Data

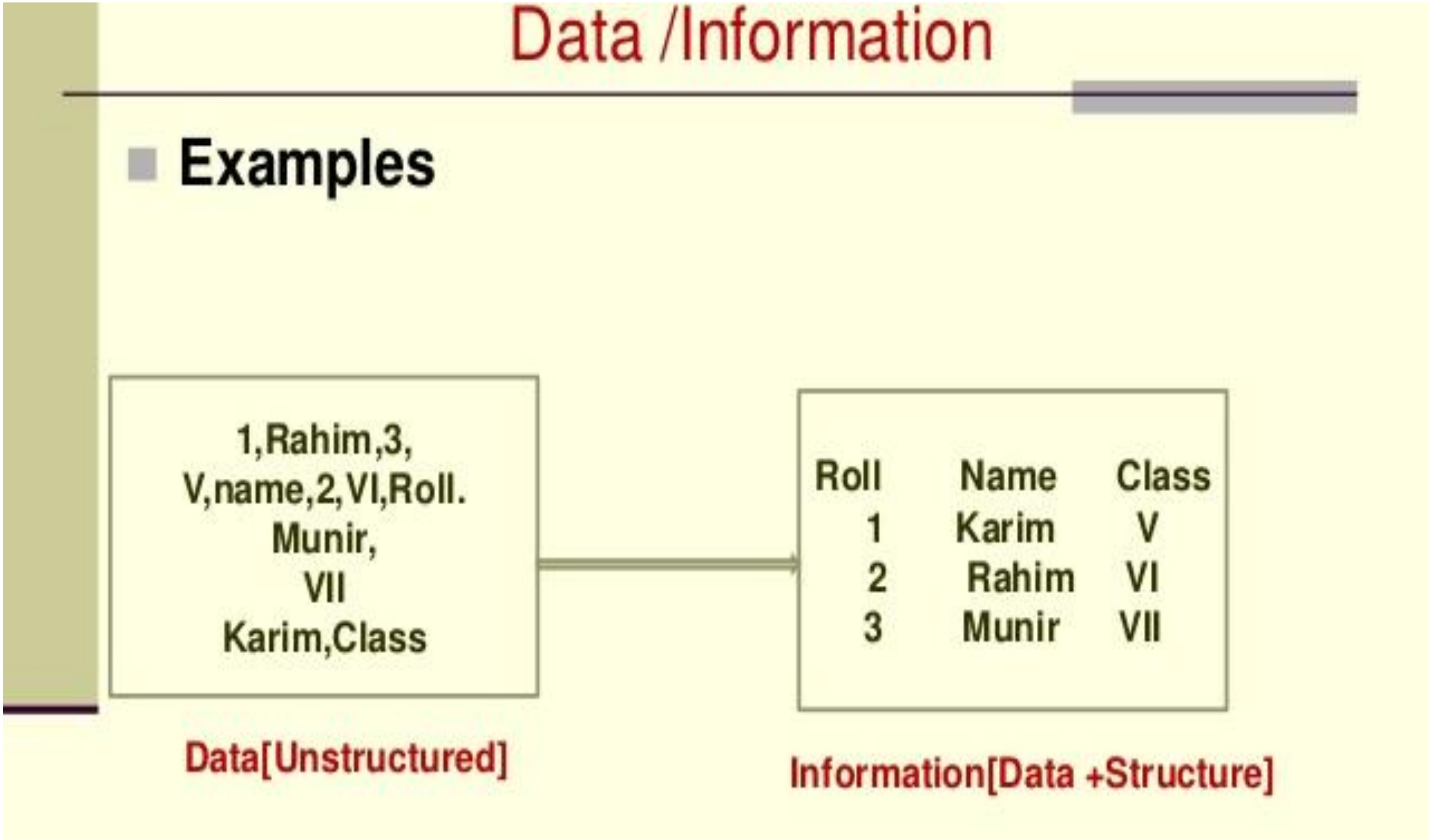
Data are simply values or sets of values. A data item refers to a single unit of values.

Data is raw, unorganized facts (ঘটনা) that need to be processed. Data can be something simple and seemingly (আপাতদৃষ্টিতে) random (এলোমেলো) and useless until it is organized.

Information

When data is processed, organized, structured or presented in a given context so as to make it useful, it is called information.

Examples of Data and Information



Data /Information

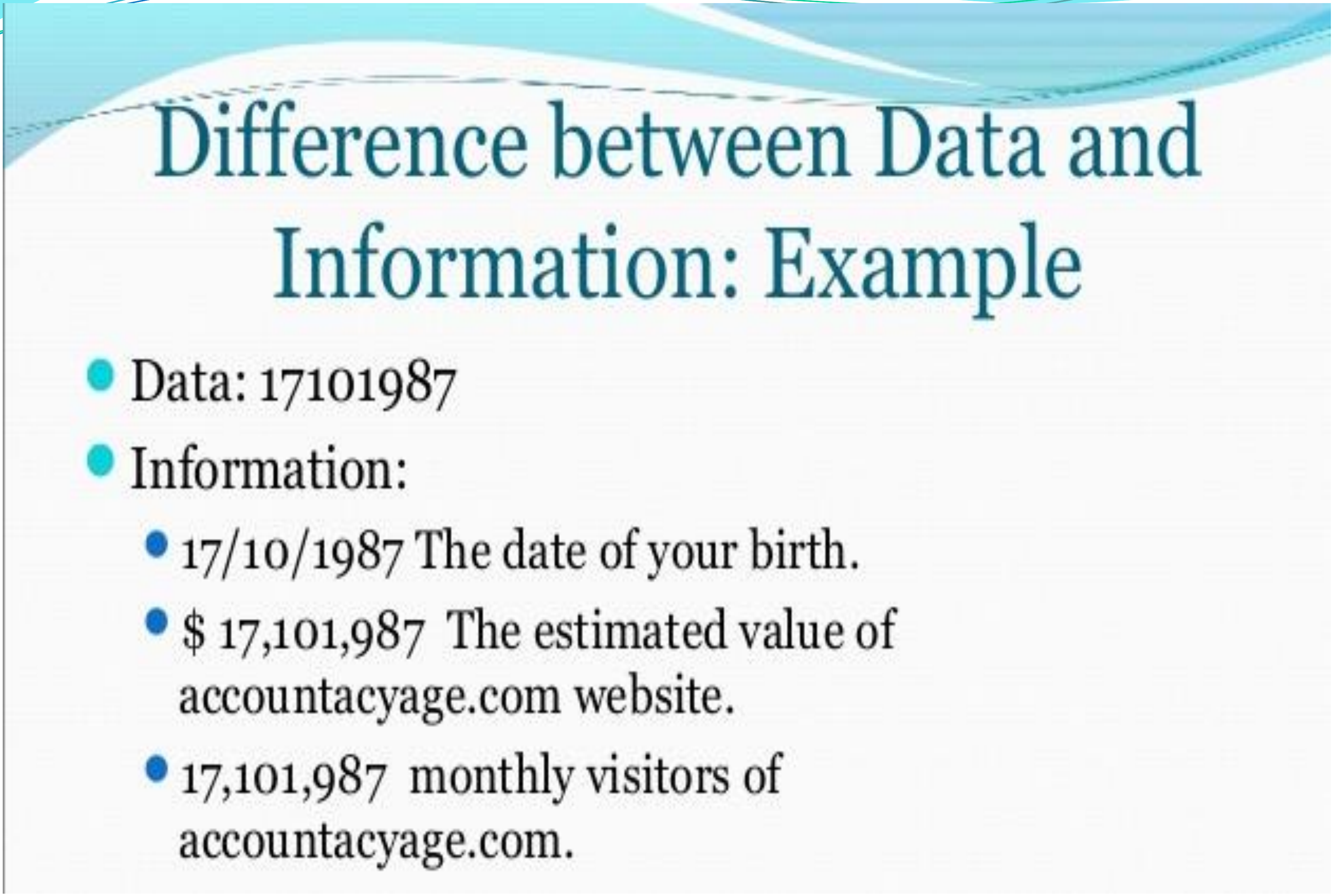
- **Examples**

1,Rahim,3,
V,name,2,VI,Roll.
Munir,
VII
Karim,Class

Data[Unstructured]

Roll	Name	Class
1	Karim	V
2	Rahim	VI
3	Munir	VII

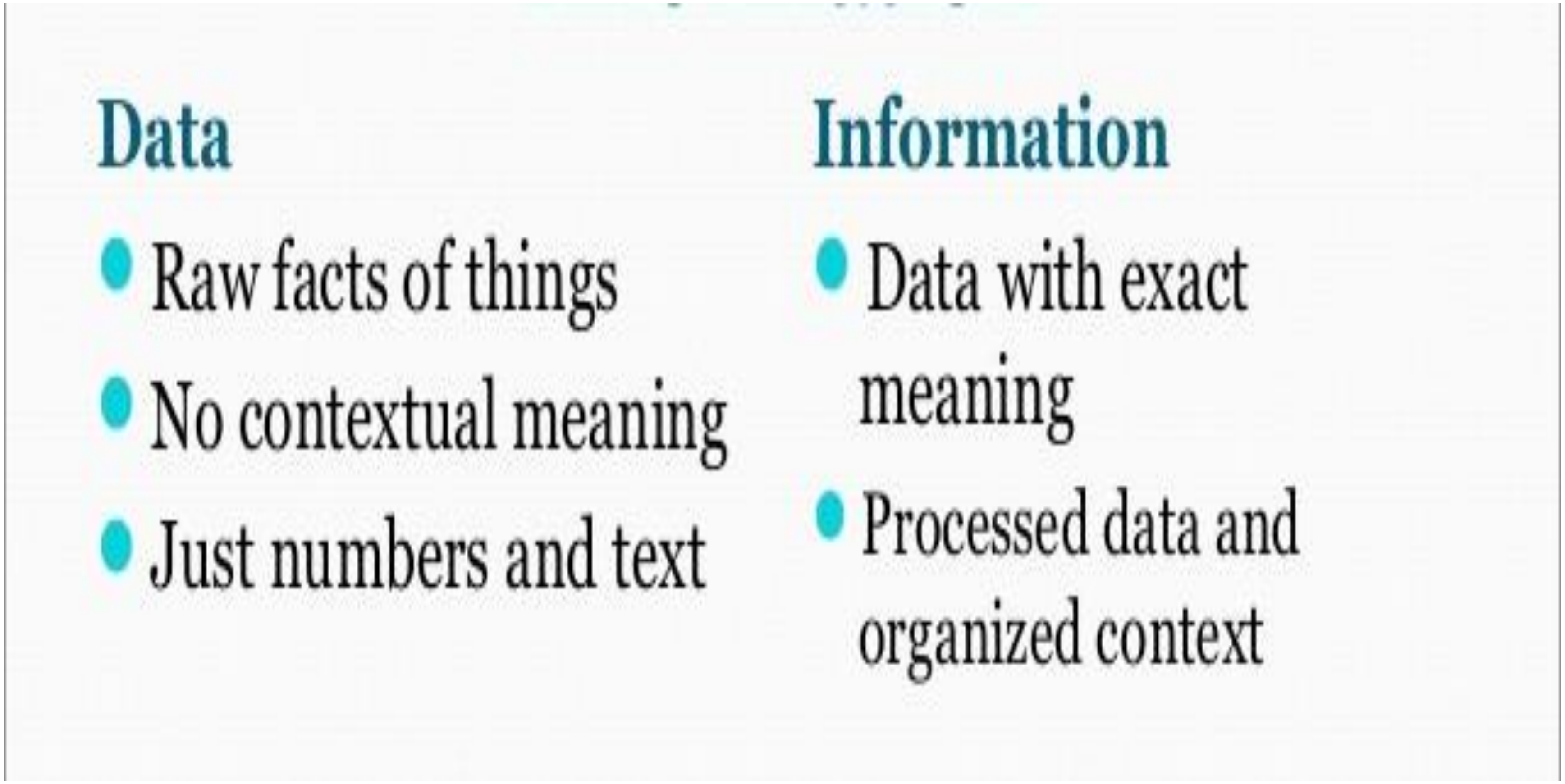
Information[Data +Structure]



Difference between Data and Information: Example

- Data: 17101987
- Information:
  - 17/10/1987 The date of your birth.
  - $ 17,101,987 The estimated value of accountacyage.com website.
  - 17,101,987 monthly visitors of accountacyage.com.

Difference between Data and Information



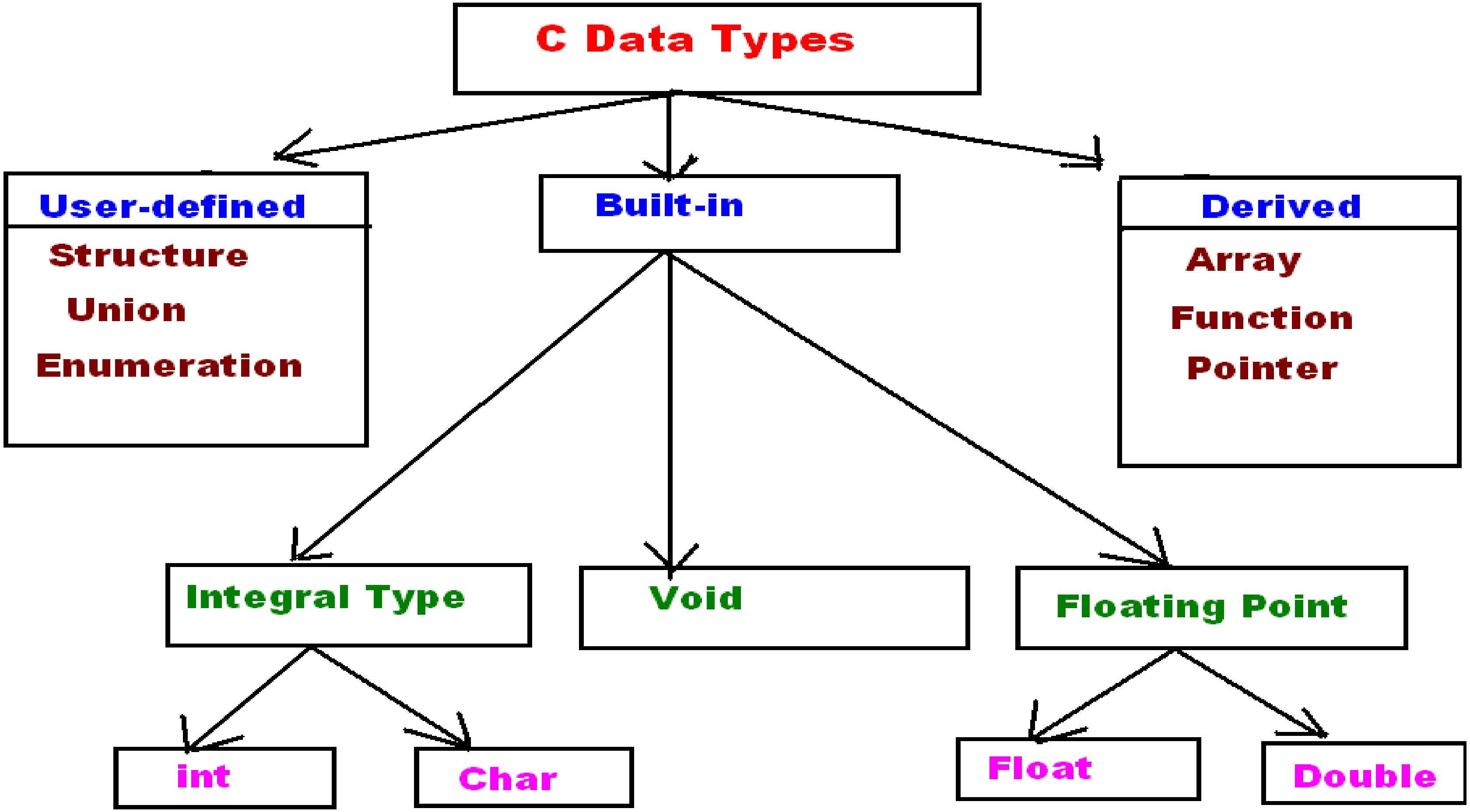
Data

- Raw facts of things
- No contextual meaning
- Just numbers and text

Information

- Data with exact meaning
- Processed data and organized context

Types of Data



C Data Types

User-defined

Structure

Union

Enumeration

Built-in

Derived

Array

Function

Pointer

Integral Type

Void

Floating Point

int

Char

Float

Double

Data Structure

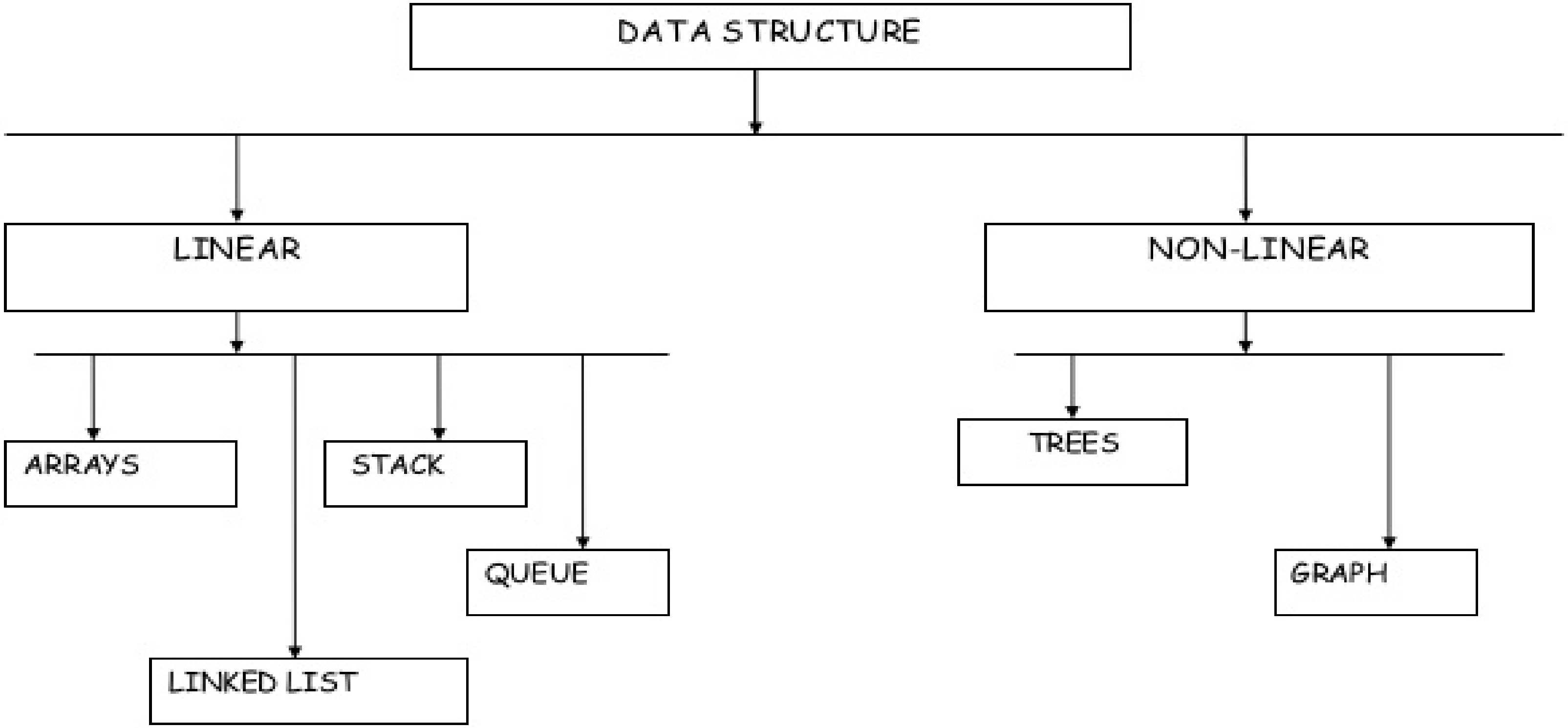
Data may be organized in many different ways; the logical or mathematical model of a particular organization of data is called a data structure.

Data structure is a particular way of organizing data in a computer so that it can be used efficiently.

A **data structure** is a specialized format for organizing and storing **data**.

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

Types of Data Structures



DATA STRUCTURE

LINEAR

NON-LINEAR

ARRAYS

STACK

QUEUE

LINKED LIST

TREES

GRAPH

Linear Data Structures

A linear data structure is a data structure that has data elements in sequential order. In a linear data structure, the adjacent elements are attached to each other. However, these data structures do not make better utilization of memory. Therefore, it can lead to memory wastage.

Array, Linked List, Stack, and Queue are some common examples of linear data structures.

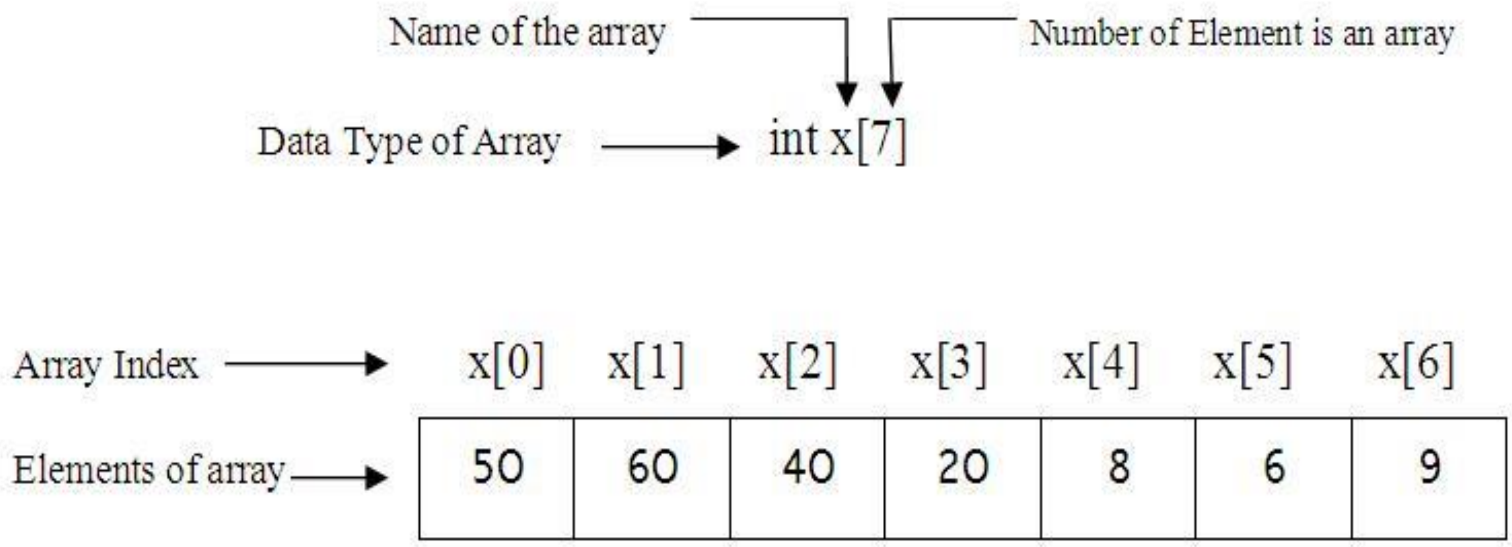
Non Linear Data Structures

Non Linear data structure stores data in a non-sequential manner. It forms a hierarchical relationship among the child elements and parent elements. In other words, the data items are attached to each other creating a relationship between them.

Trees and graphs are the most common nonlinear data structures.

Array

Arrays a kind of data structure that can store a fixed-size sequential collection of elements of the same type. An array is used to store a collection of data, but it is often more useful to think of an array as a collection of variables of the same type.

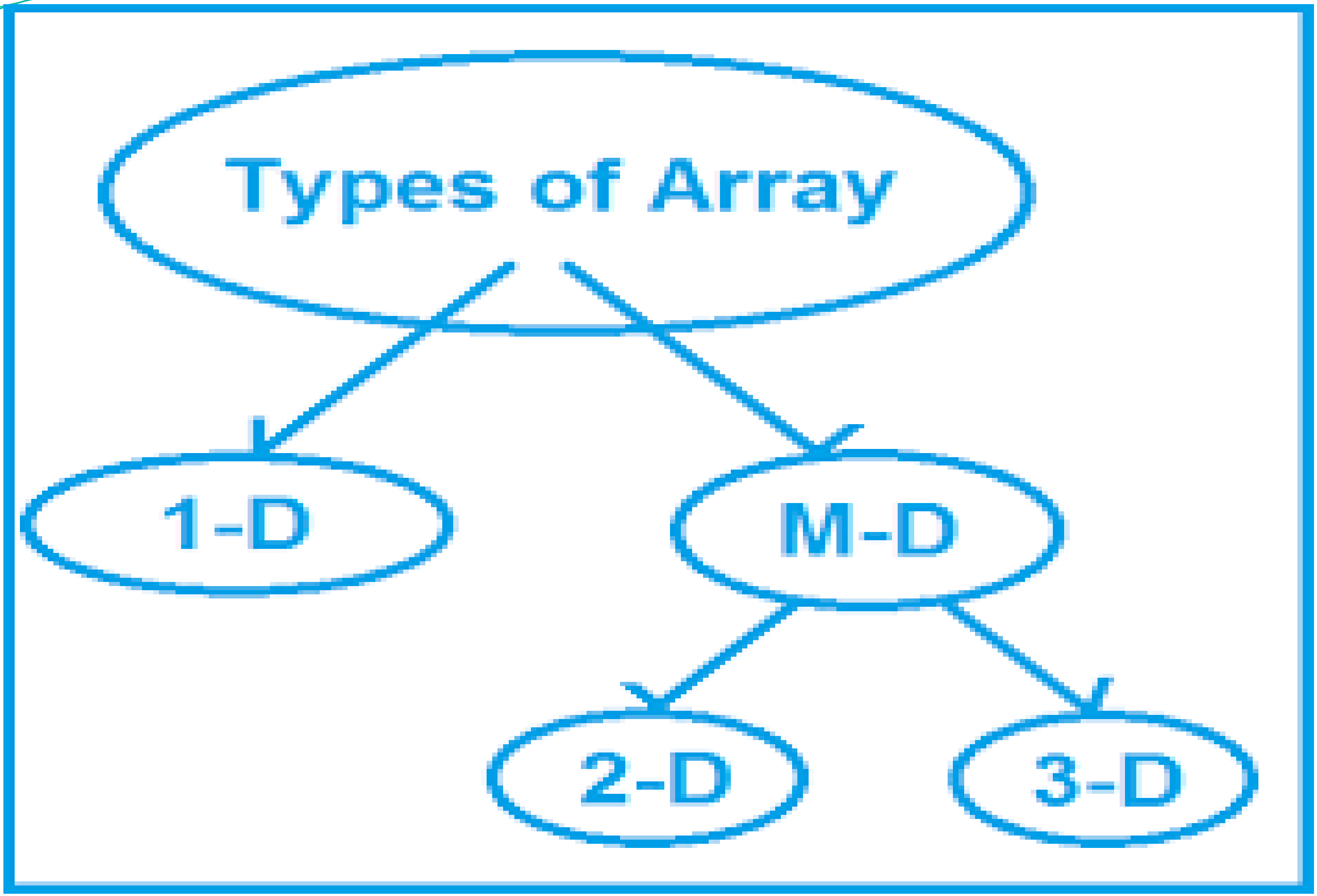


Name of the array

Number of Element is an array

Data Type of Array → int x[7]

Array Index →	x[0]	x[1]	x[2]	x[3]	x[4]	x[5]	x[6]
Elements of array →	50	60	40	20	8	6	9



Types of Array

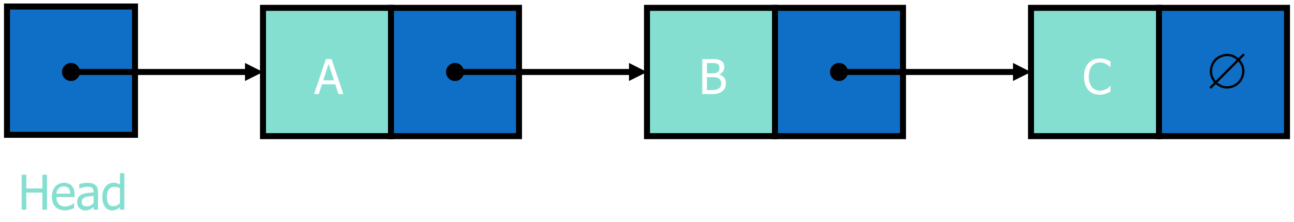
1-D

M-D

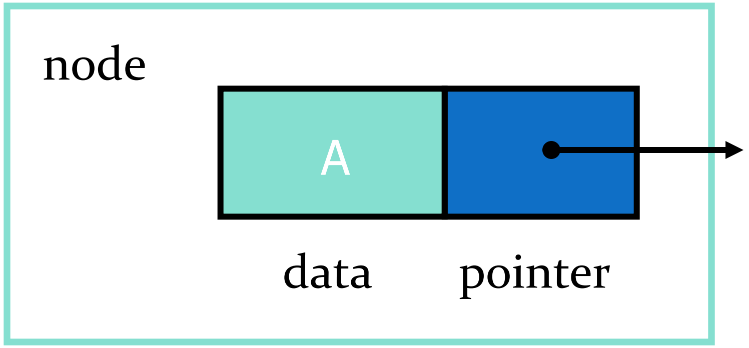
2-D

3-D

Linked List

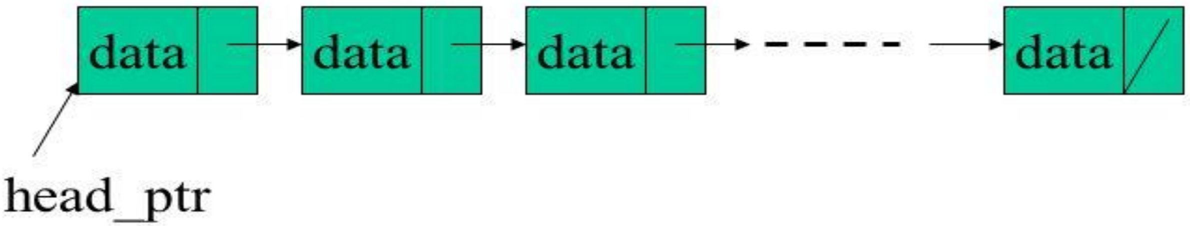


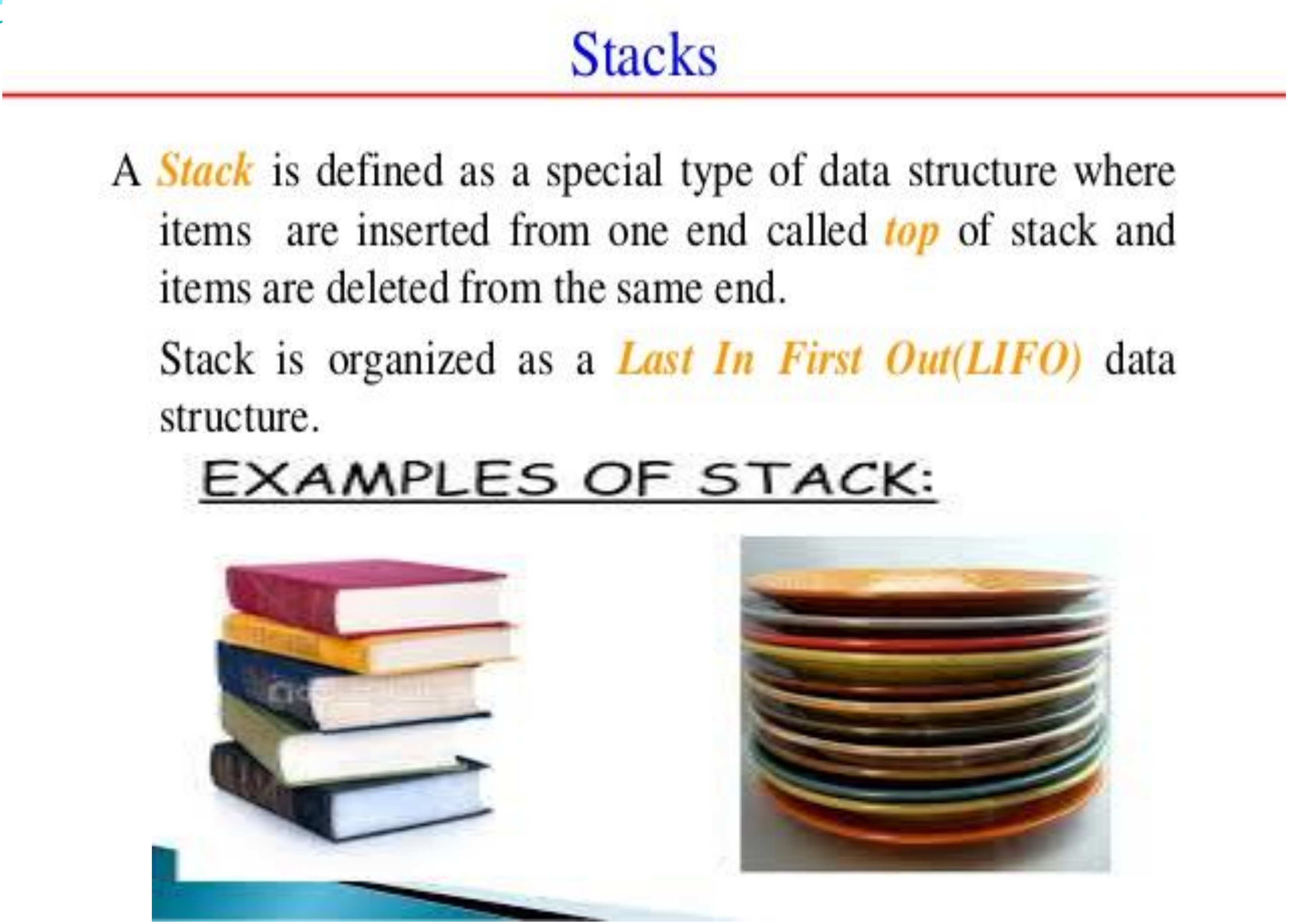
- A *linked list* is a series of connected *nodes*
- Each node contains at least
    - A piece of data (any type)
    - Pointer to the next node in the list
- *Head*: pointer to the first node
- The last node points to `NULL`



Definition of Linked Lists

- A linked list is a sequence of items (objects) where every item is linked to the next.
- Graphically:





Stacks

A *Stack* is defined as a special type of data structure where items are inserted from one end called *top* of stack and items are deleted from the same end.

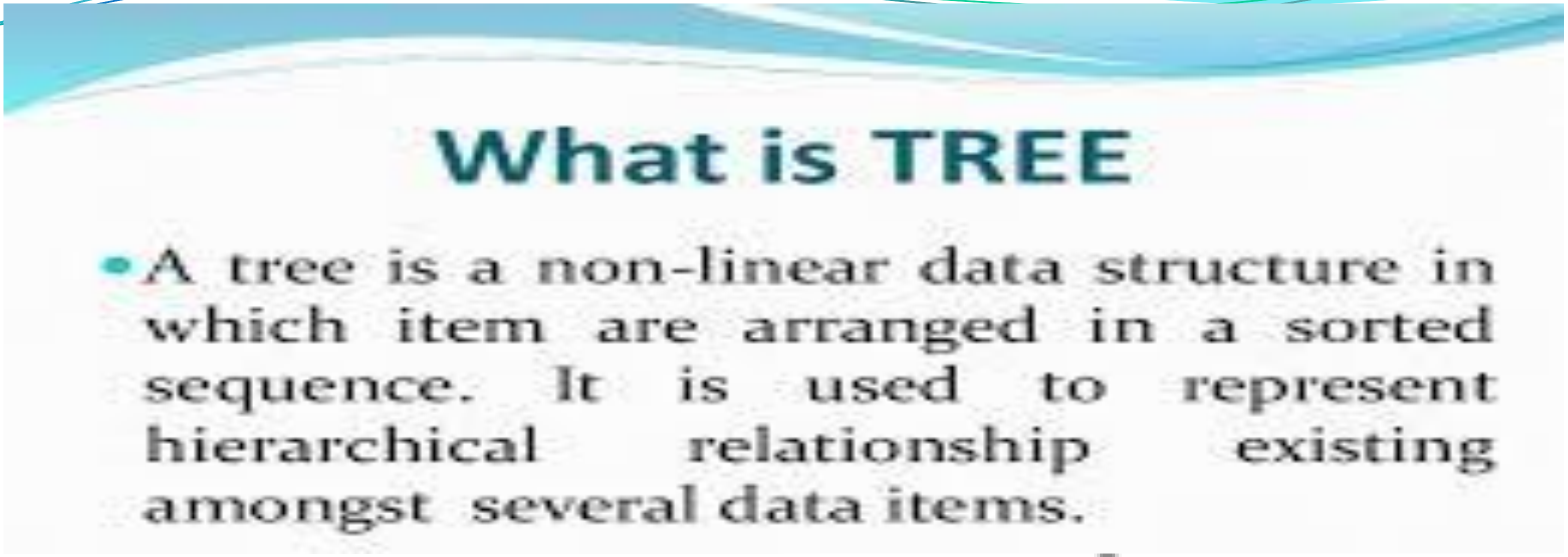
Stack is organized as a *Last In First Out(LIFO)* data structure.

EXAMPLES OF STACK:

Queues

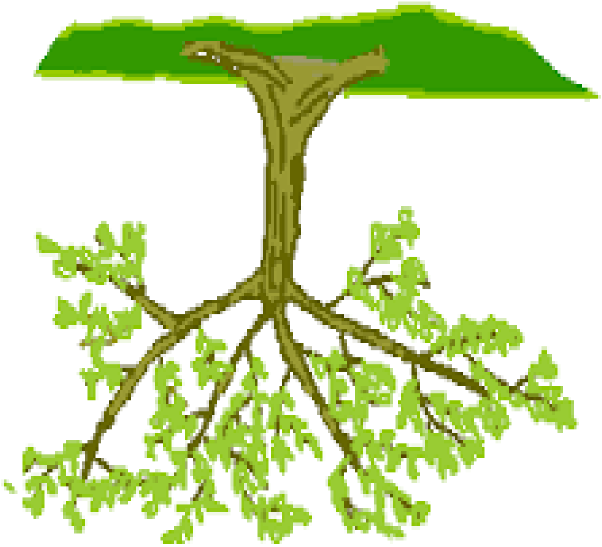
- A ***queue*** is defined as a special type of data structure where the elements are inserted from one end and elements are deleted from the other end.
- The end from where the elements are inserted is called ***REAR end.***
- The end from where the elements are deleted is called ***FRONT end***.
- Queue is organized as ***First In First Out*** (FIFO)Data Structure.

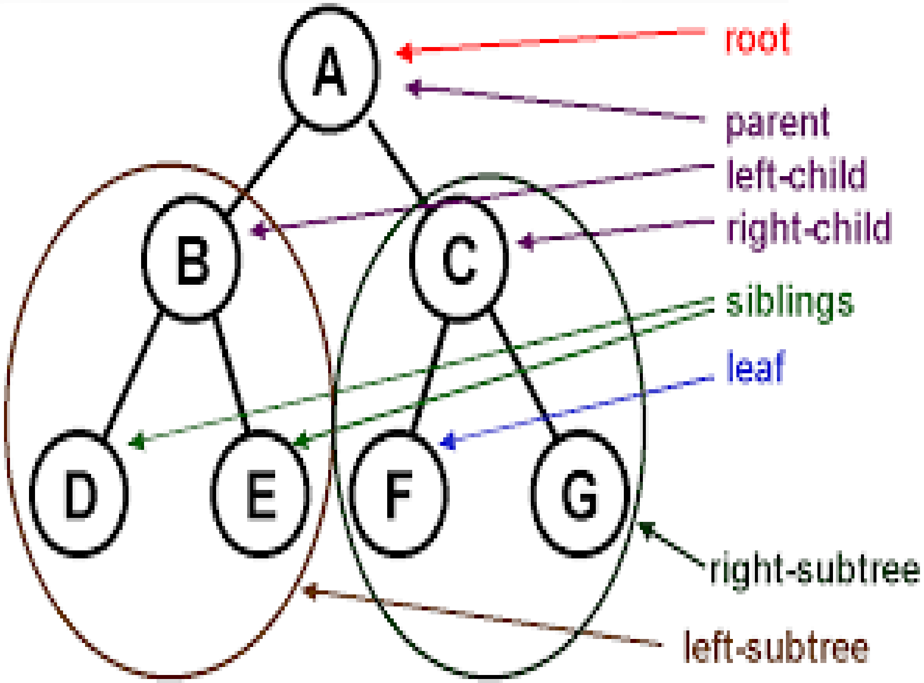
FreeDigitalPhotos.net by David Castillo Dominici

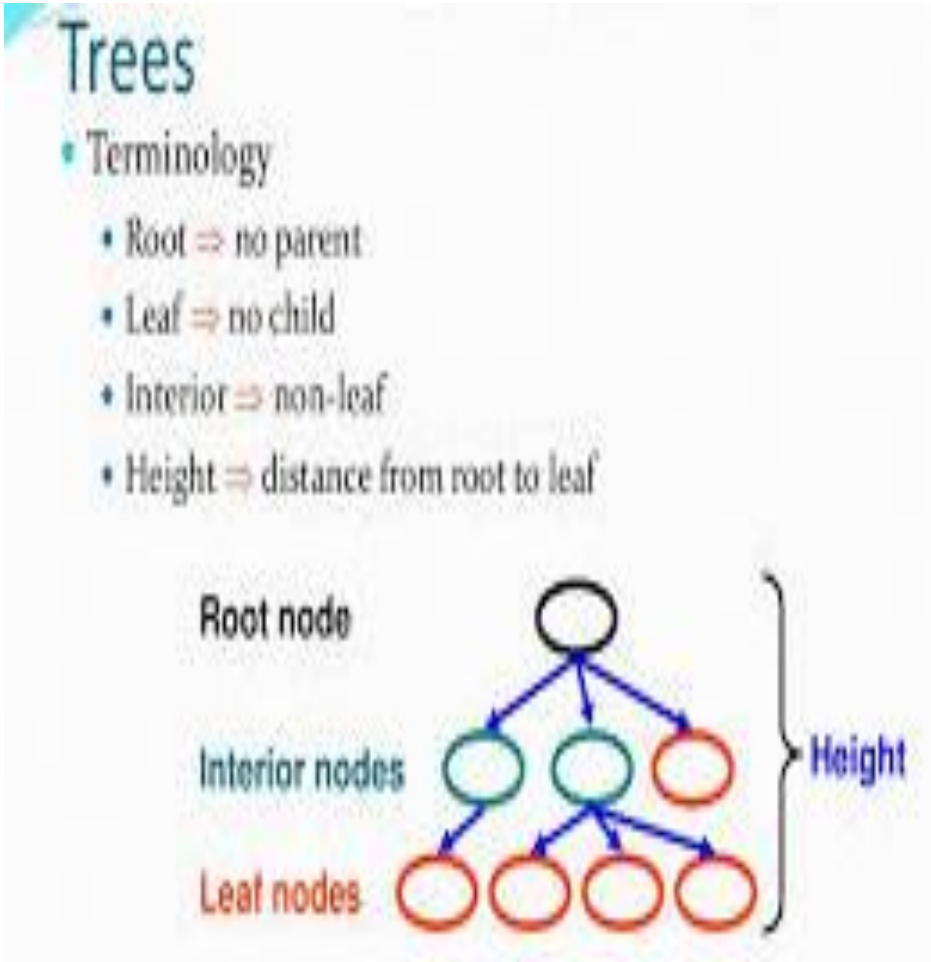


What is TREE

- A tree is a non-linear data structure in which item are arranged in a sorted sequence. It is used to represent hierarchical relationship existing amongst several data items.

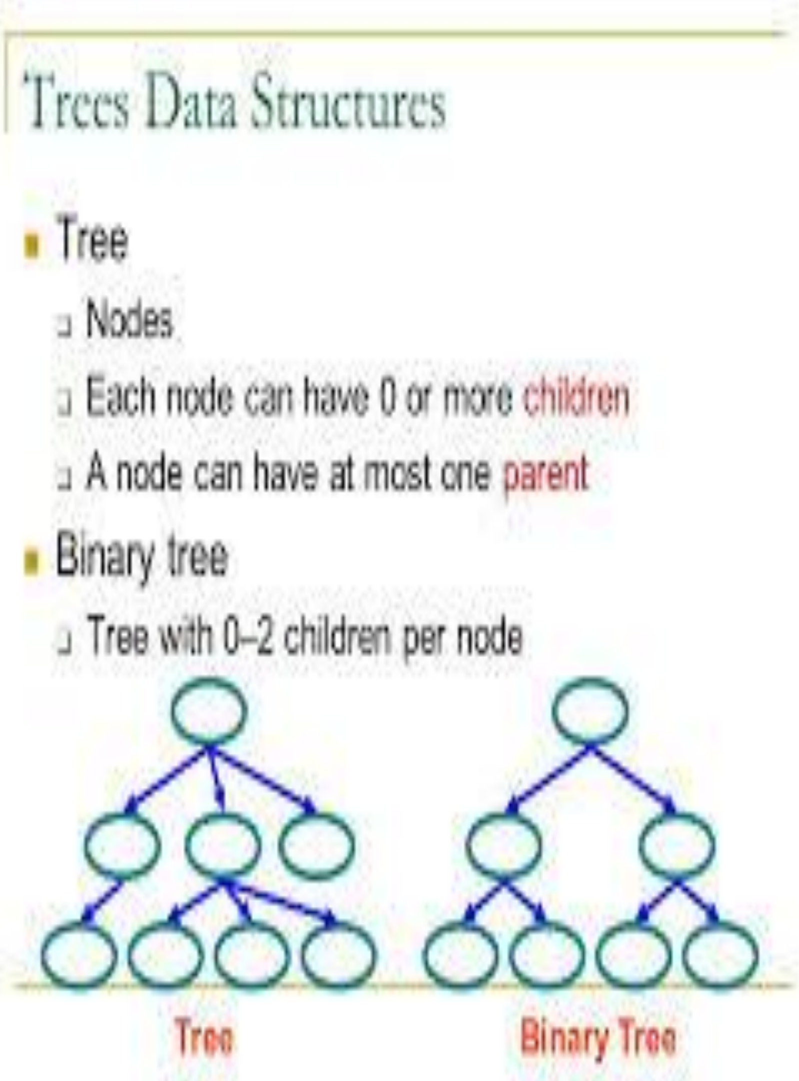






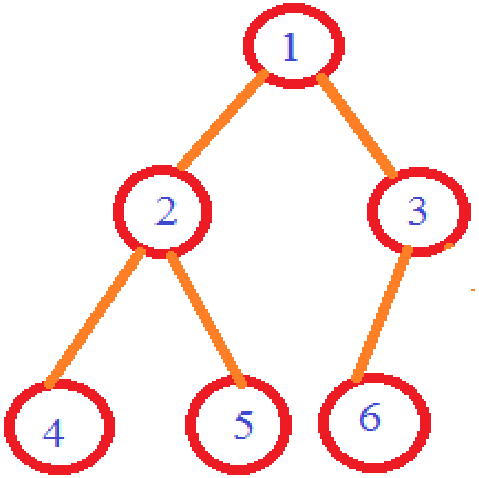
Trees

- Terminology
  - Root ⇒ no parent
  - Leaf ⇒ no child
  - Interior ⇒ non-leaf
  - Height ⇒ distance from root to leaf

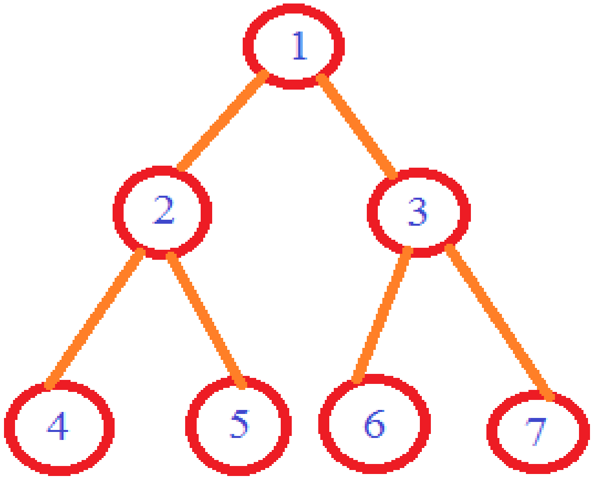


Trees Data Structures

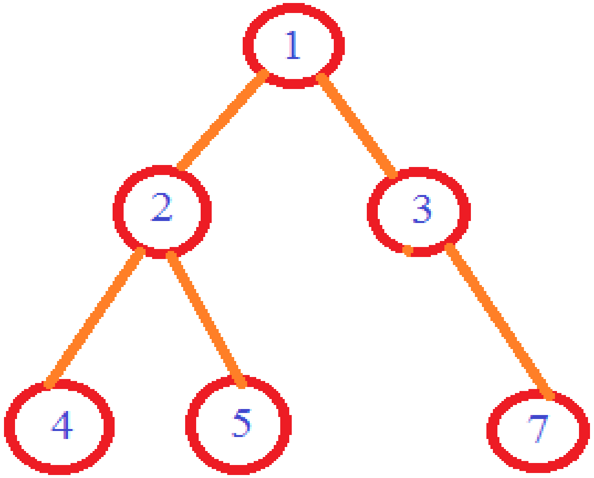
- Tree
  - Nodes
  - Each node can have 0 or more children
  - A node can have at most one parent
- Binary tree
  - Tree with 0–2 children per node



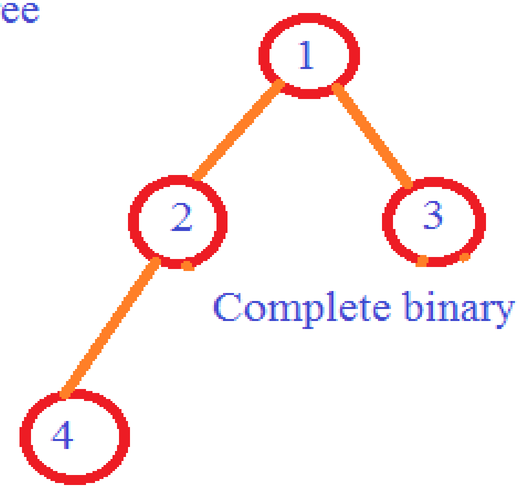
Complete binary tree



Full binary tree and complete binary tree



Neither a full binary tree nor a complete binary tree



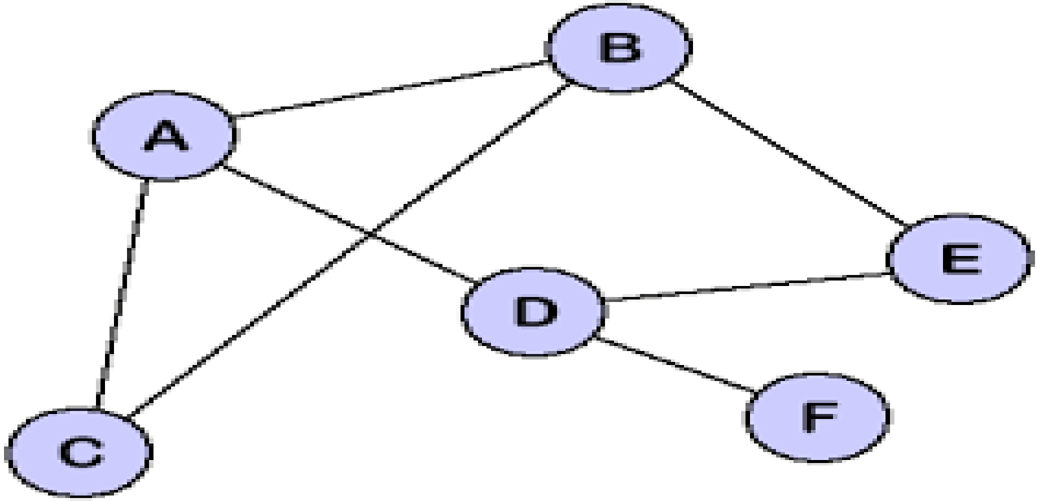
Complete binary tree

What is Graph ?

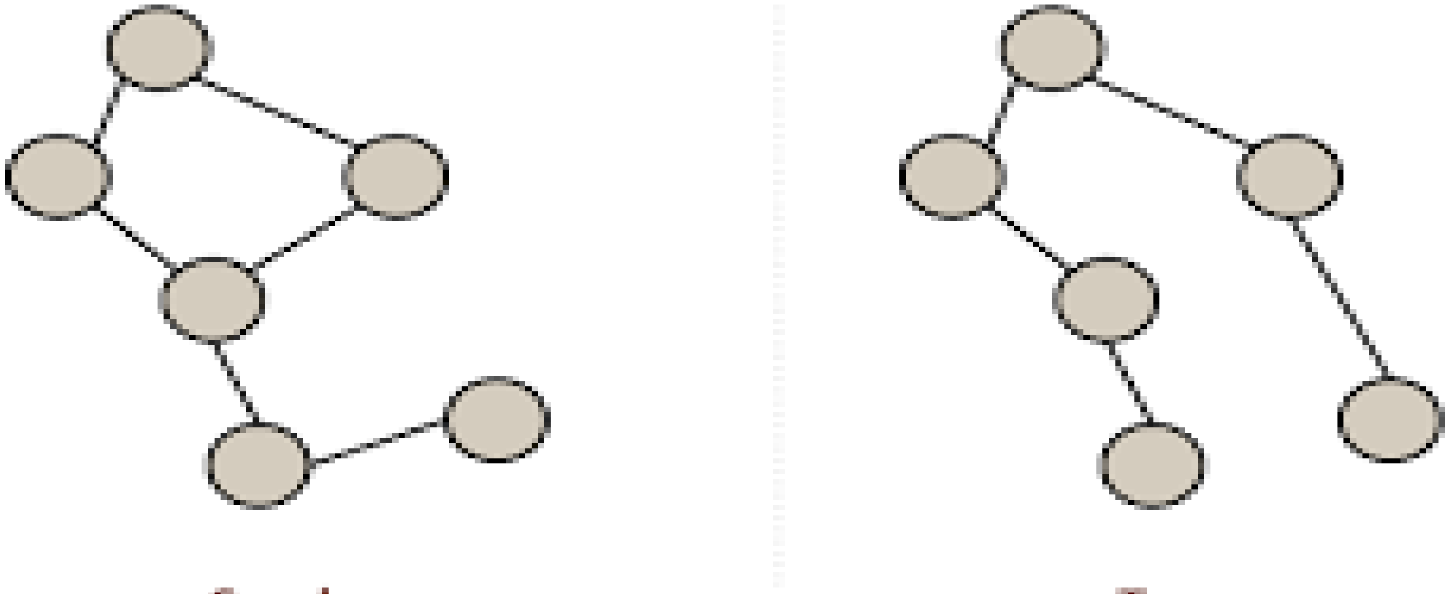
A graph G consists of a finite set of ordered pairs, called **edges E**, of certain entities called **vertices V**.

Edges are also called as arcs or links.

Vertices are also called as nodes or points.

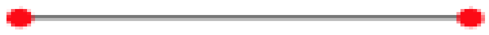


Graphs & Trees

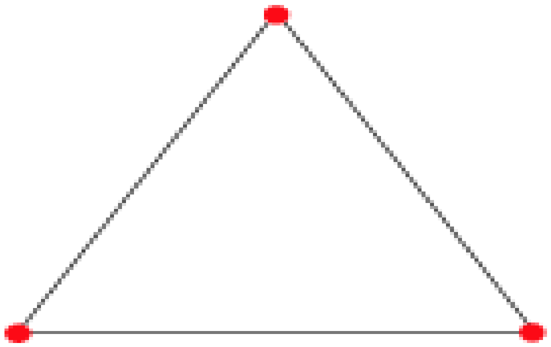


Graph

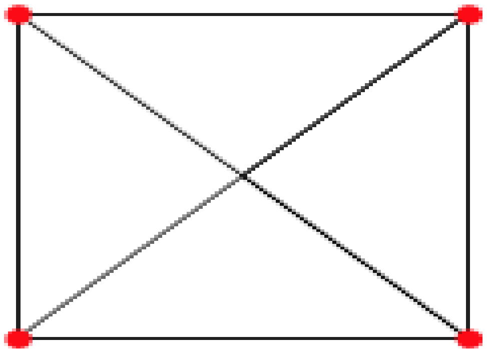
Tree



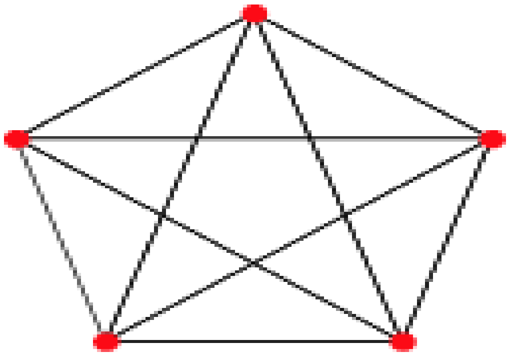
K_2



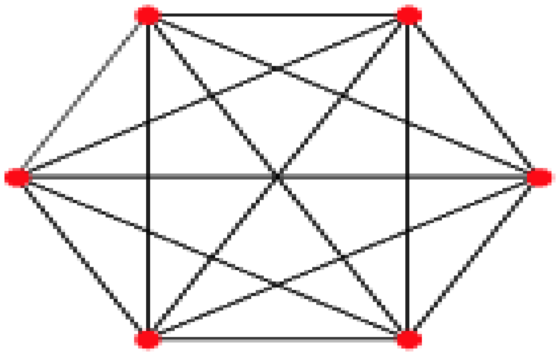
K_3



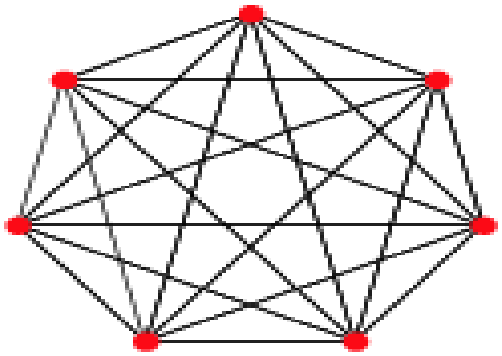
K_4



K_5

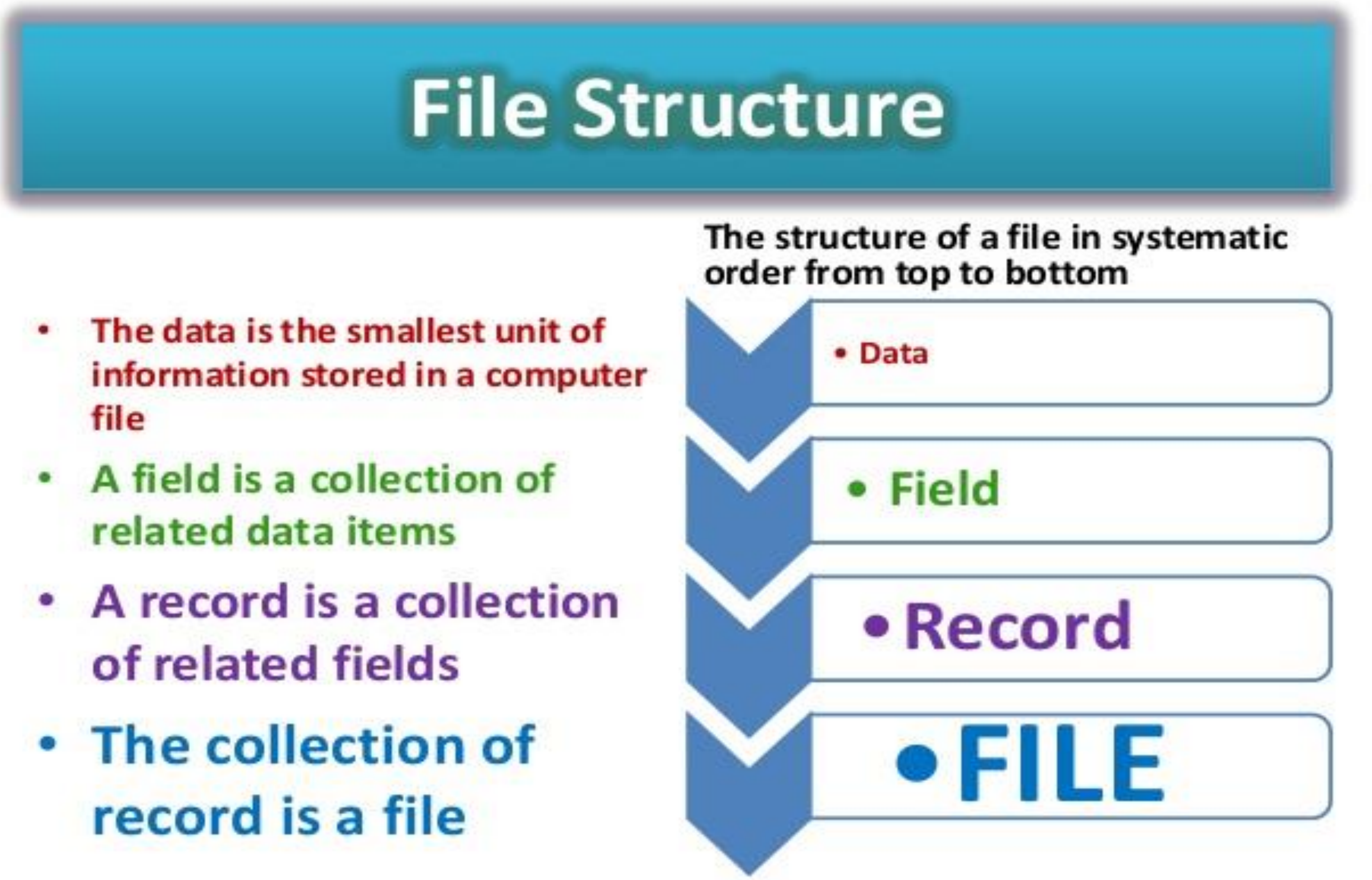


K_6



K_7

Field, Record and File

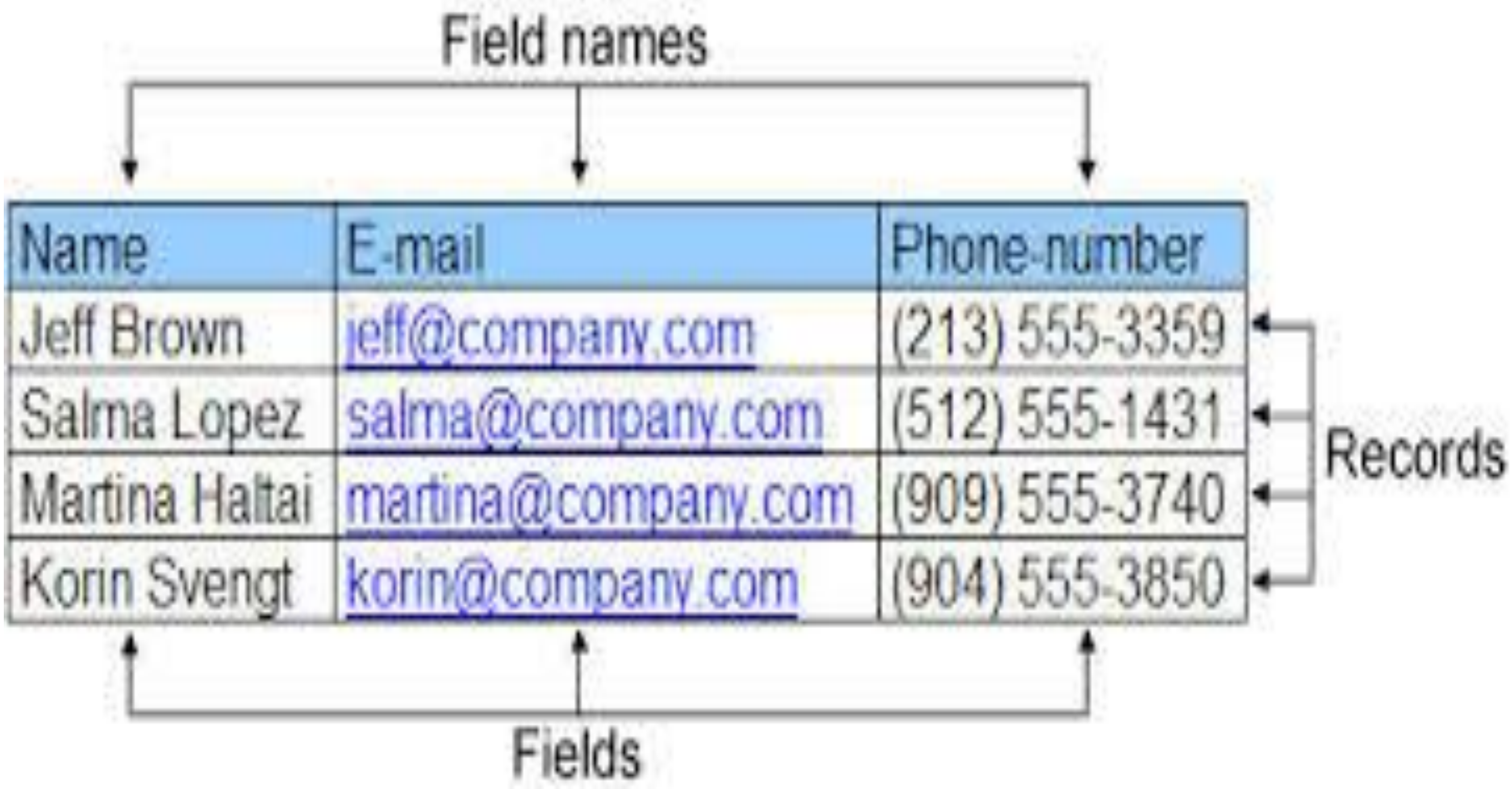


File Structure

- The data is the smallest unit of information stored in a computer file
- A field is a collection of related data items
- A record is a collection of related fields
- The collection of record is a file

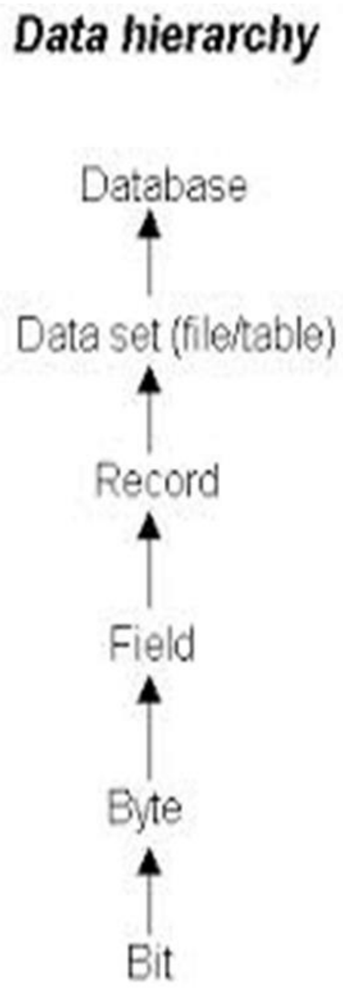
The structure of a file in systematic order from top to bottom

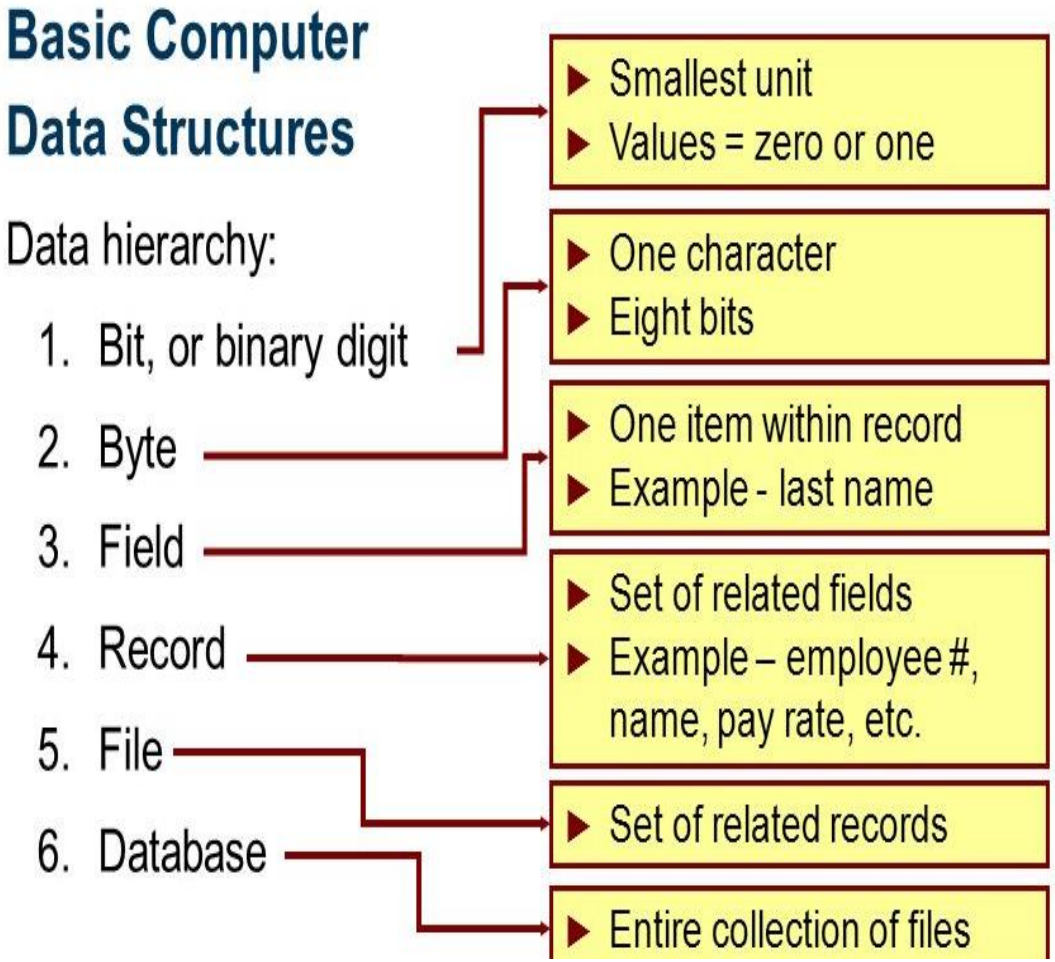
- Data
- Field
- Record
- FILE



Name	E-mail	Phone-number
Jeff Brown	jeff@company.com	(213) 555-3359
Salma Lopez	salma@company.com	(512) 555-1431
Martina Haltai	martina@company.com	(909) 555-3740
Korin Svengt	korin@company.com	(904) 555-3850

Data hierarchy





Basic Computer Data Structures

Data hierarchy:

1. Bit, or binary digit
2. Byte
3. Field
4. Record
5. File
6. Database

- Smallest unit
- Values = zero or one

- One character
- Eight bits

- One item within record
- Example - last name

- Set of related fields
- Example – employee #, name, pay rate, etc.

- Set of related records

- Entire collection of files

Data Structure Operation

Four Major Operation:

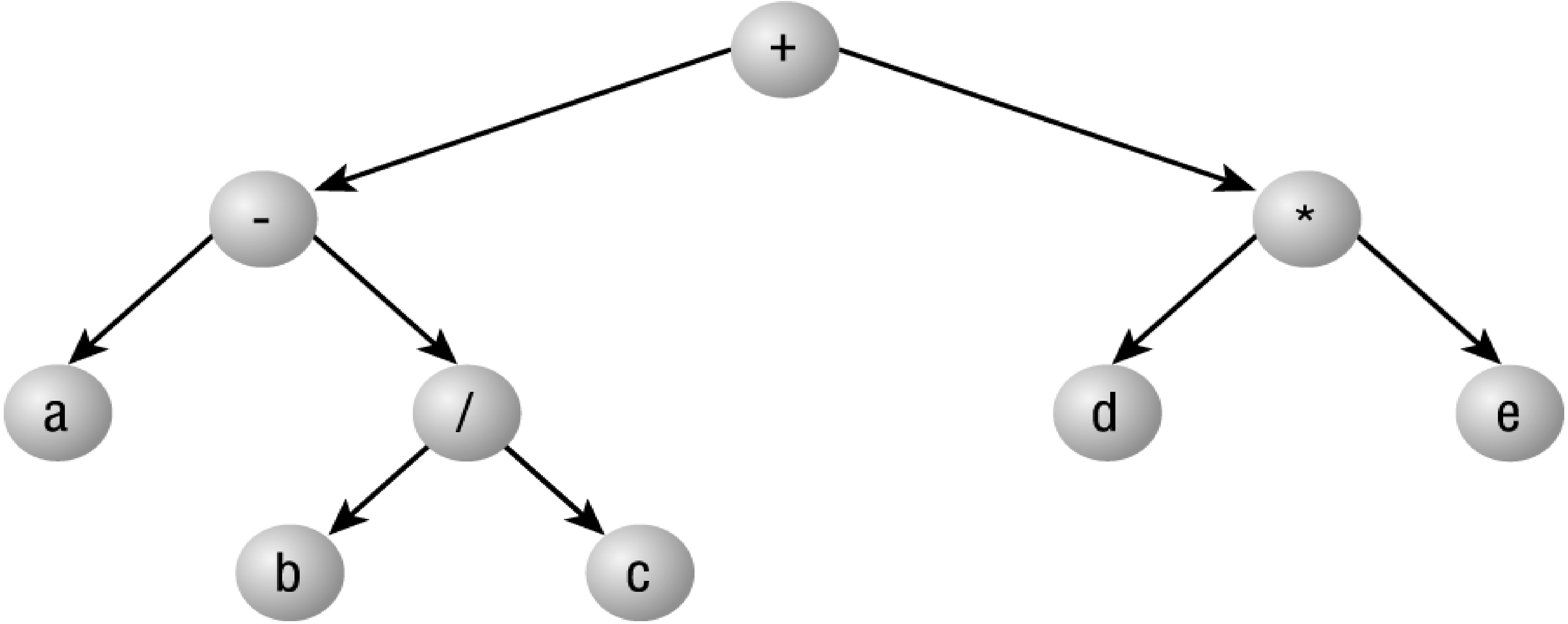
1) ***Traversing:*** Accessing /Visiting each record exactly once so that certain items in the record may be processed.
2) ***Searching:*** Finding the location of the record with a given key value, or finding the locations of all records which satisfy one or more conditions.
3) ***Inserting:*** Adding a new record to the structure.
4) ***Deleting:*** Removing a record from the structure.

Flowing two are special operations:

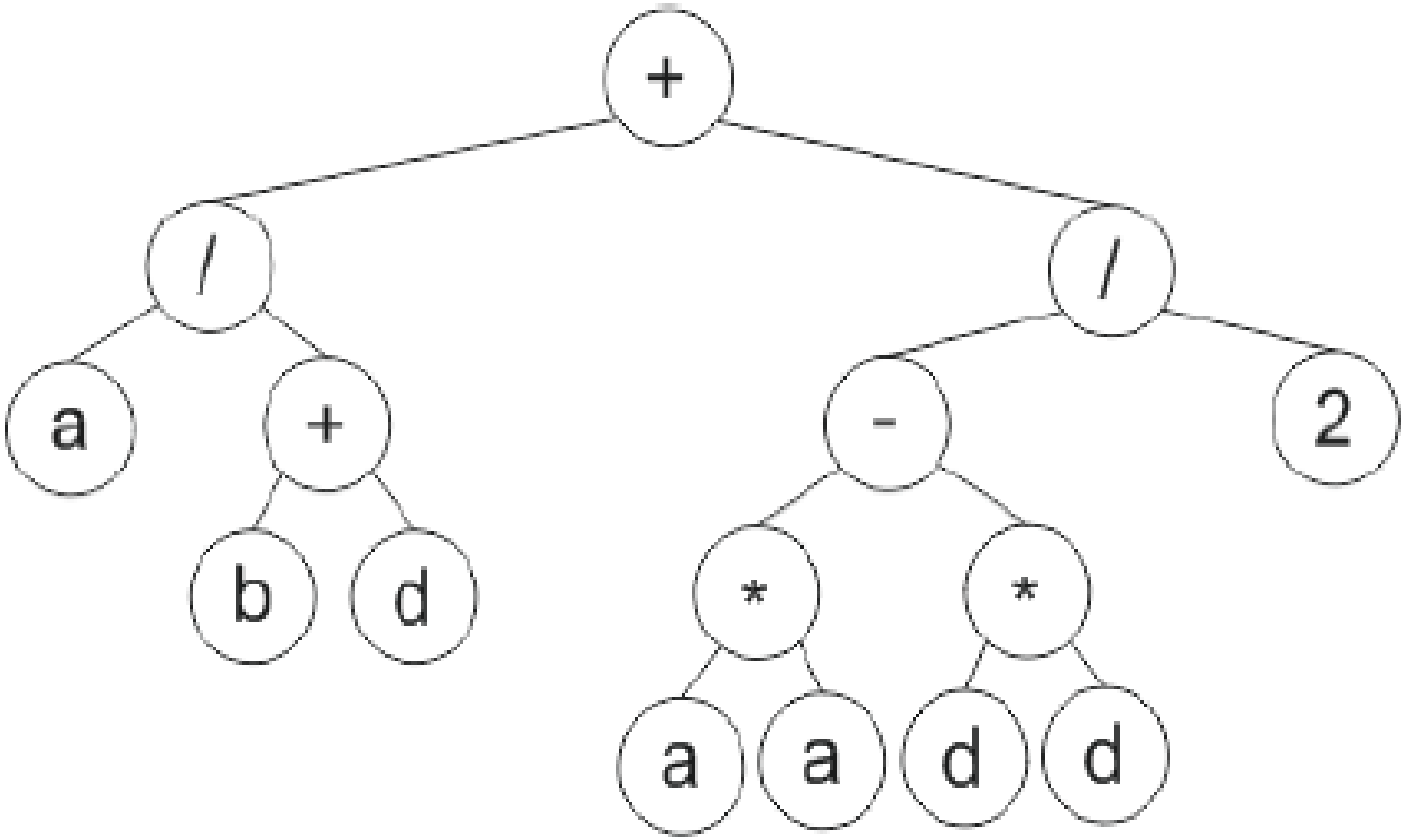
1) ***Sorting:*** Arranging the record in some logical order.
2) ***Merging:*** Combining the records in two different sorted files into a single sorted file.

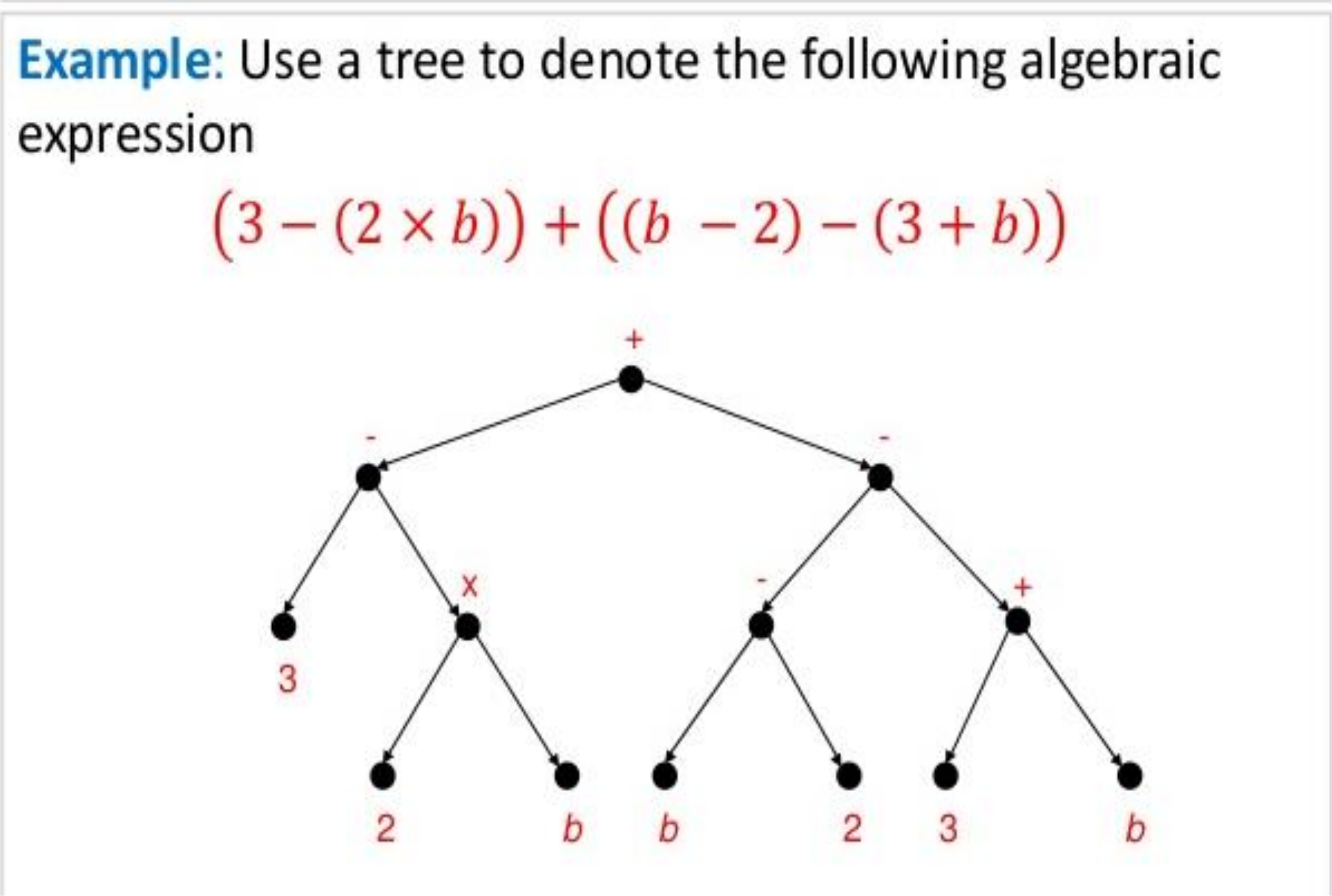
Tree from an algebraic expression

Expression: ((a - (b/c) + (d * e))



$$\frac{a}{b+d}+\frac{a^2-d^2}{2}$$





Example: Use a tree to denote the following algebraic expression

$$(3-(2\times b))+((b\ -2)-(3+b))$$

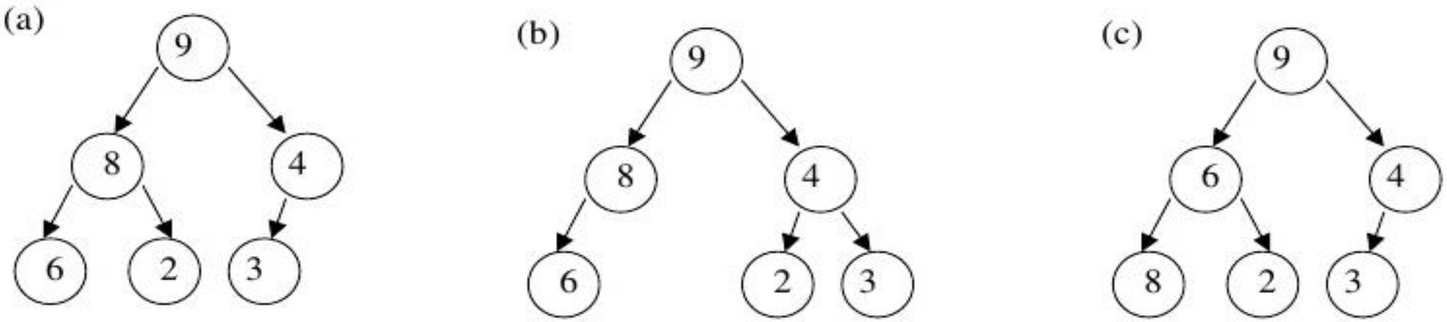
WHAT IS THE HEAP?

❑ A *heap* is a binary tree T that stores a key-element pairs at its internal nodes

❑ It satisfies two properties:

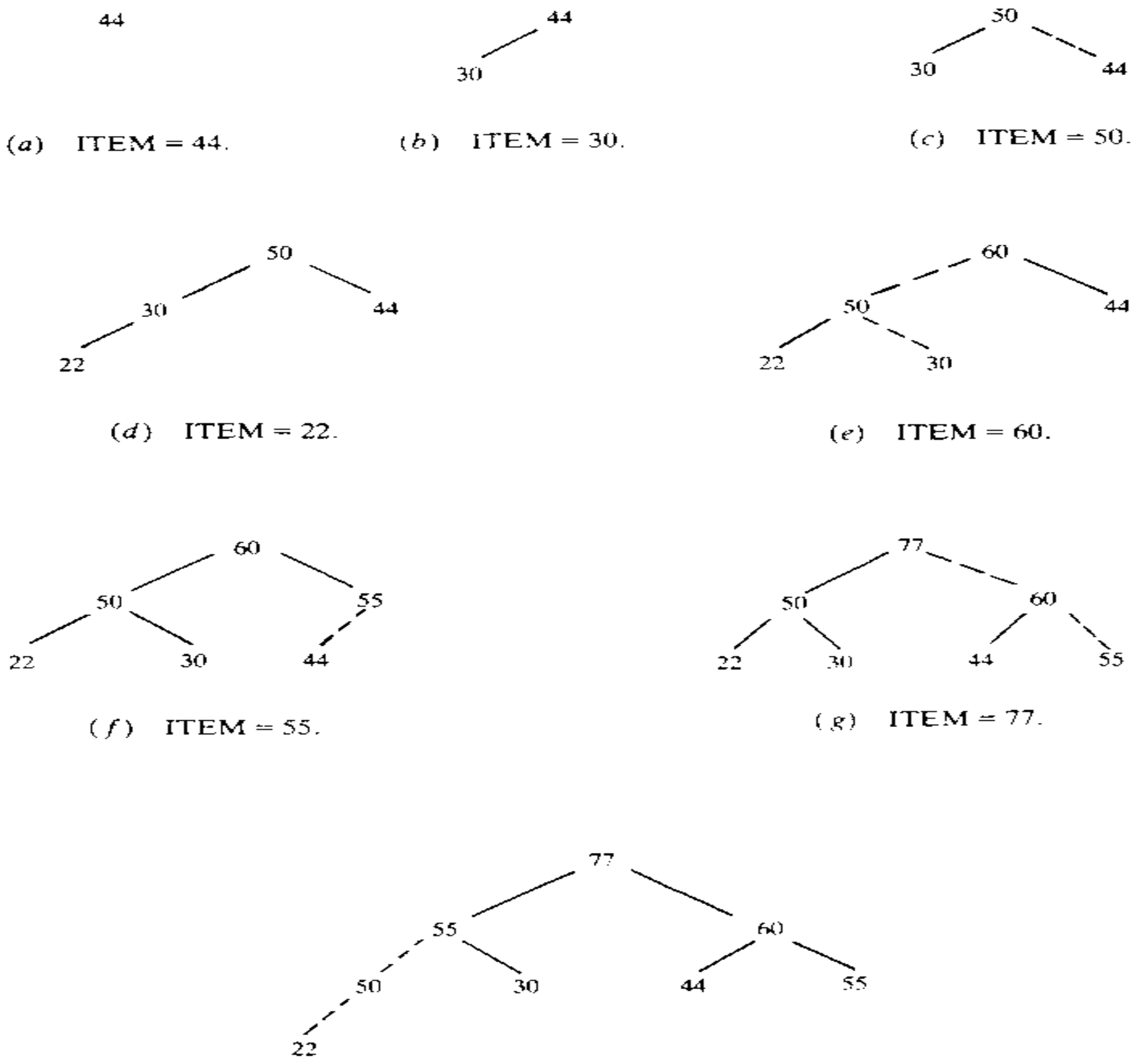
- MinHeap: key(parent) ≤ key(child)
- [OR MaxHeap: key(parent) ≥ key(child)]
- all levels are full, except the last one, which is left-filled

- Examples:



- In the above examples **only (a) is a heap**. (b) is not a heap as it is not complete and (c) is complete but does not satisfy the second property defined for heaps.

Max heap for the list of number 44,30,50,22,60,55,77,55



(*a*) ITEM = 44.

(*b*) ITEM = 30.

(*c*) ITEM = 50.

(*d*) ITEM = 22.

(*e*) ITEM = 60.

(*f*) ITEM = 55.

(*g*) ITEM = 77.