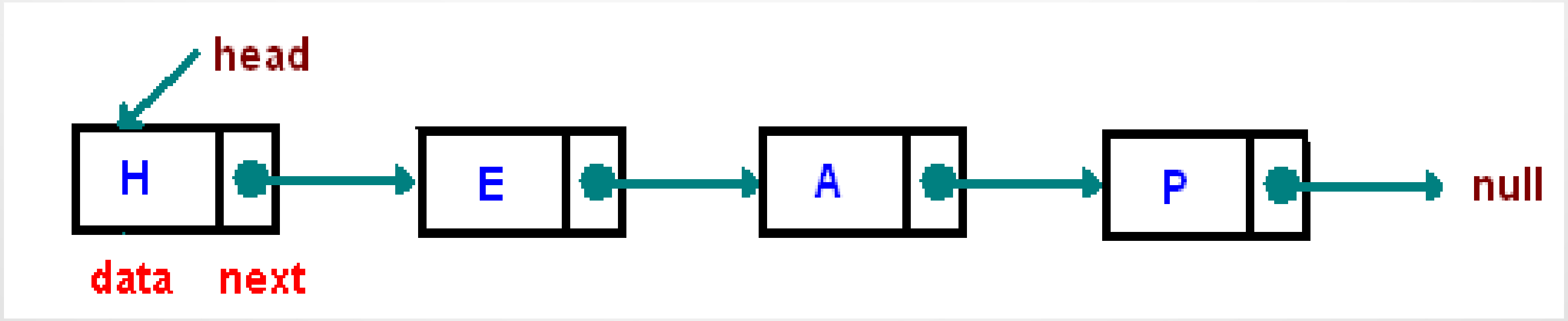
GLS UNIVERSITY

0301302 DATA STRUCTURES.
UNIT– III

- Prof. Nirav Suthar
- Prof. Ankita Shah

LINKED LIST

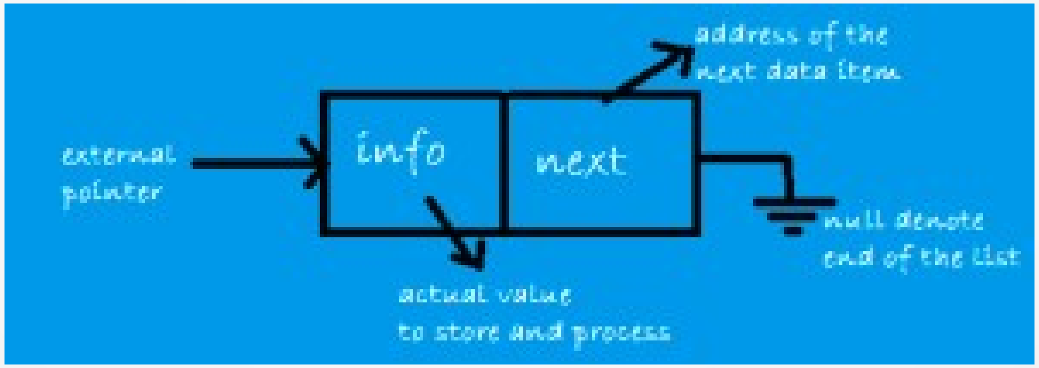
- A linked-list is a sequence of data structures which are connected together via links.
- Linked List is a sequence of links which contains items. Each link contains a connection to another link. Linked list the second most used data structure after array.
  - **Link –** Each Link of a linked list can store a data called an element.
  - **Next –** Each Link of a linked list contain a link to next link called Next.
  - **LinkedList –** A LinkedList contains the connection link to the first Link called First.



Structure of Linked list

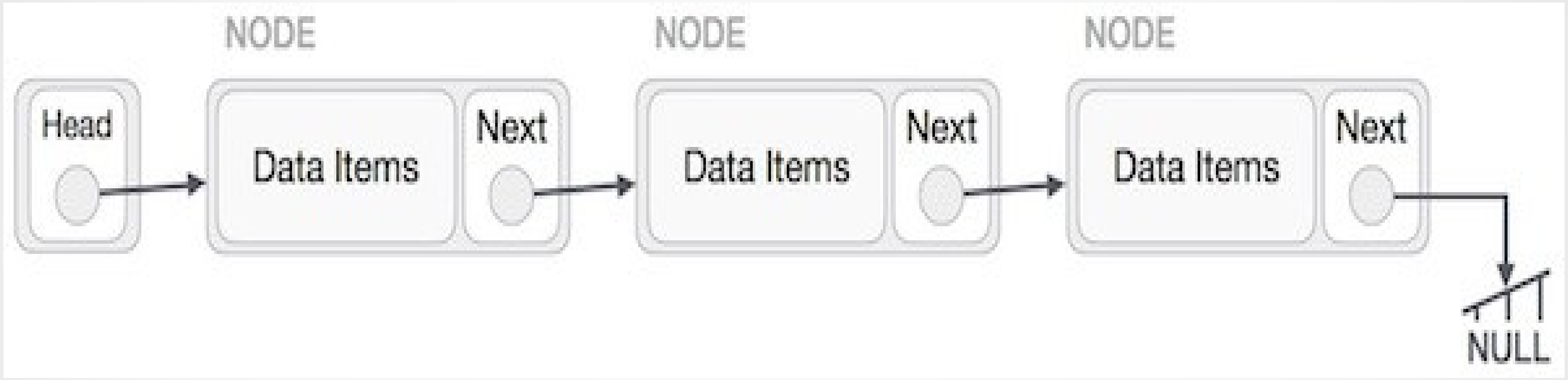
- A list consists of linearly connected train of nodes. Each node can be divided into two parts:

- Data Field:It is the actual value that is stored and processed.
- Linked Field: It is the address of the next data item in the Linked List.


address of the next data item
external pointer
info
next
null denote end of the list
actual value to store and process

LINKED LIST Representation

Linked list can be visualized as a chain of nodes, where every node points to the next node.



As per above shown illustration, following are the important points to be considered.

- LinkedList contains an link element called first.
- Each Link carries a data field(s) and a Link Field called next.
- Each Link is linked with its next link using its next link.
- Last Link carries a Link as null to mark the end of the list.

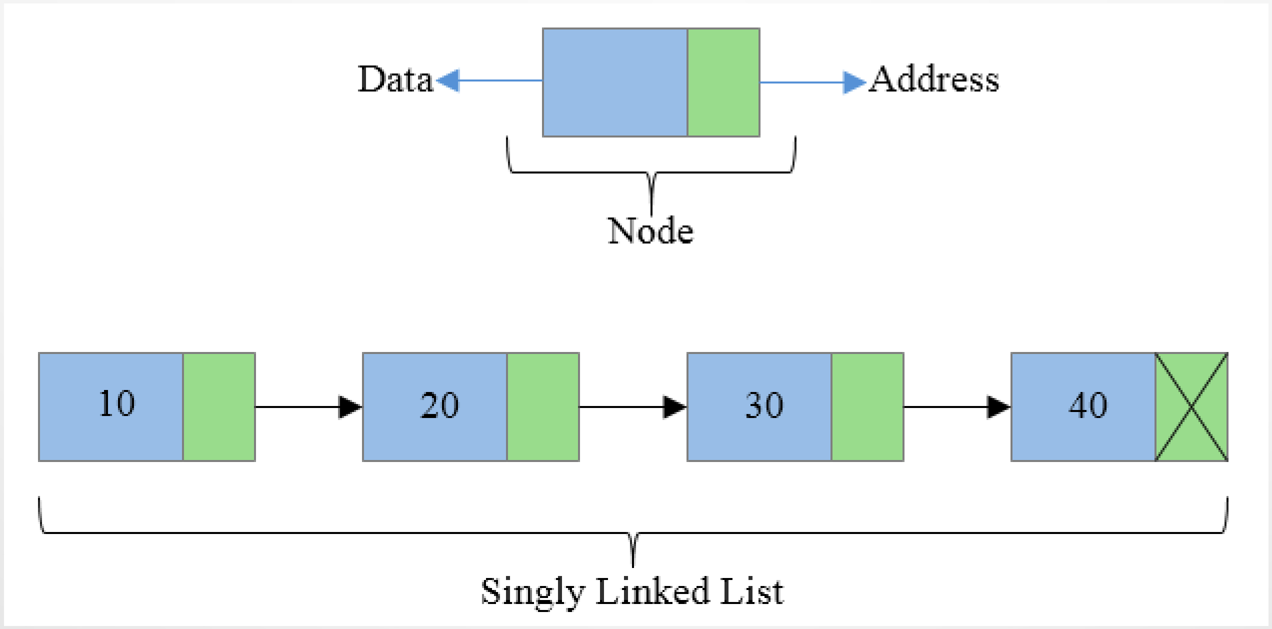
TYPES OF LINKED LIST

- Following are the various flavours of linked list.

  - **Simple Linked List −** Item Navigation is forward only.
  - **Doubly Linked List −** Items can be navigated forward and backward way.
  - **Circular Linked List −** Last item contains link of the first element as next and and first element has link to last element as prev.

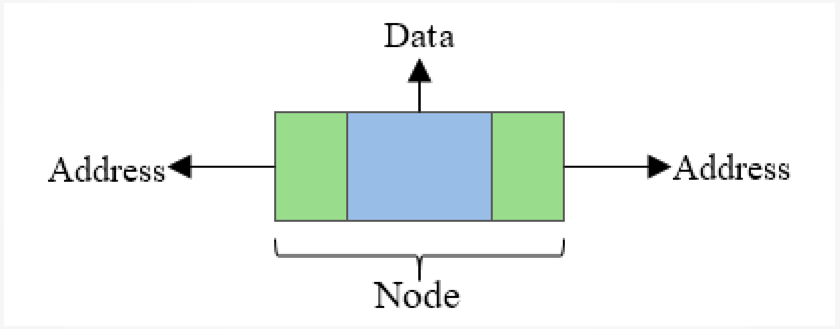
Singly Linked List

Singly linked list is a collection of nodes linked together in a sequential way where each node of singly linked list contains a data field and an address field which contains the reference of the next node.
Singly linked list can contain multiple data fields but should contain at least single address field pointing to its connected next node.

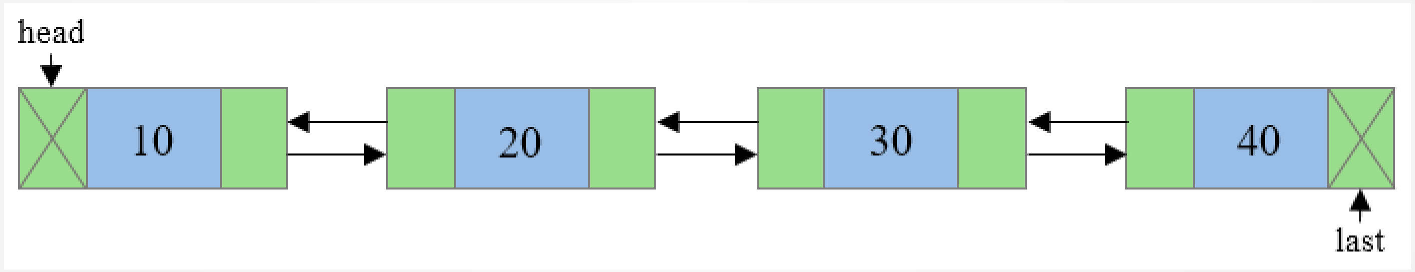


Doubly Linked List

Doubly linked list is a collection of nodes linked together in a sequential way. Each node of the list contains two parts (as in singly linked list) data part and the reference or address part.



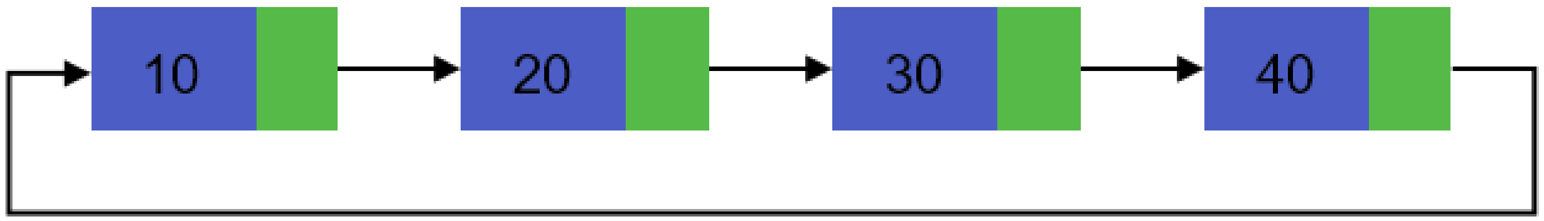
First and last node of a linked list contains a terminator generally a NULL value, that determines the start and end of the list. Doubly linked list is sometimes also referred as bi-directional linked list since it allows traversal of nodes in both direction.


head
10
20
30
40
last

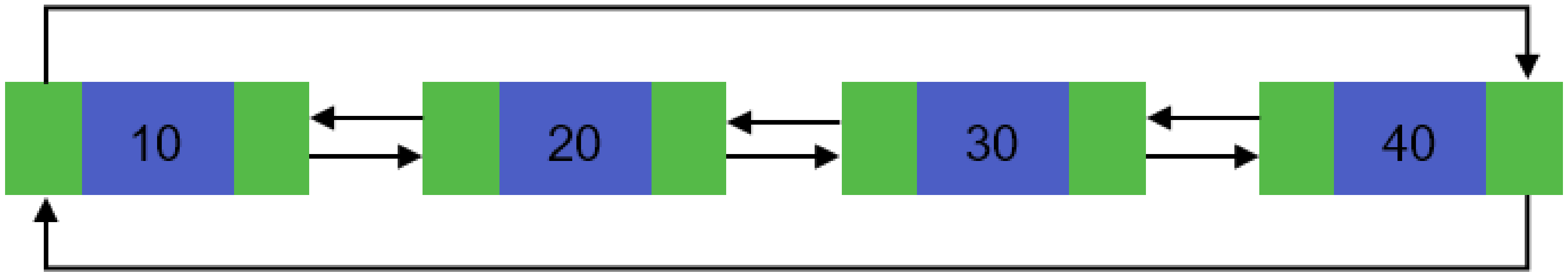
Circular Linked List

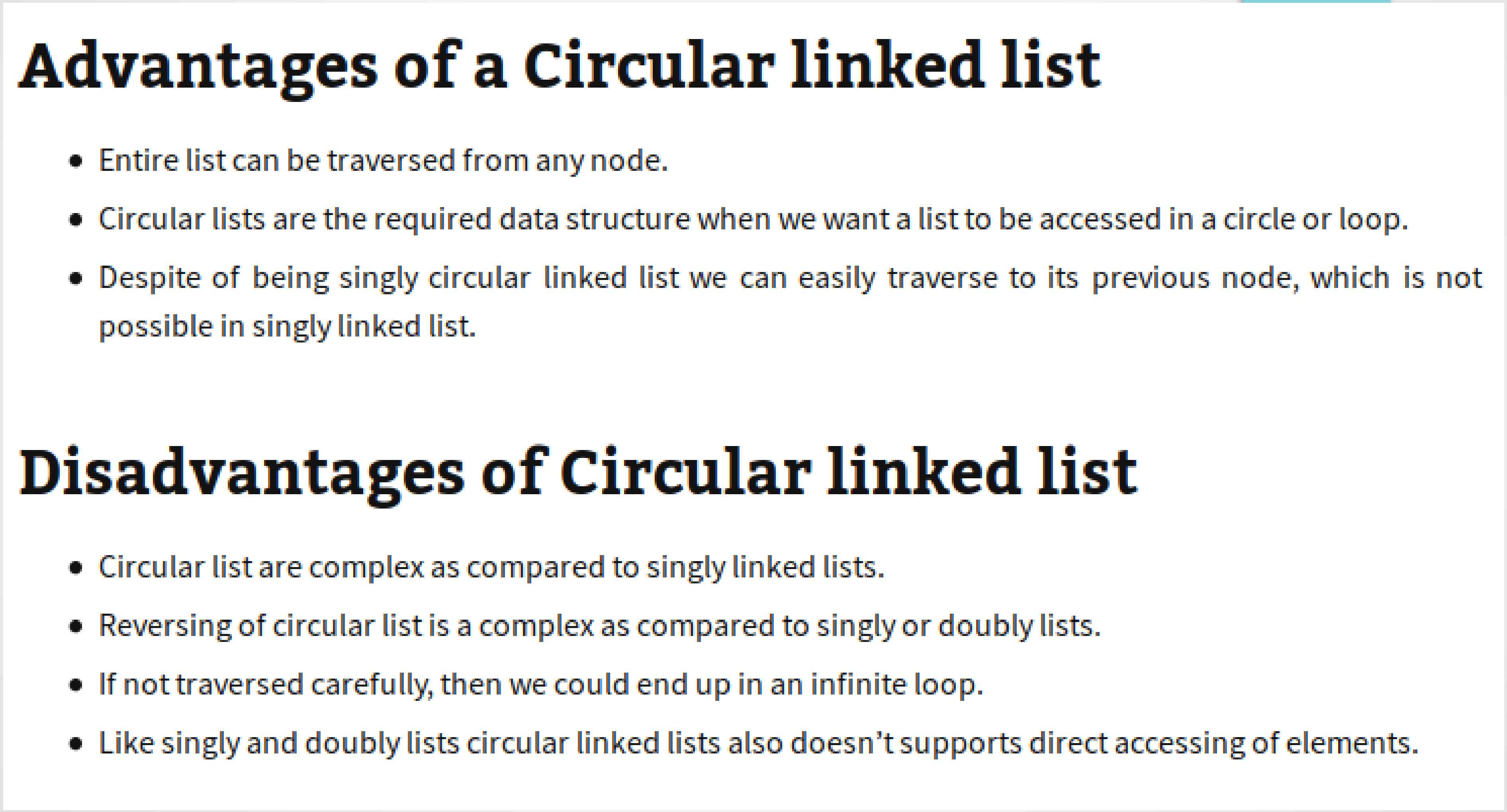
- A circular linked list is basically a linear linked list that may be singly or doubly. The only difference is that there is no any NULL value terminating the list. In fact in the list every node points to the next node and last node points to the first node, thus forming a circle.
- Since it forms a circle with no end to stop hence it is called as circular linked list.
- In circular linked list there can be no starting or ending node, whole node can be traversed from any node.
- In order to traverse the circular linked list only once we need to traverse entire list until the starting node is not traversed again.

Basic structure of Circular linked list



Circular Linked List





Advantages of a Circular linked list

- Entire list can be traversed from any node.
- Circular lists are the required data structure when we want a list to be accessed in a circle or loop.
- Despite of being singly circular linked list we can easily traverse to its previous node, which is not possible in singly linked list.

Disadvantages of Circular linked list

- Circular list are complex as compared to singly linked lists.
- Reversing of circular list is a complex as compared to singly or doubly lists.
- If not traversed carefully, then we could end up in an infinite loop.
- Like singly and doubly lists circular linked lists also doesn’t supports direct accessing of elements.

- Implementation of stacks and queues
- **Implementation of graphs** : Adjacency list representation of graphs is most popular which is uses linked list to store adjacent vertices.
- **Dynamic memory allocation** : We use linked list of free blocks.
- Maintaining directory of names
- Performing arithmetic operations on long integers
- Manipulation of polynomials by storing constants in the node of linked list representing sparse matrices