* What is data structure?

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

* One-dimensional array declaration in C++

A linear collection of elements stored in contiguous memory. Example: int arr[5];

* Multidimensional array declaration in C++

An array of arrays, used for tables or matrices. Example: int arr[3][4];

* Linked lists over arrays

A linked list uses nodes connected by pointers, unlike arrays which use continuous memory. Useful for dynamic memory management.

* Multiply linked list

A linked list where each node contains multiple pointers (e.g., doubly linked list has next and prev).

* Time complexity index

A measure of how an algorithm’s running time grows as input size increases (e.g., O(1), O(n), O(log n)).

* Time complexity

Describes the amount of time an algorithm takes to complete based on input size.

* Data structure is most efficient for implementing a queue linked list

A linked list is efficient because elements can be enqueued/dequeued without shifting data.

* Algorithm

A step-by-step procedure to solve a problem or perform a task.

* Types of algorithm

Examples: Divide and Conquer, Greedy, Dynamic Programming, Brute Force, Backtracking.

* Algorithm Development Life Cycle

Steps: Problem definition → Algorithm design → Implementation → Testing → Analysis → Maintenance.

* List of analysis method

Ways to evaluate algorithms: Worst-case, Best-case, Average-case, Amortized analysis.

* Empirical or Posteriori Analysis

Running the algorithm with actual input and measuring its performance.

* Space complexity

Amount of memory an algorithm needs to run.

* Numbering System (Decimal, Octal, Binary, Hexadecimal)
  + Decimal → Base 10
  + Octal → Base 8
  + Binary → Base 2
  + Hexadecimal → Base 16
* Binary Coded Decimal (BCD)

Represents decimal digits (0–9) in binary form (4 bits each).

* Real number

A number with fractional/decimal part (e.g., 3.14, -2.5).

* Character string

A sequence of characters stored as an array (e.g., "Hello").

* Escape sequence represents a new line in a string

Special symbols in strings, e.g., \n for newline, \t for tab.

* Immutable strings

Strings that cannot be changed after creation (like in Java, but in C++ strings are mutable).

* Bitwise operations

Operations on binary digits: AND (&), OR (|), XOR (^), NOT (~), Shift (<<, >>).

* C++ string function (length)

Returns the number of characters in a string. Example: str.length();

* Floating Number (Sign bit, Exponent, Mantissa, Base)

Representation of real numbers in memory using scientific notation.

* C++ Library ()

Predefined header files that provide functions (e.g., <iostream>, <cmath>, <string>).

* Last-In-First-Out (LIFO)

A principle where the last inserted item is the first removed (used in Stack).

* Primitive, Hybrid, Non-Primitive, Abstract
  + Primitive → Basic types (int, float, char).
  + Non-Primitive → Derived (arrays, strings, structures).
  + Hybrid → Combination (trees, graphs).
  + Abstract → Defined by behavior (stacks, queues).
* Abstract Data Types (ADTs)

A data type defined by its operations, not implementation (e.g., Stack, Queue).

* Stack

A LIFO structure where elements are added/removed from the top.

* Tree

A hierarchical structure with a root and child nodes.

* Array

Fixed-size collection of elements of the same type.

* Static data structure

Size is fixed at compile time (e.g., arrays).

* Boolean, Character, Integer, Double

Fundamental data types in C++:

* + Boolean → true/false
  + Character → single letters/symbols
  + Integer → whole numbers
  + Double → floating-point numbers (decimal).
* Hierarchical data

Data organized in a tree-like structure (e.g., file system, family tree).