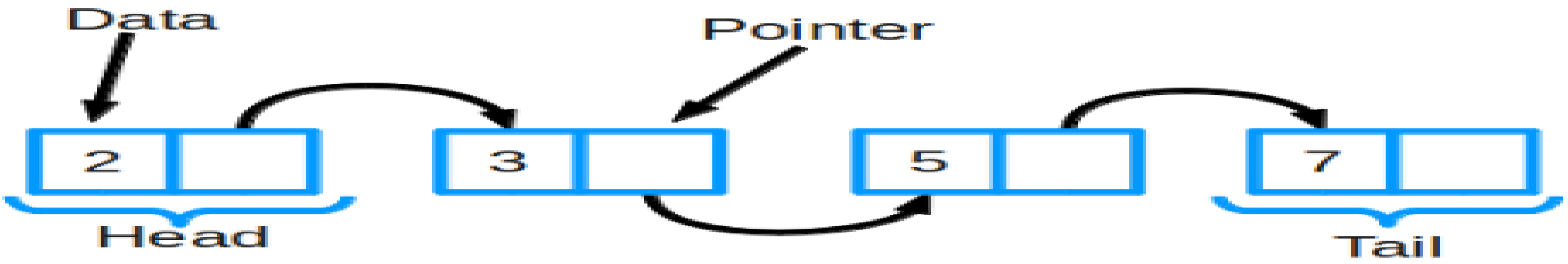
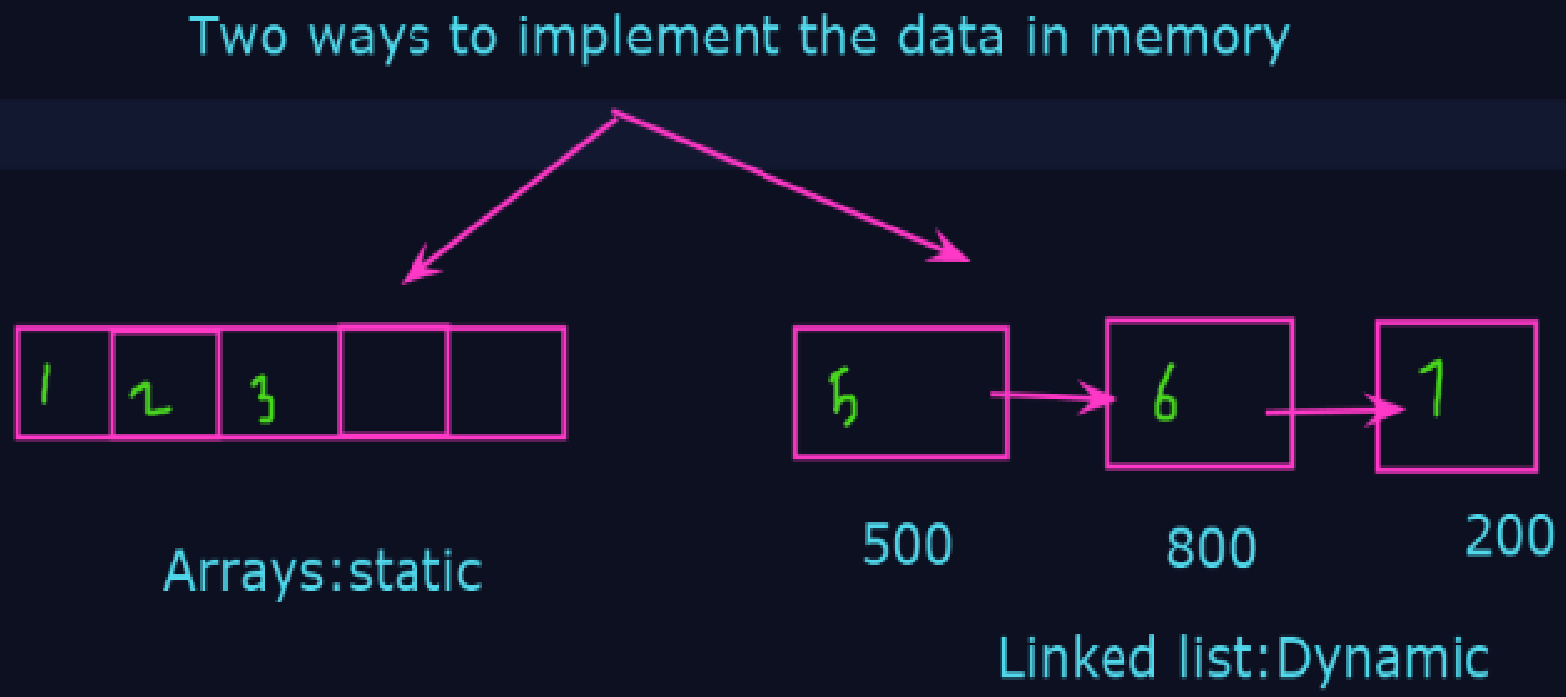
Linked list



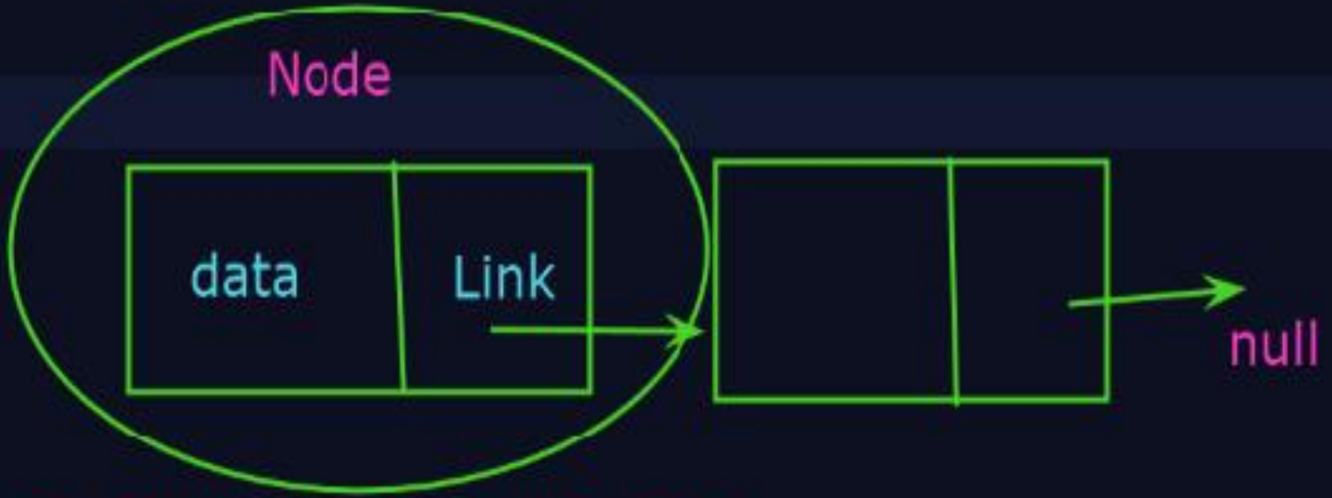
Topics:

- LinkedList



```
------------------------------------------

Topics:
    - LinkedList
```

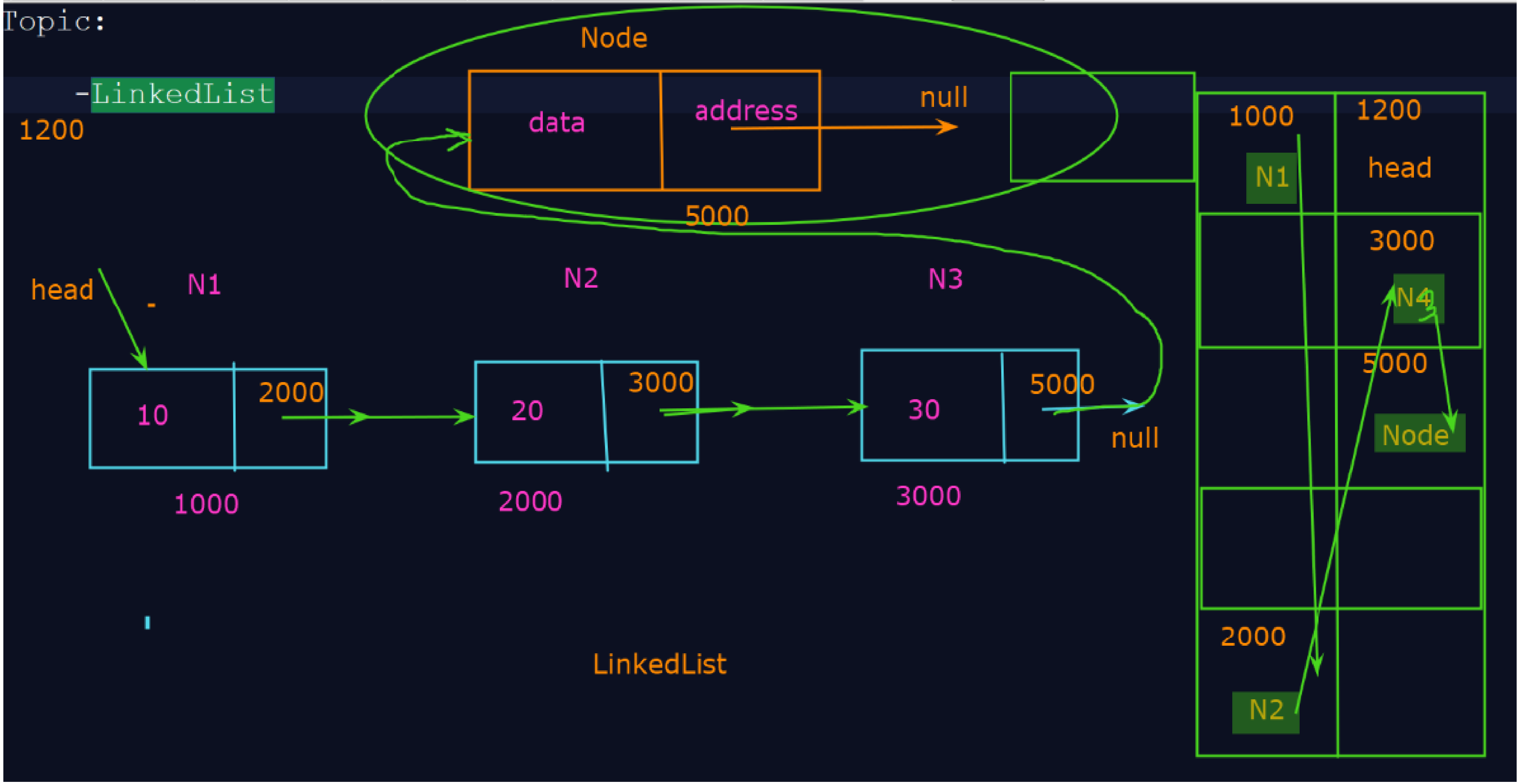


data : contains the actual value

link : contains the address of the next node of the list

Topic:

-LinkedList

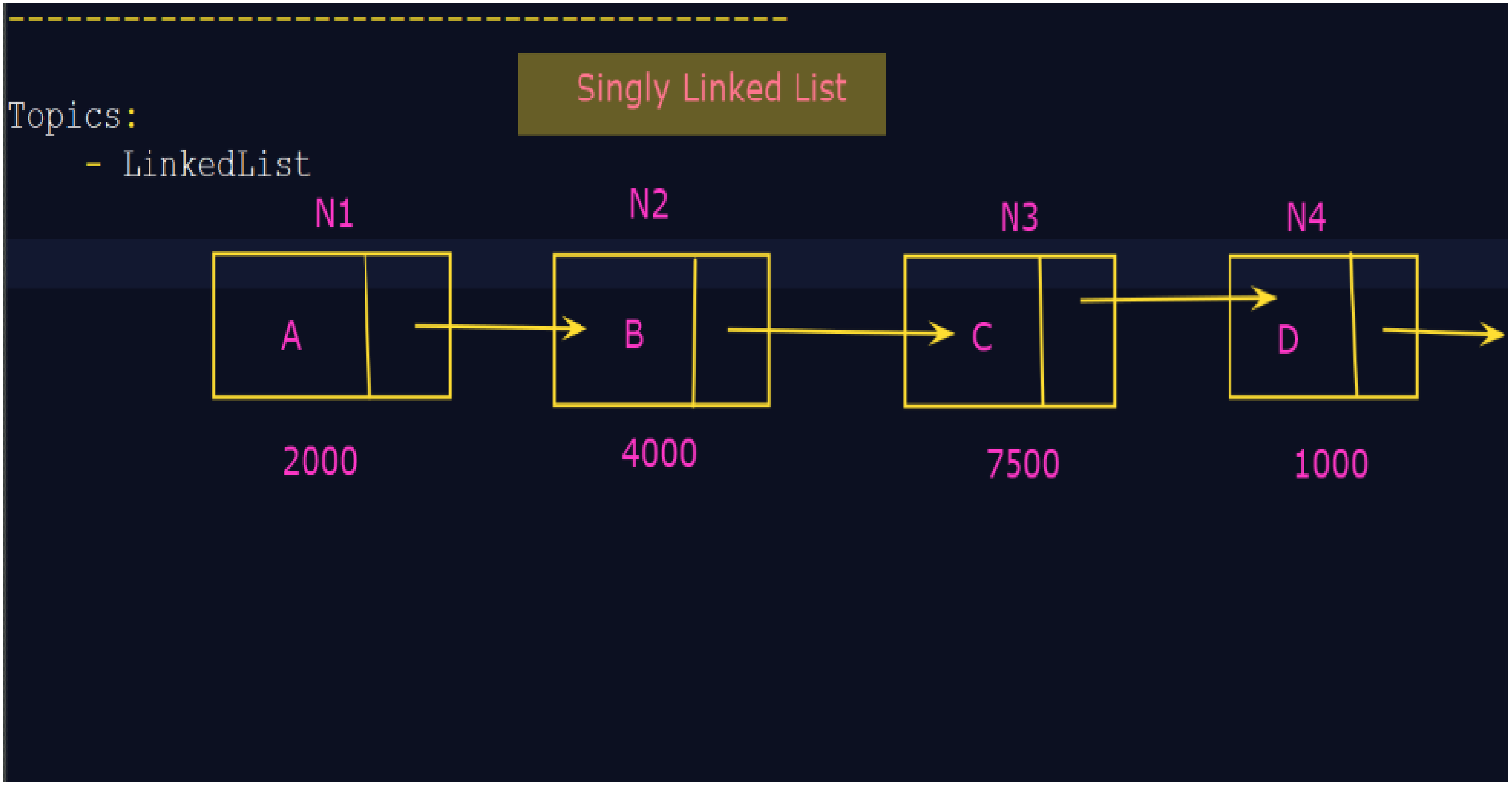


Linked List

- **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. **LinkedLis**t − A Linked List contains the **connection link** to the first link called **First**.



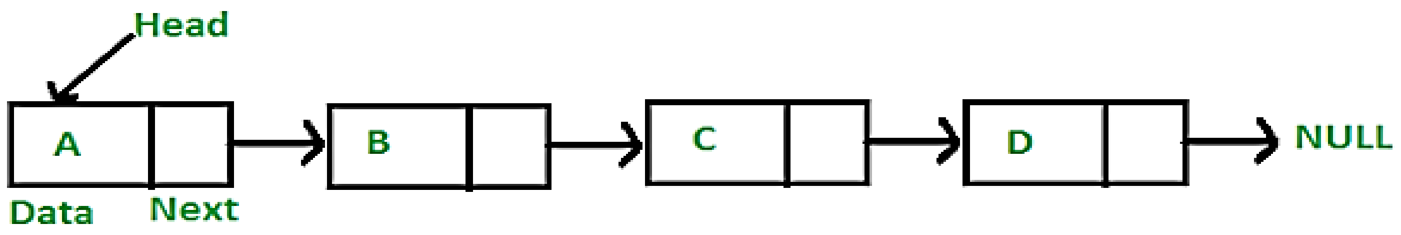
Singly Linked List

Topics:

- LinkedList

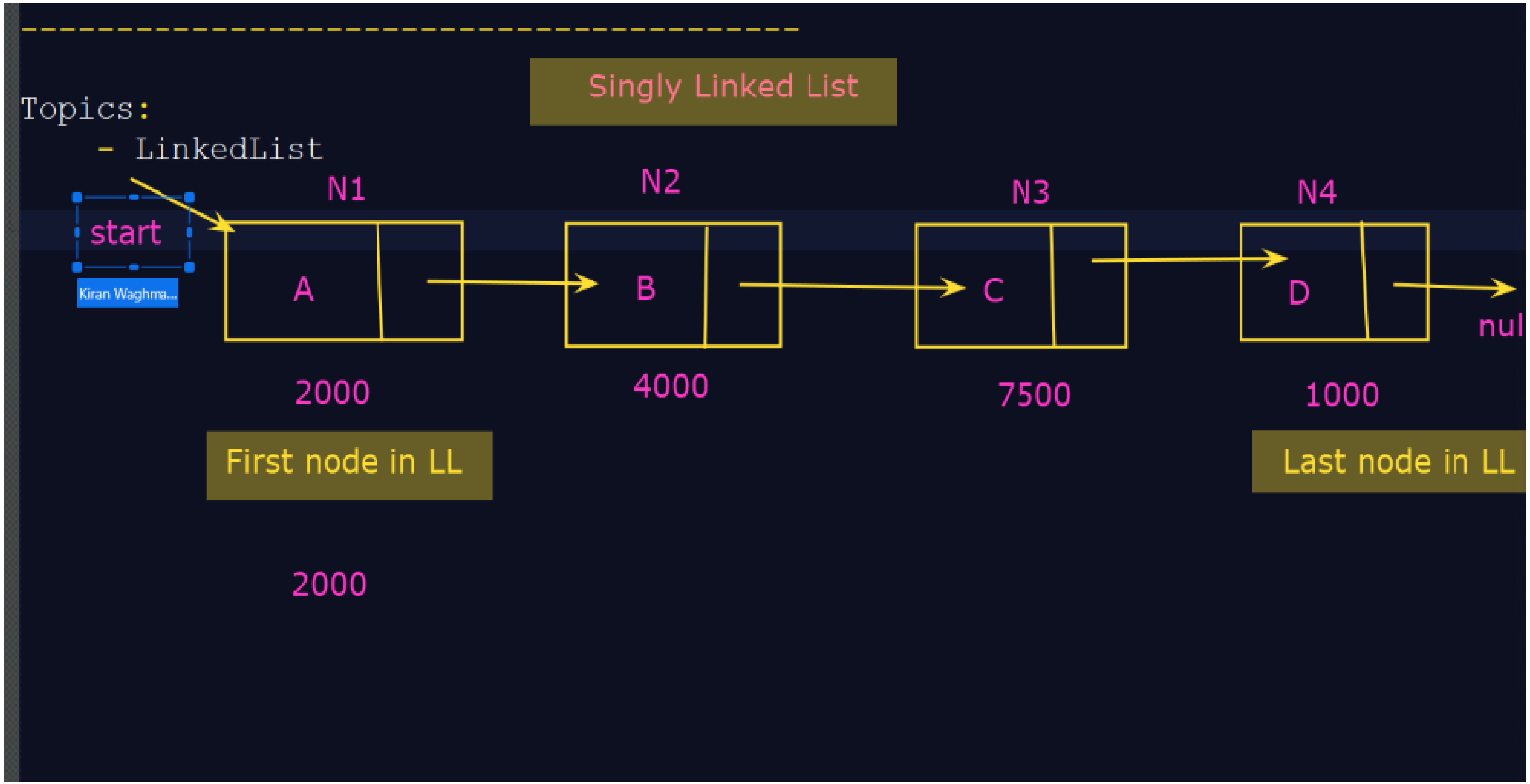
Linked List Representation

- **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.

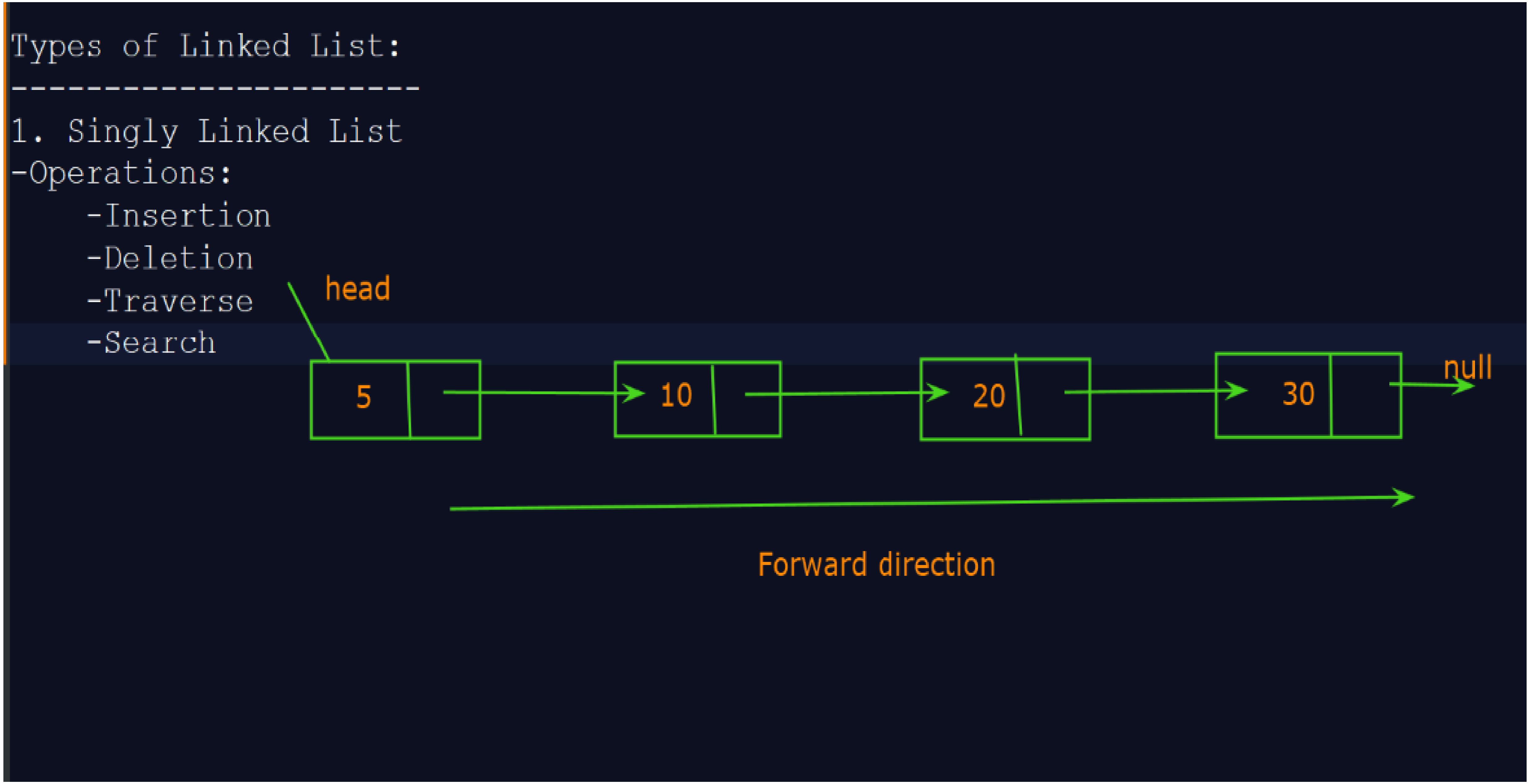


```
----------------------------------------

Topics:
    - LinkedList
```

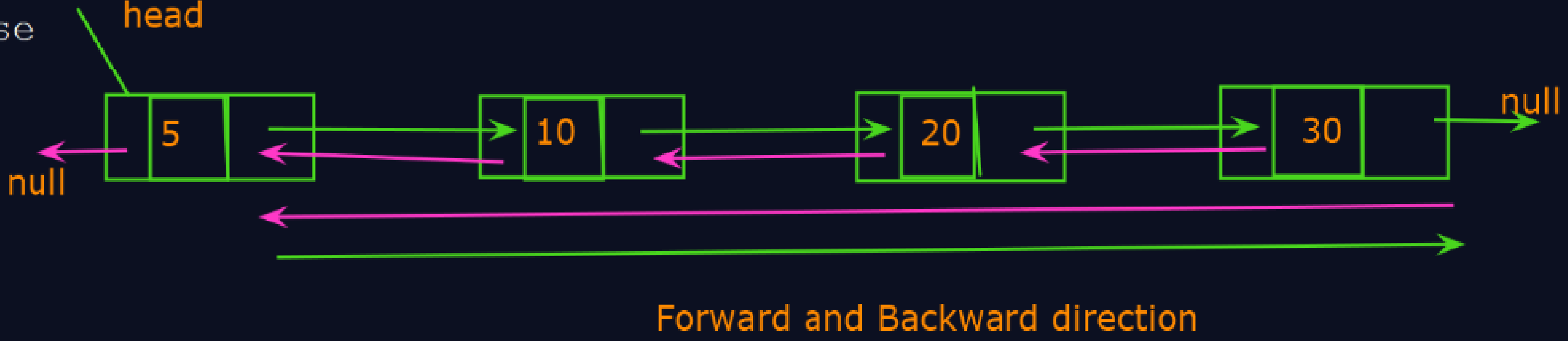
Types of Linked List

- **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.

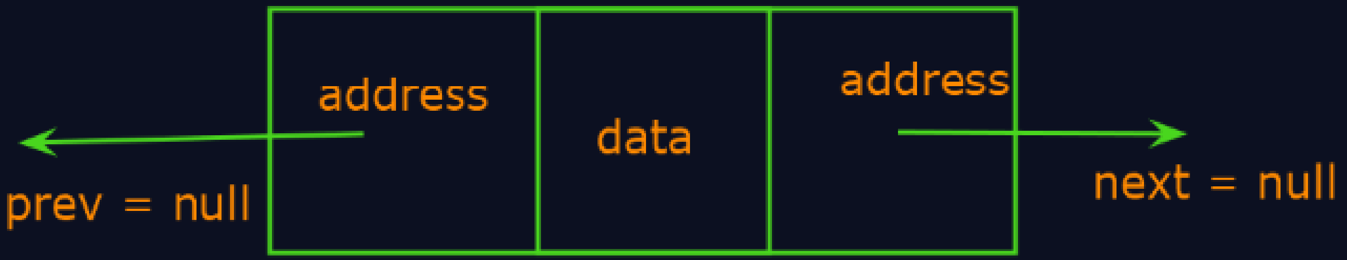


```
Types of Linked List:
----------------------
1. Singly Linked List
-Operations:
    -Insertion
    -Deletion
    -Traverse
    -Search
```

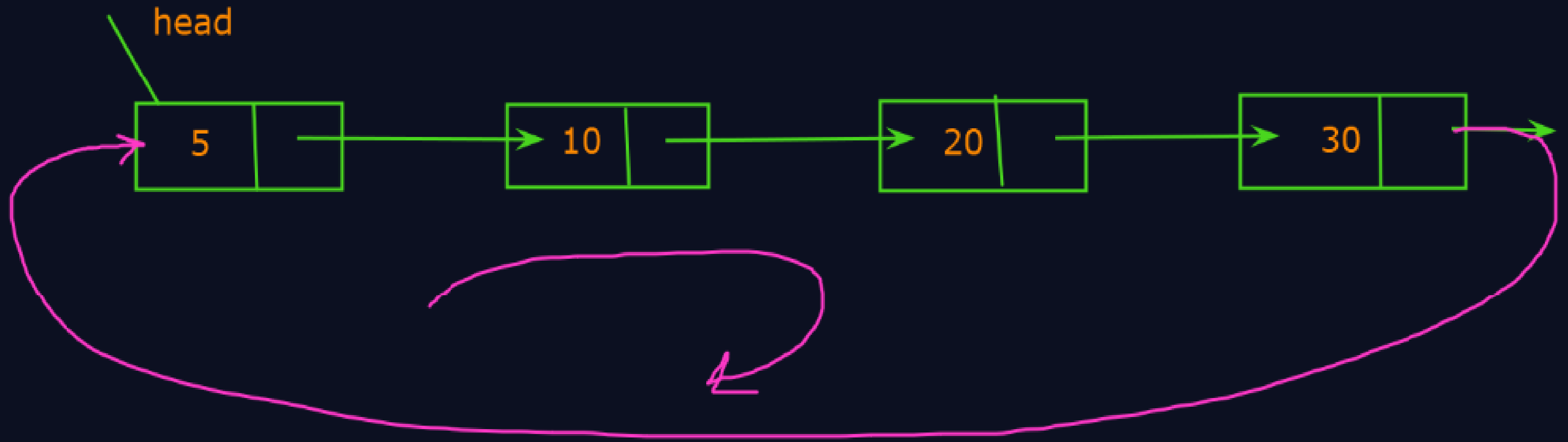
```
Types of Linked List:
-----------------------
1. Singly Linked List
-Operations:
    -Insertion
    -Deletion
    -Traverse
    -Search
```



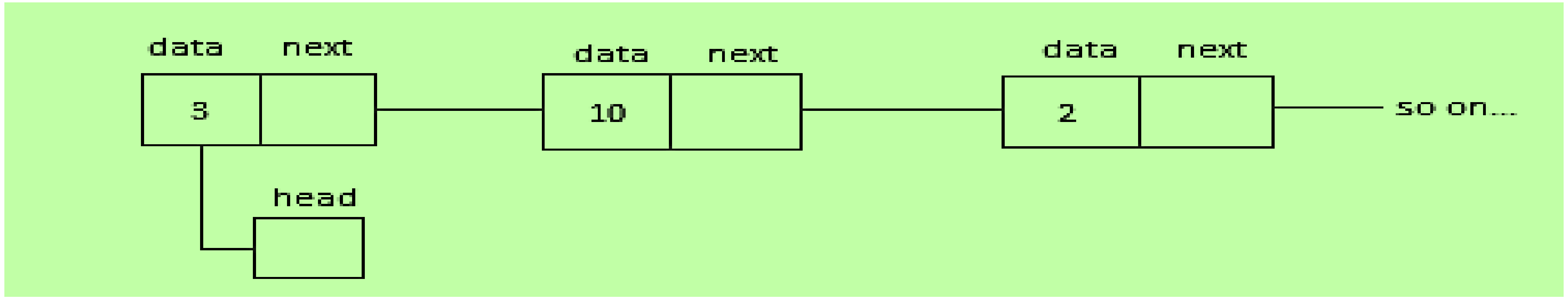
```
2. Doubly Linked List
```



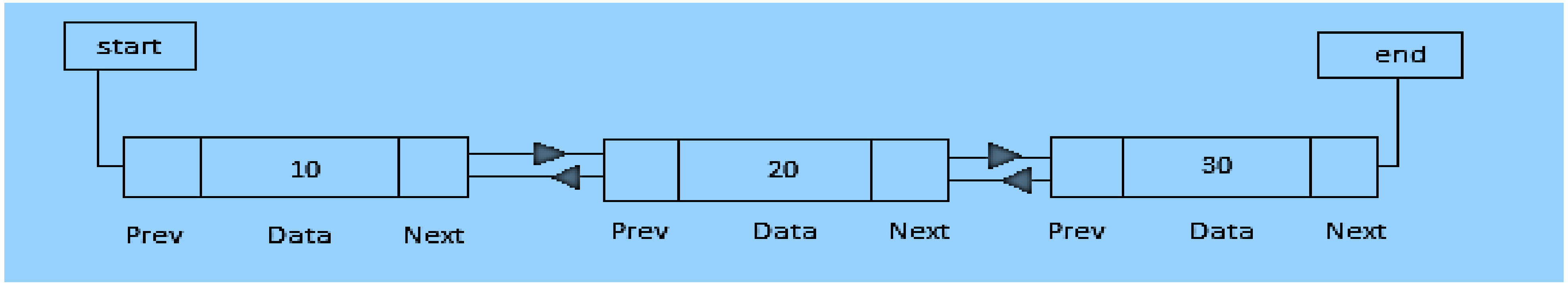
```
3. Circular Linked List:
-Operations:
    -Insertion
    -Deletion
    -Traverse
```



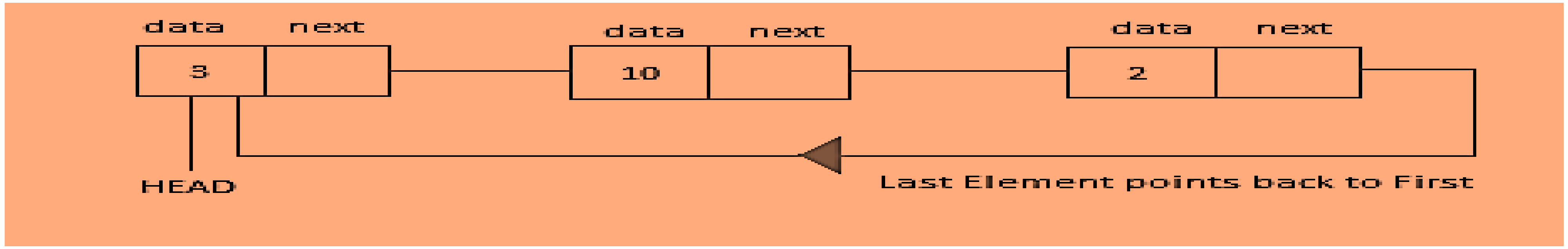
- **Simple Linked List**



- **Doubly Linked List**

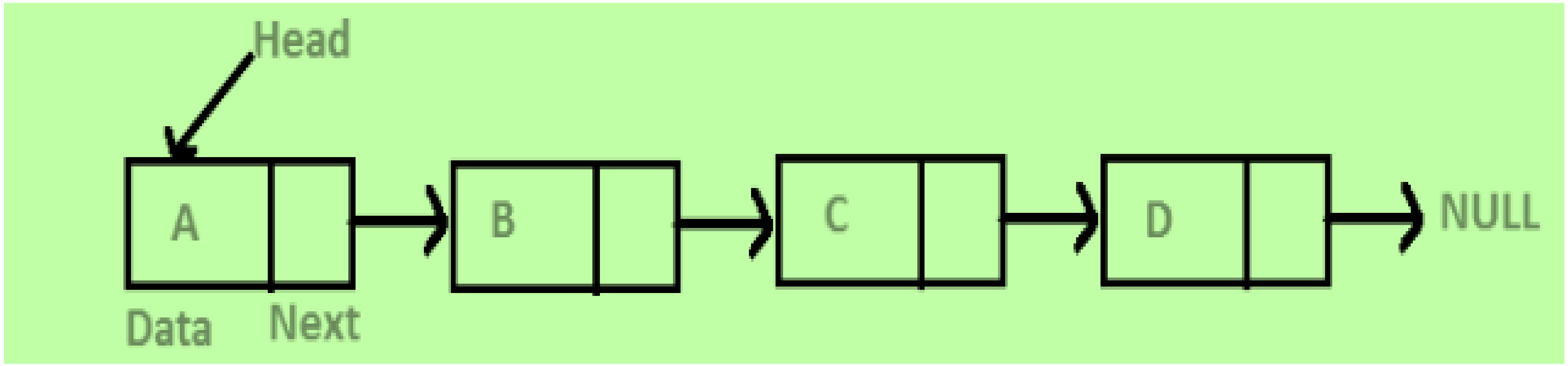


- **Circular Linked List**



Singly Linked List

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



Advantages of Linked Lists

1. They are a dynamic in nature which allocates the memory when required.
2. Insertion and deletion operations can be easily implemented.
3. Stacks and queues can be easily executed.
4. Linked List reduces the access time.

Disadvantages of Linked Lists

1. The memory is wasted as pointers require extra memory for storage.
2. No element can be accessed randomly; it has to access each node sequentially.
3. Reverse Traversing is difficult in linked list.

-

Applications of Linked Lists

1. Linked lists are used to implement stacks, queues, graphs, etc.
2. Linked lists let you insert elements at the beginning and end of the list.
3. In Linked Lists we don't need to know the size in advance.

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.