Data Structures

BS (CS) _Fall_2024

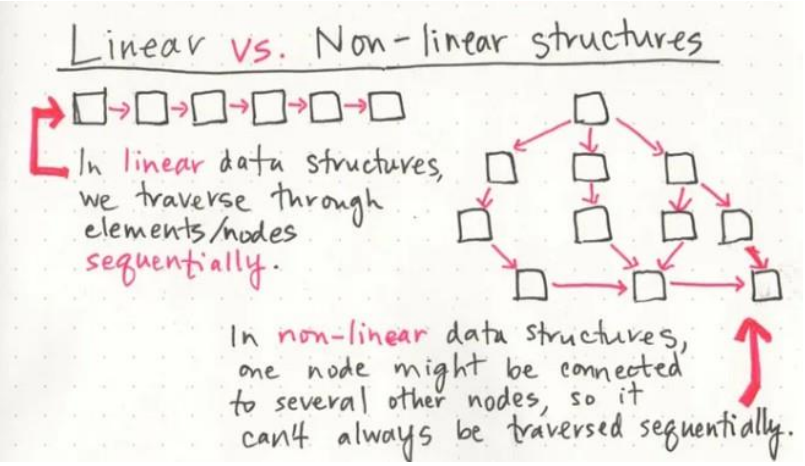
Lab_06 Manual

Learning Objectives:

1. Singly Linked List

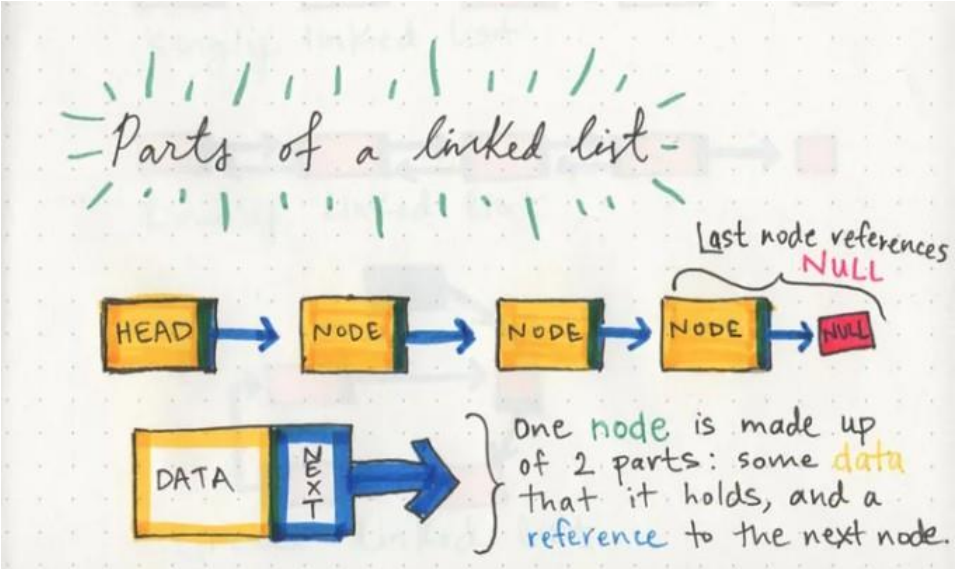
What is a Linked List?

linked lists are linear data structures, meaning there is a sequence and an order to how they are constructed and traversed. In order to get to the end of the list, we have to go through all of the items in the list in order, or sequentially. Linear structures, however, are the opposite of non-linear structures. In non-linear data structures, items don't have to be arranged in order, which means that we could traverse the data structure non-sequentially.



Parts of a Linked List:

A linked list is made up of a series of nodes, which are the elements of the list.The starting point of the list is a reference to the first node, which is referred to as the head. Nearly all linked lists must have a head, because this is effectively the only entry point to the list and all of its elements, and without it, you wouldn't know where to start! The end of the list isn't a node, but rather a node that points to null, or an empty value.



A single node is also pretty simple. It has just two parts: data, or the information that the node contains, and a reference to the next node.

A Simple Linked List Class:

- We use two classes: **Node** and **List**
- Declare Node class for the nodes
  - data: double-type data in this example
  - next: a pointer to the next node in the list

```
class Node {
public:
    double  data;    // data
    Node*   next;    // pointer to next
};
```

- Declare List, which contains
  - head: a pointer to the first node in the list.

Since the list is empty initially, head is set to NULL

```
class List {
public:
    List(void) { head = NULL; }   // constructor
    ~List(void);                  // destructor

    bool IsEmpty() { return head == NULL; }
    Node* InsertNode(int index, double x);
    int FindNode(double x);
    int DeleteNode(double x);
    void DisplayList(void);
private:
    Node* head;
};
```