**📘 Doubly Linked List (DLL)**

**🔹 Definition:**

A **Doubly Linked List** is a linear data structure in which each node contains **three parts**:

1. **Data** – the actual value
2. **Prev** – a pointer to the **previous** node
3. **Next** – a pointer to the **next** node

**🔹 Node Structure:**

None ← [Prev | Data | Next] ⇄ [Prev | Data | Next] ⇄ [Prev | Data | None]

Each node is connected **both forward and backward**.

**🔹 Key Components:**

* **Head**: The first node in the list
* **Prev**: Points to the previous node
* **Next**: Points to the next node
* **None**: Used for the first node's prev and last node's next

**🔹 Operations:**

**✅ 1. Traversal**

* Forward (from head to end)
* Backward (from tail to head)

**✅ 2. Insertion**

* At beginning
* At end
* At a specific position

**✅ 3. Deletion**

* From beginning
* From end
* From specific position

**🔹 Python Code Example**

# Define a Node class

class DNode:

def \_\_init\_\_(self, data):

self.data = data # data part

self.prev = None # pointer to previous node

self.next = None # pointer to next node

# Define Doubly Linked List

class DoublyLinkedList:

def \_\_init\_\_(self):

self.head = None # initially list is empty

# Append a node at the end

def append(self, data):

new\_node = DNode(data)

if not self.head: # if list is empty

self.head = new\_node

return

curr = self.head

while curr.next: # traverse to the last node

curr = curr.next

curr.next = new\_node # link current last node to new node

new\_node.prev = curr # link new node back to last node

# Display the list forward

def display(self):

curr = self.head

while curr:

print(curr.data, end=" ⇄ ")

curr = curr.next

print("None")

**🔹 Example Usage**

dll = DoublyLinkedList()

dll.append(100)

dll.append(200)

dll.append(300)

dll.display()

**🔹 Output**

100 ⇄ 200 ⇄ 300 ⇄ None

**🔹 Advantages:**

* Can be traversed in both directions
* Easier to delete a node (no need to traverse from head)

**🔹 Disadvantages:**

* More memory per node (due to extra pointer)
* Slightly more complex to implement