What is a Linked List?

A **linked list** is a data structure where **each element** (called a **node**) points to the next element in the list. It consists of a sequence of nodes where each node holds:

- **Data:** The actual value.
- **A Reference (Pointer):** To the next node in the list

Visual Example:

If you add values 50, 5, 1, 10 to the linked list in that order, the linked list will look like this:

```
50 -> 5 -> 1 -> 10 -> None
```

Each arrow (->) represents the **next reference pointing** to the next node. **None** means the end of the list.

Code Breakdown Node Class

```python
class Node:
    def __init__(self, data):
        self.data = data  # Store the data value
        self.next = None    # Pointer to the next node
(initially None)
```

 Node is a blueprint for creating elements of the linked list.

 When a Node is created, it:

- **Stores the provided data value.**
- **Sets next to None**, meaning it doesn't yet point to another node.

Example:

```
node = Node(10)
print(node.data)   # Output: 10
print(node.next)   # Output: None
```

LinkedList Class

```
class LinkedList:
    def __init__(self):
        self.head = None  # Initialize the head of the list as
None
```

 LinkedList **manages the collection of Node** objects.

 self.head keeps **track** of the **first node** in the list. **When the list is empty, head is None.**

Method to Insert at the Beginning

```
def insertAtBeginning(self, new_data):
    new_node = Node(new_data)   # Create a new node
with the given data
    new_node.next = self.head    # Link new node to
current head
    self.head = new_node          # Update head to the
new node
```

Step-by-Step:

1. **Create a Node:** new_node is created with the value new_data.
2. **Link to the List:** new_node.next = self.head connects the new node to the previous first node.
3. **Update the Head:** self.head = new_node sets the new node as the first element in the list.

Example:

Let's insert 10, then 1, then 5, then 50:

- **Insert 10:** 10 -> None
- **Insert 1:** 1 -> 10 -> None
- **Insert 5:** 5 -> 1 -> 10 -> None
- **Insert 50:** 50 -> 5 -> 1 -> 10 -> None

Method to Print the List

```
def printList(self):
    temp = self.head #Start from the head of the list
    while temp:
        print(str(temp.data) + " ", end=" ")
        temp = temp.next  # Move to the next node
```