SUBJECT CODE: CS301

SUBJECT NAME: DATA STRUCTURES AND ALGORITHMS

UNIT - 3

SEMESTER: 3

Introduction to Data Structures

Data Structure:

A way of organizing and storing data so it can be accessed and modified efficiently.

🔷 Types of Data Structures

a) Primitive

- int, float, char, pointer

b) Non-Primitive

- Linear → Array, Stack, Queue, Linked List
- Non-Linear → Tree, Graph

🔷 Classification

- Static Data Structure → size fixed (Array)
- Dynamic Data Structure → size changes (Linked List)

🔷 Why Data Structures?

- Efficient memory usage
- Faster searching & sorting
- Better program organization

Algorithms

Algorithm: A finite set of instructions to solve a problem.

🔷 Characteristics

- Input
- Output
- Definiteness
- Finiteness
- Effectiveness

🔷 Example (Simple Algorithm)

Steps to find largest number:

1. Start
2. Read A, B
3. If A > B print A
4. Else print B
5. Stop

Analysis of Algorithms

Used to measure efficiency of an algorithm.

🔹 Types of Analysis

- Best Case
- Average Case
- Worst Case

Time Complexity

Measures execution time based on input size **n**.

🔹 Common Time Complexities

- O(1) → Constant
- O(log n) → Logarithmic
- O(n) → Linear
- O(n log n)
- $O(n^2)$ → Quadratic
- $O(2^n)$ → Exponential

👉 Example:

```
for(i=0;i<n;i++)
    print(i);
```

Time Complexity = **O(n)**

Space Complexity

Memory used by an algorithm.

Includes:

- Fixed part (variables, constants)
- Variable part (dynamic memory)

Asymptotic Notations

Used to represent algorithm growth.

🔷 Big-O (O)

Upper bound / Worst case
Example: $O(n^2)$

🔷 Omega (Ω)

Lower bound / Best case

🔷 Theta (Θ)

Average case / Tight bound

Recursion (Basic Concept)

Function calling itself to solve smaller problems.

🔷 Parts of Recursion

- Base Case → stopping condition
- Recursive Case → function calls itself

Example:

```
factorial(n):
if n==1 return 1
else return n * factorial(n-1)
```

Abstract Data Type (ADT)

Logical description of a data structure.

Example:

- Stack ADT → push(), pop(), peek()

Focus:

- What operations are performed