

DSA stands for Data Structures and Algorithms. Data structures manage how data is stored and accessed. Algorithms focus on processing this data. Examples of data structures are Array, Linked List, Tree and Heap, and examples of algorithms are Binary Search, Quick Sort and Merge Sort.

Foundation for almost every software like GPS, Search Engines, AI ChatBots, Gaming Apps, Databases, Web Applications, etc

Top Companies like Google, Microsoft, Amazon, Apple, Meta and many other heavily focus on DSA in interviews. Learning DSA boosts your problem-solving abilities and make you a stronger programmer.

Before beginning the DSA journey, it is recommended to learn at-least one programming language (C++ , Java , Python , JavaScript or any other language of your choice).

Below are the recommended step by step topics to learn complete DSA.

Once you have learned basics of a programming language, it is recommended that you learn basic logic building To analyze algorithms, we mainly measure order of growth of time or space taken in terms of input size. We do this in the worst case scenario in most of the cases.

Array is a linear data structure where elements are allocated contiguous memory, allowing for constant-time access.

Searching algorithms are used to locate specific data within a large set of data. It helps find a target value within the data.

Sorting algorithms are used to arrange the elements of a list in a specific order, such as numerical or alphabetical.

Hashing is a technique that generates a fixed-size output (hash value) from an input of variable size using mathematical formulas called hash functions. Hashing is commonly used in data structures for efficient searching, insertion and deletion.

In Two Pointer Technique, we typically use two index variables from two corners of an array. We use the two pointer technique for searching a required point or value in an array.

In Window Sliding Technique, we use the result of previous subarray to quickly compute the result of current.

In Prefix Sum Technique, we compute prefix sums of an array to quickly find results for a subarray.

A sequence of characters, typically immutable and have limited set of elements (lower case or all English alphabets).

A programming technique where a function calls itself within its own definition. It is usually used to solve problems that can be broken down into smaller instances of the same problem.

A two-dimensional array of elements, arranged in rows and columns. It is represented as a rectangular grid, with each element at the intersection of a row and column.

A linear data structure that stores data in nodes, which are connected by pointers. Unlike arrays, nodes of linked lists are not stored in contiguous memory locations and can only be accessed sequentially.

A linear data structure that follows the Last In, First Out (LIFO) principle. Stacks are used for managing function calls, in algorithms like stock span problem, next greater element and largest area in a histogram.

Queue is a linear data structure that follows the First In, First Out (FIFO) principle. Queues play an important role in managing tasks or data in order, scheduling and message handling systems.

A Deque or double-ended queue is a data structure that allows elements to be added or removed from both ends efficiently.

A non-linear, hierarchical data structure consisting of nodes connected by edges, with a top node called the root and nodes having child nodes. It is widely used in file systems, databases, decision-making algorithms, etc.

A complete binary tree that satisfies the heap property. Heaps are usually used to implement priority queues , where the smallest or largest element is always at the root of the tree.

A non-linear data structure consisting of a finite set of vertices(or nodes) and a set of edges(or links)that connect a pair of nodes. Graphs are widely used to represent relationships between entities.

Greedy Algorithm builds up the solution one piece at a time and chooses the next piece which gives the most obvious and immediate benefit i.e., which is the most optimal choice at that moment.

Dynamic Programming is a method used to solve complex problems by breaking them down into simpler subproblems. By solving each subproblem only once and storing the results, it avoids redundant computations, leading to more efficient solutions.

Advanced Data Structures like Trie, Segment Tree, Red-Black Tree and Binary Indexed Tree offer significant performance improvements for specific problem domains.

Bitwise Algorithms: Operate on individual bits of numbers.

Backtracking Algorithm : Follow Recursion with the option to revert and traces back if the solution from current point is not feasible.

Divide and conquer: A strategy to solve problems by dividing them into smaller subproblems, solving those subproblems, and combining the solutions to obtain the final solution.

Branch and Bound : Used in combinatorial optimization problems to systematically search for the best solution. It works by dividing the problem into smaller subproblems, or branches, and then eliminating certain branches based on bounds on the optimal solution.

Geometric algorithms are a set of algorithms that solve problems related to shapes, points, lines and polygons.

Randomized algorithms are algorithms that use randomness to solve problems. They make use of random input to achieve their goals, often leading to simpler and more efficient solutions.