Linked list Data Structure

In this tutorial, you will learn about linked list data structure and it's implementation in Python, Java, C, and C++.

A linked list is a linear data structure that includes a series of connected nodes. Here, each node stores the **data** and the **address** of the next node. For example,

Linked list Data Structure

You have to start somewhere, so we give the address of the first node a special name called `HEAD`. Also, the last node in the linked list can be identified because its next portion points to `NULL`.

Linked lists can be of multiple types: **singly**, **doubly**, and **circular linked list**. In this article, we will focus on the **singly linked list**. To learn about other types, visit Types of Linked List (https://www.programiz.com/dsa/linked-list-types).

> **Note:** You might have played the game Treasure Hunt, where each clue includes the information about the next clue. That is how the linked list operates.

- An address of another node

We wrap both the data item and the next node reference in a struct as:

```
struct node
{
  int data;
  struct node *next;
};
```

Understanding the structure of a linked list node is the key to having a grasp on it.

Each struct node has a data item and a pointer to another struct node. Let us create a simple Linked List with three items to understand how this works.

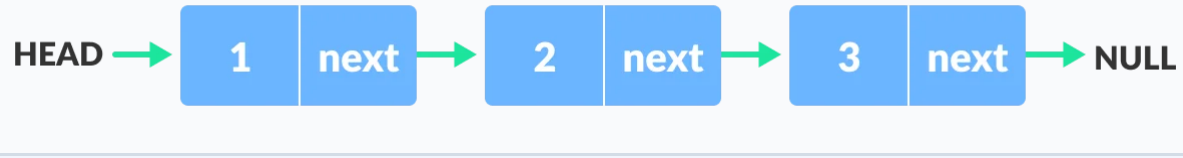
```
/* Initialize nodes */
struct node *head;
struct node *one = NULL;
struct node *two = NULL;
struct node *three = NULL;

/* Allocate memory */
one = malloc(sizeof(struct node));
two = malloc(sizeof(struct node));
three = malloc(sizeof(struct node));

/* Assign data values */
one->data = 1;
two->data = 2;
three->data=3;

/* Connect nodes */
one->next = two;
two->next = three;
three->next = NULL;

/* Save address of first node in head */
head = one;
```



Linked list Representation

The power of a linked list comes from the ability to break the chain and rejoin it. E.g. if you wanted to put an element 4 between 1 and 2, the steps would be:

- Create a new struct node and allocate memory to it.
- Add its data value as 4
- Point its next pointer to the struct node containing 2 as the data value
- Change the next pointer of "1" to the node we just created.

Doing something similar in an array would have required shifting the positions of all the subsequent elements.

In python and Java, the linked list can be implemented using classes as shown in [the codes below](https://www.programiz.com/dsa/linked-list#code).

Apart from that, linked lists are a great way to learn how pointers work. By practicing how to manipulate linked lists, you can prepare yourself to learn more advanced data structures like graphs and trees.

Linked List Implementations in Python, Java, C, and C++ Examples

Python | Java | C | C++

```
    def __init__(self, item):
        self.item = item
        self.next = None


class LinkedList:

    def __init__(self):
        self.head = None


if __name__ == '__main__':

    linked_list = LinkedList()

    # Assign item values
    linked_list.head = Node(1)
    second = Node(2)
    third = Node(3)

    # Connect nodes
    linked_list.head.next = second
    second.next = third
```

Linked List Complexity

Time Complexity

	Worst case	Average Case
Search	O(n)	O(n)
Insert	O(1)	O(1)
Deletion	O(1)	O(1)

Space Complexity: `O(n)`

- Implemented in stack and queue
- In **undo** functionality of softwares
- Hash tables, Graphs

Recommended Readings

1. Tutorials

- Linked List Operations (Traverse, Insert, Delete) (/dsa/linked-list-operations)
- Types of Linked List (/dsa/linked-list-types)
- Java LinkedList (/java-programming/linkedlist)

2. Examples

- Get the middle element of Linked List in a single iteration (/java-programming/examples/get-middle-element-of-linkedlist)
- Convert the Linked List into an Array and vice versa (/java-programming/examples/linkedlist-array-conversion)
- Detect loop in a Linked List (/java-programming/examples/detect-loop-in-linkedlist)

Next Tutorial:
Linked List Operations → **(/dsa/linked-list-operations)**

Previous Tutorial:
Deque **(/dsa/deque)**

Did you find this article helpful?

Related Tutorials

DS & Algorithms

Types of Linked List - Singly linked, doubly linked and circular

(/dsa/linked-list-types)

DS & Algorithms

Linked List Operations: Traverse, Insert and Delete

(/dsa/circular-linked-list)

DS & Algorithms

Doubly Linked List

(/dsa/doubly-linked-list)