

**Your Ultimate Guide To Landing  
Top AI roles**



2.2

## Introduction to DSA



→ DSA → Data Structures and Algorithms

→ DSA involves using data structures (list, stacks, trees etc) to store and access data and applying algorithms (step by step procedures) to solve problem efficiently.

→ Course structure

- ① Learn about Time and Space Complexity
- ② Data Structures in detail + Implementation in Python
- ③ Algorithms in details + Implementation in Python
- ④ Problem Practice in Python

Problem: Search for a number  $x$

```
def linear_search(nums, x):
    for val in nums:
        if val == x:
            return True

    return False
```

→ Data Structure → List

Algorithm → Linear Search

```
nums = [5, 15, 10, 6]
x = 12
is_found = linear_search(
    nums, x)
if is_found == True:
    print("number exist")
else:
    print("no. don't exists")
```

→ what is a better program?

↳ A program which takes less memory and less time

→ How to find how much **space** and **time** a program takes?

↑  
Space Complexity

↑  
Time Complexity.

→ A better program takes

① Less memory in RAM (Space complexity ↓)

② Less time for execution (Time complexity ↓)

①  $P_1 \rightarrow 10 \text{ ms}, 20 \text{ KB}$

$P_2 \rightarrow 10 \text{ ms}, 10 \text{ KB}$

②  $P_1 \rightarrow 10 \text{ ms}, 50 \text{ KB}$

$P_2 \rightarrow 15 \text{ ms}, 50 \text{ KB}$

# Data structures

\* Python



→ A data structure is a way of storing and organizing data so that it can be used efficiently.

## \* Important data structures

- |                         |                      |
|-------------------------|----------------------|
| ① Array                 | ⑧ Tree               |
| ② Linked lists          | ⑨ Graph              |
| ③ Strings               | ⑩ Trie (prefix tree) |
| ④ stack                 | ⑪ Fenwick Tree (BIT) |
| ⑤ Queue                 | ⑫ Segment Tree       |
| ⑥ Hash Table (Hashing)  | ⑬ Disjoint Sets      |
| ⑦ Priority Queue (Heap) |                      |

# Algorithms



→ An algorithm is a step-by-step set of instructions used to solve a problem or perform a task.

## \* Important Algorithms

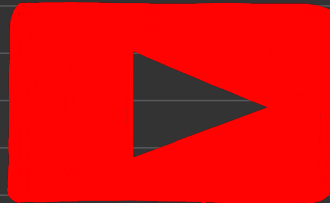
- ① Searching Algorithms
- ② Sorting Algorithms
- ③ Greedy Algorithms
- ④ Divide and Conquer Algorithms
- ⑤ Dynamic Programming Algorithms
- ⑥ Recursion & Backtracking
- ⑦ BFS/DFS Traversal
- ⑧ Minimum Spanning Tree
- ⑨ Single source shortest path
- ⑩ All pair shortest path
- ⑪ Bridges / Articulation Point
- ⑫ string pattern matching

## \* Algorithm Vs Program



- Algorithm : A step-by-step procedure or set of rules to solve a problem.
- Program : A set of instructions written in a programming language.
- First we decide the algorithm. Then we write the Program.
- Computer can only execute the program. It can't execute the algorithms.

Like



Subscribe