**Data Structures and Algorithm**

* **Data Structure** is a way to organize & store data efficiently.

Performance

**Efficiently**

Memory

**Primitive Data Types**

* Number 🡪 int
* Text 🡪 String
* Character 🡪 char
* Point values 🡪 float
* True/False 🡪 Boolean
* char – single character
* String – consists of multiple chars
* An **Algorithm** is a set of instructions for completing a task or solving a problem.

Data

Object

Name/Model No

Brand

Configuration



int A = 5;

int B = 7;

int num[] = {5, 7, 12, 16}

int C = 12;

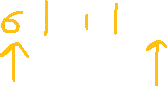


int D = 16;

**Store bunch of values**

List, Set, Queue, Stack

**Queue 🡪 first in first out**



**Stack 🡪 Last in first out**



**Array**



101 102 103 104 105

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **5** | **6** | **17** | **12** | **9** |

nums

Index value

0 1 2 3 4



nums[3] = 12

Time consuming



**Complexity**



Time Complexity

Space Complexity

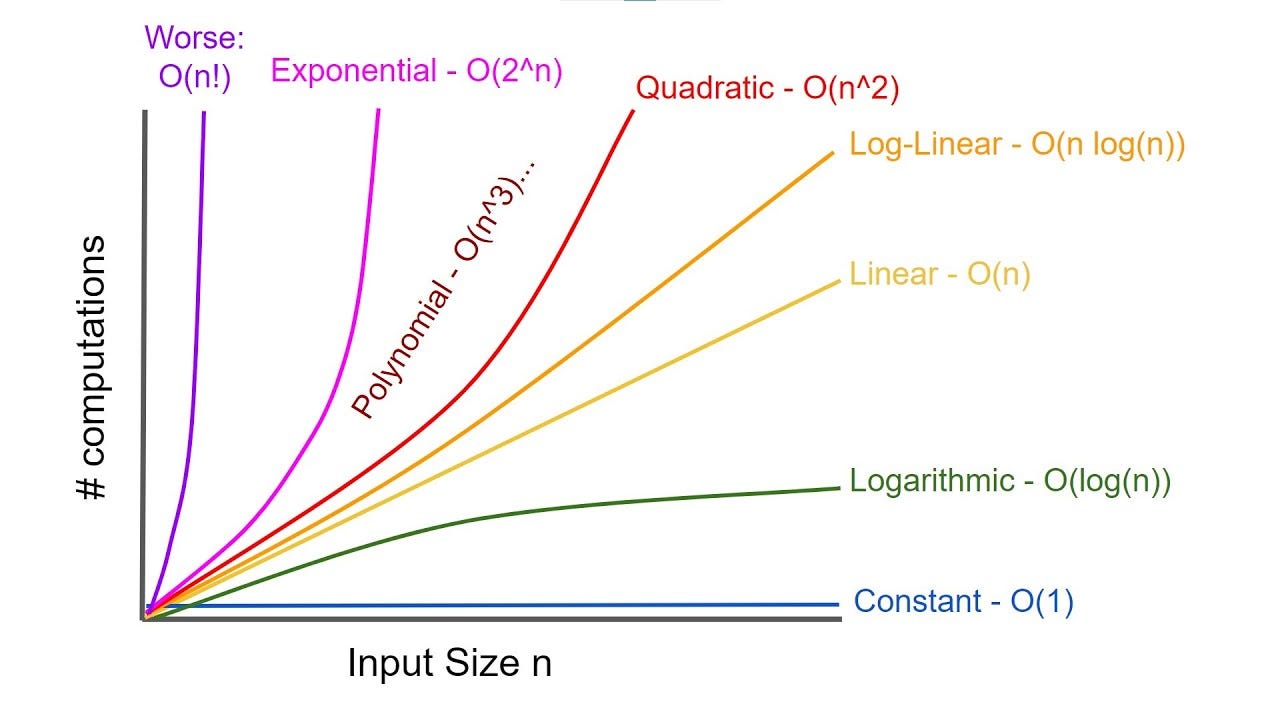
**Time Complexity**

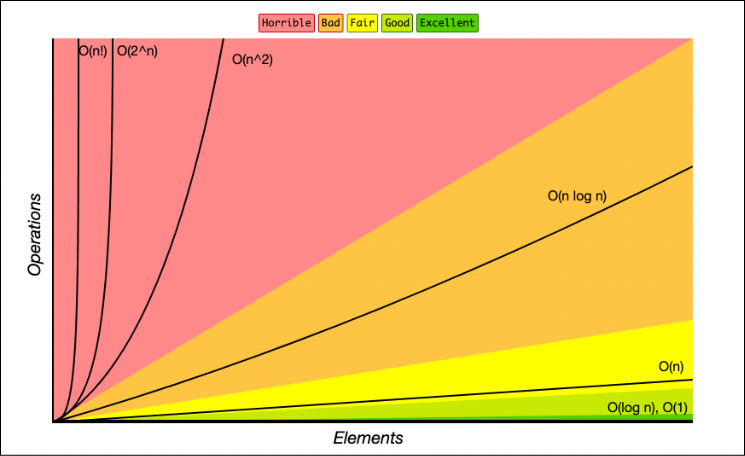
Measure of how the running time of an algorithm increases with the size of the input data.

**Big O Notation**

* O(1) 🡪 Constant time
* O(log n) 🡪 Logarithmic time
* O(n) 🡪 Linear time
* O(n log n) 🡪 Linearithmic time
* O(n^2) 🡪 Quadratic time
* O(2^n) 🡪 Exponential time
* O(n!) 🡪 Factorial time

**Big – O Complexity Chart**





Searching an element in a sorted array

Binary Search >>>>>> Linear Search

Linear Search 🡪 O(n)

Binary Search 🡪 O(log n)