“DSA: The overpowered cheat code behind every coding superhero.”

Ever wondered how your phone keeps a list of your favorite contacts or how websites quickly find your Saved preferences? The magic starts with how data is organized using arrays, lists and HashMap.

DATA STRUCTURE AND ALGORITHM

1.Introduction to data structure

-> Data structure is a way to store and organize data so that it can be used efficiently.

-> The data structure is not any programming language like C, C++, Java etc. It is a set of algorithms that we can use in any programming language to structure data in memory.

-> Two types of data structures: -

A. Linear data structure: - The arrangement of data in Sequential manners is known as linear data structure. The data structure used for this purpose is: - Arrays, Linked list, stacks and queues.

-> In this data structure, one element is connected to only one another element in a linear form.

B. Non – Linear data structure: - When one element is connected to the ‘n’ number of elements known as non – linear data structures. In this case, elements are arranged in a random manner. Example: - Trees and Graphs.

2. Array

A. 1D Array: -

-> Arrays are defined as collections of similar types of data items stored at contiguous memory locations.

-> Array is the simplest data structure where each data element can be randomly accessed by using its index number.

-> Array declaration: -

Int arr[10] ; char arr[10]; float arr[10];

B. 2D Array: -

-> 2D arrays can be defined as an array of arrays. The 2D array is organized as matrices which can be represented as a collection of rows and columns.

-> Declaration of 2D Array

Ex – int arr[max_rows][max_coloumns];

-> The size of the 2D array is equal to the multiplication of the number of rows and number of columns present in the array.

Application of array in real world

-> Array stores large database for (e.g., temperature readings, Stock prices).

-> Arrays are used in games development for (e.g., A 2D array in a chess game for board layout.

-> Arrays are used to store characters in a document.

-> 2D Array is used for representing an image in pixels.

3.Searching and sorting

A. Searching: -

-> Searching is the process of finding a particular element in a data structure (like an array, list, tree, etc.).

Types of searching algorithm: -

a. Linear search:

-> Linear search is the simplest sequential search algorithm and often called sequential search. In this type of searching, we simply traverse the list completely and match each element of the list with the item where the location is to be found.

-> Linear search is mostly used to search for an unordered list In which items are not sorted.

b. Binary search:

-> Binary search is a search technique which works an Efficiently on sorted lists. Here to search for an element into some list by using binary search technique, we must ensure that list is sorted.

-> Binary search follows divide and conquer approach in which, list is divided into two halves and item

-> Binary search follows a divide and conquer approach in which, list is divided into two halves and the item is compared with the middle element of the list.

B. Sorting: -

-> Sorting is the process of arranging elements in an order typically ascending or descending.

Types of sorting algorithm: -

a. Bubble sort: -

-> Swaps adjacent elements if they are in the wrong order.

-> Time complexity: O(n^2).

b. Selection sort: -

-> Repeatedly select the smallest (or largest) element and move it to the correct position.

->Time complexity: O(n^2).

c. Insertion sort: -

-> Build the sorted array one item at a time by Inserting elements into their correct position.

-> Time complexity: O(n^2).

d. Merge sort: -

-> Divide-and-conquer algorithm; divides arrays into halves, sort them, and merge them.

-> Time complexity: O (n log n).

Application for searching and sorting: -

-> Binary search is used to look up a word in an electronic dictionary or translation app, it usually keeps its entries in its sorted array or trees so it can rapidly locate your words.

-> Searching algorithms are also used in scrolling through your contacts list on a phone whether you tap “J” to jump to “john”, or you type in the search bar relies on indexed data structures and fast search algorithms.

4. Linked lists

-> A linked list is a collection of the nodes in which one node is connected to another node and nodes consist of two parts, one is the data part and another one is the address part..

Types of Linked list: -

a. Singly linked list: - The singly linked list is the most common list which consists of data parts and address parts. The address part in the node is called a pointer.

b. Doubly linked list: - As name suggests, the doubly linked list contains two pointers. We define it in three parts: the data part and the two address parts.

c. Circular linked list: - A circular linked list is a variation of a singly linked list the only difference is the last node point to any node in a singly linked list.

d. Doubly circular linked list: - The last node is attached to the first node and thus creates a circle. The main difference is that the doubly circular linked list does not contain NULL value in the previous field of the node.

Application of linked list

-> Using a linked list, when we are playing songs in a playlist, each song points to the next one.

-> Linked lists manage the web browser history, clicking “back” or “forward” takes you through your browsing history.

-> Linked lists are used in undo/redo functionality, used in Applications like MS word or photoshop.

5. Stack

-> A stack is a linear data structure that follows LIFO (Last In First Out) principle having one end.

-> A stack is a container in which insertion and deletion can be done from the end(one) known as the top of the stack.

Operation on the stack: -

Push (): - When we insert an element in a stack then the operation is known as push. If the stack is full overflow conditions occur.

Pop (): - When we delete an element from the stack, the operation is called pop (). If a stack is empty means no element exists in the stack; this state is known as an overflow state.

Top (): - It returns the element at the top of the stack.

Application of stack

-> when functions are called, they are added to the stack. When a function returns, it is removed from the stack.

-> Every web page is pushed onto a stack. Pressing “back”pops the last visited page.

-> Used in compilers to check for balanced parentheses using a stack.

6. Queues

-> Queues can be defined as an ordered list which enables Insert operation to be performed at end and delete operation to be performed at another end called front.

-> Queues can be referred to as being first in the first out list. Operation on queues:

-> Enqueue: - Enqueue is used to insert elements at the rear end of the queue. It returns void.

-> Dequeue: - Dequeue operation performs the deletion from the front end of the queue. The dequeue operation can also be designed to be void.

Application of queues

-> Banking and finance: - Ticket number system in banks/post Offices. ATM lines.

-> cultural and Entertainment Events: - Ticket lines for Concerts and cinemas.

-> Food supply and Aid Distribution: - Food bank or ration distribution lines. water collection points in areas lacking plumbing.

7. Binary Trees

-> A tree is a data structure defined as a collection of objects or entities known as nodes that are linked together to represent the data.

-> A tree is a non-linear data structure because it does not store in a sequential manner. It is a hierarchical structure as elements in tree are arranged at multiple levels.

-> In trees data structures the topmost node is called a root Node.

Types of Traversals in binary Tree: -

Pre-order (NLR): -

N -> Fetch the root data.

L -> Recursive call in left part of root.

R-> Recursive call in right part of root.

Pre-order (LNR): -

L -> Recursive call in left part of root.

N-> Fetch the root data.

R-> Recursive call in right part of root.

Post-order (LRN): -

L -> Recursive call in left part of root.

R-> Recursive call in right part of root.

N-> Fetch the root data.

Level-order: - Visit nodes level by level from the root outward, typically using a queue.

Morris's traversal: - It is a way to traverse a binary tree without using a recursion or a stack, and with O (1) extra space. It temporarily modifies the tree structure by creating "threads" to remember where it came from-then later restores the original tree.

Conclusion: -

Data structure and algorithms form the backbone of efficient programming. From arrays and linked lists to binary trees. Each topic plays a critical role in solving complex computational problems. A solid understanding of these topics not only helps in building optimal solutions but also prepares for real-world scenarios and technical interviews.