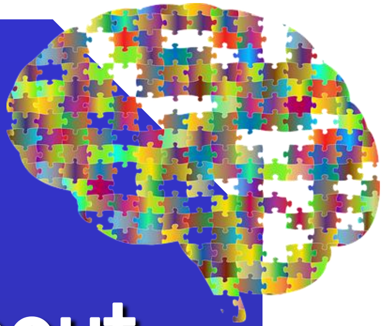
Topic

Algorithmic Time Complexity



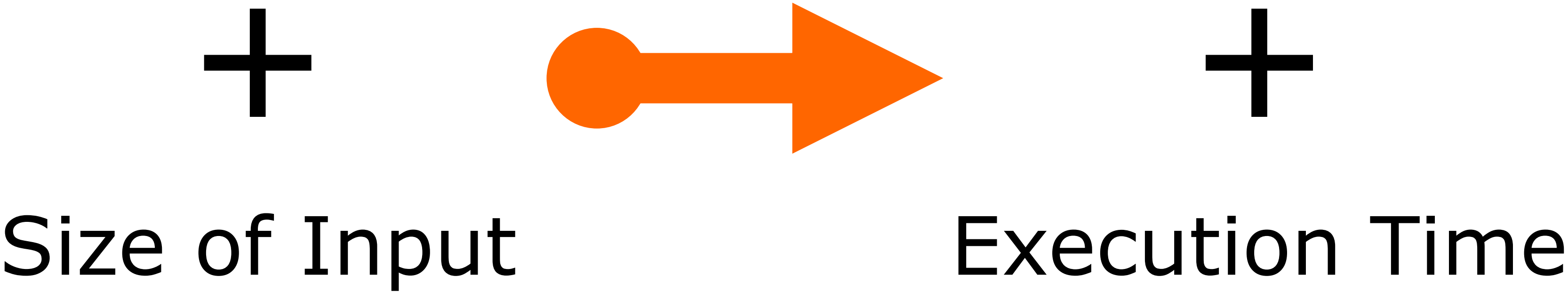
Let’ s think about…
Algorithms

Efficiency

Fast

<u>Size</u> of Input

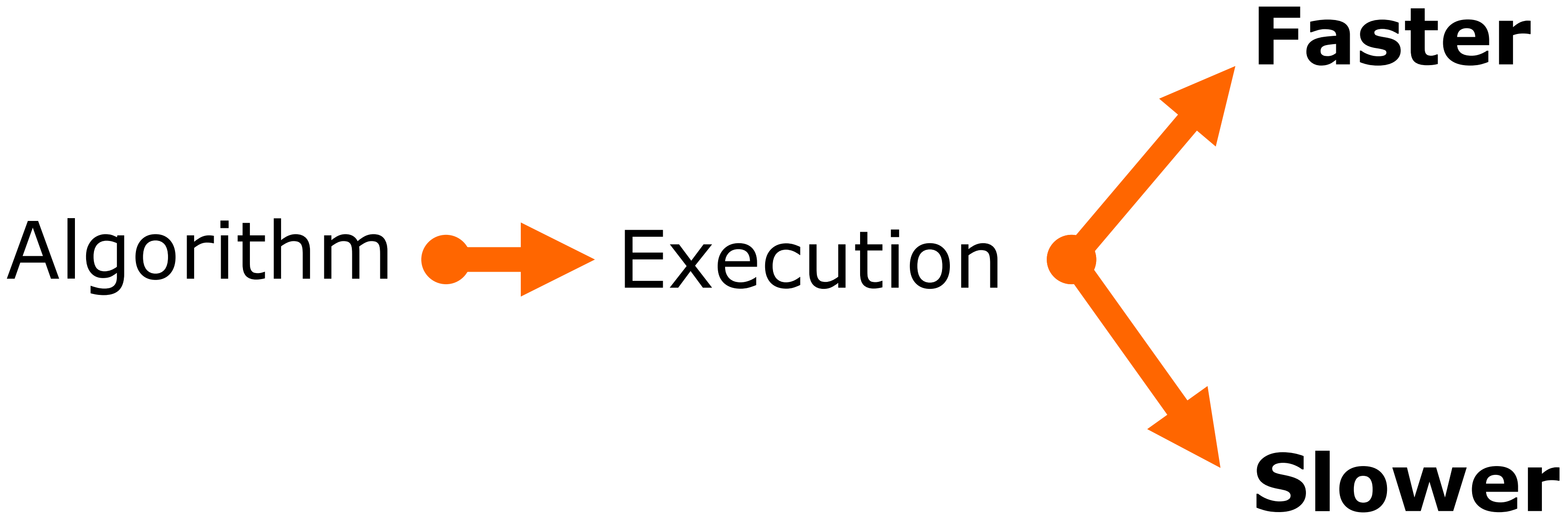
Time Complexity 101



+ Size of Input

+ Execution Time

Time Complexity 101



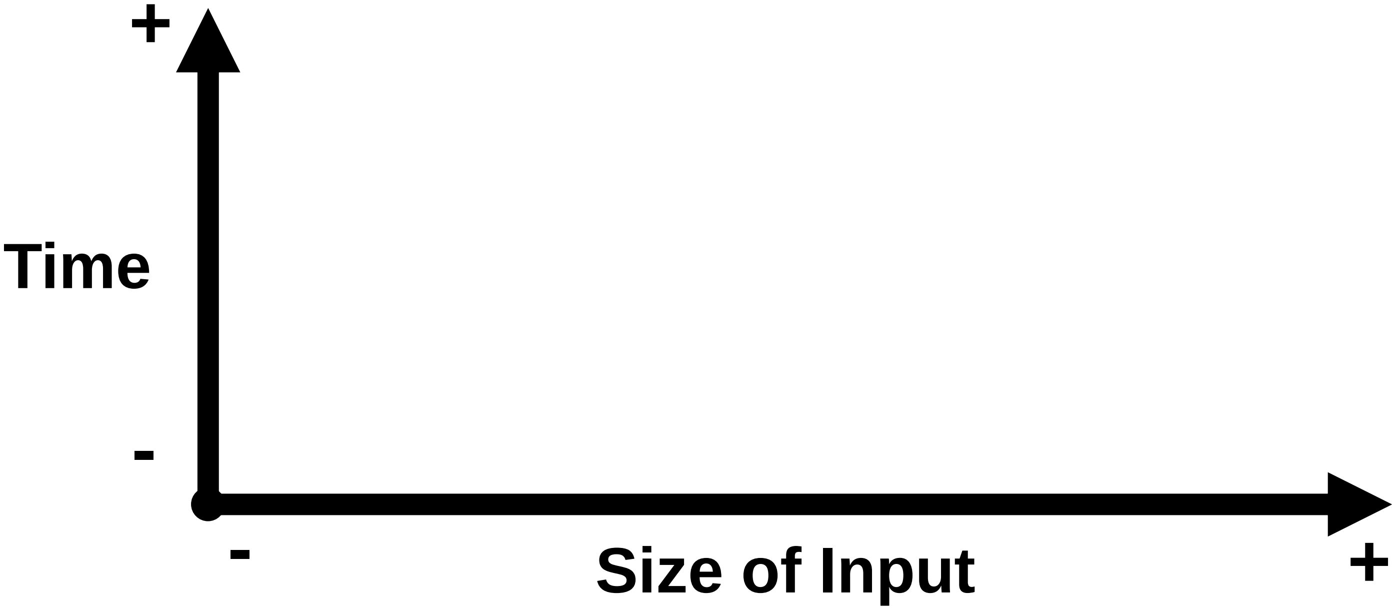
Algorithm

Execution

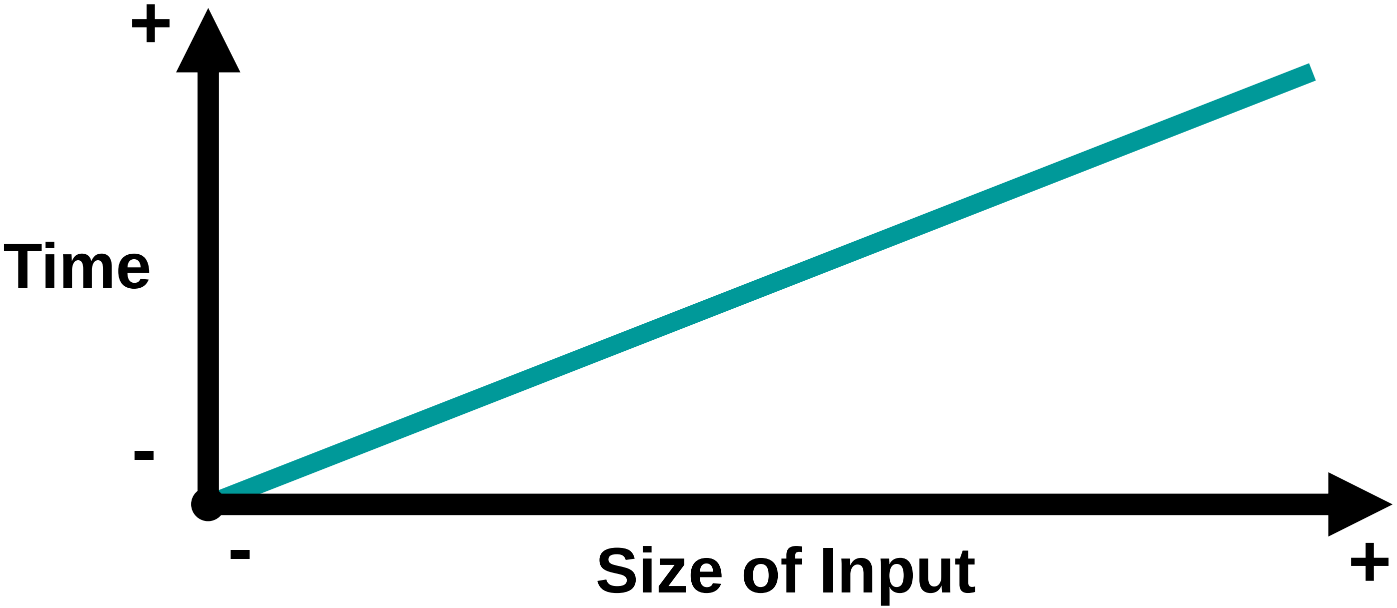
Faster

Slower

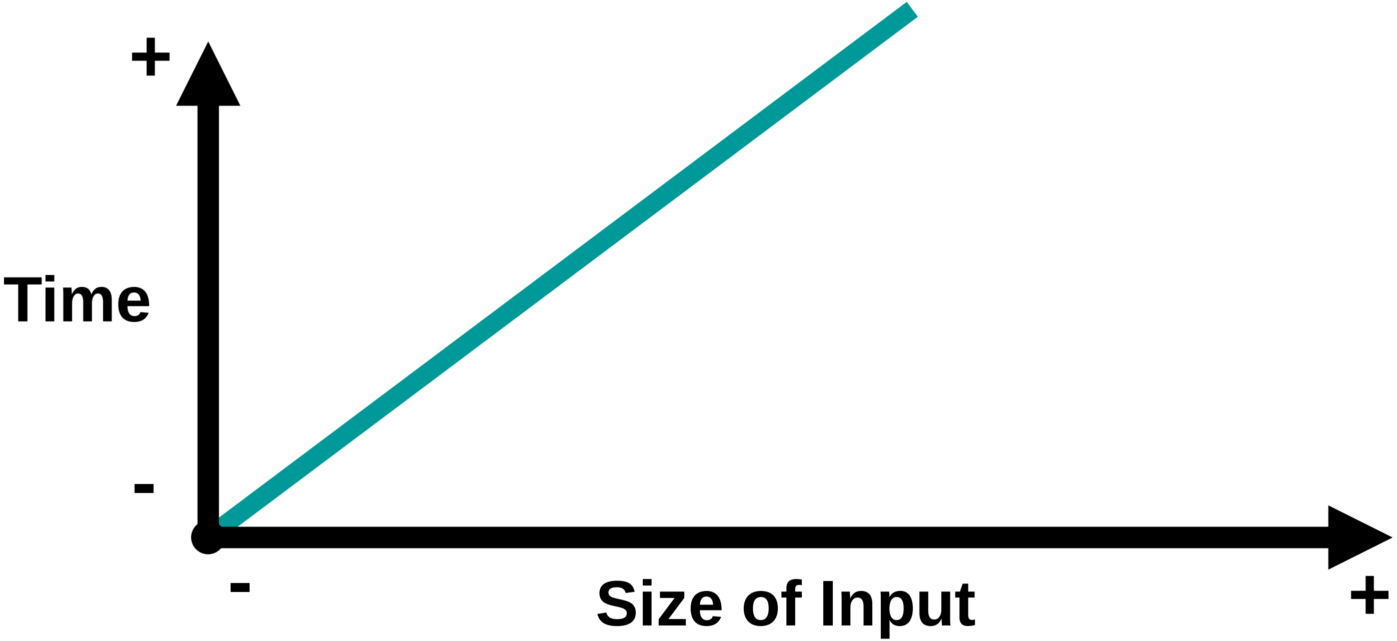
Time Complexity 101



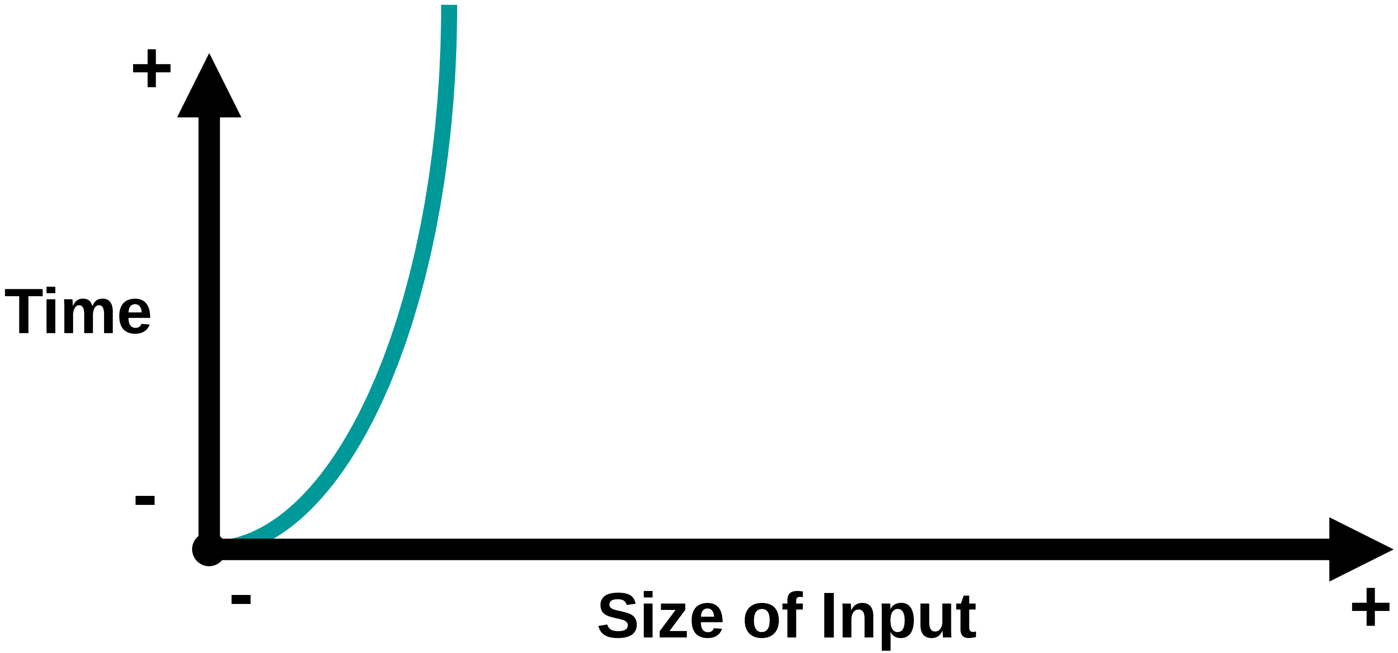
Time Complexity 101



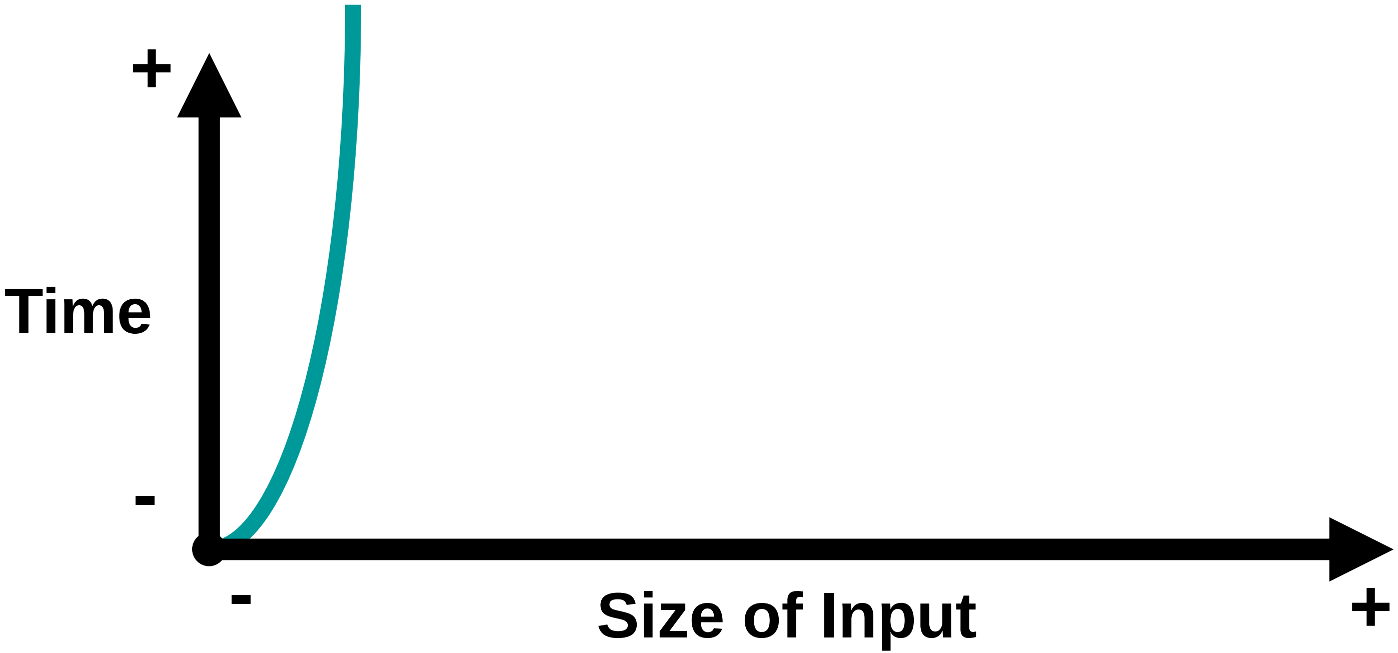
Time Complexity 101



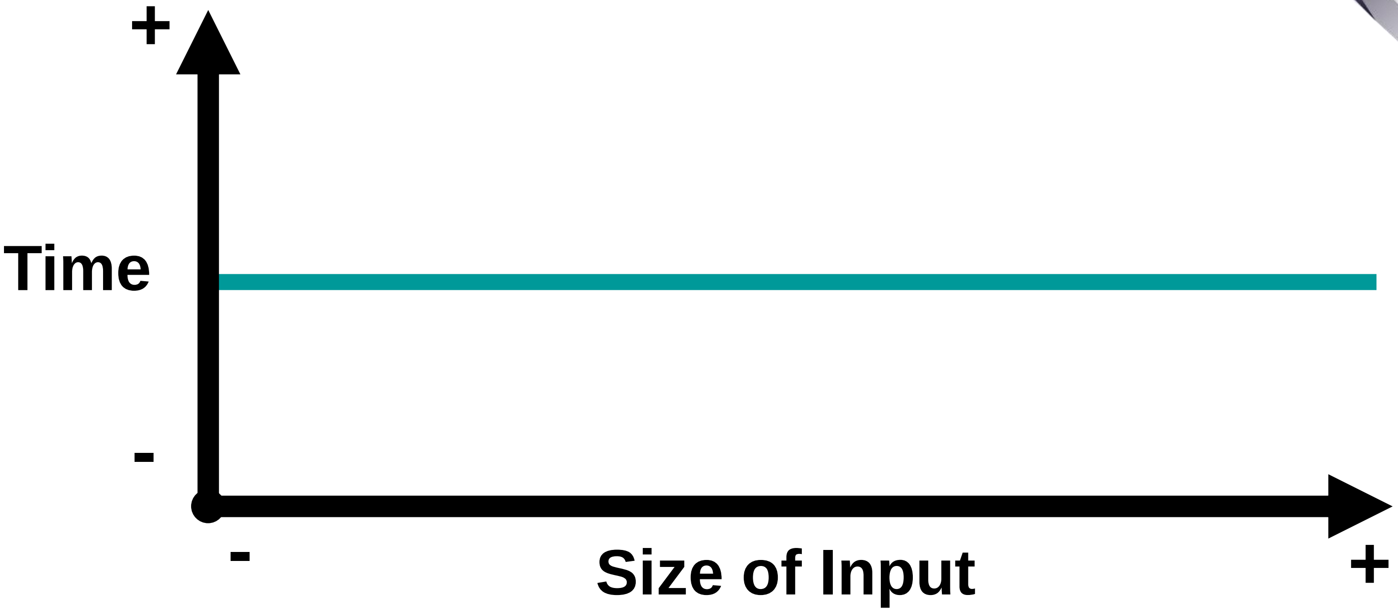
Time Complexity 101



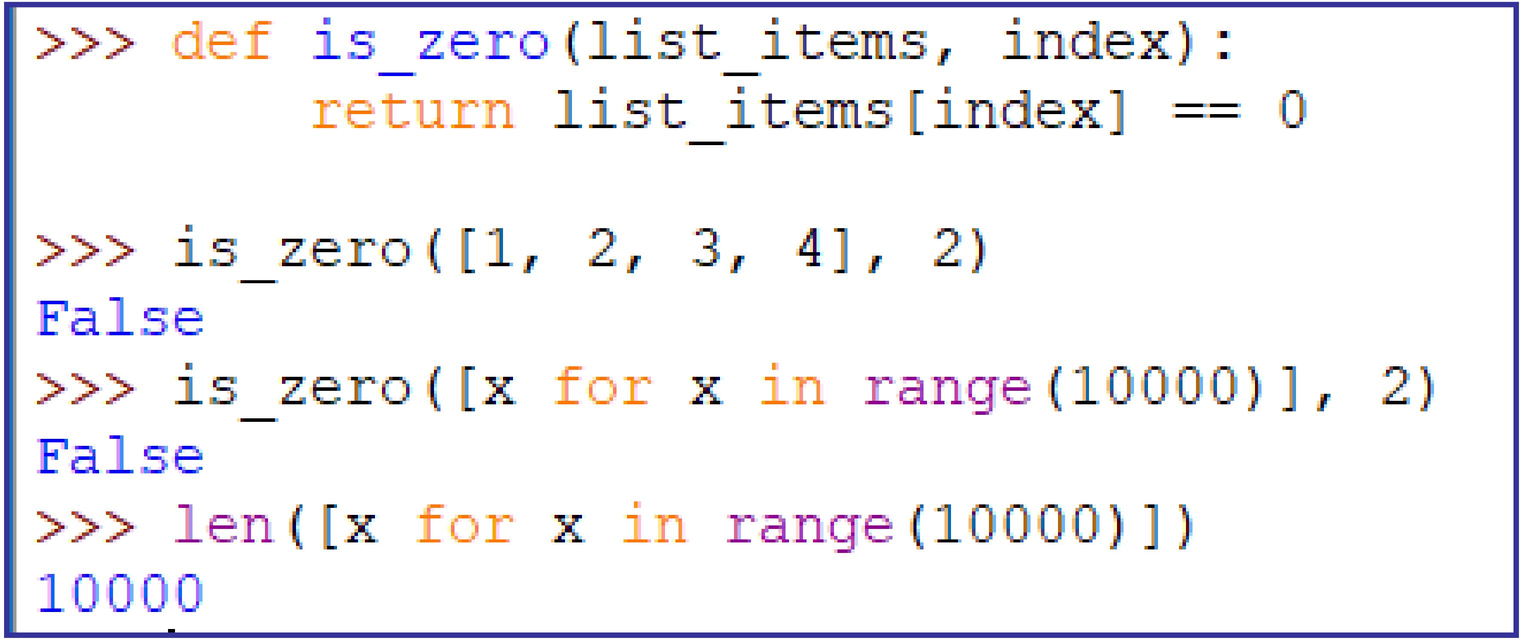
Time Complexity 101



Time Complexity 101



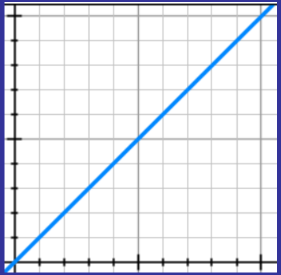
Time Complexity 101



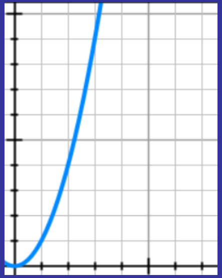
```
>>> def is_zero(list_items, index):
        return list_items[index] == 0

>>> is_zero([1, 2, 3, 4], 2)
False
>>> is_zero([x for x in range(10000)], 2)
False
>>> len([x for x in range(10000)])
10000
```

Time Complexity 101

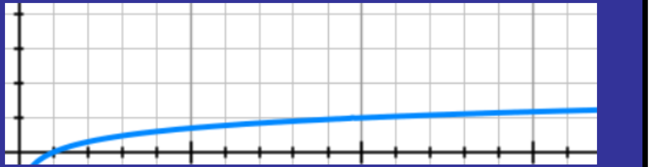
Orders of Growth



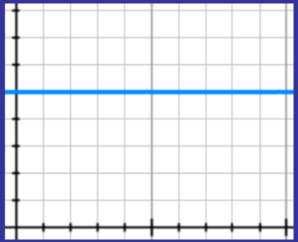
Linear



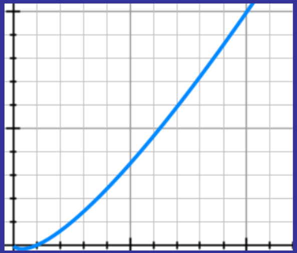
Quadratic



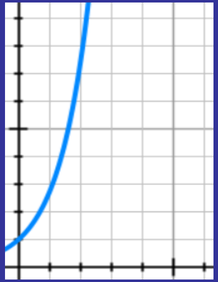
Logarithmic



Constant



nlog(n)



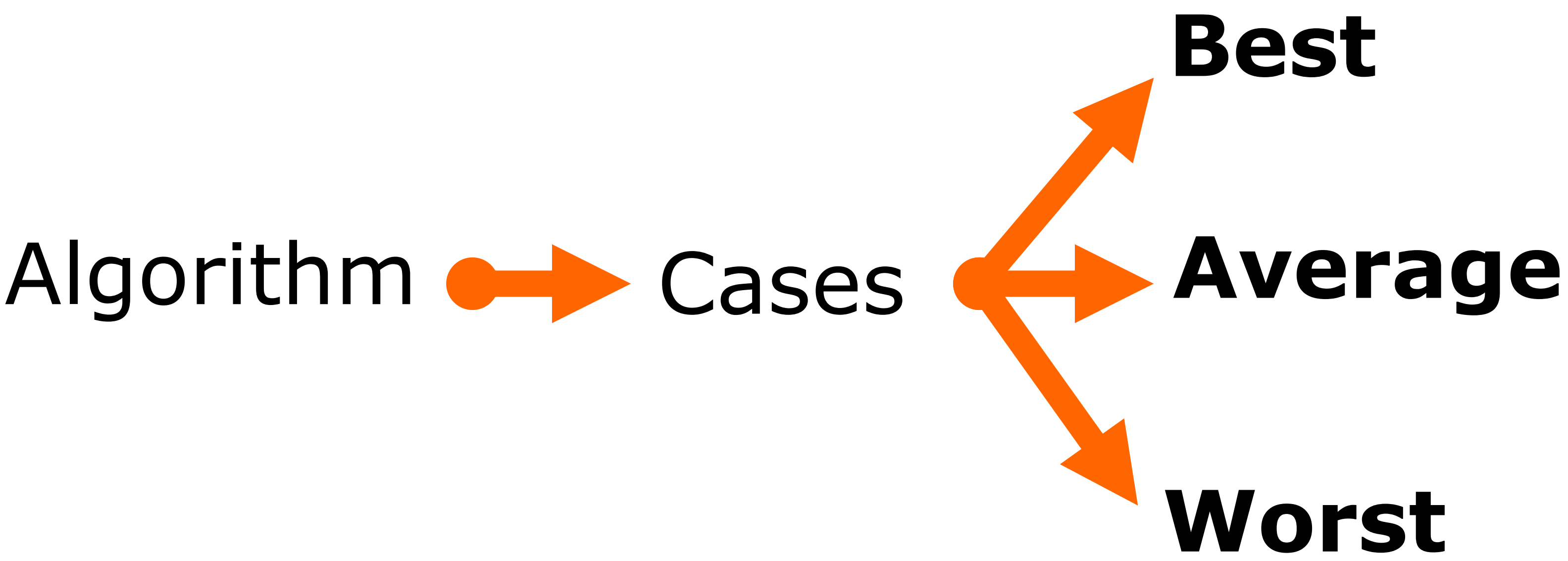
Exponential

Time Complexity 101

?

Big O!

Time Complexity 101



Algorithm

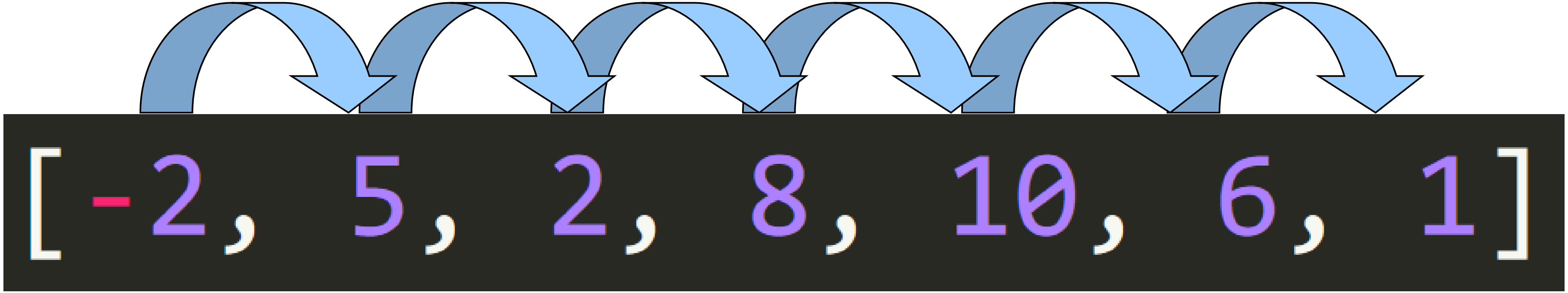
Cases

Best

Average

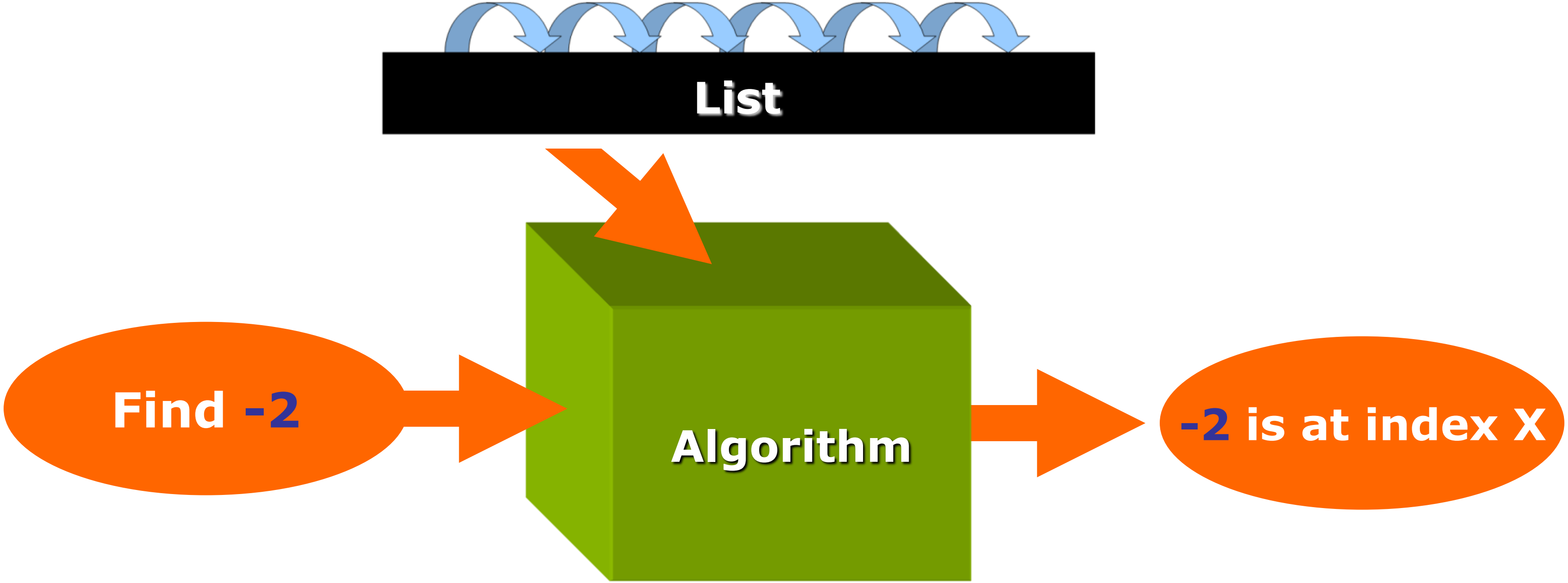
Worst

Time Complexity 101 – Best Case



```
[-2, 5, 2, 8, 10, 6, 1]
```

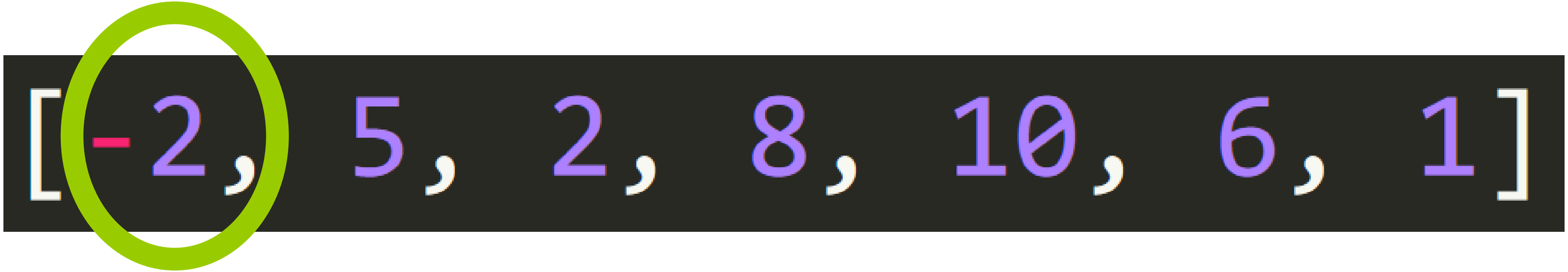
Time Complexity 101 – Best Case



Let's think about…

Best Case

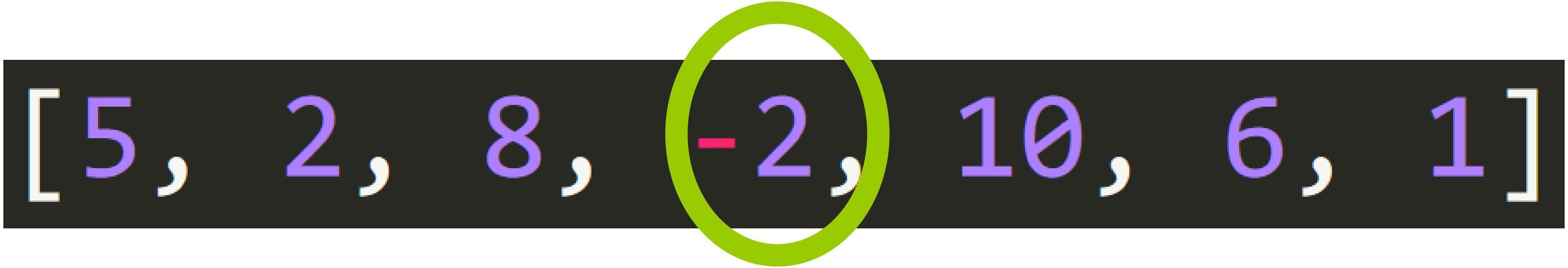
Time Complexity 101 – Best Case



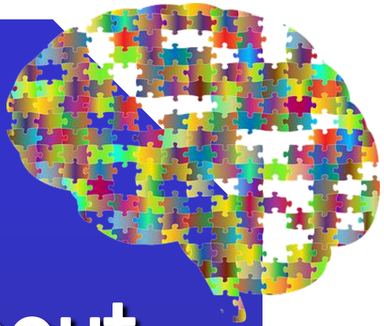
```
[-2, 5, 2, 8, 10, 6, 1]
```

Let’s think about…
Average Case

Time Complexity 101 – Best Case

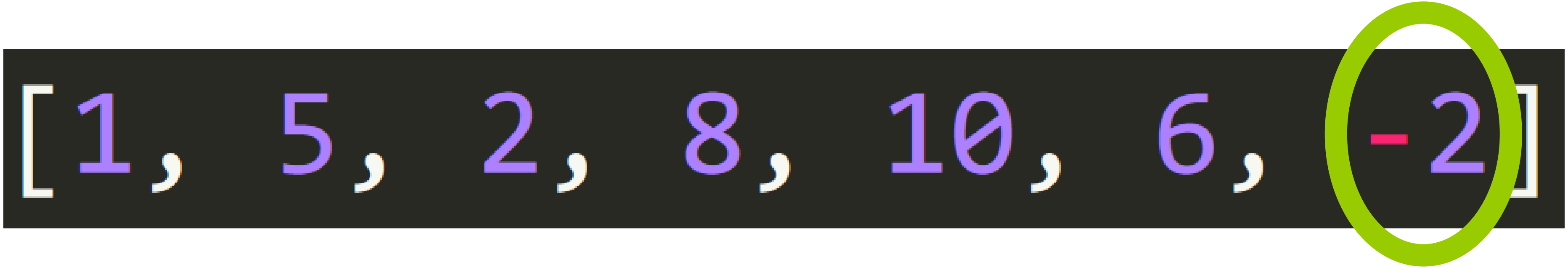


```
[5, 2, 8, -2, 10, 6, 1]
```



Let’s think about…
Worst Case

Time Complexity 101 – Worst Case



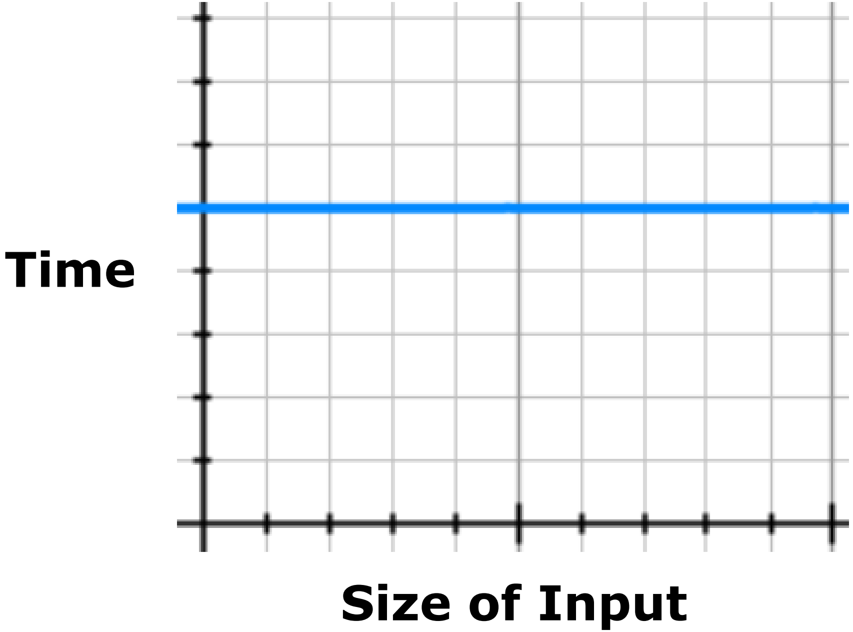
```
[1, 5, 2, 8, 10, 6, -2]
```

Big O = Upper Bound

Time Complexity 101 – Big O Notation

O(1)

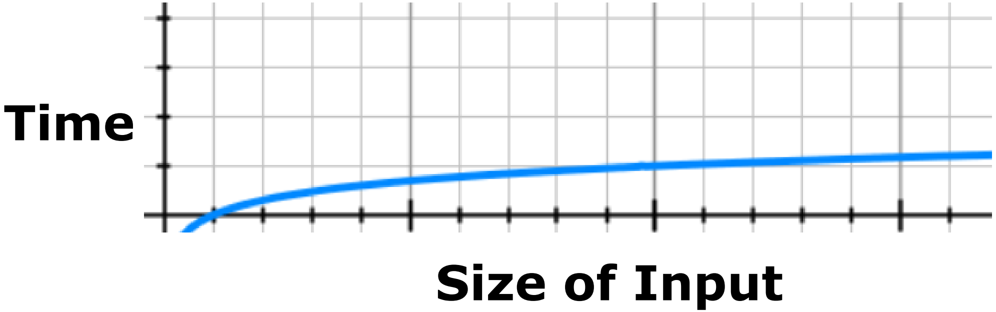
Constant



Time Complexity 101 – Big O Notation

O(log(n))

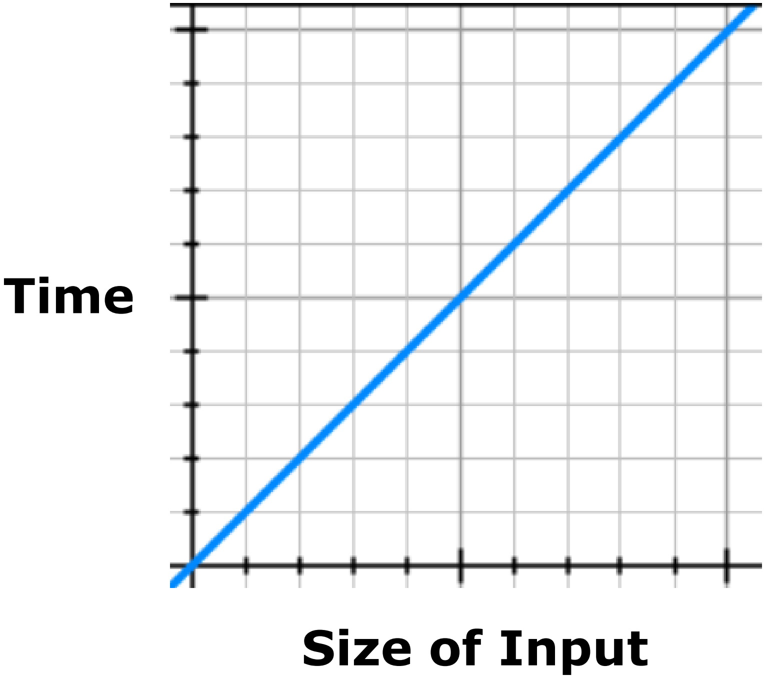
Logarithmic



Time Complexity 101 – Big O Notation

O(n)

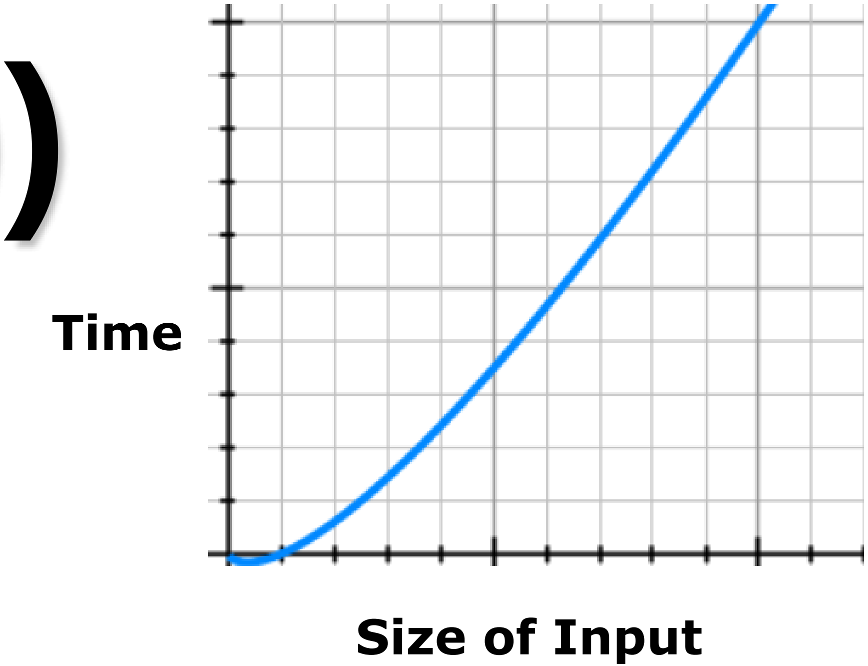
Linear



Time Complexity 101 – Big O Notation

O(nlog(n))

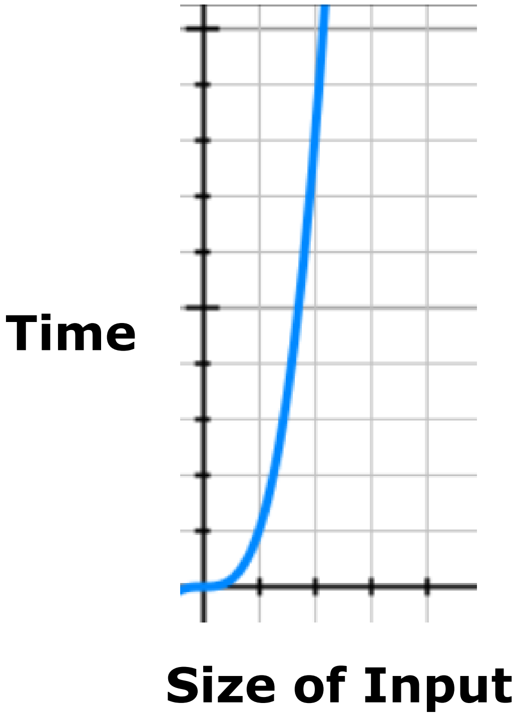
Log-Linear



Time Complexity 101 – Big O Notation

O(n^c)

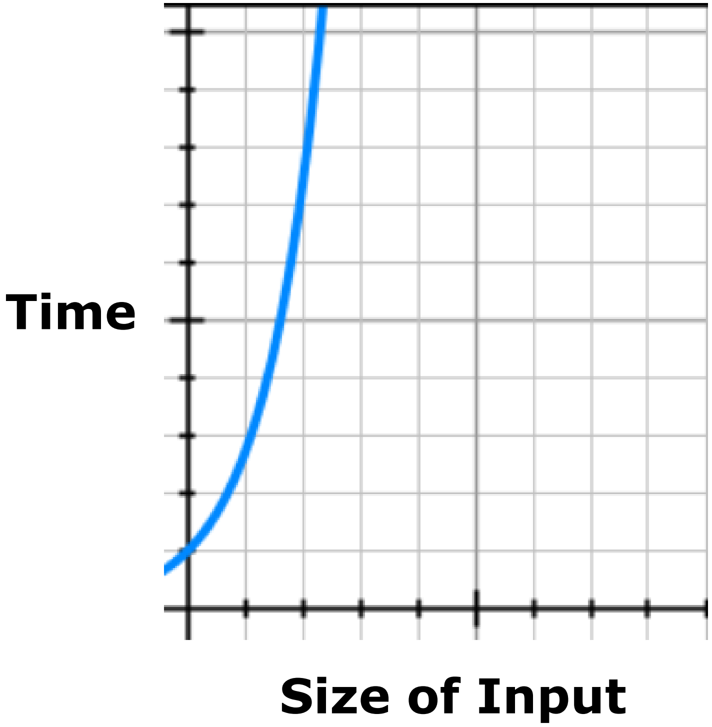
Polynomial



Time Complexity 101 – Big O Notation

O(c^n)

Exponential



Time Complexities → Searching & Sorting Algorithms in Python

Time to Practice!