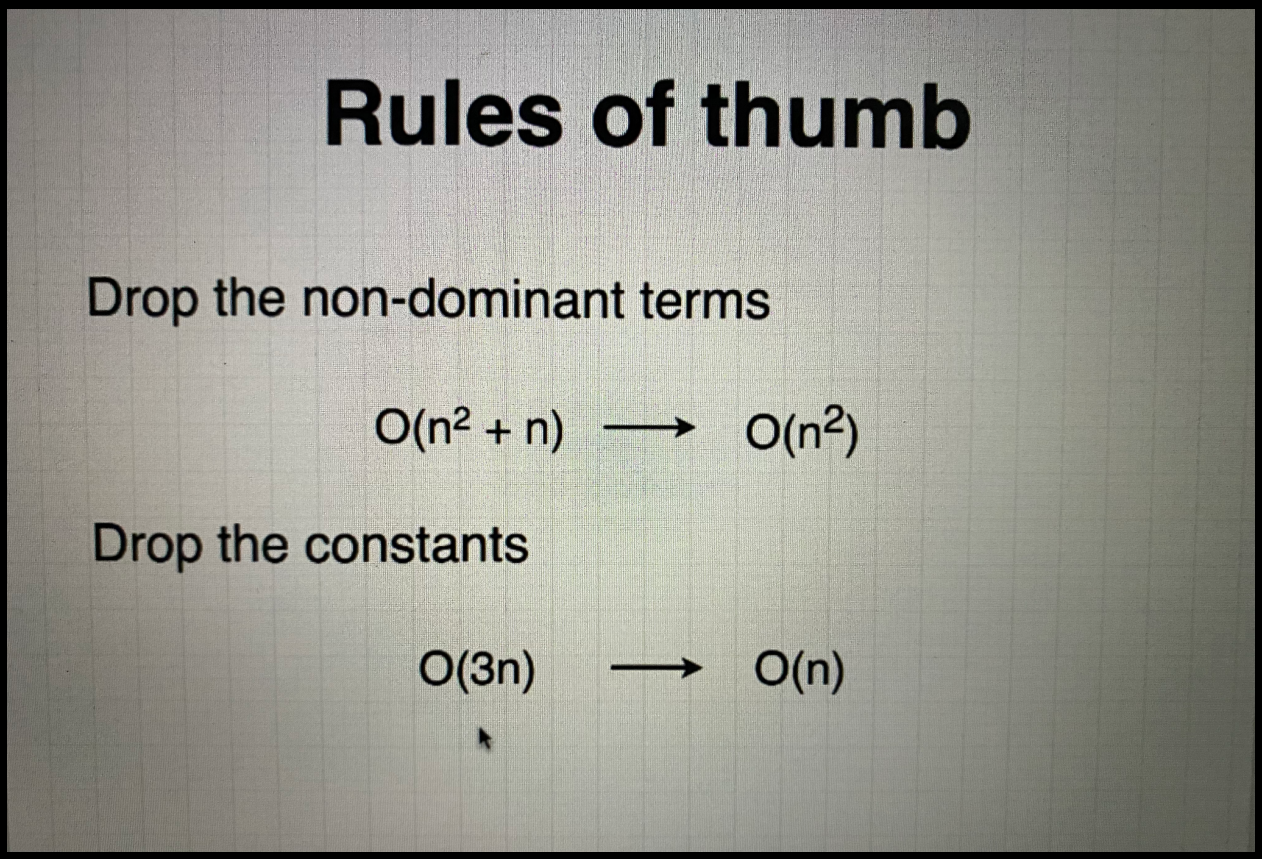
Scenario	Big O Notation
Operation takes same amount of time (regardless of the number of elements)	O(1) Constant
Iterating through a collection (for loop, while loop)	O(n) Linear
Looping through same collection ×2 (nested)	$O(n^2)$ Quadratic
Fibonacci series - recursion O (branch depth)	2^n Exponential
Searching sorted array (i.e. binary tree)	O(Log n) Logarithmic
Sorting	O(n log n) Quasilinear

Rules of Thumb



Rules of thumb

Drop the non-dominant terms

$O(n^2 + n) \longrightarrow O(n^2)$

Drop the constants

$O(3n) \longrightarrow O(n)$

Rules of thumb

Drop the non-dominant terms

$O(n^2 + n) \longrightarrow O(n^2)$

Drop the constants

$O(3n) \longrightarrow O(n)$

Remember Don’t Misunderstand Below:

Add Runtimes:

```
O(n + m)
for (int n: arrayN)
{
        print(n)
)
for (int m: arrayM) {
        print(m)
}
```

This will be O(n+m) and ~~not O(n)~~

Multiply Runtimes:

```
for(int n:arrayN){
        for(int m:arrayM){
                print(n+” ”+m)
        }
}
```

This will be O(n*m) and ~~not O(n^2)~~