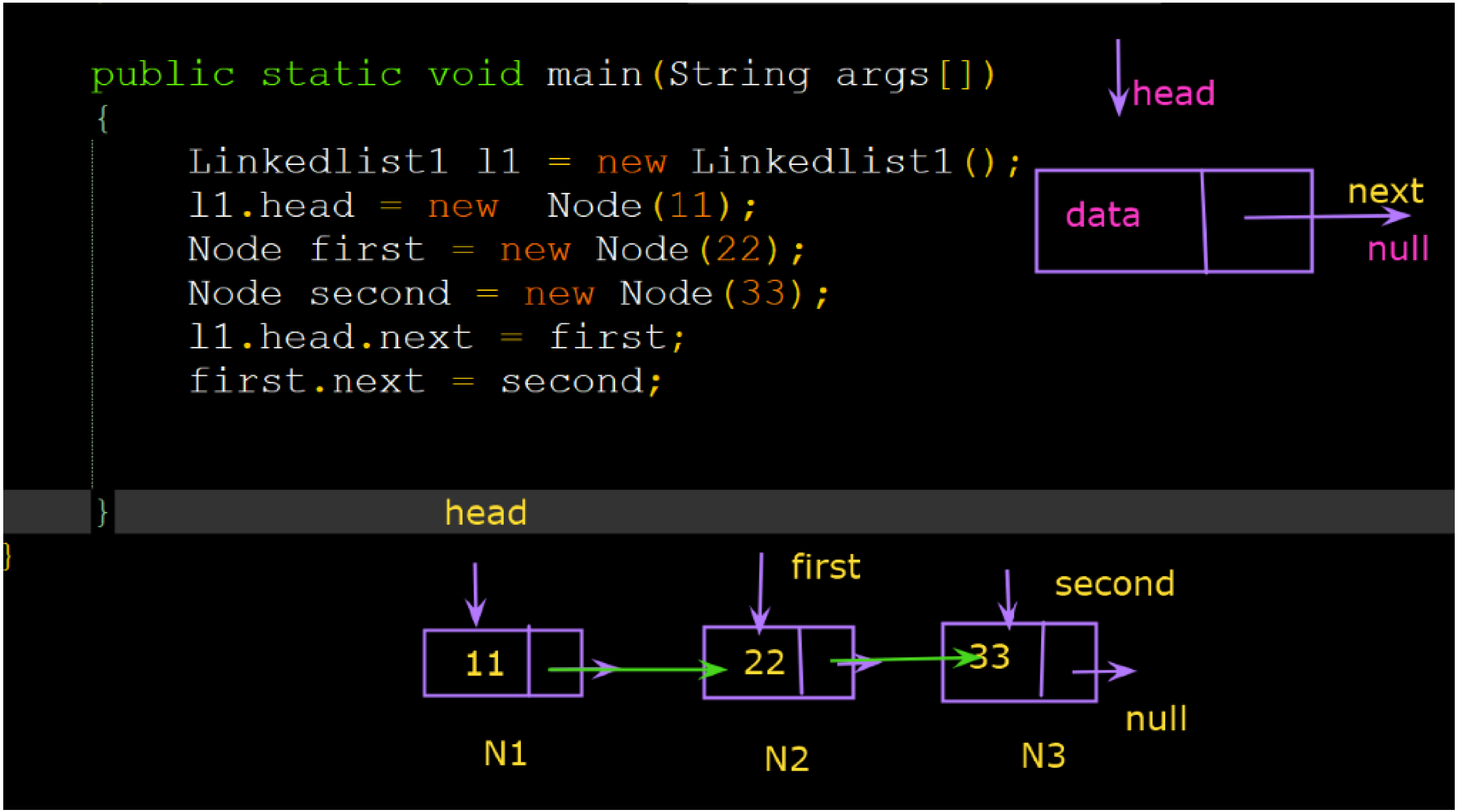
Singly Linked List

- Singly Linked Operations: Insert, Delete, Traverse, search, Sort, Merge

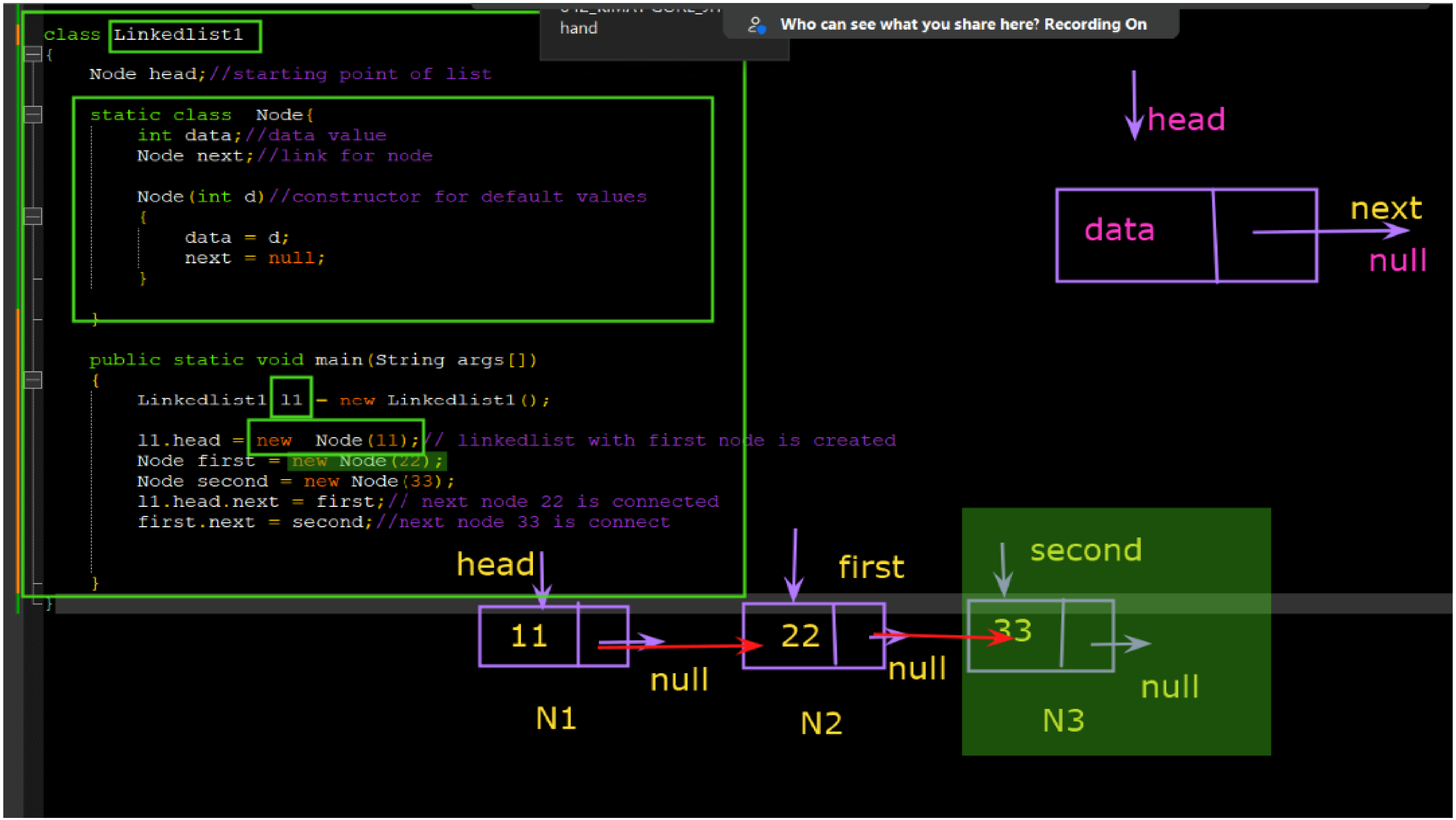


Basic Operations

- **Following are the basic operations supported by a list.**
  1. **Insertion** − Adds an element at the beginning of the list.
  2. **Deletion** − Deletes an element at the beginning of the list.
  3. **Display** − Displays the complete list.
  4. **Search** − Searches an element using the given key.
  5. **Delete** − Deletes an element using the given key.



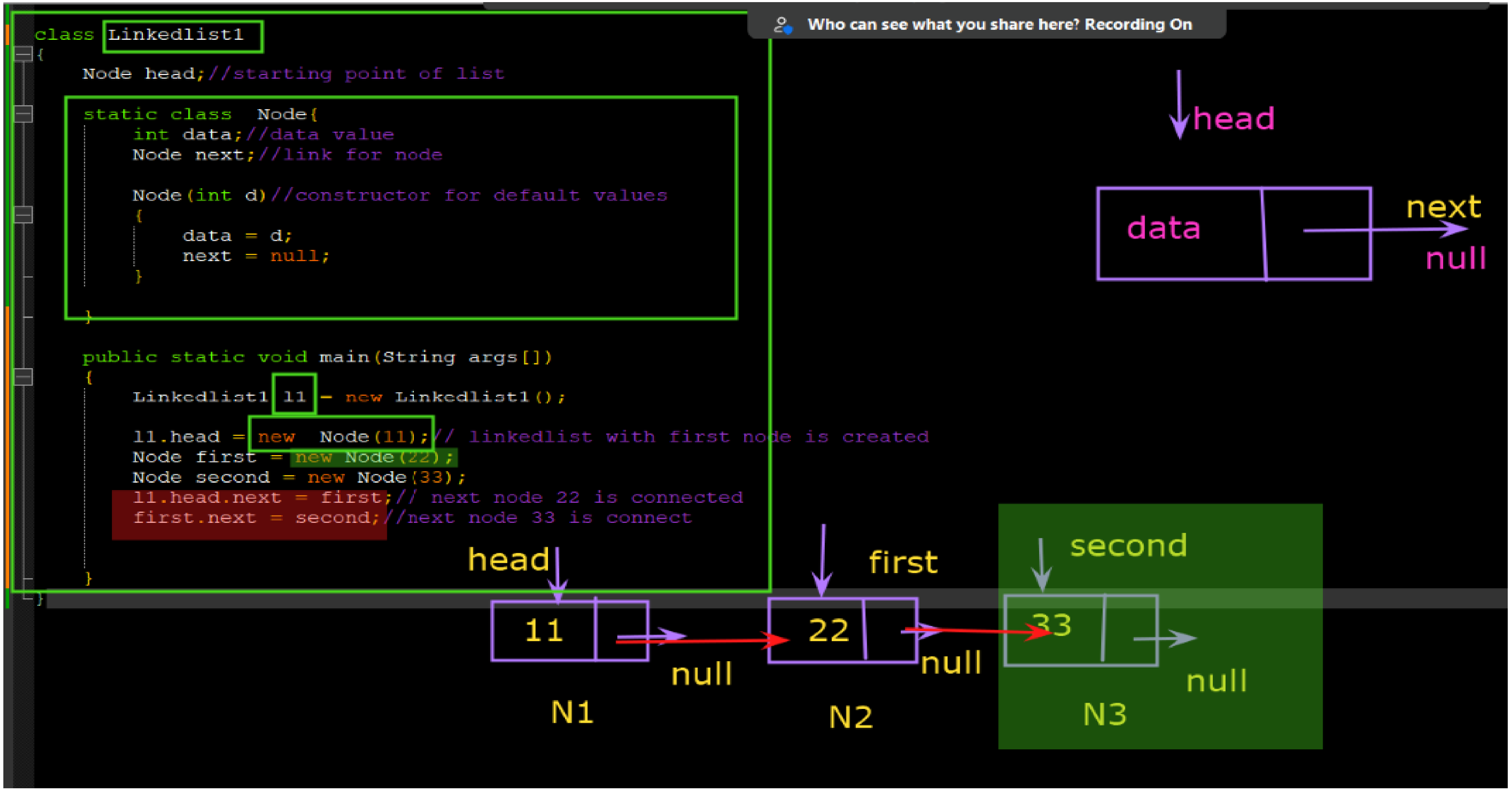
```
public static void main(String args[])
{
        Linkedlist1 l1 = new Linkedlist1();
        l1.head = new  Node(11);
        Node first = new Node(22);
        Node second = new Node(33);
        l1.head.next = first;
        first.next = second;


}
}
```



hand

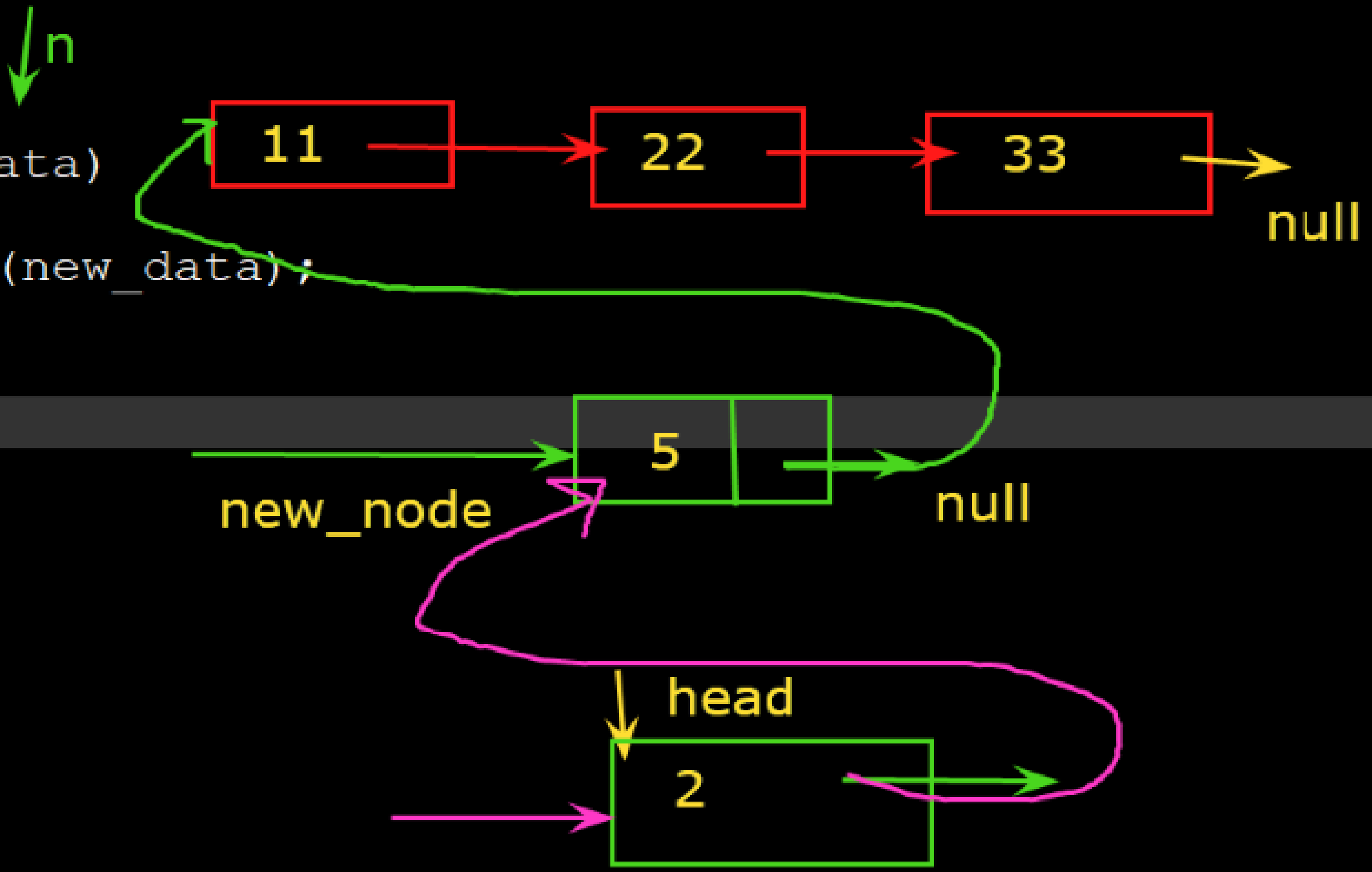
Who can see what you share here? Recording On

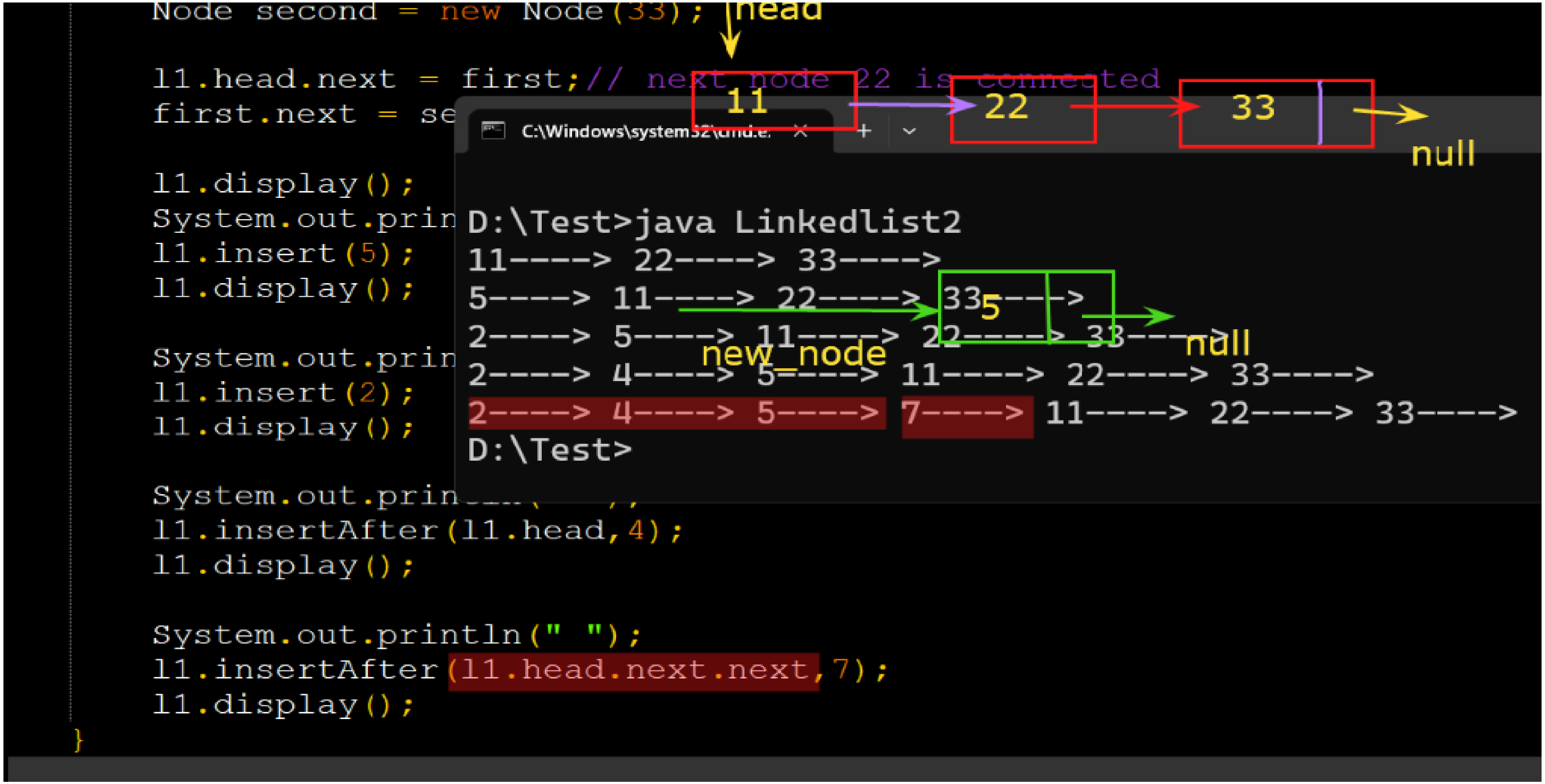
```
class Linkedlist1
{
    Node head;//starting point of list

    static class  Node{
        int data;//data value
        Node next;//link for node

        Node(int d)//constructor for default values
        {
            data = d;
            next = null;
        }

    }

    public static void main(String args[])
    {
        Linkedlist1 l1 = new Linkedlist1();

        l1.head = new  Node(11);// linkedlist with first node is created
        Node first = new Node(22);
        Node second = new Node(33);
        l1.head.next = first;// next node 22 is connected
        first.next = second;//next node 33 is connect

    }
}
```



```
class Linkedlist1
{
    Node head;//starting point of list
    static class  Node{
        int data;//data value
        Node next;//link for node

        Node(int d)//constructor for default values
        {
            data = d;
            next = null;
        }

    }

    public static void main(String args[])
    {
        Linkedlist1 l1 = new Linkedlist1();

        l1.head = new  Node(11);// linkedlist with first node is created
        Node first = new Node(22);
        Node second = new Node(33);
        l1.head.next = first;// next node 22 is connected
        first.next = second;//next node 33 is connect

    }
}
```

```
Insertion at the begining:
----------------------------

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


}
```





```
        Node second = new Node(33);
        
        l1.head.next = first;// next node 22 is connected
        first.next = se

        l1.display();
        System.out.prin
        l1.insert(5);
        l1.display();

        System.out.prin
        l1.insert(2);
        l1.display();

        System.out.prin
        l1.insertAfter(l1.head,4);
        l1.display();

        System.out.println(" ");
        l1.insertAfter(l1.head.next.next,7);
        l1.display();
}
```

```
D:\Test>java Linkedlist2
11----> 22----> 33---->
5----> 11----> 22----> 33---->
2----> 5----> 11----> 22----> 33---->
2----> 4----> 5----> 11----> 22----> 33---->
2----> 4----> 5----> 7----> 11----> 22----> 33---->
D:\Test>
```

Problem Statement 1 : Delete a Linked List node at a given position.

Given a singly linked list and a position, delete a linked list node at the given position.

Example:

Input: position = 1, Linked List = 18->12->13->11->17
Output: Linked List = 18->13->11->17

Input: position = 0, Linked List = 98->24->32->17->74
Output: Linked List = 24->32->17->74

Program for Nth node from the end of a Linked List

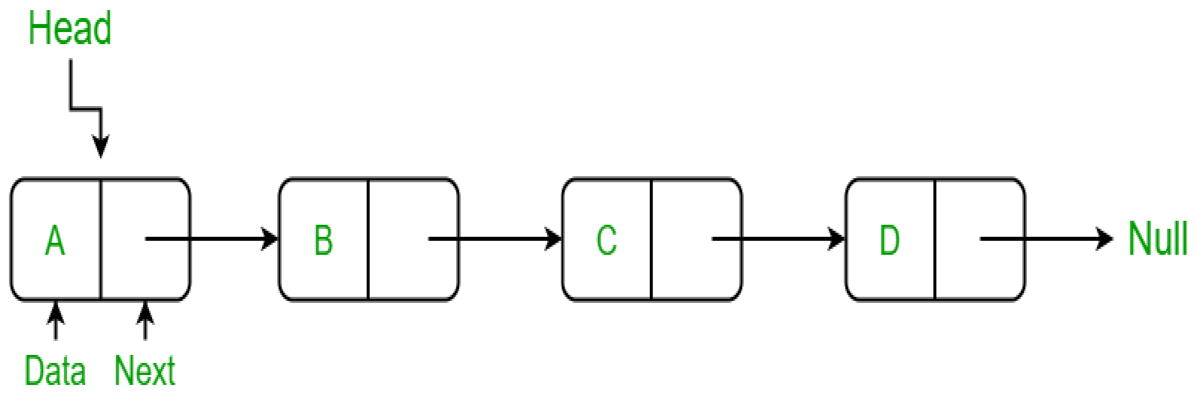
Given a Linked List and a number N, write a function that returns the value at the Nth node from the end of the Linked List.

Linked-List

Examples:

Input: 1 -> 2 -> 3 -> 4, N = 3
Output: 2

Input: 35 -> 15 -> 4 -> 20, N = 4
Output: 35



Thanks