CS ACADEMY BY KETAKI

# DATA STRUCTURE IN ONE VIDEO

For Beginner

# ( TOPICS OF DATA STRUCTURE )

- Introduction of data structure
- Data structure operation
- Algorithmic notations
- Control structures
- Arrays – representation in memory
- Searching techniques
- Linked list
- Tree : binary tree

# LECTURE 1 ( INTRODUCTION TO DS )

- *Data :  simply value or set of value*

- *Group item : data item which are divided into subitems*
- *Ex: date : day/month/year*

- *Elementary item : items which are not divided into subitems ex: pin code*

# INTRODUCTION TO DS

- Entity : something which has certain properties

| Attributes | values |
| --- | --- |
| name | ram |
| age | 22 |

- Record : collection of field (single elementary unit)

- File : collection of record

# EXAMPLE

field

| name | address | number | Roll no |
|---|---|---|---|
| A | XYZ | 234 | 1 |
| B | ABC | 23456 | 2 |

Record

# WHAT IS DATA STRUCTURE

- Way to organized data
- Forming an organization characterized by accessing function .
- Collection of data
- It should be simple
- It should show simple  relationship

# LET'S REVISE

- Data
- Group item
- Elementary item
- Entity
- Field
- Record
- File
- What is data structure
- Types of data structure

*The data structure operations :*

1 Traversing
2 Inserting
3 deleting
4 searching
5 sorting
6 merging

# DELETING

Defination :
removing record from existing structure

Ex :

**Searching : finding a location of record**

**Sorting : arranging record in some logical order**

**Merging : combining of records in two different sorted files into a single sorted file**

# LECTURE 3

# ALGORITHMIC NOTATION

# Algorithm :

Defination : algorithm is a finite step by step list of well define instruction for solving a particular problem

Ex :2 + 3 = 5

- ✓ step 1 =2
- ✓ step 2=2+3
- ✓ step 3=2+3=5

# ALGORITHM

Algorithm consists of 2 parts :

1st part : paragraph which tells the purpose of algorithm

2nd part : it consist of steps of actual algorithm

# Points :

- In 1st part we define variable in algorithm and list the input data
- In 2nd part the control can be transferred to step n.
- The algorithm is completed when the statements “exit” or “stop” is encountered .

# Algorithm Example :

Algorithm for addition of two number i.e. 2 and 3

- X=2 ;
- Y=3;
- Set z=1;
- Z = x+y;
- Then Z = 5 ;
- STOP ;

# DEFINE ALGORITHM AND GIVE CHARACTERISTICS OF ALGORITHM

- ✓ Algorithm is an effective way to solve any problem
- ✓ Algorithm is used for calculation , data processing .
- ✓ Each algorithm is list of well define instruction for complete task
- ✓ Algorithm is not only base of data structure but also the base of good programming

## Characteristics of properties of algorithm

1. Input : each algorithm require input, input is necessary.

2. Output : each algorithm will generate result after processing the input

3. Definiteness : each algorithm must be simple and clear

4. Finiteness : each algorithm must have proper starting and end

5. Effectiveness : each algorithm firstly sketch on paper so that instruction must be feasible

# CONTROL STRUCTURE

LECTURE 4

# CONTROL STRUCTURE

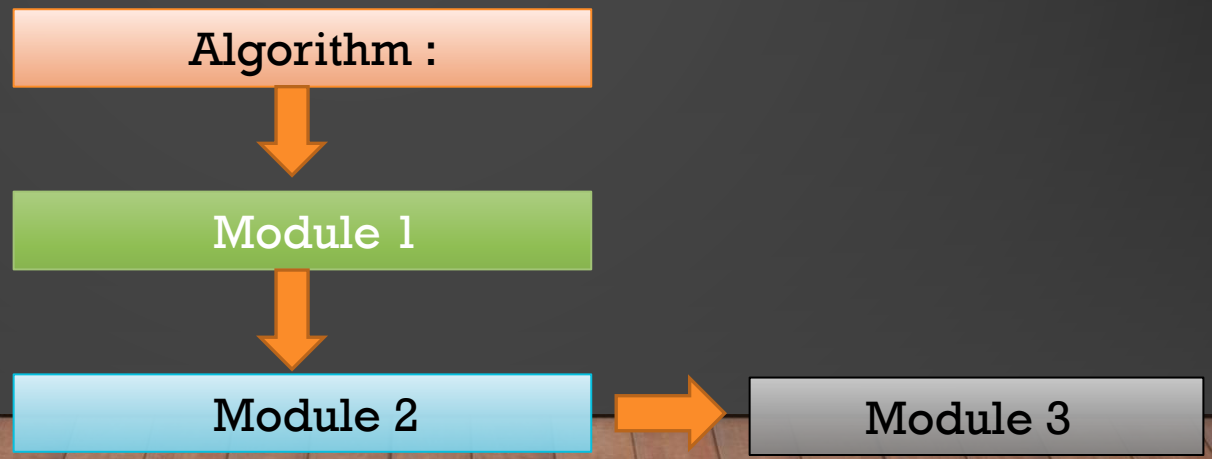
1. Control structure are just a way to specify flow of control in program.
2. Any algorithm can be more clear and understood if they use self contained module called as a **control structure or logic**
3. Basically it analyze and choose the direction of programon certain condition
4. There are main three main types of control structures :
    1. Sequence flow
    2. Selection flow
    3. Iteration flow

# SEQUENCE LOGIC ( SEQUENCE FLOW )

Sequence flow = serial flow

Defination : sequence means modules are executed sequentially means one after another

Ex :



# SEQUENCE FLOW

- In the sequence flow modules are executed in serial i.e one after another
- In this flow it follow the sequence of modules in which the modules are written.
- Most of the processing even some complex program will follow this type of sequence flow pattern.

# SELECTION FLOW

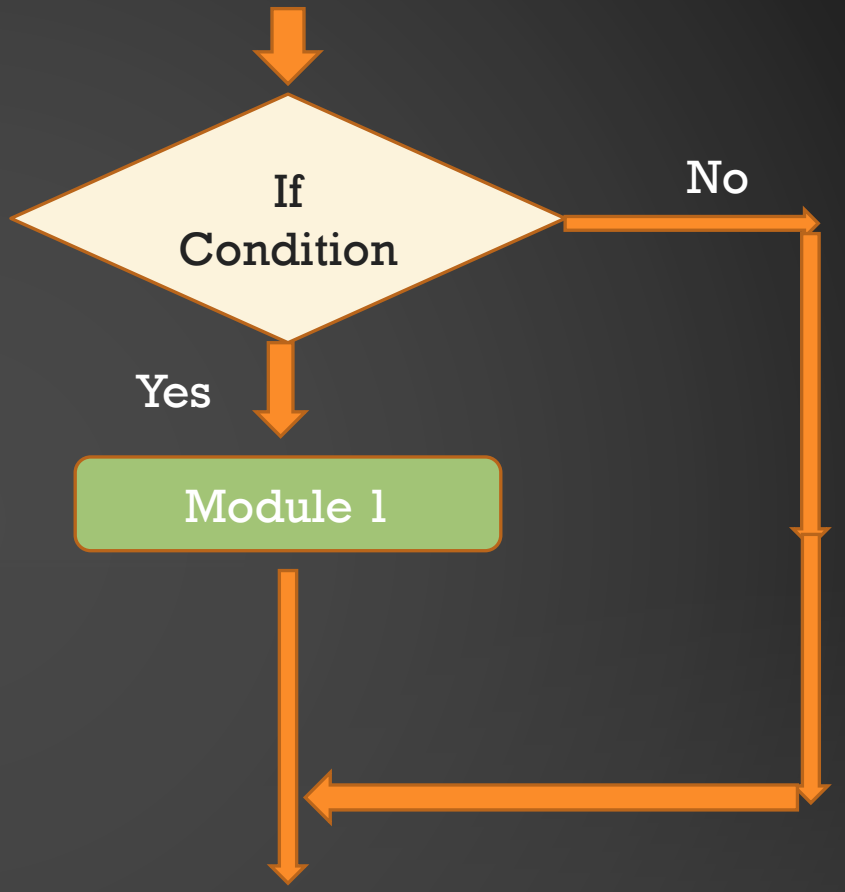
- Selection flow uses number of condition
- In this it use one out of several alternative modules or condition
- The structure which implement this type of logic or flow known as selection flow or conditional flow
- There are main three types of conditional structure
  - Single alternative
  - Double alternative
  - Multiple alternative

# SINGLE ALTERNATIVE

If condition ,then

[module 1 ]

[ end of if structure ]



# DOUBLE ALTERNATIVE

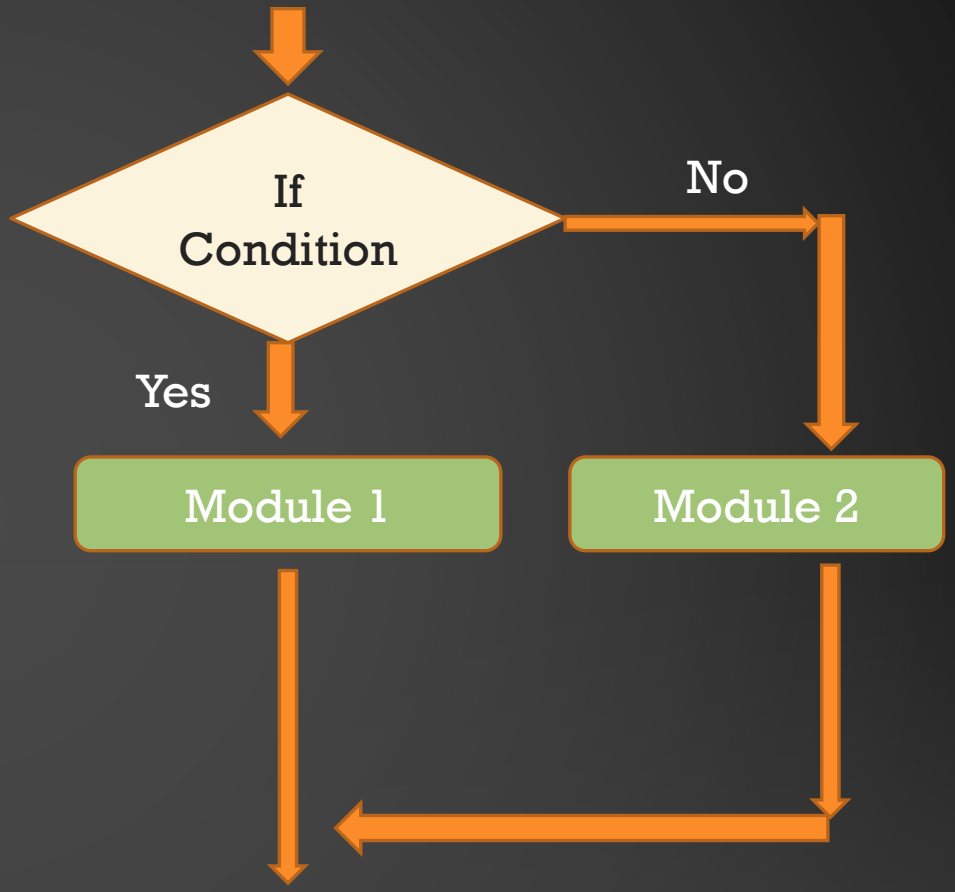
If condition ,then

[module 1 ]

Else

[ module 2 ]

[ end of if structure ]



# MULTIPLE ALTERNATIVE

If condition (1),then :

[module A1 ]

Else if condition (2), then ;

[ module A2 ]

-

-

-

Else if condition(n), then ;

[ module An ]

Else :

[ module B ]

[ end of if structure]

- The logic of this structure allows only one module to be executed .
- The following the condition which is satisfied the condition will be be executed .
- If no condition is satisfied then the module which follow last else statement will be executed

# ITERATION LOGIC OR ITERATION FLOW

Iteration : processing one code again and again called as iteration .

There are 2 type of iteration logic :

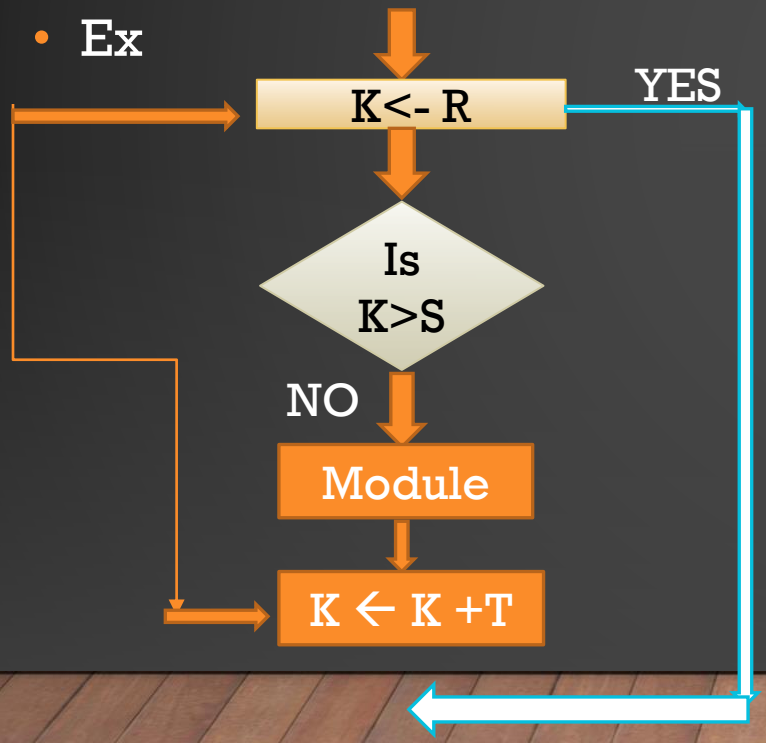
1. Repeat-for –loop

2. Repeat-while-loop

# ITERATION FLOW

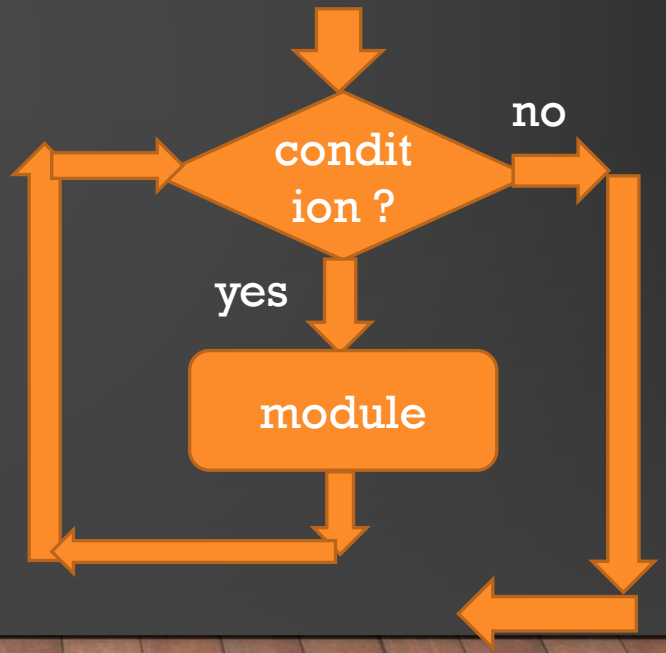
## REPEAT-FOE-LOOP

- Ex




## REPEAT-WHILE-LOOP

EX



# ARRAY

# WHAT IS ARRAY ?

- Array is collection of similar type of data element.
- A single array variable  is capable to hold number of values at a time depending upon size of array .
- Array is a linear type of data structure.
- Element are stored in successive memory locations.
- Ex a[4] = 40.

# REPRESENTAION OF ARRAY IN MEMORY

- Elements if linear array are stored in consecutive memory location
- Computer dose not need to stored or to keep a track of address of every element of array
- It just requires the address of first element of array called as a base address

- Formula :

Address of any element (k) = base address (B) + memory per element (M) ( K- lower bound (LB) )

FORMULA : K = B + M * ( K – LB )

# EXAMPLE

- Formula :

**Address of any element (k) = base address (B) + memory per element (M) ( K- lower bound (LB) )**

Q. Consider the array A which have the number of automobiles from 100 through 350 .
Suppose that base address is 200 and memory per element is 4 .then calculate the address
At which 250's record is stored .

Ans : k =250
B= 200
m =4
Lb =100

K = 200 + 4 *( 250 -100)

=200 + 4*( 150)
200+ 600
800

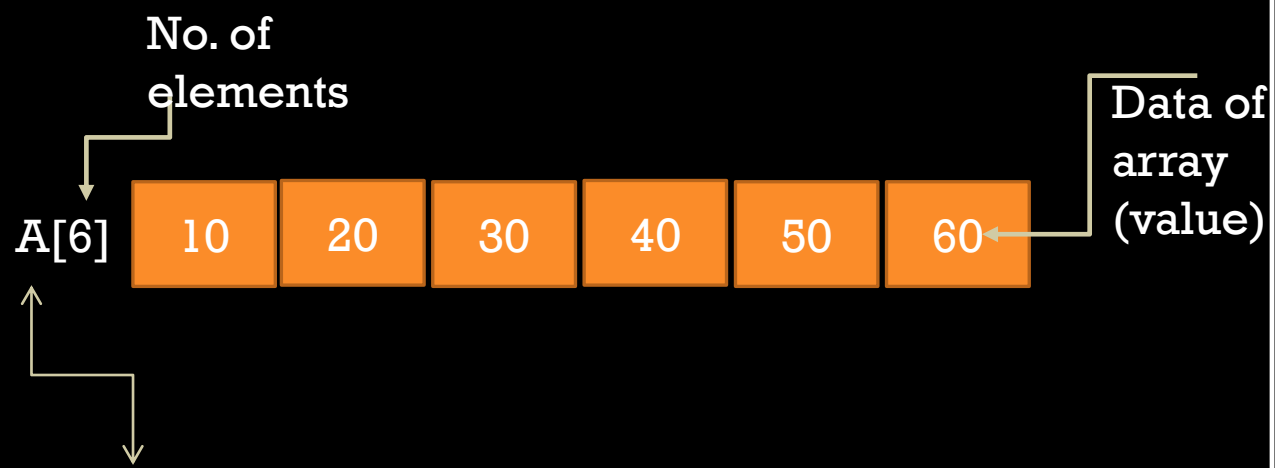
# ARRAY TRAVERSING

LECTURE 6

# Previous lecture :

## ARRAY

- DEFINATION : array is collection of similar data elements
- Ex of ARRAY




1. What is array ?
2. Details about array .
3. Representation of array in memory
4. Formula to find the address of particular element
5. Ex.

# IN THIS LECTURE :

- What is traversing ?
- What is Traversing an array ?
- Algorithm for traversing of linear array .

# TRAVERSING

Defination : accessing each record or element exactly once so that it can be processed is called as traversing

Ex :

# TRAVERSING AN ARRAY

- Traversing an array means accessing with each element of array at onceso that it can be processed.

- Ex .

# ALGORITHM FOR TRAVERSING OF ARRAY

1. [ Initialize counter ]  set k = LB
2. Repeat step 3 amd 4 while k <= UB
3. Visit element ( apply process to E[k]
4. Increment counter set K =K+1
5. END of loop
6. Exit

- Example
- Array B[ 10 ,20 ,30 ,40 ,50 ]
- Set  k=1
- Repeat step 3 & 4 up to K<=UB
- Visit E [K]
- Increment K = K+1
- End of loop
- exit

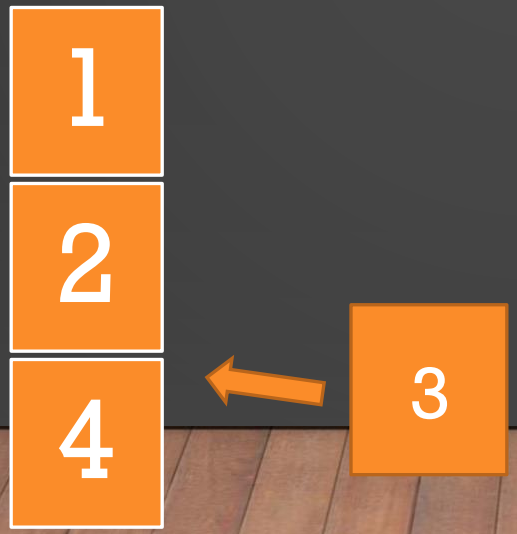
# INSERTING IN ARRAY

Lecture 7

# INSERTING

Defination : adding new record to existing structure

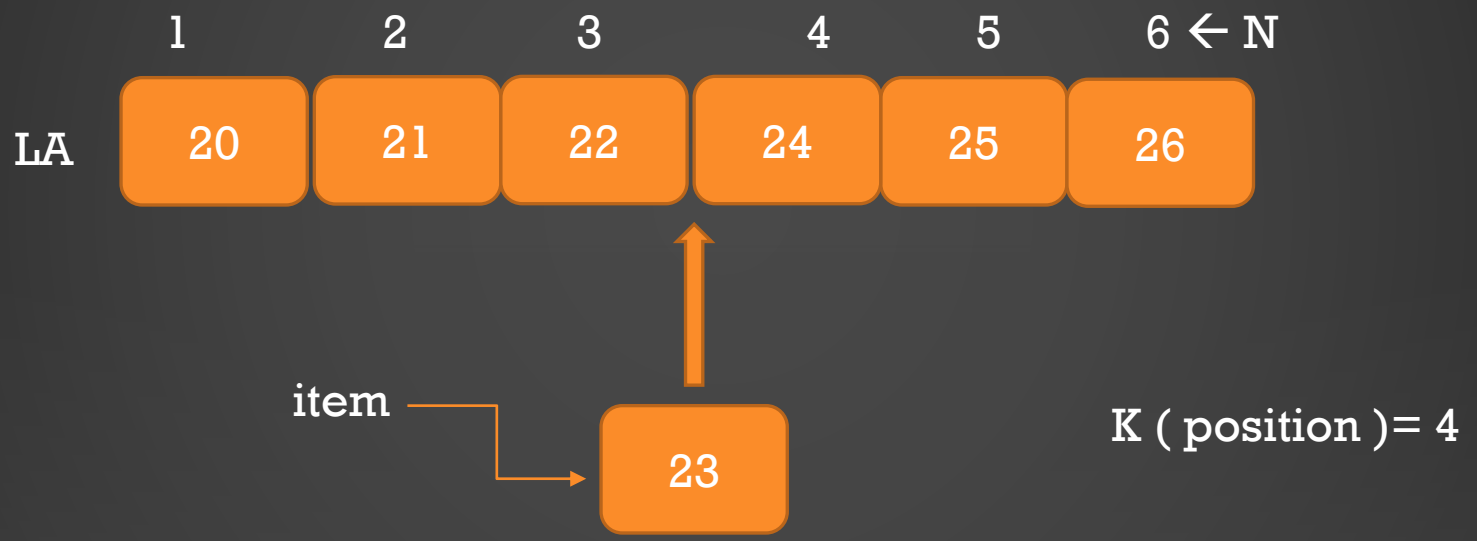
Ex :



# WHAT IS INSERTING ?

- Inserting refer to the operation of adding an element to the existing element of array.
- The element can be easily inserted at the end of array
- But for insertion in the middle of array it require to move the elements of array one byte forward.

# ALGORITHM FOR INSERTION



# ALGORITHM FOR INSERTING

ALGORITHM FOR INSERTING AN ELEMENT TO ARRAY AT K POSITION ……

LA = Linear array
N =no of element
K = positon
ITEM = element
Steo 1 : initialize counter
[set j=N]
Step 2 :repeat step 3 & 4 while j>=K

Step 3 : move jth element forward
[set LA[ J+1] =LA [ j ]]

Step 4 : decrement counter
[ j =j -1]
Step 5 : insert an element set LA [ K] = ITEM

Step 6 : set N = N+1

STEP 7: exit [ end ]

# DELETION OF ELEMENT FROM AN ARRAY

Lecture 8

# WHAT IS DELETING ?

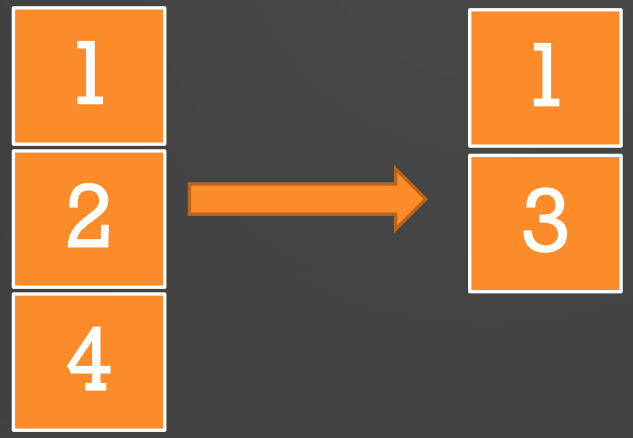
- Deleting means removing an element from existing element of array .
- Deletion at the end of array is easy.
- If to delete an element from mid of array then move the element of array one location upward.

# DELETING

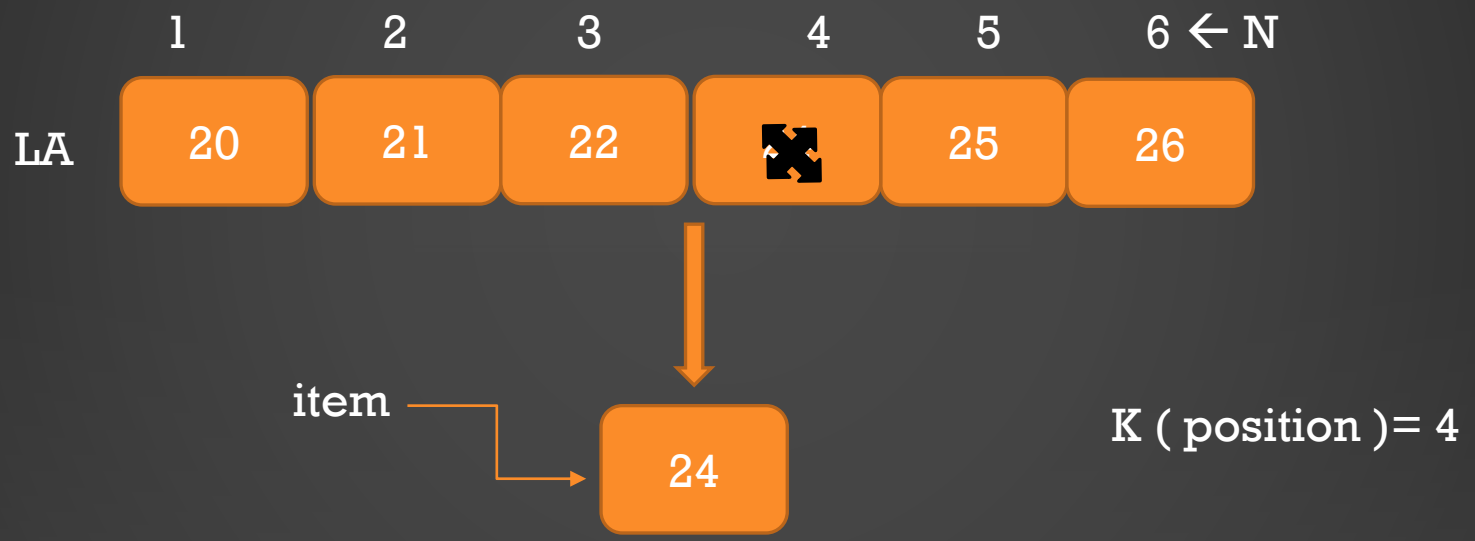
Defination :

removing record from existing structure

Ex :



# ALGORITHM FOR DELETION



# ALGORITHM FOR DELETING

ALGORITHM FOR DELETING AN ELEMENT TO ARRAY AT K POSITION ……

LA = Linear array
N =no of element
K = positon
ITEM = element
Steo 1 : initialize counter
[ set ITEM=LA[K] ]

Step 2 :repeat step 3 FOR j=K to N-1

[ MOVE (j+1)th element backward ]

Step 3 [set LA[ J] =LA [ j+1 ]]

Step 4 end of loop

Step 5 : set N = N-1 [ reset number of element of array ]

STEP 7: exit [ end ]

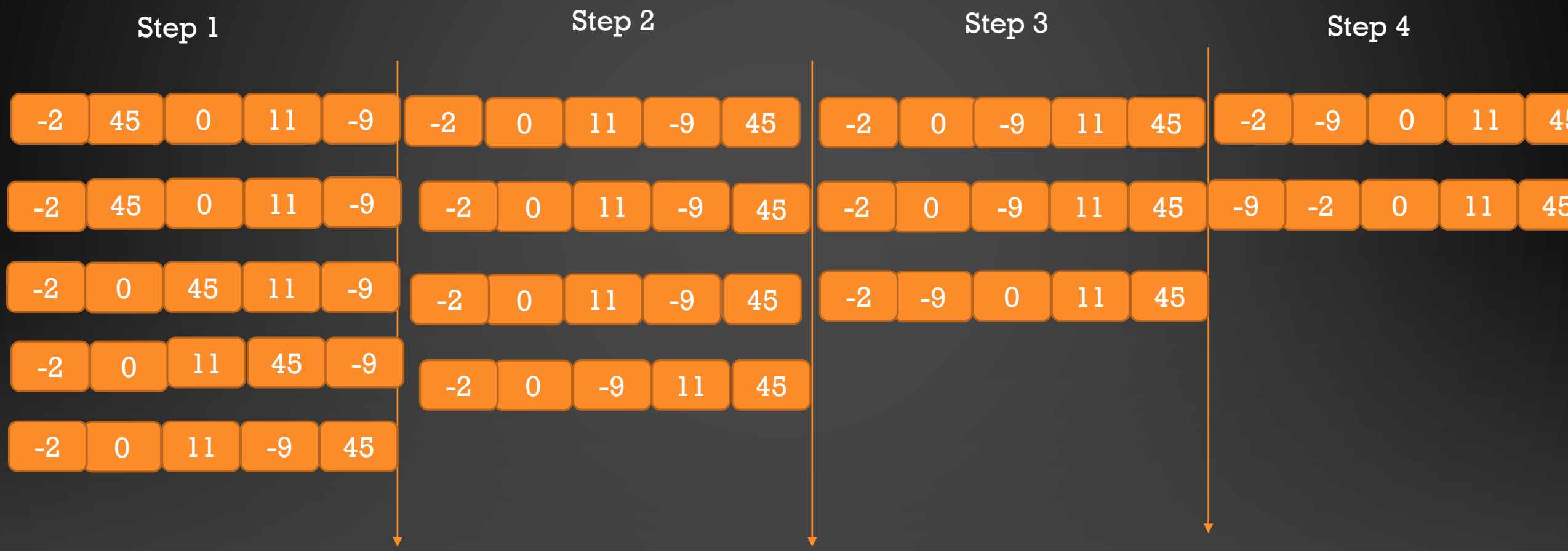
# BUBBLE SORT

LECTURE 9

# WHAT IS BUBBLE SORT ?

- ✓ Bubble sort algorithm starts by comparing the two elements of an array .
- ✓ If you want to sort the element of array in ascending order if the 1st element is greater than 2nd then you need to swap the element.
- ✓ Then again 2nd and 3rd are compared and swapped . If it is necessary and this process go on until last and second last element is compared and swapped .

# EXAMPLE


Step 1
Step 2
Step 3
Step 4
-2 45 0 11 -9
-2 45 0 11 -9
-2 0 45 11 -9
-2 0 11 45 -9
-2 0 11 -9 45
-2 0 11 -9 45
-2 0 11 -9 45
-2 0 -9 11 45
-2 0 -9 11 45
-2 -9 0 11 45
-2 0 -9 11 45
-9 -2 0 11 45
-2 -9 0 11 45

# ALGORITHM TO SORT AN ARRAY USING BUBBLE SORT

Step 1 : repeat step 2 and 3 for i=1 to n

Step 2 : set j=1

Step 3 : repeat while j<=n

1 2 3

if ( a[ j ] < a[ j+1 ] )

interchange a[ i ] and a[ j ]

[end of if structure]

set j = j+1

[end of inner loop ]

[end of outer loop ]

Step 4 : end

for (i=1 to n)

{ j = 1

while( j <= n)

# SEARCHING TECHNIQUE
# [ LINEAR SEARCH ]

Lecture 10

# WHAT IS SEARCHING ?

- Searching means to find out a particular element from a given list of elements.
- It also  check the whether the require element is present in array or not .
- There are two types of searching technique  as follow :
    - Linear search
    - Binary search

# LINEAR SEARCH

- In linear searching technique the given element is compared with each element of list one by one
- This one is very easy method to find out the position of given element or to find out whether the given element is present in array or not.
- It require more time than binary search .

# EXAMPLE OF LINEAR SEARCH


N =no of element

DATA = name of array

ITEM = given element

# ALGORITHM OF LINEAR SEARCH

Step 1 : item =given element
n = no of element
data = name of array

Step 2 : set i=1
[ initialize counter ]

Step 3 : repeat while ( data[i] =! item )
{

i = i+1 ;

}

Step 4 : if i= n+1
set i=0

Step 5 : end

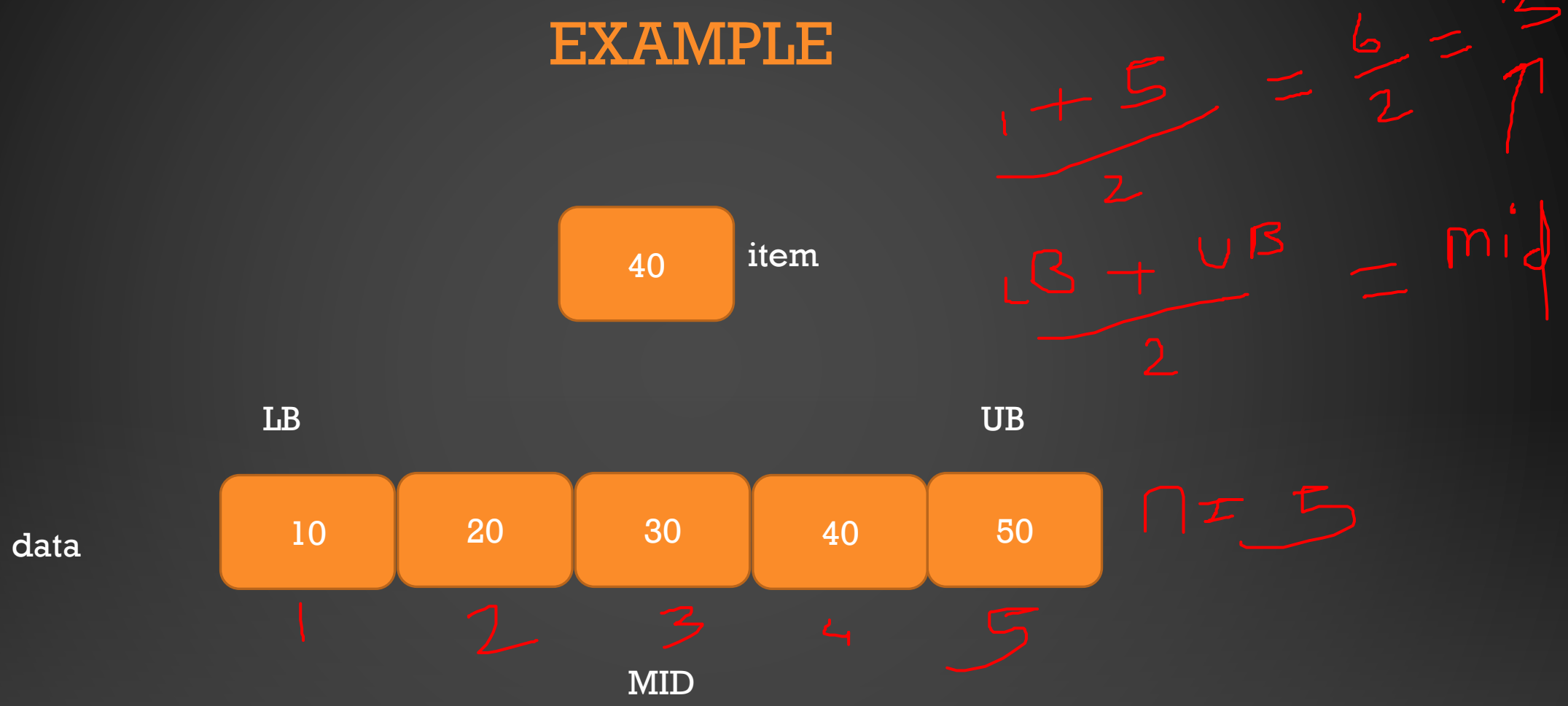
# BINARY SEARCH ALGORITHM

Lecture 11

# WHAT IS BINARY SEARCH ?

- Binary search is a very useful technique of searching .
- Binary search is used to search an element from sorted array
- For binary search array must be in sorted form

# EXAMPLE



# ALGORITHM

Step 1: initialize vaiable

LB =data[ 1 ]

UB =data[ n ]

Step 2 : while ( LB<=UB)

MID= [ ( LB+UB)/2 ]

if ( data[ MID ] = item)

goto end step.

else if ( item < data[ MID ] )

UB= MID-1

else

LB = MID+1

Step 3 : stop

# ADVANTAGES AND DISADVANTAGES

## ADVANTAGES :

- It is efficient as the search scope get reduced to half the size of array with each iteration.
- Required less time to search an element

## DISADVANTAGES :

- The given list must be in sorted form
- The middle element can be accessed
- In each iteration middle entry calculation is required

# POINTER ARRAY

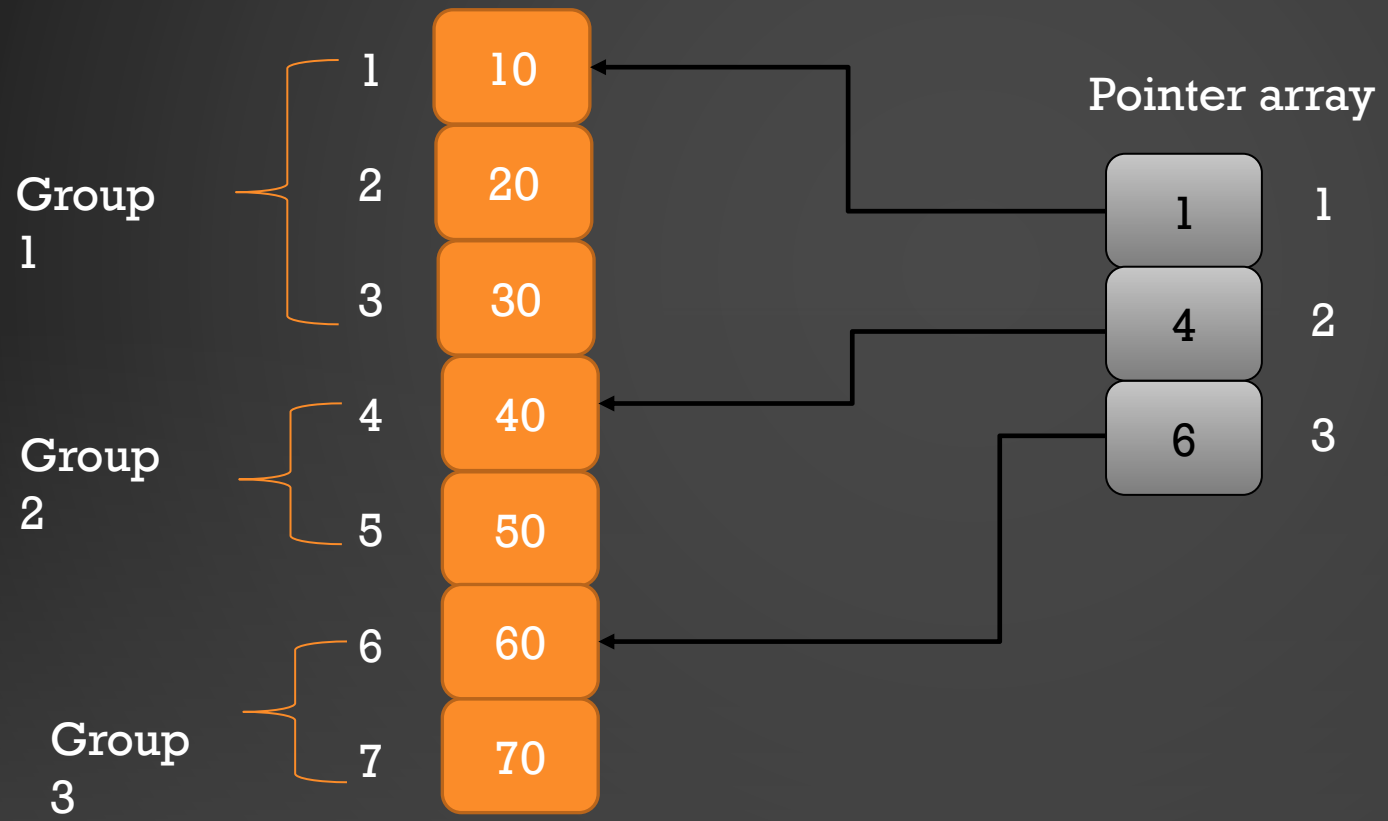
Lecture 12

# WHAT IS POINTERS ARRAY ?

[ OCT-2003 , MAR-2012,15 JULY-16]

- The array is called pointer array if each element of that array is pointer.
- The variable is called as pointer variable if it points to another variable
- Pointer variable contains the memory address of another variable.
- Each element of pointer array is pointer which hold the address of another array.
- pointer array which contains the starting address of each group i. e. Each array

# EXAMPLE


Group 1
Group 2
Group 3
1
2
3
4
5
6
7
10
20
30
40
50
60
70
Pointer array
1
4
6
1
2
3

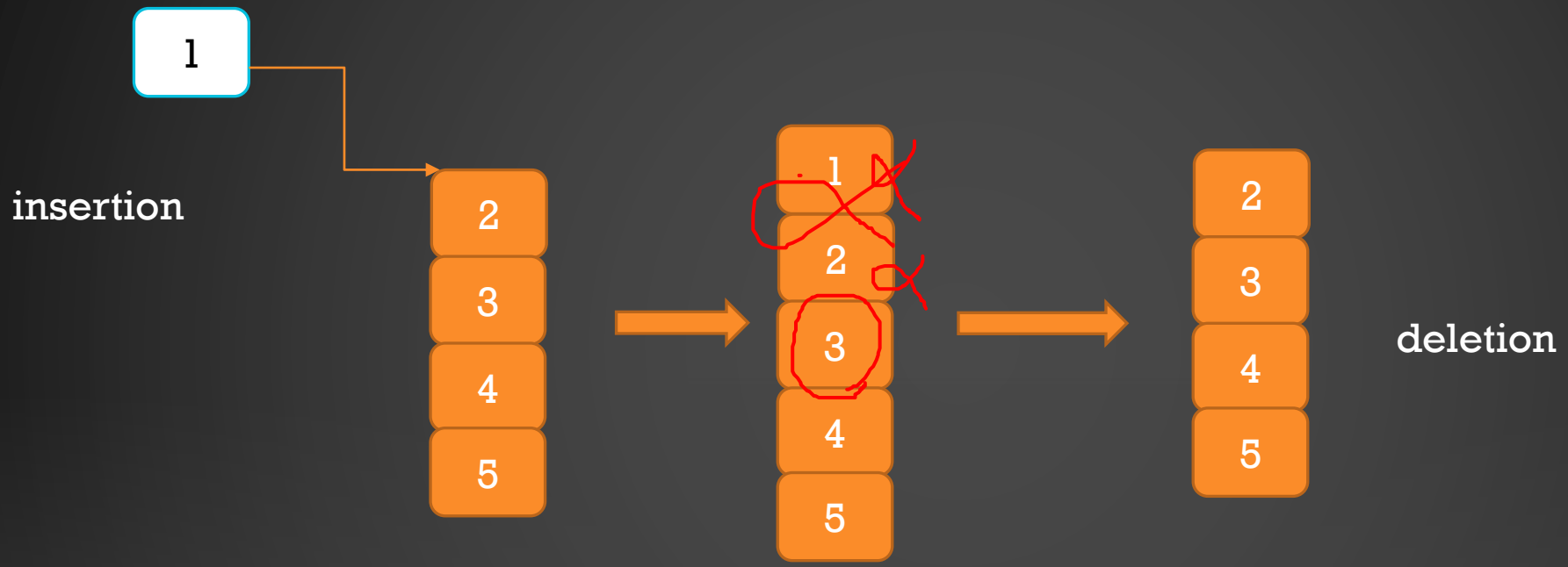
# STACK AND QUEUE

Lecturer 13

# WHAT IS STACK ?

- Stack is a linear type of data structure .
- In stack insertion and deletion takes place at only at one end that end called as top of stack
- In stack what ever element is inserted last is deleted first this system is called as LAST IN FIRST OUT [ LIFO ]
- Stack perform two operation :
    - PUSH : insertion of element
    - POP : deletion of element

# PUSH AND POP




Ex : stack of dishes .

# QUEUE

- Queue is linear data structure .
- Queue have two end in which insertion and deletion can takes place at two separate end.
- The end which perform inseretion operation called as RARE END.
- The end which perform deletion operation called as FRONT END .
- The element which inserted first is deleted first so that why the queue call as a  FIFO system.
- Ex .a queue for ticket in cinema hall.

# TREE DATA STRUCTURE

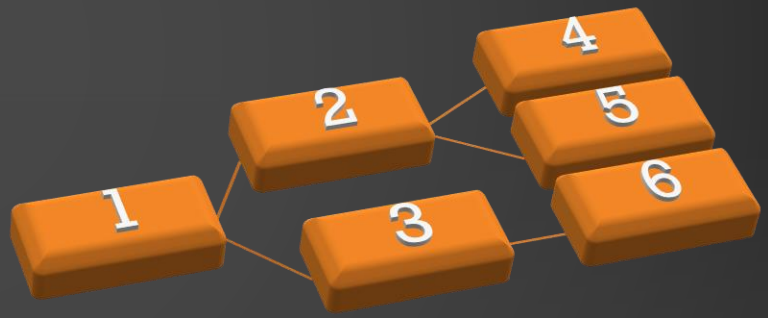
Lecture 14

# TYPES OF DATA STRUCTURE

- Linear data structure


Ex : arrays , linked list

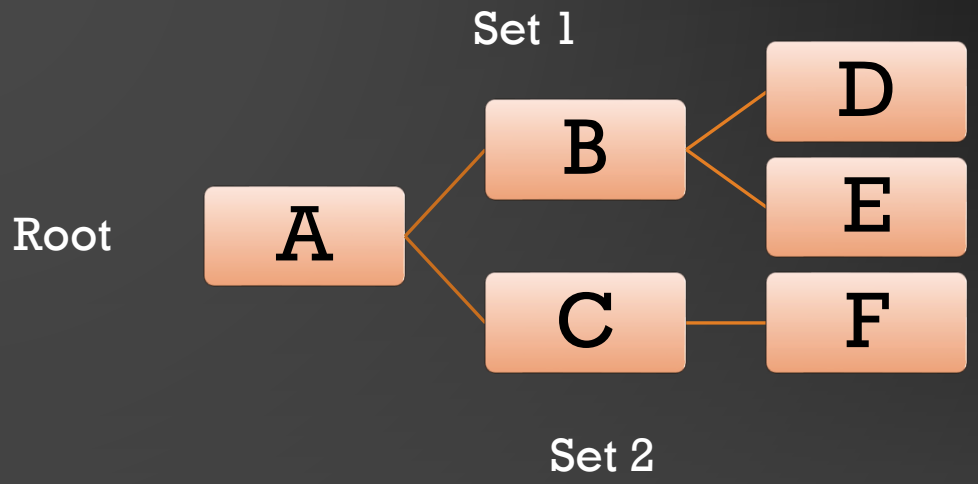
- Non-linear data structure




Ex : tree , graph

# WHAT IS TREE ? WHAT DO YOU MEAN BY ROOT , LEAF ,SIBLINGS AND CHILD ABOUT TRERE [ MAR 2010 , OCT 2006 ]

- Tree is a non - linear hierarchical data structure which contain collection of data items.
- there is special design node called as a node .
- Remaining nodes are partitioned into finite disjoint sets.



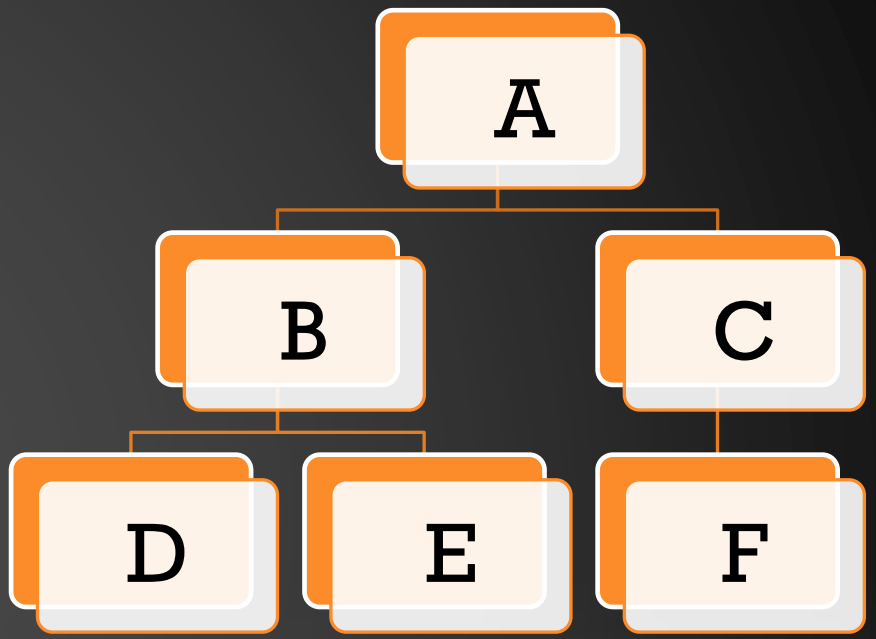
# DEFINATION

ROOT : Node which has no parent . Generally 1st nodeis called as root node . from ex A is a root node

LEAF : the node which has no child or children such node have degree zero. D , E , F are leaf node.

CHILD : the node which are reachable from a node through a single edge are called a children . Ex B & C are child of root A.

SIBLING : child of same parents called as a siblings . EX D and F are sibling.

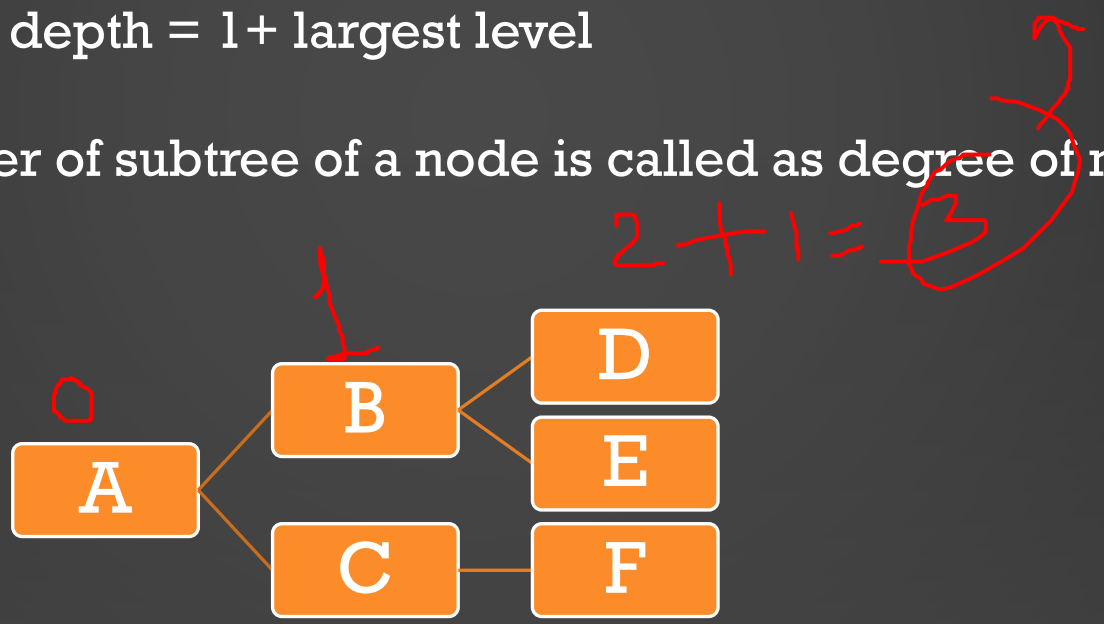


# DEFINATIONS

1. LEVEL OF TREE : each node in tree is assigned a level number. Generally levelof root node is zero and every other node is assigned to level number one more tthen level of its parent.

2. DEPTH / HEIGHT : Depth of tree means a maximum level of any node in tree. the depth = 1+ largest level

3 . DEGREE : the number of subtree of a node is called as degree of node

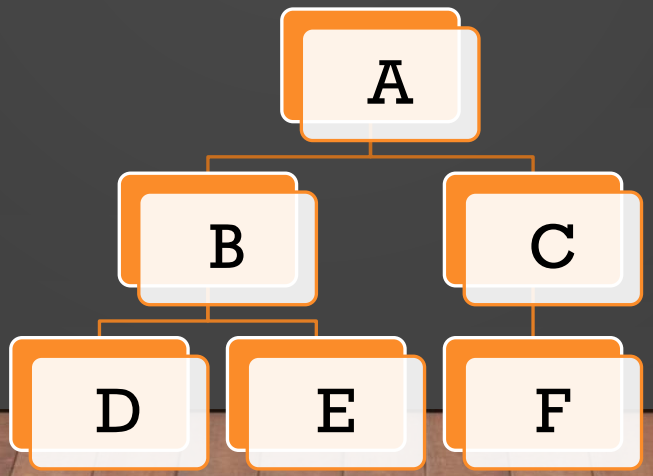


# BINARY TREE

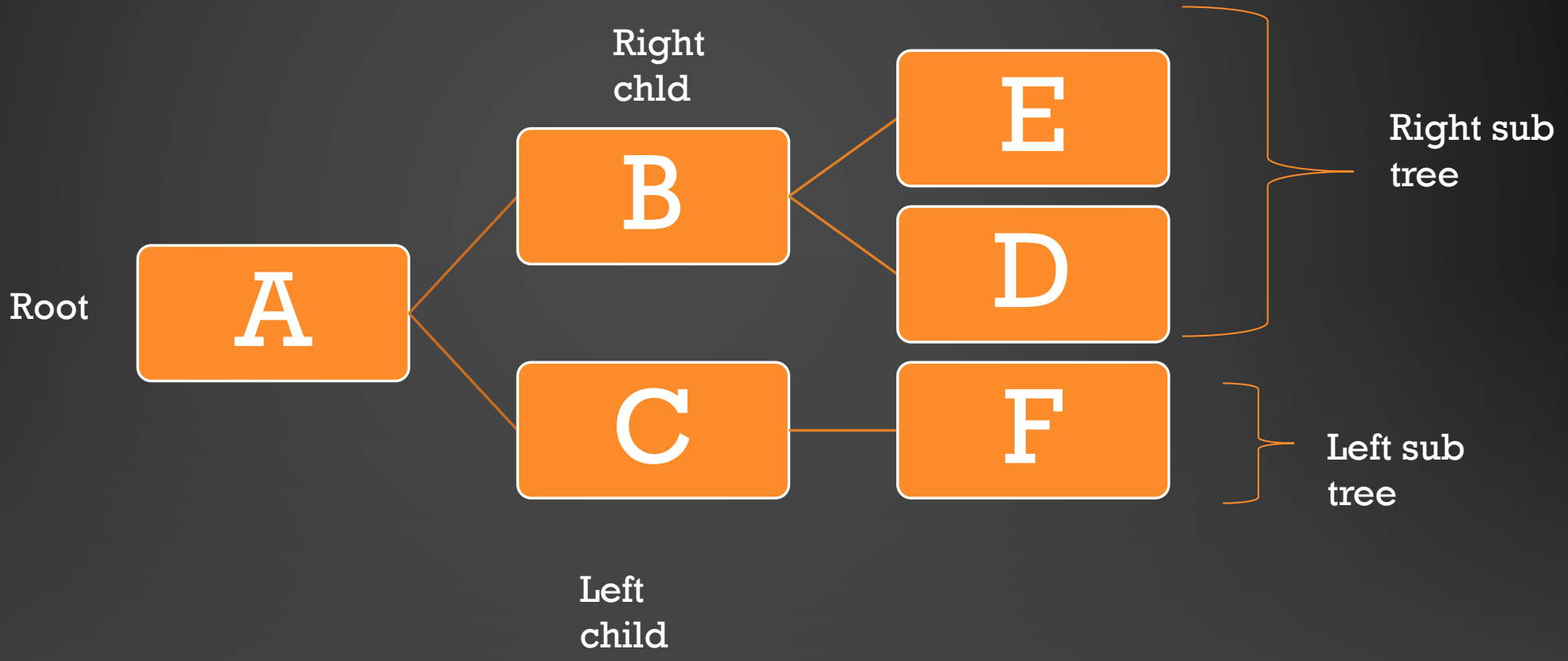
Lecture 15

# WHAT IS BINARY TREE ?

- Binary tree is a finite set of element called nodes.
- The root node have two disjoint set or subtree called as a right sub tree and left subtree.
- A left and right subtree can be empty .
- In binary tree there is no node with degree greater than two.



# EXAMPLE

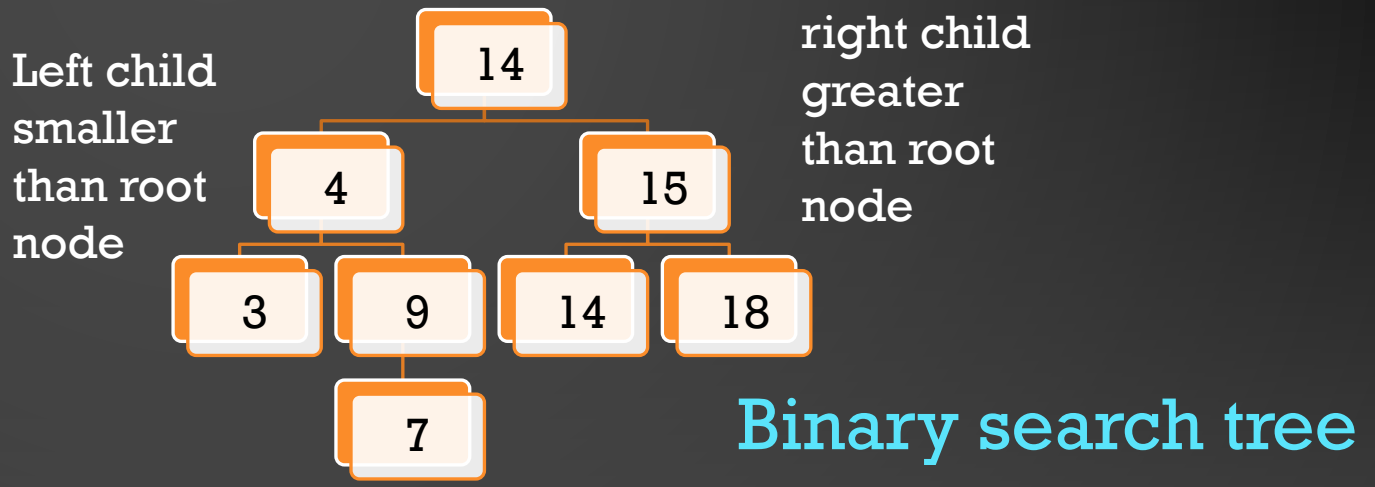

Right
chld
B
E
D
Right sub
tree
Root
A
C
F
Left sub
tree
Left
child

# BINARY TREE PROBLEM AND BST

Lecture 16

# BST ( BINARY SEARCH TREE )

- It is a tree in symmetric order .
- Binary tree is made up of a finite set of elements called nodes .
- It is a binary tree in which each node N of tree has the property that the value at N is greater than every node value in the left subtree of N and is less than or equal to every node value in the right subtree of N .



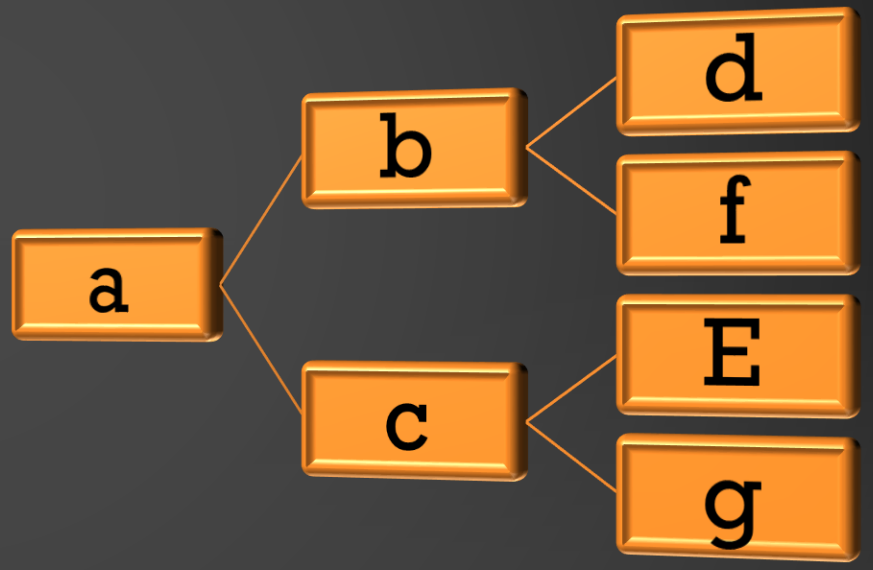
# RELATIONSHIP BETWEEN TOTAL NUMBER OF NODE AND DEPTH OF TREE

DEPTH / HEIGHT : Depth of tree means a maximum level of any node in tree.
the depth = 1+ largest level

Depth = n

Formula :

No of node = $2^n - 1$




The tree with depth n having $2^n - 1$ number of total nodes .

This formula is use for only for binary search tree .

# DRAW A TREE STRUCTURE FOR THE FOLLOWING EXPRESSION

1) E = (a + b) / [(c * d) – e]

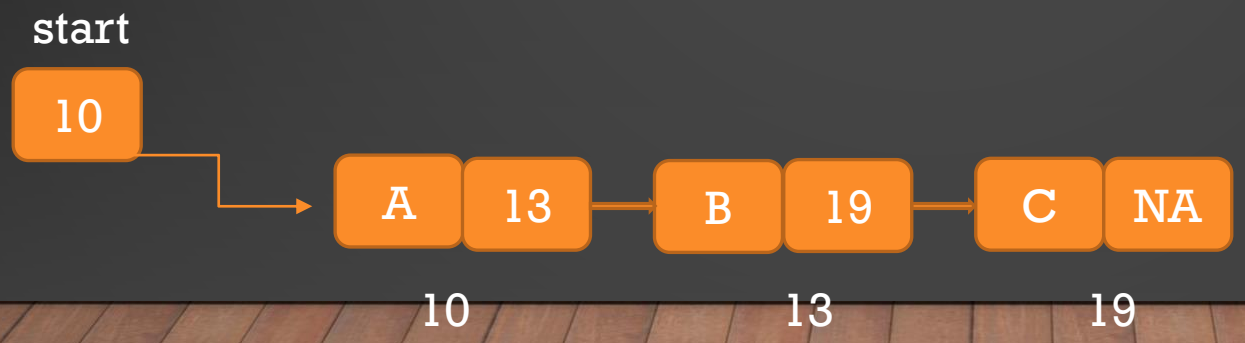
2) (2A + B) (5F - D)

# LINKED LIST

LECTURE 17

# LINKED LIST

- A linked list is a linear collection of data element called nodes where the linear order is maintained with the help of pointer.
- The linked list is also called as one-way-list.
- Each node is divided into two part :
  - 1st part : information part ( contain actual information or data)
  - 2nd part : link part ( link of next pointer field contain address of next node )



# LINKED LIST

- The address of the list is stored in start or name of list.
- the link part of last node is NULL pointer i.e. it contains nothing.
- To trace the linked list we just required the address of start.

# ADVANTAGES OF LINKED LIST

- ✓ To store arrays in memory , require consecutive memory locations , while to store linked list consecutive memory location are not required.
- ✓ Linked list can be easily extended
- ✓ easy to insertion and deletion of element.
- ✓ Easily implemented in computer memory .

# THANK YOU

CS ACADEMY