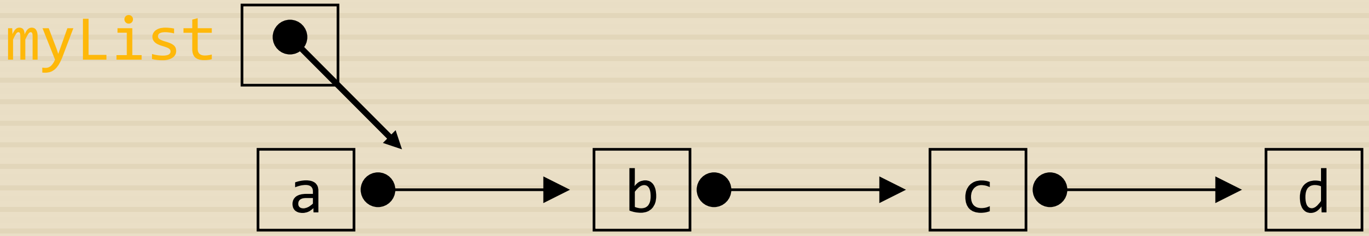
LINKED LIST

A **LINKED LIST** IS A DATA STRUCTURE CONSISTING OF A GROUP OF NODES WHICH TOGETHER REPRESENT A SEQUENCE.

Linked List

- A linked list consists of:
  - A sequence of nodes



Each node contains a value

and a link (pointer or reference) to some other node

The last node contains a null link

The list may (or may not) have a header

Basic Terminology

A node's successor is the next node in the sequence

The last node has no successor

A node's predecessor is the previous node in the sequence

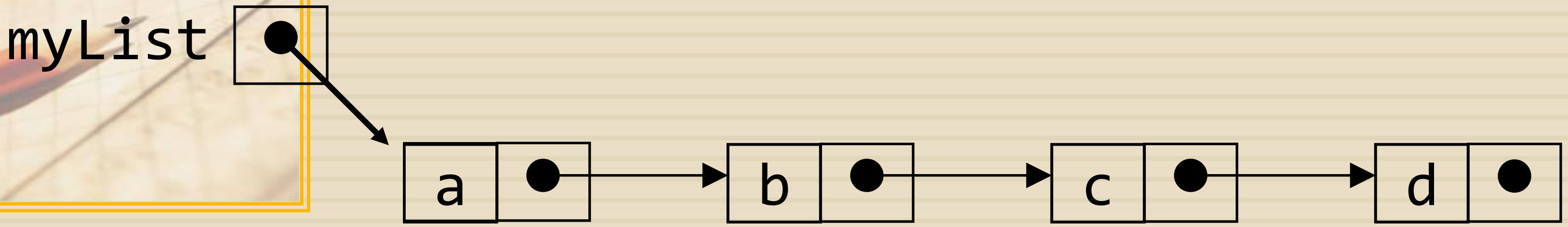
The first node has no predecessor

A list's length is the number of elements in it

A list may be empty (contain no elements

Singly-linked lists

- Here is a singly-linked list (SLL)

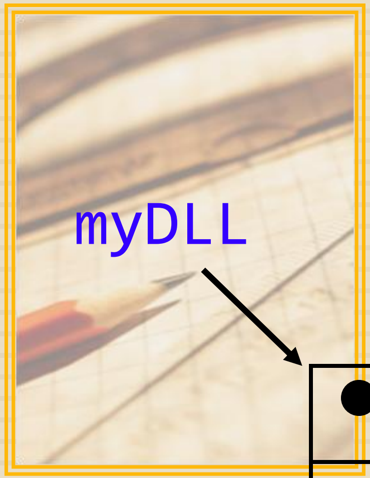


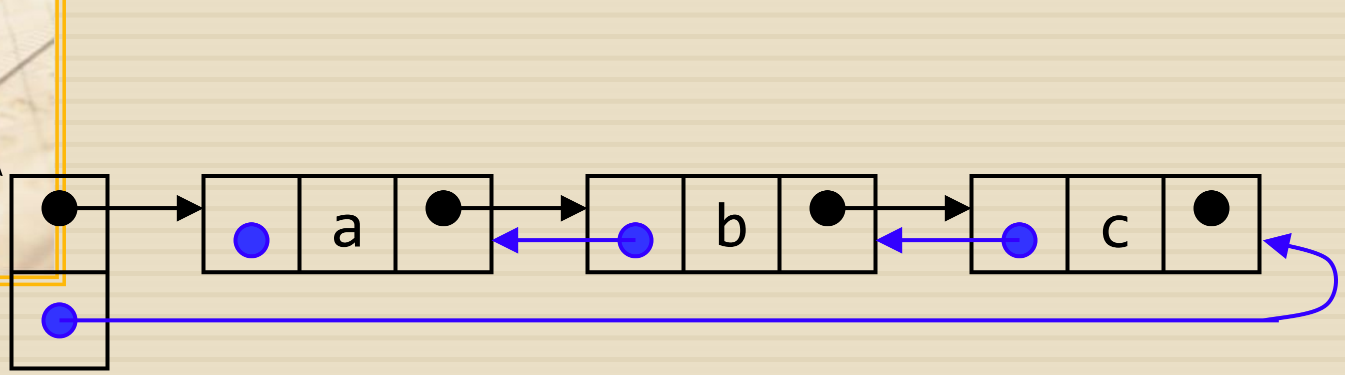
Each node contains a value and a link to its successor (the last node has no successor).

The header points to the first node in the list (or contains the null link if the list is empty)

Doubly Linked List

- DLL



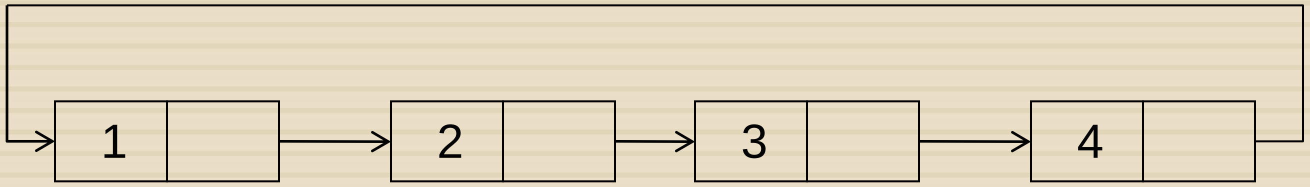


Each node contains a value, a link to its successor (if any), *and* a link to its predecessor (if any)
The header points to the first node in the list *and* to the last node in the list (or contains null links if the list is empty)

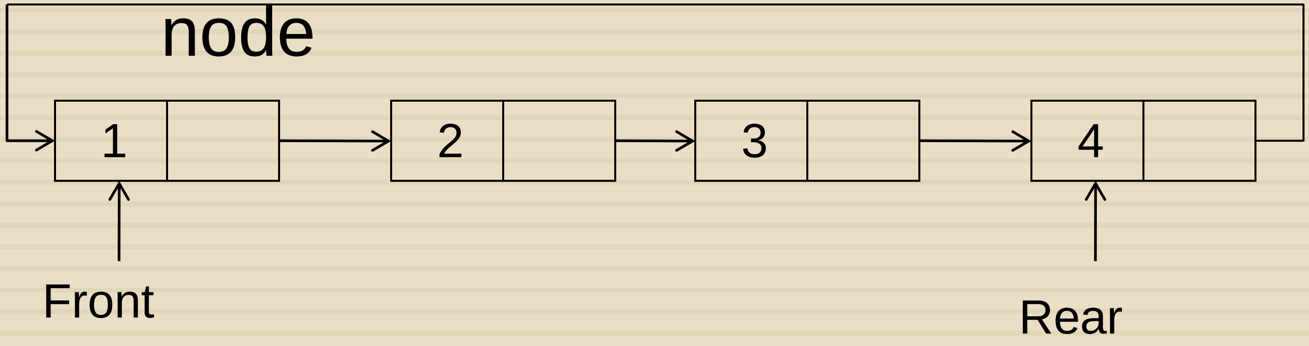
- info: the user's data
- next, back: the address of the next and previous node in the list

Circular Linked List

- Circular linked list is a linked list where the last node of the linked list is pointing to the first node of the linked list



- We need two pointers
  - A head pointer – pointing to the first node
  - A rear pointer – pointing to the last node

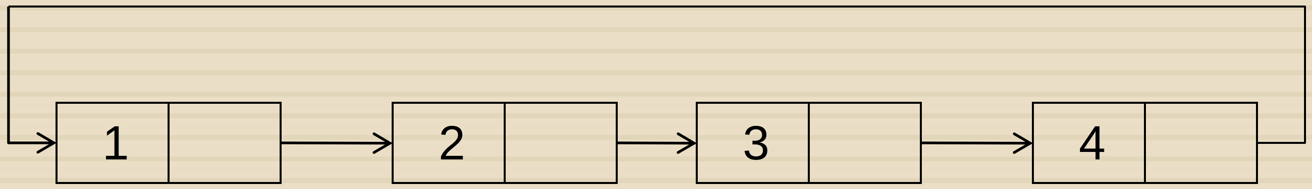


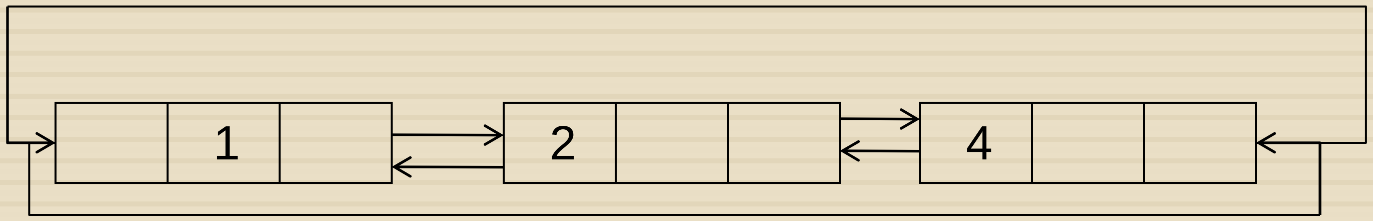
When is a circular LL is used?

- When we need to traverse from the last node to first node with out having to traverse backwards.
- Overcome the limitation of Doubly Linked List.

Implementation

- Can be implemented using
  - Singly linked list
  - Doubly linked list





Operation On Linked List

- Insertion : we add one node to a list
- Deletion: we delete an node from a list
- Traversing : we traverse a node.