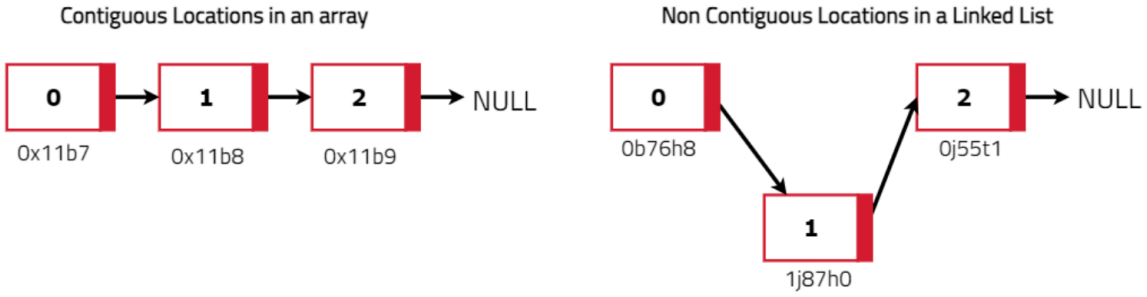
Imagine a train with a series of carriages connected together. Each carriage can be added or removed independently without disturbing the others. This is similar to a linked list, where each element (carriage) points to the next, allowing for flexible addition and removal.
In contrast, think of an array as a train where all carriages are welded together. Adding or removing a carriage in the middle would require shifting all the subsequent carriages, making the process laborious and inefficient.
So, if you frequently need to add or remove carriages (elements) in your application, a linked list is like having a train with detachable carriages, providing greater flexibility and efficiency.

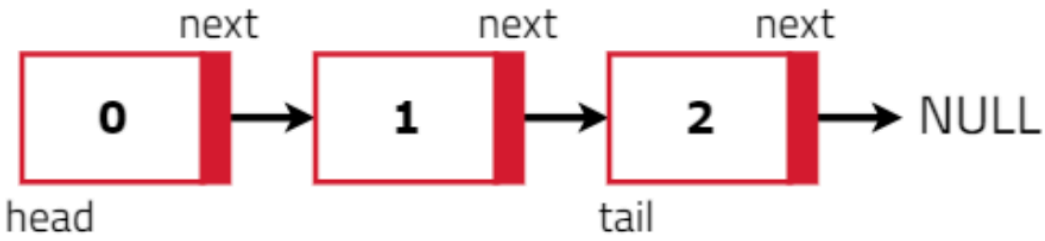
What is a Linked List ?

A linked list is a linear data structure resembling a chain, where each node is connected to the next, and each node represents an individual element. Unlike arrays, the elements in a linked list are not stored in contiguous memory locations.

In arrays, adding a new element requires the next memory location to be empty, which cannot always be guaranteed. Therefore, expanding an array beyond its initial size can be challenging and inefficient. This limitation is not present in linked lists, which can dynamically grow and shrink as needed.



A linked list is a data structure containing two crucial pieces of information, the first being the data and the other being the pointer to the next element. The ‘head’ is the first node, and the ‘tail’ is the last node in a linked list.



A Linked List

Why Linked List over arrays ?

Unlike arrays, the size of the Linked List can be decreased or increased at any location and at any point of time efficiently.

Difference Between Struct and Class in Linked Lists

Aspect	Struct	Class
Definition	A struct is a user-defined data type that groups together different data types to form a single unit. It is primarily used for simple data structures.	A class is a blueprint for creating objects, providing more advanced features such as inheritance, encapsulation, and polymorphism.
Access Control	By default, all members of a struct are public.	By default, all members of a class are private.
Usage in Linked Lists	Often used for defining the nodes in a simple linked list where each node typically contains data and a pointer to the next node.	Used for defining more complex linked list structures that require encapsulation and additional functionality, such as methods for insertion, deletion, and traversal.
Memory Management	Typically used in simpler scenarios with straightforward memory management, often using stack allocation.	More suitable for dynamic memory management using heap allocation, allowing for more control over the lifecycle of linked list nodes.

Creating a Linked List

```
import java.util.*;

class Node {
    public int data;
    public Node next;

    public Node(int data, Node next) {
        this.data = data;
        this.next = next;
    }
}

public class Main {
    public static void main(String[] args) {
        ArrayList<Integer> arr = new ArrayList<>();
        arr.add(2);
        arr.add(5);
        arr.add(8);
        arr.add(7);

        /*
         * Assigning values to
         * the nodes
         */
        Node y1 = new Node(arr.get(0), null);
        Node y2 = new Node(arr.get(1), null);
        Node y3 = new Node(arr.get(2), null);
        Node y4 = new Node(arr.get(3), null);

        /*
         * Linking of
```

```
        * Nodes
        */
        y1.next = y2;
        y2.next = y3;
        y3.next = y4;

        /*
        * Printing Nodes with their
        * values and data
        */
        System.out.println(y1.data + " " + y1.next);
        System.out.println(y2.data + " " + y2.next);
        System.out.println(y3.data + " " + y3.next);
        System.out.println(y4.data + " " + y4.next);
    }
}
```

Let's break this code to understand how it works:

The struct has two data types: data which contains the value of the node and a pointer next, which points to the next node in the list.
There is a constructor which assigns the values to a new node.
A new keyword is used to dynamically allocate memory to a node with data as arr[0].
The combination of the given parameters and functions initializes a linked list.