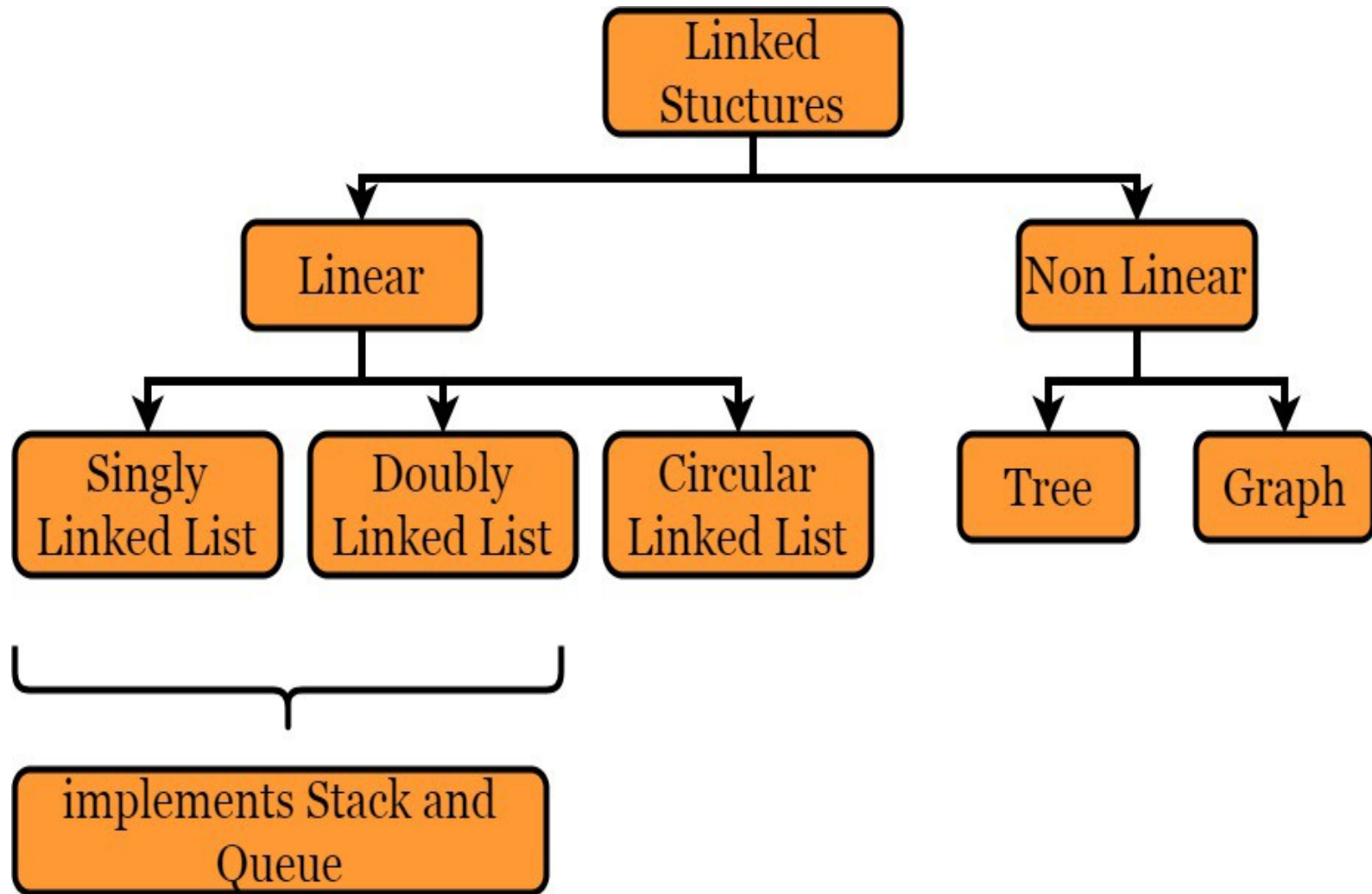# Data Structure

Data Structure

Data Structure is a way of collecting and organising data in such a way that we can perform various kind of operations on these data in an effective way.
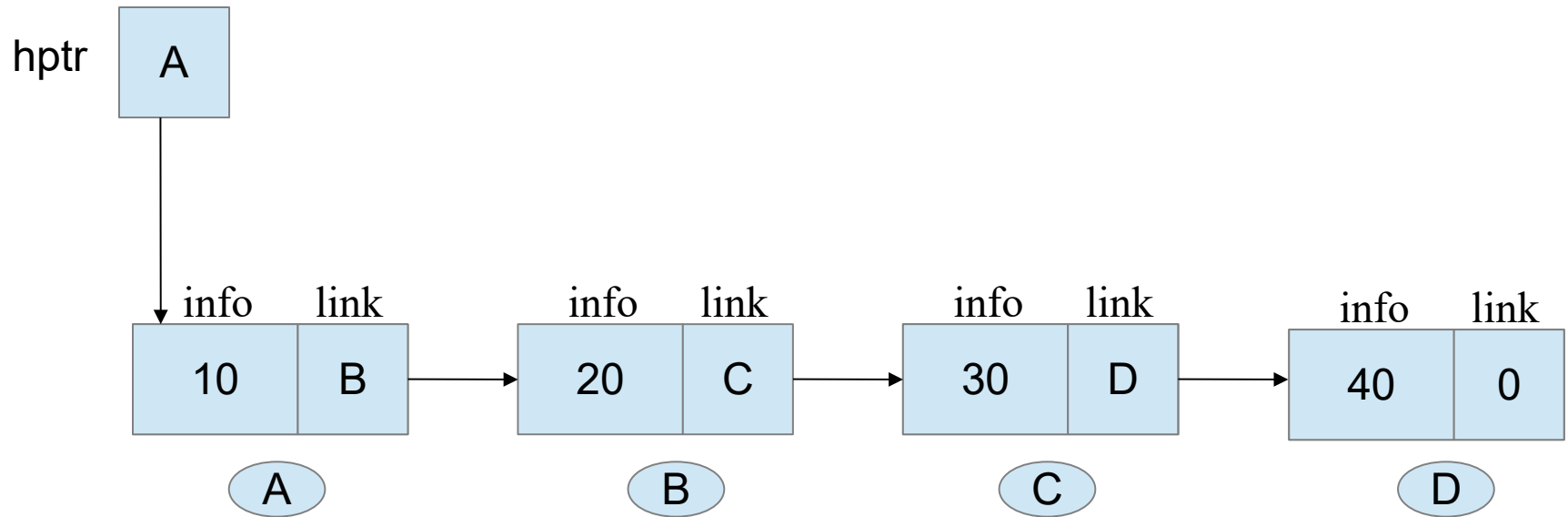
Data Structure

Collecting (store) and **Organising** (arrange) the data to **perform** various kind of operations (display, search, insert, delete etc.........) in **effective way** (faster and using less memory).

- Memory
- Speed

```
                    ┌─────────────┐
                    │   Linked    │
                    │  Stuctures  │
                    └─────────────┘
              ┌───────────┴───────────────┐
              ▼                           ▼
         ┌─────────┐                ┌──────────────┐
         │ Linear  │                │  Non Linear  │
         └─────────┘                └──────────────┘
       ┌─────┼──────────┐           ┌──────┴──────┐
       ▼     ▼          ▼           ▼             ▼
  ┌────────┐┌────────┐┌──────────┐┌──────┐    ┌───────┐
  │ Singly ││ Doubly ││ Circular ││ Tree │    │ Graph │
  │ Linked ││ Linked ││  Linked  │└──────┘    └───────┘
  │  List  ││  List  ││   List   │
  └────────┘└────────┘└──────────┘
```

Linked Stuctures

Linear   Non Linear

Singly Linked List   Doubly Linked List   Circular Linked List   Tree   Graph

implements Stack and Queue

Format of Linked list

Ex:

Linked List

A linked list is a linear data structure, in which the elements are not stored at contiguous memory locations(mendatory).

- Efficient memory management.

Node

Node – Most basic building block of Link List
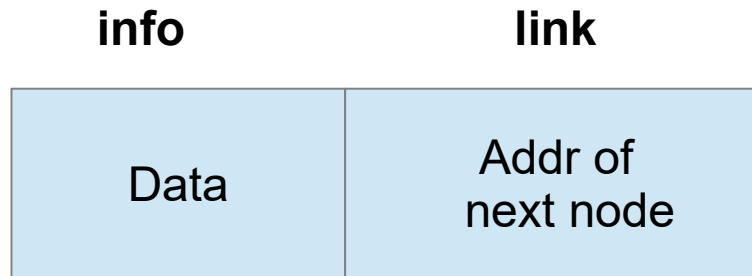
It contains two parts

 Data part (store the information)

 A**ddress** part (pointer-(store the address of next
   node).

Node

Node : It is a memory region, which containes 2 fields.
    1) information field
    2) link field

info	link
Data	Addr of next node

Advantages of Lists:

1) They are a dynamic in nature which allocates the memory when required.
2) Insertion and deletion operations can be easily implemented.
3) Linked List reduces the access time.
4) Linked lists are used to implement stacks, queues, graphs, etc.
5) Linked lists let you insert elements at the beginning and end of the list or in the middle to get sorted data storage.
6) In Linked Lists we don't need to know the size in advance.

Disadvantages of Lists:

1) The memory is wasted as pointers require extra memory for storage.
2) No element can be accessed randomly; it has to access each node sequentially.
3) Reverse Traversing is difficult in linked list.

Points to Revise

- Pointers
- Functions, call by value,call by address
- DMA
- Structure,self referential structure
- Structure pointers
- typedef

Important Points to Remember

1, Need to declare a self referential structure

  (for nodes – basic buliding block)

struct list{

int data;

struct list *next;

};

2, Need make a pointer (structure pointer)- called head pointer for storing the starting node address.