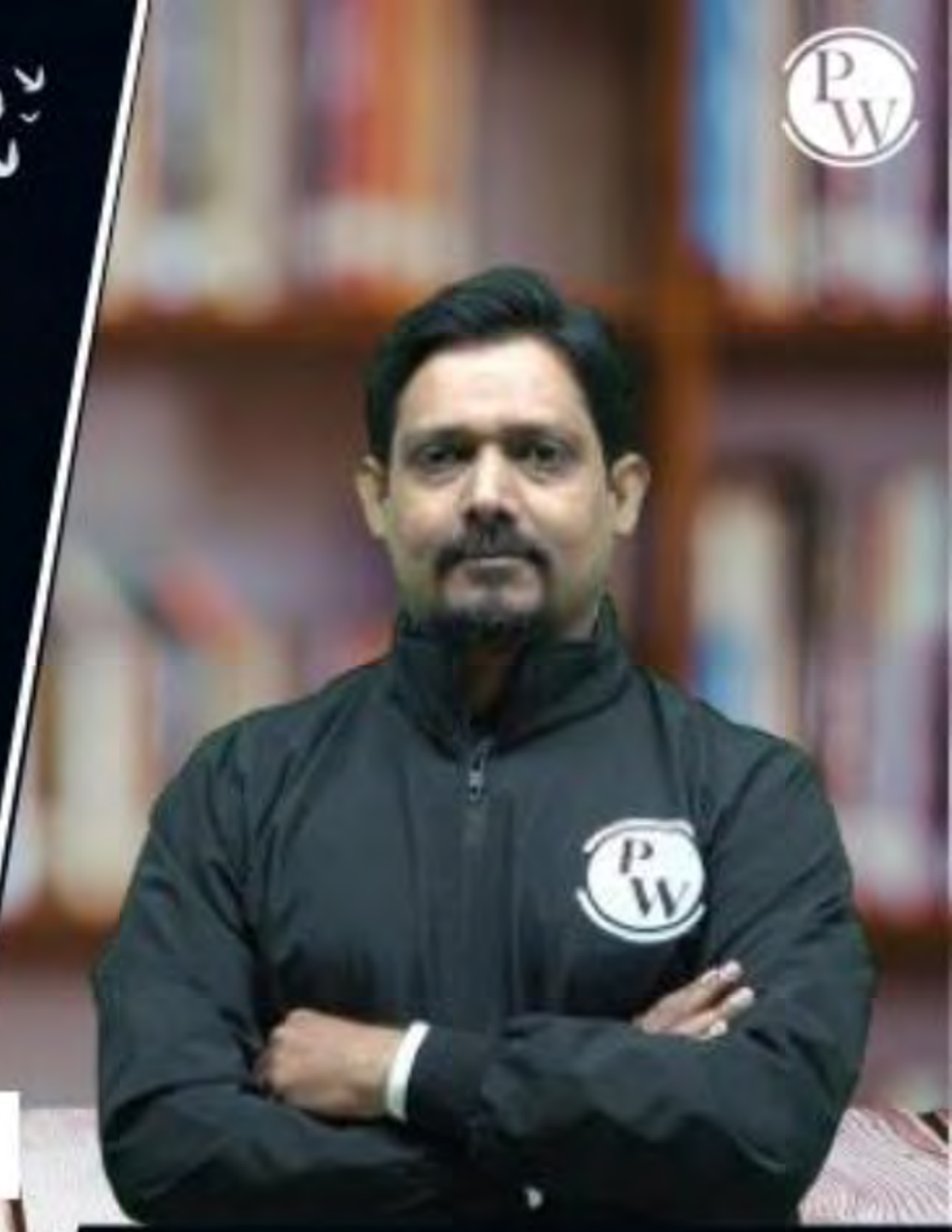
Computer Science & IT

Data Structure & Programming

Array

Lecture No. 01



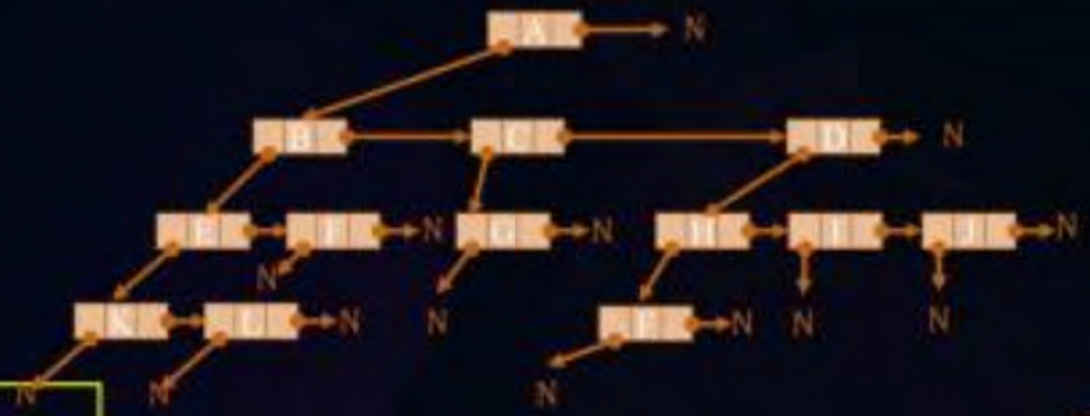
By- Abhishek Sir

Topics to be Covered

- Topic — Introduction
- Topic — Syllabus
- Topic — Marsk distribution
- Topic — Array
- Topic

Slide

Topic : Data Structure



Slide

Topic : Syllabus

C programming

Basics →

Structure
+
pointer

Slide

Topic : Syllabus

Syllabus : programming & Data structure

programming in C ←

Linear Data structure : Array, stack, queue, Linked List

Non Linear Datastructure : Trees, Binary tree, Binary Search Tree, Binary Heaps

Graph

Slide

Topic : Syllabus

Graph - Representation

* Hashing (Algorithm)

Mathematical Aspect DM

Algorithms of Graph

Algorithm

Topic : Syllabus

Linked List ⟶ programming

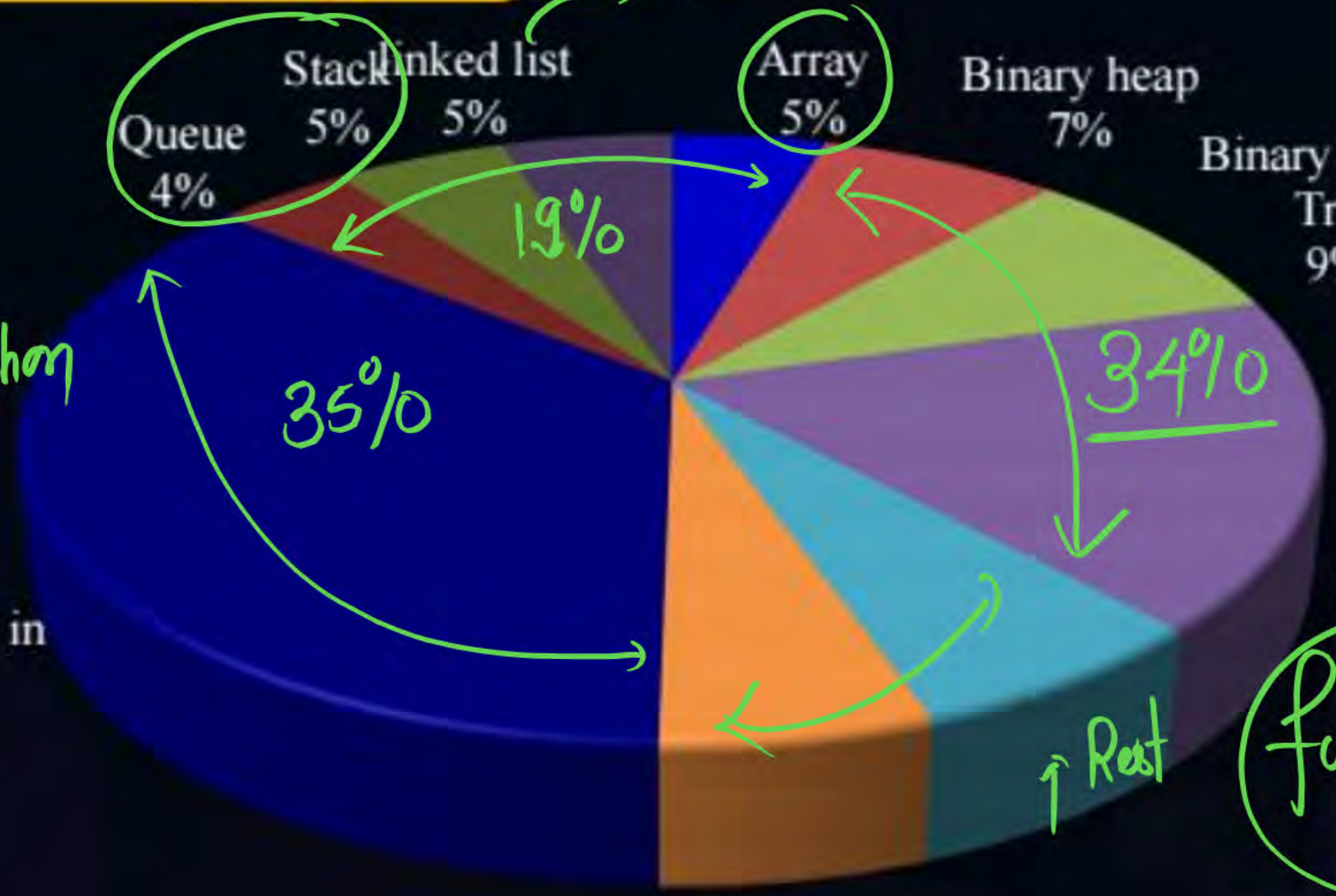
Structure

pointer

while loop

Slide

Topic : Weightage



Code

19%

35%

34%

Rest

PYQ Classification

Mathematical Aspect

fundamental

Slide

Topic : Definition

Data structure :- Data structure is about

* Data organization
* Data Management
* Storage format

that enables "efficient Access" of Data & operation performed on them.

→ Complexity

Slide

Topic : Definition

Linear Data structure

L

* Student Data :- Array Linear arrangment
* Employ Data -:- Linear Arrangment

Linear arrangment :- In memory we can store them in Sequential manner.

Slide

Topic : Definition

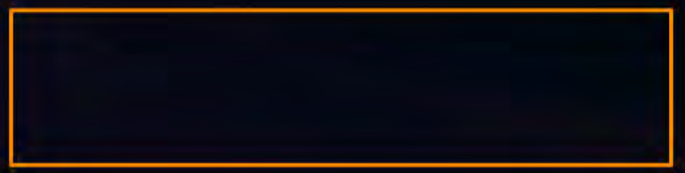
Non Linear :

University

Engg. Arts Commerce Medical

folder structure

folder



Slide

Topic : Array

Linear Data structure : Array, stack, queue, Linked List

Array :- C-programming · Array & pointer

How C-Language Implements array using pointer

Slide

Topic : Array

Array : Array is Sequential Collection of Similar Data type

In Memory elements will be stored one after another

1. Base Address :- Address of first element of array
2. Lower Bound : Index of first element of array
3. Upper Bound : Index of last element of array
4. Size of each element : Each element occupie How many memory location.

Slide

Topic : Array

A [1 ... 10]

Index

Size = 2 Memory Location/2 Byte

A [1]

↑

Index

↑

Base Address

1 is lower bound

10 is upper bound

Slide

Topic : Array

A[1...10]

A[9] — No of element
A[1]...A[8]

1000+8x2 = 1016

1000 + 6x2

1000+12 = 1012

Slide

Topic : Question

1154

#Q. Consider the above array A[1....100]. Base address of the array is 1000 and each element occupies 2 Bytes of space. What is the address of A[78]?

1. Befor A[7]

No. of element arranged

A[1].... A[6]

2 Memory Location

Before A[78] No. of elements arranged

A[1].. A[77] total. 77 elements

Size of each element is 2B

Address of A[78] = 1000 + 77 x 2B = 1154

Slide

Topic : Answer

We need the address of A[78], before reaching A[78] we have already arranged A[1.....77] each occupies 2B of space hence 77× 2 = 154B space , The final address is 1000+154 = 1154.

Topic : Question

C-Language

#Q. Consider the above array A[0...99]. Base address of the array is 1000 and each element occupies 2 Bytes of space. What is the address of A[78].

No. of elements arranged ✓

A[0] .. A[77] = 78

Address of A[78] = 1000 + 78×2

= 1156

int A[100]

A[0...99]

lower bound - 0

upper Bound

A[78] = 1000 + (78-0)×2

1000 + 78×2

1156

Slide

Topic : Question

```
for (i=15; i<=39; i++)
    Stmt;
```

Lower Bound

Upper bound

25 times

No. of times Stmt will execute.

formula : upper bound − lowerbound + 1

39 − 15 + 1 = 24 + 1 = 25

```
for (i=1; i<=10; i++)
    stmt
```

⟶ 10 − 1 + 1 = 10

```
for (i=0; i<=10; i++)
    Stmt;
```

10 − 0 + 1 = 11

Slide

Topic : Question

#Q. Consider the above array A[-5.....100]. Base address of the array is 1000 and each element occupies 2-Bytes of space. What is the address of A[78].

A[78]

A[-5....100] Theoritically

No. of elements arranged : A[-5] .. A[77]

A[-5], A[-4] A[-3] A[-2], A[-1], A[0] A[1] ... A[77]

6+77 = 83

Slide

Topic : Question

#Q. Consider the above array A[-5.....100]. Base address of the array is 1000 and each element occupies 2-Bytes of space. What is the address of A[78].

$1000 + (78-(-5)) \times 2$

$1000 + 83 \times 2 = 1166$

A[78]

A[-5....100]

Theoritically

No. of elements arranged : A[-5] .. A[77]

upper bound - lower bound + 1 = 77 - (-5) + 1

= 83

Address of A[78] = 1000 + 83x2 = 1000 + 166

1166

Slide

Topic : Solution

We need the address of A[78], before reaching A[78] we have already arranged A[-5 …..77] each occupies 2B of space hence 77-(-5) +1 = 83 elements already arranged. With each element occupy 2B total space is 83 × 2 = 166B. The final address is 1000+166 = 1166.

Topic : Number of element in array

A[LB... UB] No. of element in array is

UB - LB + 1

A[-7...92] ⟶ 92 - (-7) + 1 = 100

Slide

Topic :Generalized formula

$A[LB \ldots UB]$

BA is Base Address

S is Size

Address of $A[i]$

No. of elements arrange

$A[LB] \ldots A[i-1]$

total = $i - LB + 1 = (i - LB)$

Address of $A[i]$ =

$$BA + (i - LB) \times Size$$

Slide

2- Dimensional Array

Topic : 2 D Array

$A[1...3][1...3]$

Row

Column

	$Column_1$	$Column_2$	$Column_3$
$Row\ 1 \rightarrow$	$A[1][1]$	$A[1][2]$	$A[1][3]$
$Row\ 2 \rightarrow$	$A[2][1]$	$A[2][2]$	$A[2][3]$
$Row\ 3 \rightarrow$	$A[3][1]$	$A[3][2]$	$A[3][3]$

When 2-D arranged in memory then its mapped into 1-D array by following way

1. Row major order (C - Language)
2. Column major order

Slide

Topic : 2 D Array

A[1...3][1...3]

Row

Column

Column$_1$ Column$_2$ Column$_3$

Row 1 →	A[1][1] 100	A[1][2] 102	A[1][3] 104
Row 2 →	A[2][1] 106	A[2][2] 108	A[2][3] 110
Row 3 →	A[3][1] 112	A[3][2] 114	A[3][3] 116

Row major order

Consecutive Rows will be arranged one after another

Row-1 Row 2 Row-3

A_{11}	A_{12}	A_{13}	A_{21}	A_{22}	A_{23}	A_{31}	A_{32}	A_{33}
100	102	104	106	108	110	112	114	116

Slide

Topic : 2 D Array

A[1..3][1..3] Address of A[3][2]. BA: 100 Size is 2B

A[3][2]. Before 3rd Row No. of Rows arranged : 2 Rows

(3 - 1) = 2 Rows

No. of elements in each Row = 3.

total elements = 3 x 2 = 6

In third Row : before second column (2 - 1) columns arranged

1

total = 6 + 1 = 7

Address of A[3][2] = 100 + 7 x 2 = 114

Slide

Topic : 2 D Array

No. of elements in a Row decided by Size of column

No. of elements in a column decided by Size of Row

Slide

Topic : Question

#Q. Consider the following array A[1…100][1…100].Base address is 1000 and each element occupies 2-Bytes of space. What is the address of A[50][49] in row major order?

A [1..100] [1..100]

A [50] [49]

Before 50th Row No. of Rows arranged

(50 − 1) = 49

Each Row consists of 100 elements

total = 49 × 100 = 4900 ✓

Slide

Topic : Question

#Q. Consider the following array A[1…100][1…100].Base address is 1000 and each element occupies 2-Bytes of space. What is the address of A[50][49] in row major order?

A[1..100][1..100]

A[50][49]

In 50th Row we are in 49th column

No. of columns completed 49-1 = 48 ✓

Total = 4900+48 = 4948

Address of A[50][49] = 1000 + 4948x2

1000 + 9896 = 10896

Slide

Topic : Question

We need to find the address of A[50][49]. Before reaching to 50th row 1…49 rows are already been arranged and each row consists of 100 elements. So total 49× 100 =4900 elements are arranged.

For the 50th row we have already arranged 1…48 i.e. 48 elements (look at the second dimension) are already arranged.

So total number of elements arranged are = 4900+48 = 4948.

The address is 1000+4948× 2=10896

2 mins Summary

Topic

1-D array

Topic

1-D array generalized formula

Topic

2D array

Topic

Row major order

Topic

Slide

THANK - YOU

Slide