What is Algorithm Analysis ?

Simple Definition –

- Algorithm analysis is a study to provides theoretical estimation for the required resources of an algorithm to solve a specific computational problem. i.e. calculating efficiency.

Generally, the efficiency an algorithm is related to the input length (number of steps), known as **time complexity**, or volume of memory, known as **space complexity**.

Why do we need Algorithm Analysis ?

- *Knowing efficiency of an algorithm is very vital on mission critical tasks.*
- *Generally there are **multiple** approaches/method/algorithms to solve one problem statement. Algorithm analysis is performed to figure out which is the better/optimum approaches/method/algorithms out of the options.*

What does a BETTER Algorithm mean ?

- **Faster ? (Less execution time) – Time Complexity**
- **Less Memory ? – Space Complexity**
- **Easy to read ?**
- **Less Line of Code ?**
- **Less Hw/Sw needs ?**

**Note: Algorithm Analysis does not give you accurate/exact values(time, space etc), however it gives estimates which can be used to study the behavior of the algorithm.*

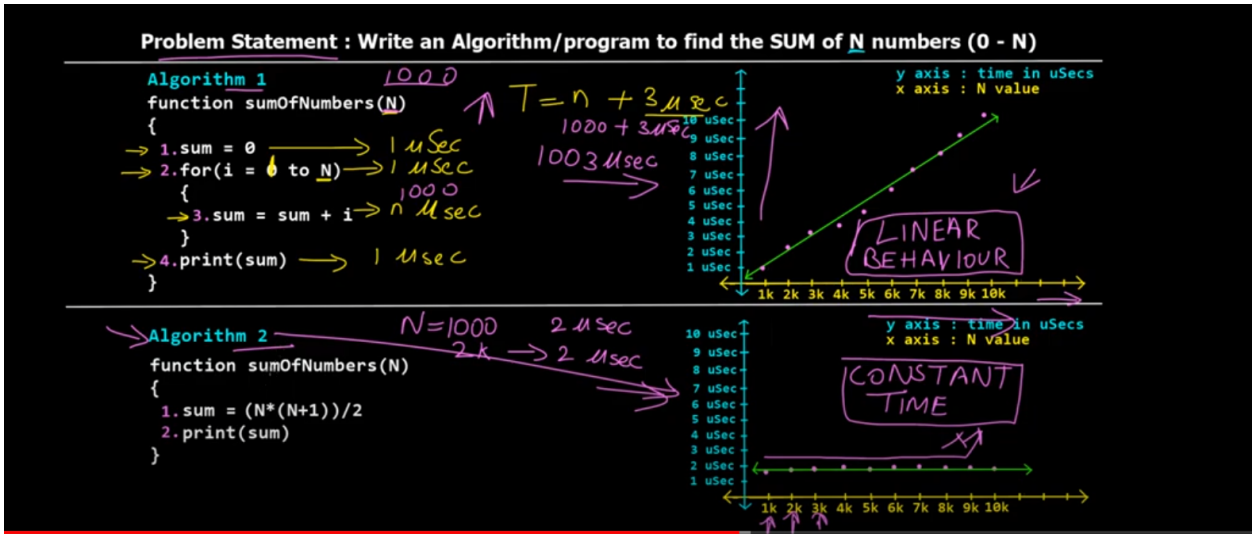
What is Asymptotic Algorithm Analysis ?

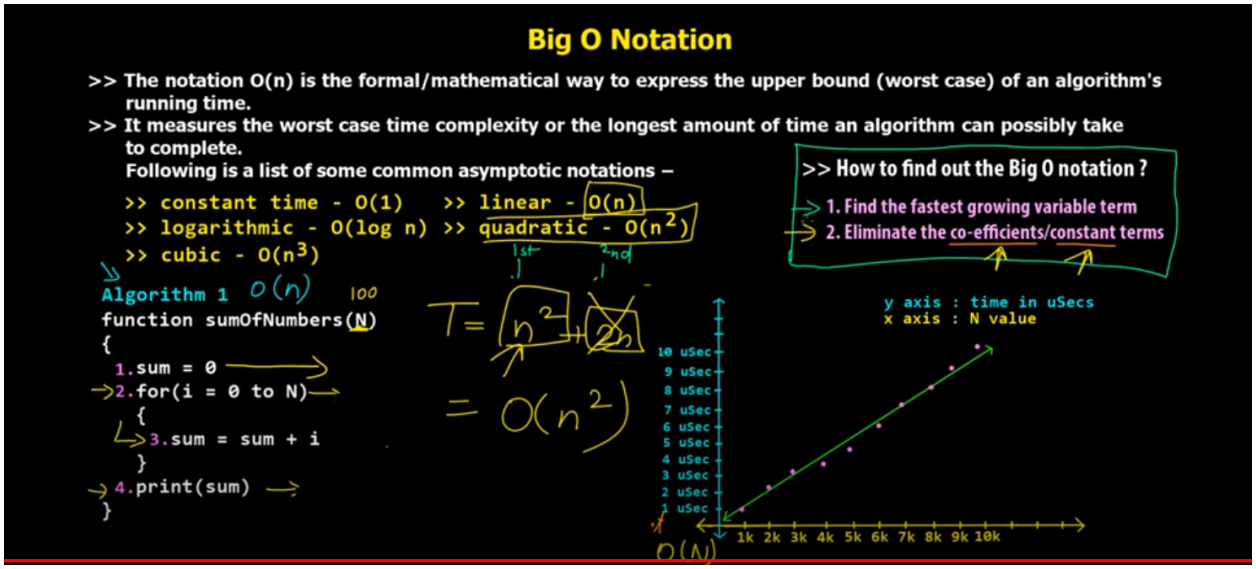
- Definition: *In mathematical analysis, asymptotic analysis of algorithm is a method of defining the mathematical boundaries of its **run-time** performance.*
- Using the asymptotic analysis, we can easily estimate about the average case, best case and worst case scenario of an algorithm.

Simple words: It is used to mathematically calculate the running time of any operation inside an algorithm.

Asymptotic Algorithm analysis is to estimate the time complexity function for arbitrarily large input.

*Time Complexity : is a computational way to show **how(behavior)** runtime of a program increases as the size of its input increases.*

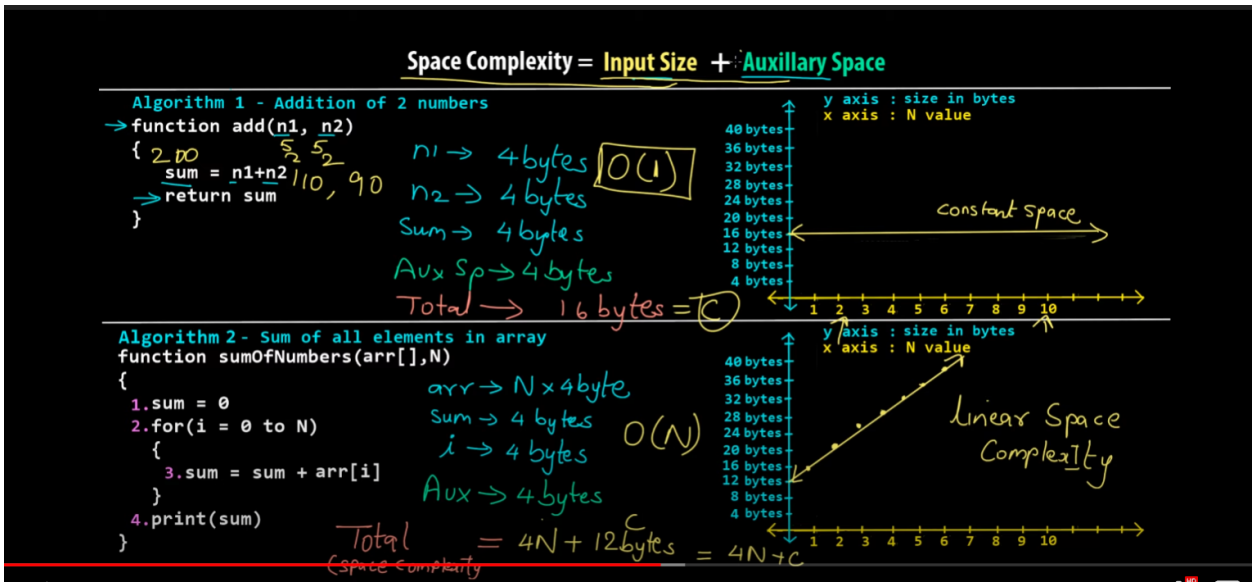




Space Complexity

What is Space complexity?

- Definition: *The space complexity of an algorithm or a computer program is the amount of memory space required to solve an instance of the computational problem as a function of the size of the input.*
- Simple words : It is the memory required by an algorithm to execute a program and produce output.
- Similar to time complexity, Space complexity is often expressed asymptotically in big O notation, such as O(n), O(nlog(n)), O(n^2) etc., where n is the input size in units of bits needed to represent the input.



→ Space Complexity = Input Size + Auxillary Space

Algorithm 3 - Factorial of a number(iterative)

```
int fact = 1;
for (int i = 1; i <= n; ++i)
{
      fact *= i;
}
return fact;
```

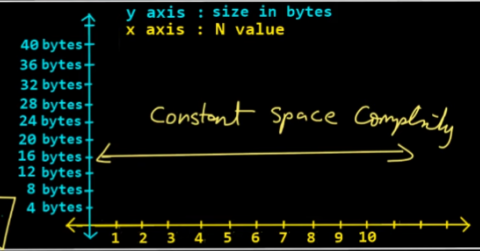
n=5

fact → 4 bytes
n → 4 byte
i → 4 byte.

O(1)

Aux → 4 bytes

S = 12 bytes + 4 byte = 16 byte



Algorithm 4 - Factorial of a number(recursive)

```
factorial2(n)
{
    if(n<=1)
    {
       return 1;
    }
    else
    {
       return (n*factorial2(n-1));
    }
}
```

n → 4byte
Aux → 4byte
= 8byte × 5 = 40byte

fn(5)
120

1 byte
1 2 × f(1)
2 3 × 2
3 4 × 6
4 5 × 24

S = 4byte + (n × byte)