Topic _____ Exp. PG. #_____

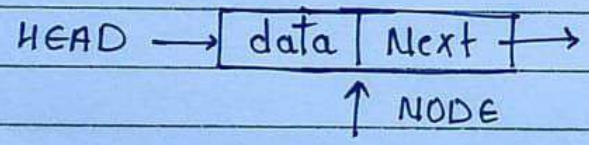
Date ___/___/___

Linked lists In Data-Structure

By: @ Curious_.Programmer

* **What is linked list?**

It is a linear data structure that consist of Nodes. In which elements are not stored at contiguous memory location. The element is linked using pointer.



* **Why we use linked list?**

(I) → It is a dynamic data structure. It can easly (grow/shrink) during runtime.

(II) → No Memory Wastage.

(III) → Insertion and deletion is easy as no shift is needed.

Types of linked list.

1. Singly linked list.
2. Doubly linked list.
3. Circular linked list.

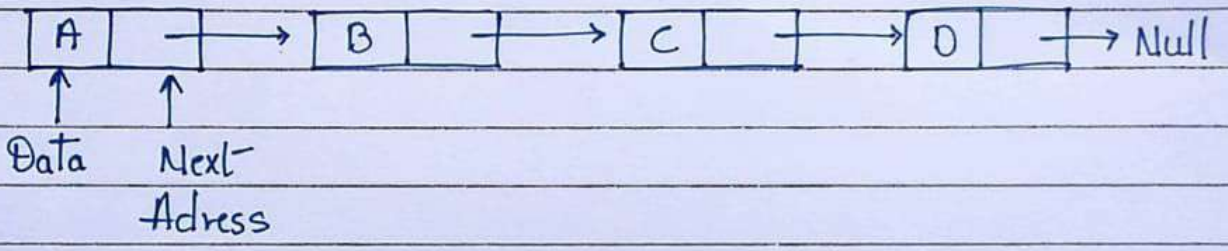
Teacher's Signature_____

Exp. PG. #_____

Topic _____ Date__/__/__

1. What is Singly linked list.

A Singly linked list is a linear data structure in which the elements are not stored in contiguous memory location and each element is connected only to its next element using a pointer.

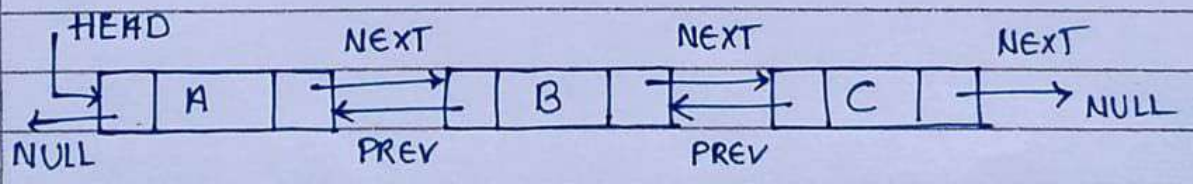


Singly linked lists.

curious_.programmer

2. What is doubly linked list.

A doubly linked list (DDL) is a special type of linked list in which each node contain a pointer to the previous node as well as the next node of the linked list.



Doubly linked list.

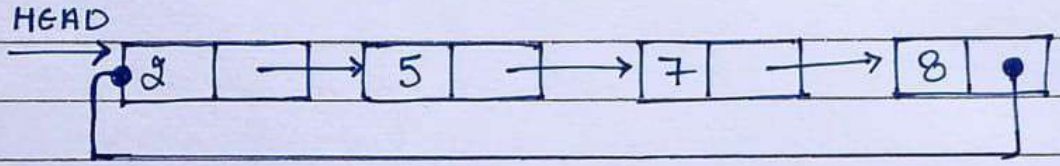
Teacher's Signature_____

Topic _____ Exp. PG. #_____

Date__/__/__

3. What is Circular linked list:

The Circular linked list is a linked list where all nodes are connected to form a Circle. In Circular linked list, the first node and the last node are connected to each other which forms a Circle. There is no NULL at the end.



Curious_programmer

* Operation on linked lists.

1. Insertion in linked list.

It is used to insert new node at a specific positions.

You can add a node at the beginning, middle, and end.

* Insert at the beginning

1. Create a memory for a new node.
2. Store data in a new node.
3. Change Next to the new node to point to start.
4. Change starts to tell the recently created node.

Teacher's Signature_____

Topic _____ Exp. PG. #_____

Date ___/___/___

```
struct node * NewNode;
NewNode = malloc (sizeof (structnode));
NewNode -> data = 40;
NewNode -> next = start;
start = NewNode;
```

* Insert at the End. Curious_.programmer

1. Insert a new node and store data in it.
2. Traverse the last node of a linked list.
3. Change the next pointer of the last node to the newly created node.

```
struct node * NewNode;
NewNode = malloc (sizeof (struct node));
NewNode -> data = 40;
NewNode -> next = NULL;
struct node * temp = start;
while (temp -> next != NULL) {
    temp = temp -> next;
}
temp -> next = NewNode;
```

* Insert at the Middle.

1. Allocate Memory and store data in the new node.
2. Traverse the Node, which is just before the new node.

Teacher's Signature_____

Topic _____ Exp. PG. #_____

Date ___/___/___

3. Change the next pointer to add a new node in between.

```
struct node * NewNode;
NewNode = malloc ( sizeof ( struct node));
NewNode -> data = 40;
Struct node -> temp = start;
for (int i= 2; i < position ; i++ ) {
    if (temp -> next ! = NULL)
        temp = temp -> next;
}}

NewNode -> Next = temp -> next;
temp -> next = NewNode;
```

Curious_.programmer

2. Deletion in linked list.

It is used to delete nodes from a specific positions. You can also do deletion in the linked list in three ways either from the end, beginning, or from a specific position.

* Delete from the beginning.

1. The point starts at the second node.

```
start = start -> next;
```

Teacher's Signature_____

Topic _____ Exp. PG. #_____

Date ___/___/___

* Delete from the End.

1. Traverse the second last element in the linked list.
2. Change its next pointer to null.

```
struct node *temp = start;
While (temp -> next -> next != NULL){
        temp = temp -> next;
}
temp -> next = NULL;
```

* Delete from the Middle.

Curious_programmer

1. Traverse the element before the element to be deleted.
2. Change the next pointer to exclude the node from the linked list.

```
for (int i= 2; i, position; i++){
  if (temp -> next != NULL)
      temp = temp -> next;
}
}
temp -> next = temp -> next -> next;
```

Teacher's Signature_____

Topic _____ Exp. PG. #_____

Date ___/___/___

3. Traversing in linked list.

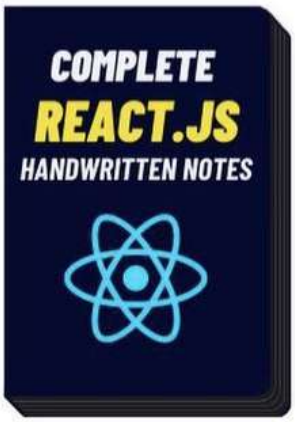
It is used to traverse all nodes one by one. In this operation, you will display all the nodes in the linked list.

When the temp is null, it means you traversed all the nodes, and you reach the end of the linked list and get out from the while loop.

```
Struct node * temp = start;
printf("\n list empty are-");
    While (temp != NULL)
        {
printf("%d", temp -> data)
    temp = temp -> next;
        }
```

Teacher's Signature_____

Buy This Complete Premium Notes @ Rs. 149



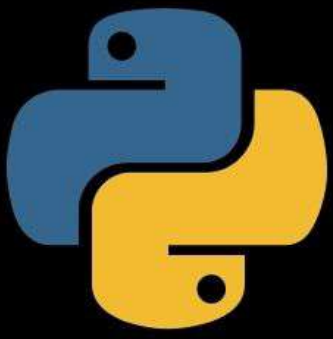
LINK IN BIO

5. List, Dictionary, Set, Tuples & Type Conversion

Made By:

Yadneyesh (Curious Coder)
CodWithCurious.com

Find More PDFs on Our Telegram Channel
Search Curious_Coder on Telegram



Chapter 5 will Be Upload In Some Days

Find All PDFs in Our Telegram Channels