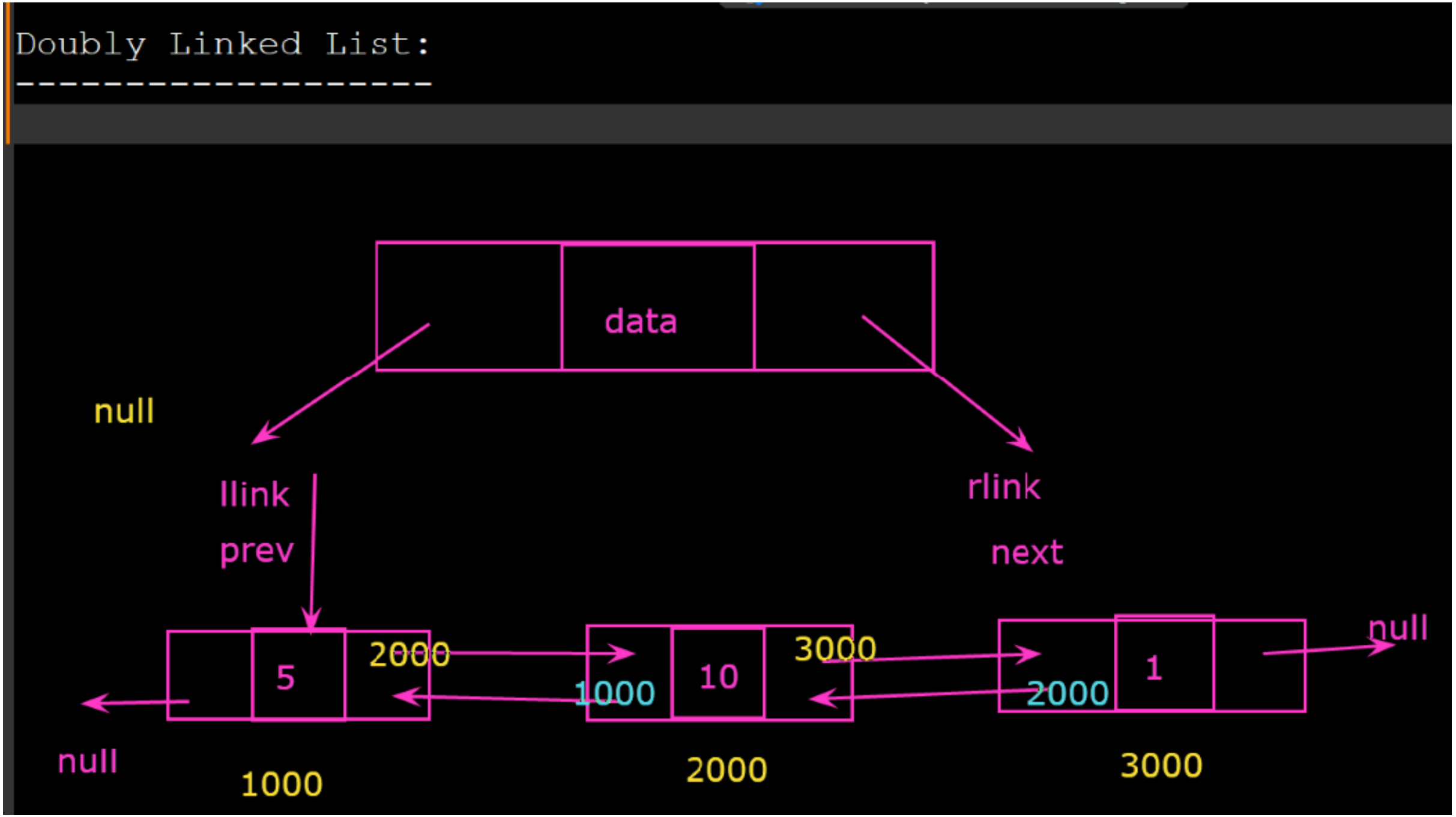
Data Structure

Sep23 : Day 4

Kiran Waghmare

CDAC Mumbai

Doubly Linked List:

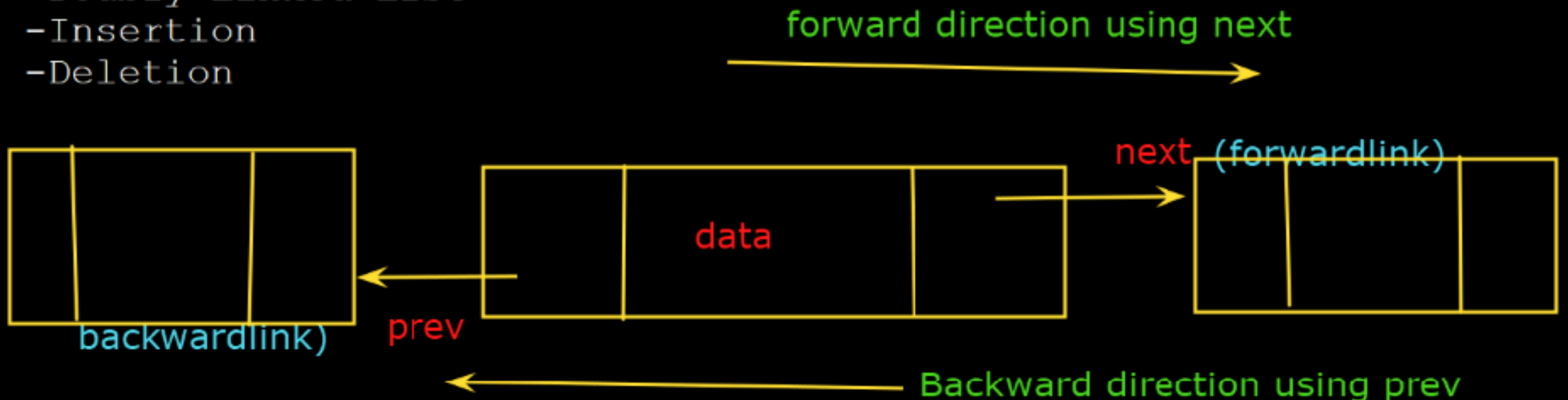


Date: 22-10-2023
Topic: Doubly Linked list

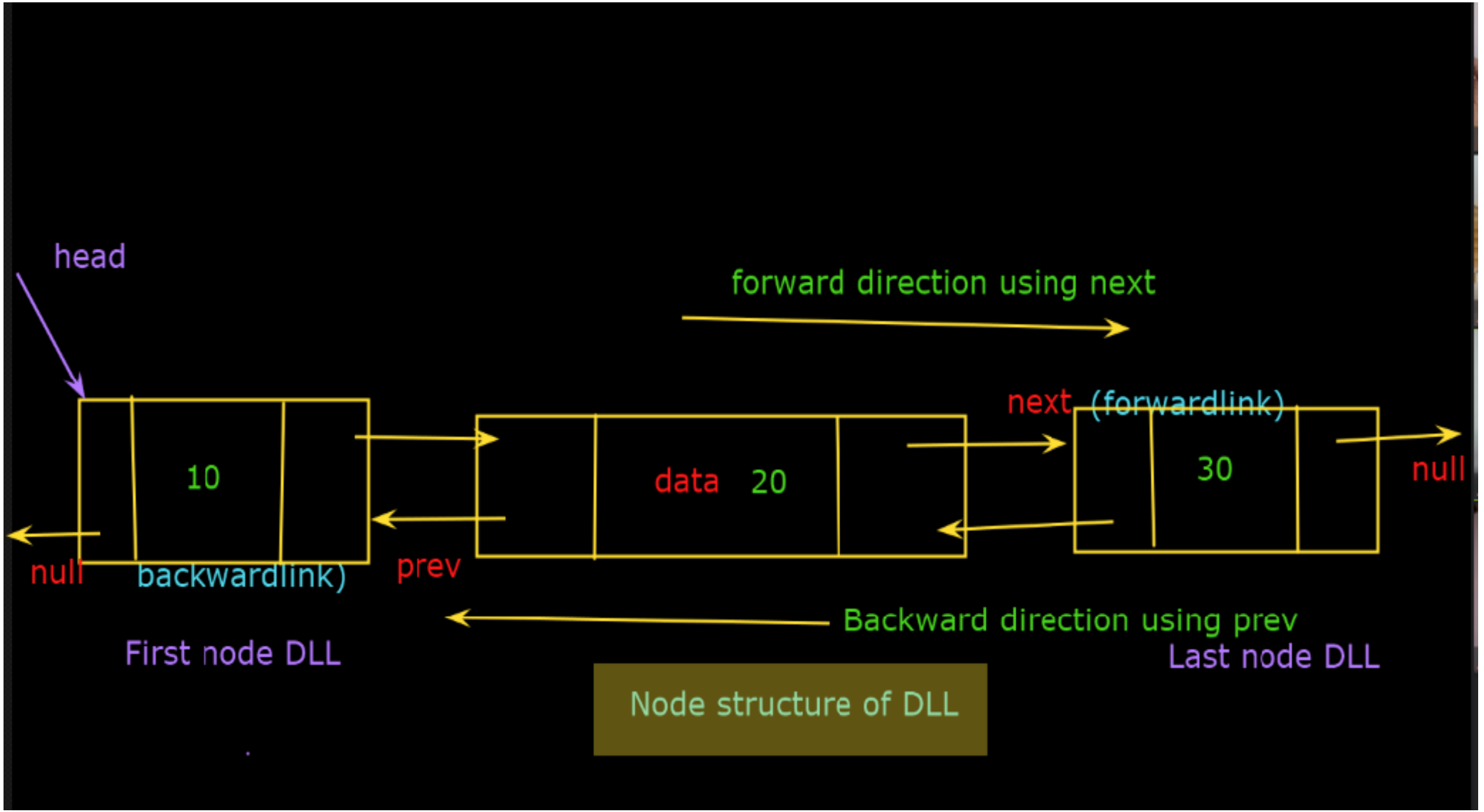
EVERYDAY IS A NEW BEGINING. TAKE A DEEP BREATH, SMILE AND START AGAIN.

Agenda:

- Doubly Linked List
- Insertion
- Deletion



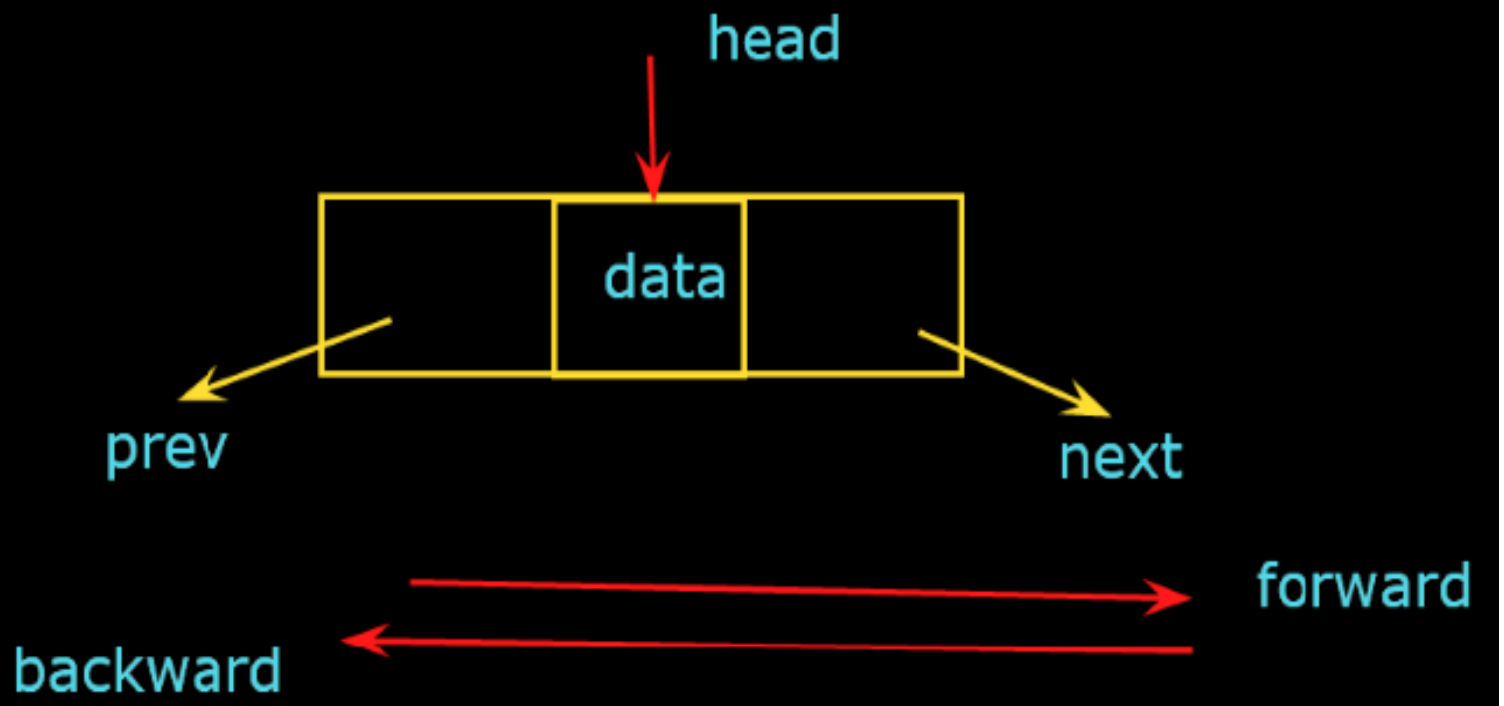
Node structure of DLL


head
forward direction using next
next (forwardlink)
10
data 20
30
null
null
backwardlink)
prev
Backward direction using prev
First node DLL
Last node DLL
Node structure of DLL

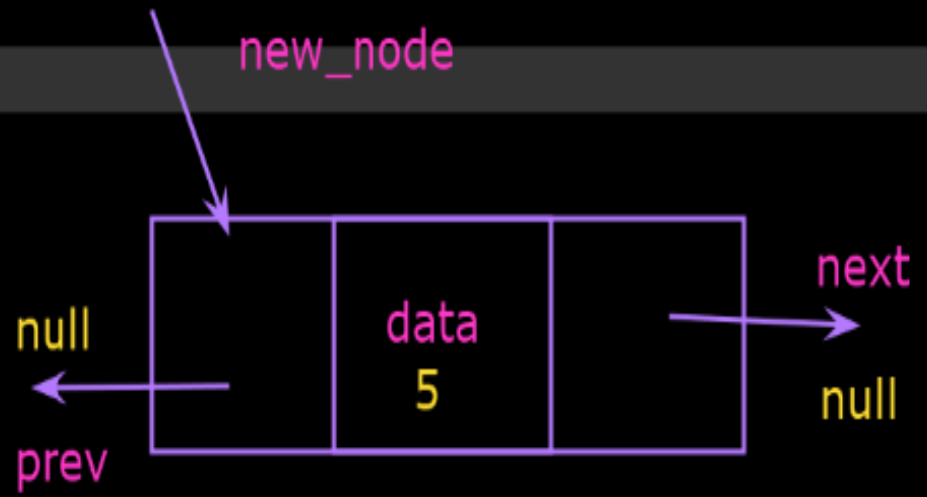
Doubly Linked List:

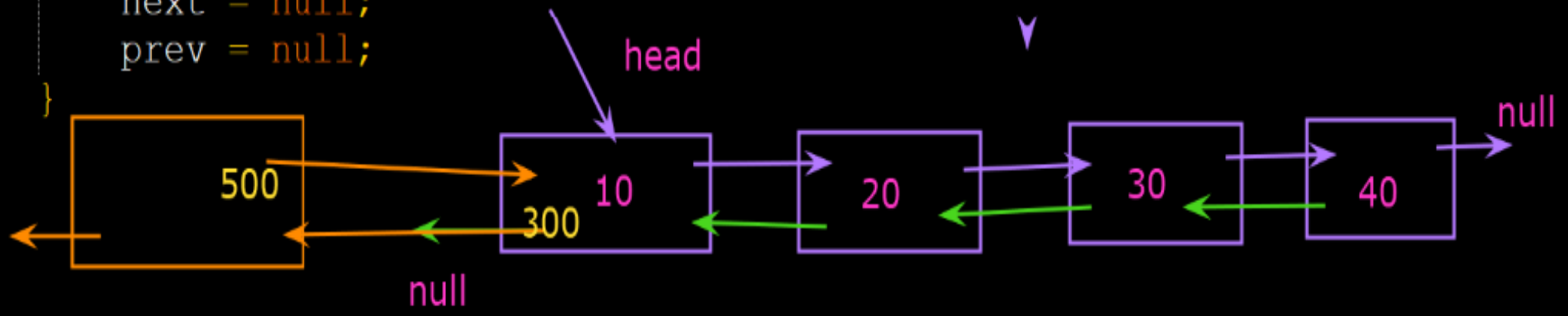
Node structure:

```
class Node{
    int data;
    Node prev;
    Node next;

    Node(int d)
    {
        data = d;
        prev=null;
        next=null;
    }
}
```



```
class DLL{

    Node head;

    static class Node{
        int data;
        Node next,prev;

        Node(int d)
        {
            data = d;
            next = null;
            prev = null;
        }
    }
}
```



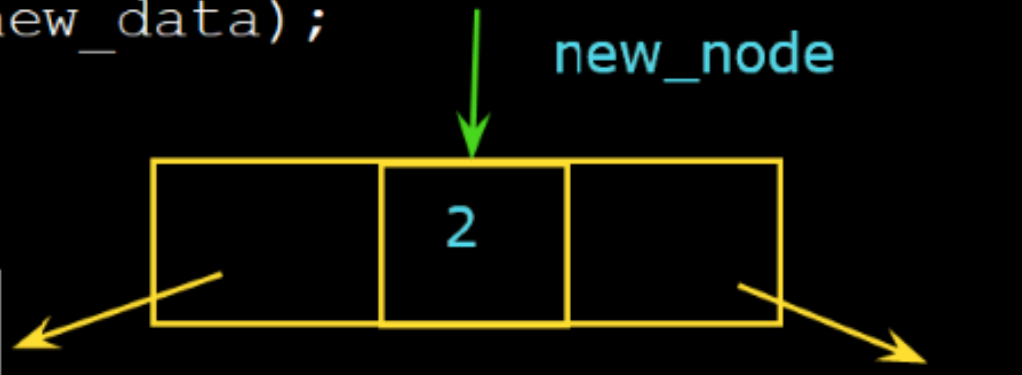


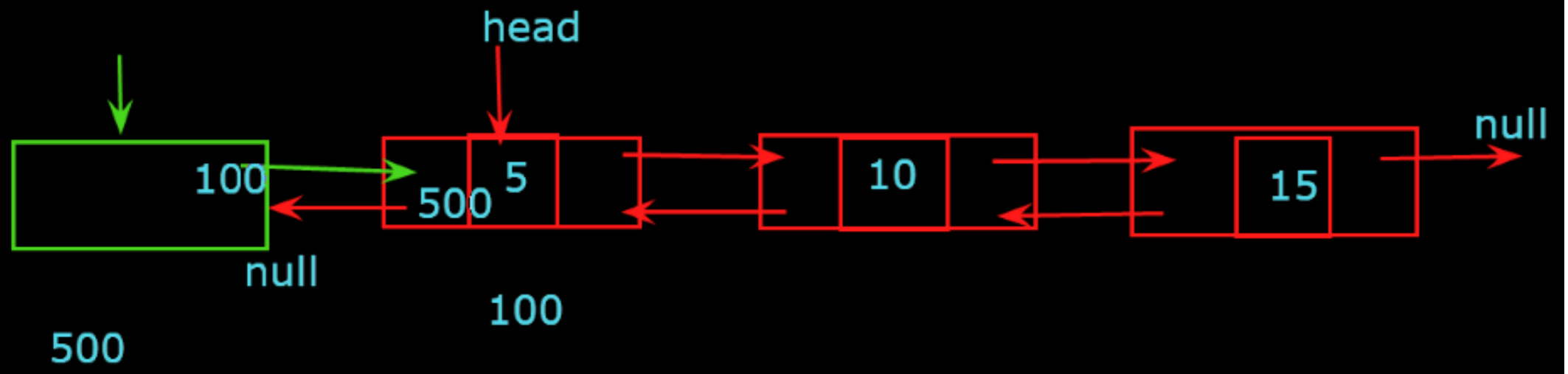
300 500

```java
public static void main(String args[])
{

    DLL1 d1 = new DLL1();
    d1.display(d1.head);
    System.out.println("-----------------------

    d1.insert(10);
    d1.display(d1.head);


    System.out.println();

    d1.insert(5);
    d1.display(d1.head);


}
```

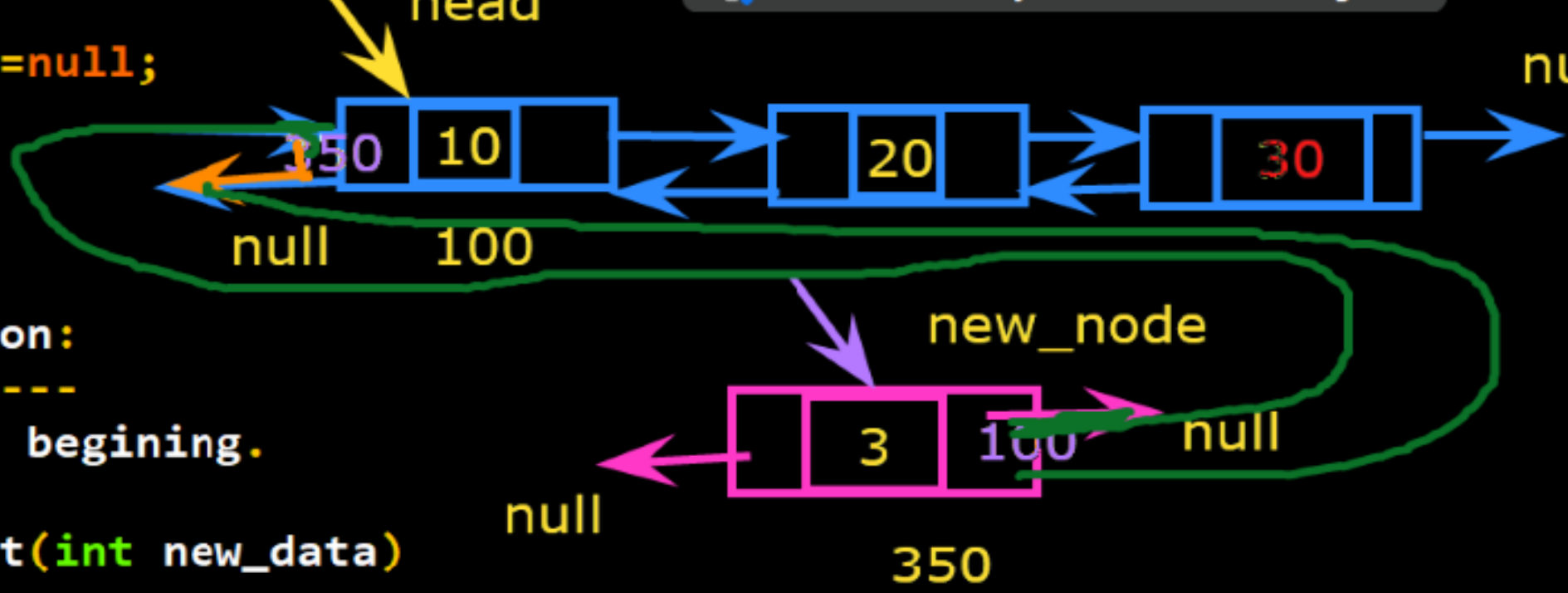

```
C:\Windows\system32\cmd.e:
D:\Test>javac DLL1.java

D:\Test>java DLL1
Forward direction:
----------------------------

Backward direction:
------------------------------**
Forward direction:
10---> ----------------------------

Backward direction:
10--->
Forward direction:
5---> 10---> ----------------------------

Backward direction:
10---> 5--->
D:\Test>
```

```
Node new_node = new Node(new_data);

new_node.next = head;
new_node.prev = null;

if(head != null)
    head.prev = new_node;

head = new_node;
```





```
        data = d;
        prev=next=null;
    }

}
```



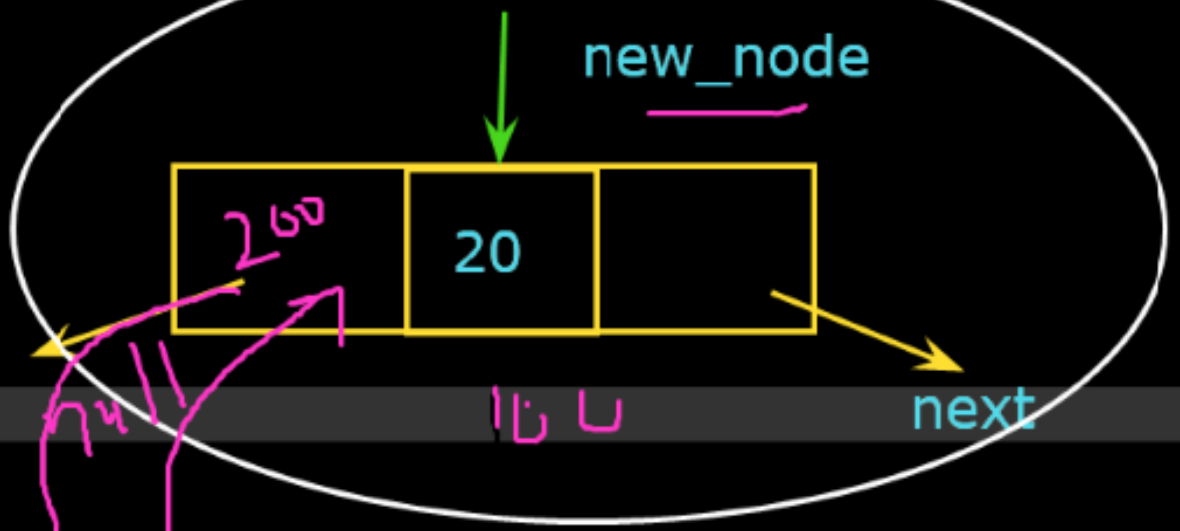
Insertion Operation:

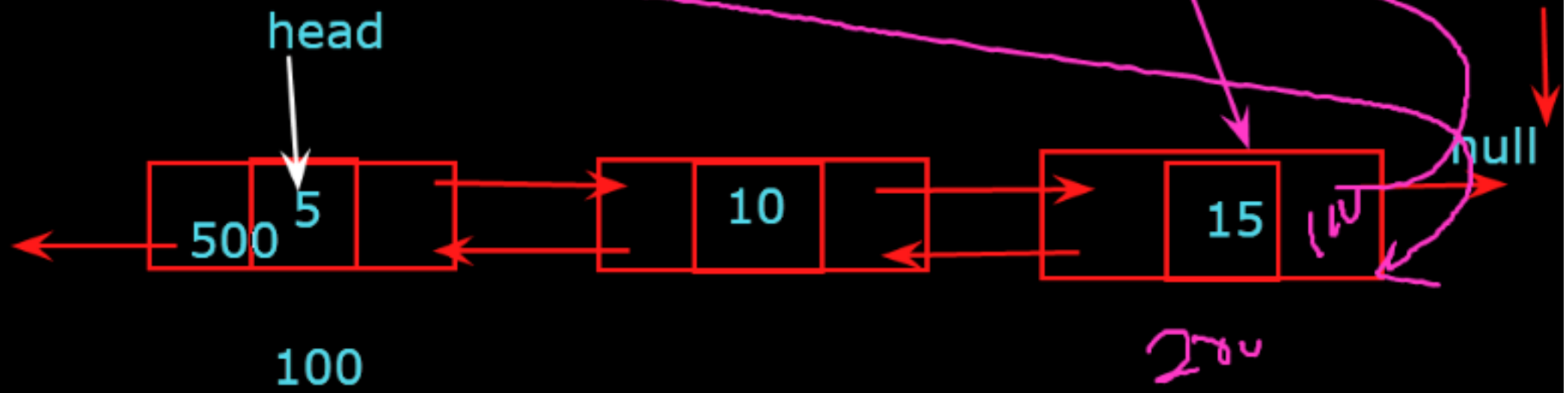
---

case 1: Insert at begining.

```
static void insert(int new_data)
{
    Node new_Node = new Node(new_data);
    new_Node.next = head;
    new_Node.prev = null;
    if(head != null)
        head.prev = new_node;
    head = new_Node;

}
```

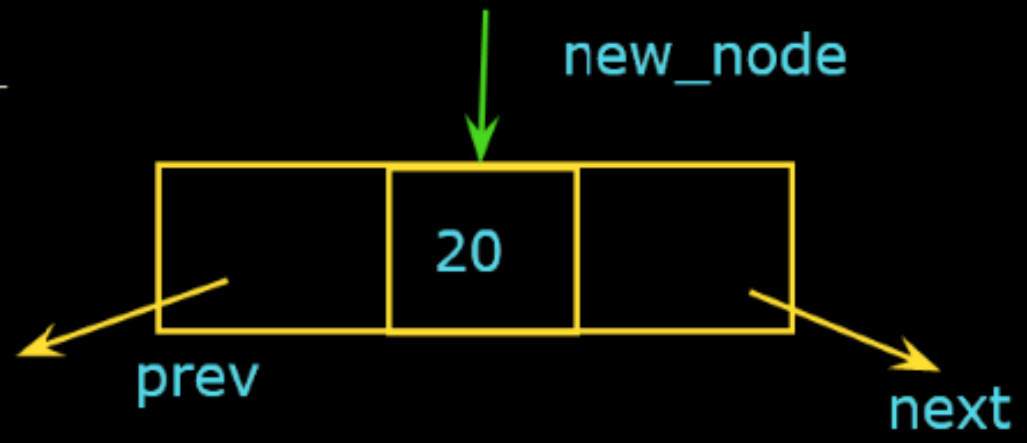
```
void append(int new_data)
{
    Node new_node = new Node(new_data);
    Node n = head;

    if(head == null)// list is empty
    {
        new_node.prev = null;
        head = new_node;

    }

    while(n.next != null)
    {
        n=n.next;
    }
    n.next = new_node;
    new_node.prev = n;

}
```

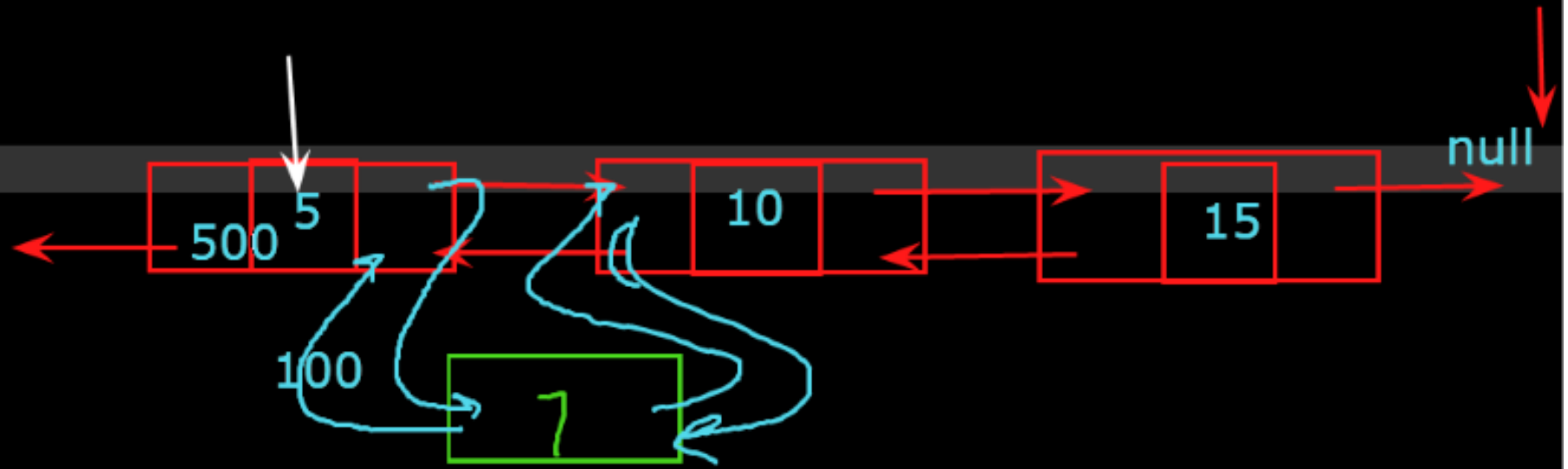




case 3: Insertion in between two nodes.

```
void insertAfter(Node n, int new_node)
{
    if(n == null)
        return;

    Node new_node = new Node(new_data);
    new_node.next = n.next;
    n.next.prev = new_node;
    new_node.prev = n;
    n.next = new_node;


}
```





Thanks