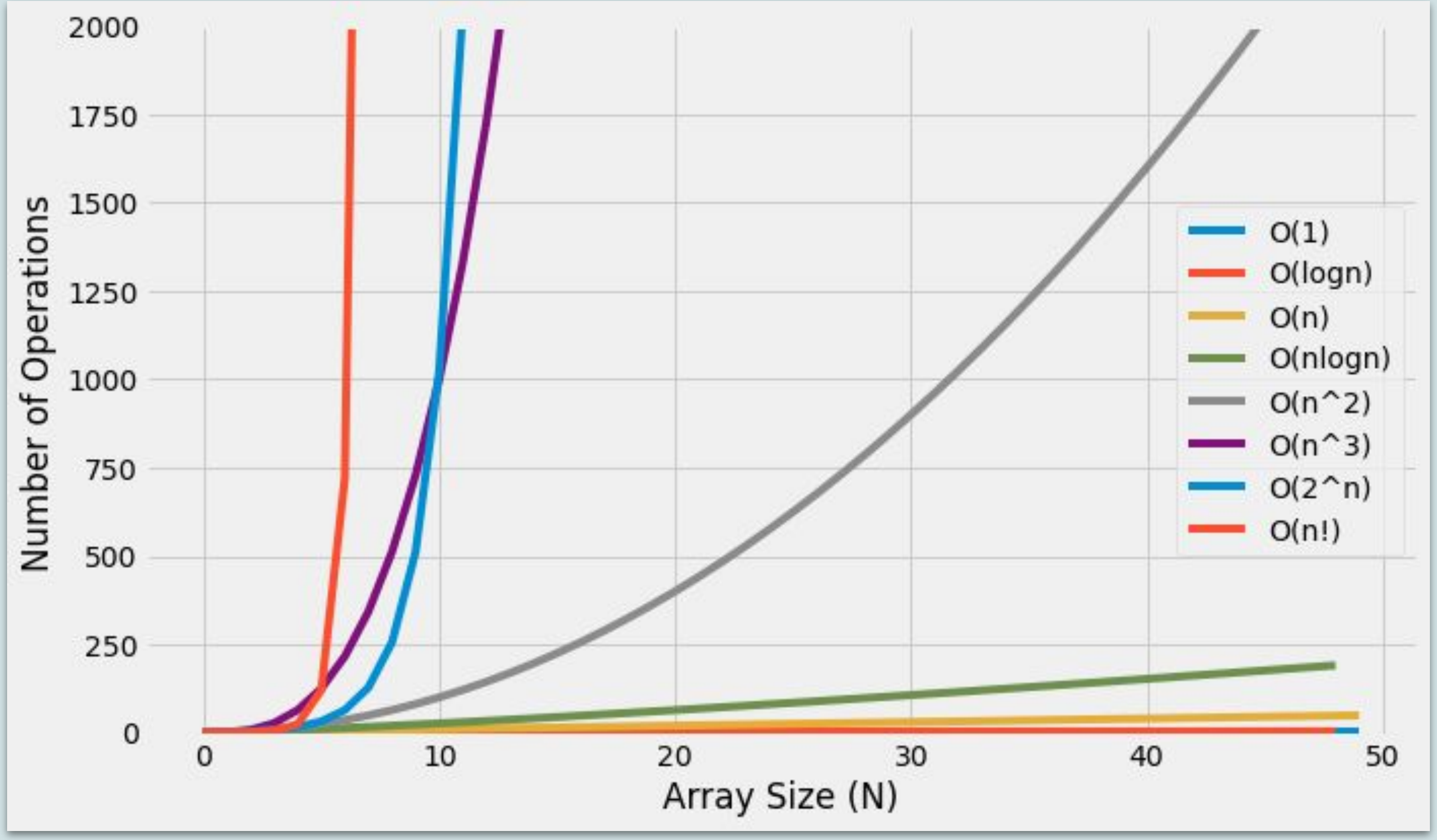
Algorithm Complexity II

Sorting Algorithms

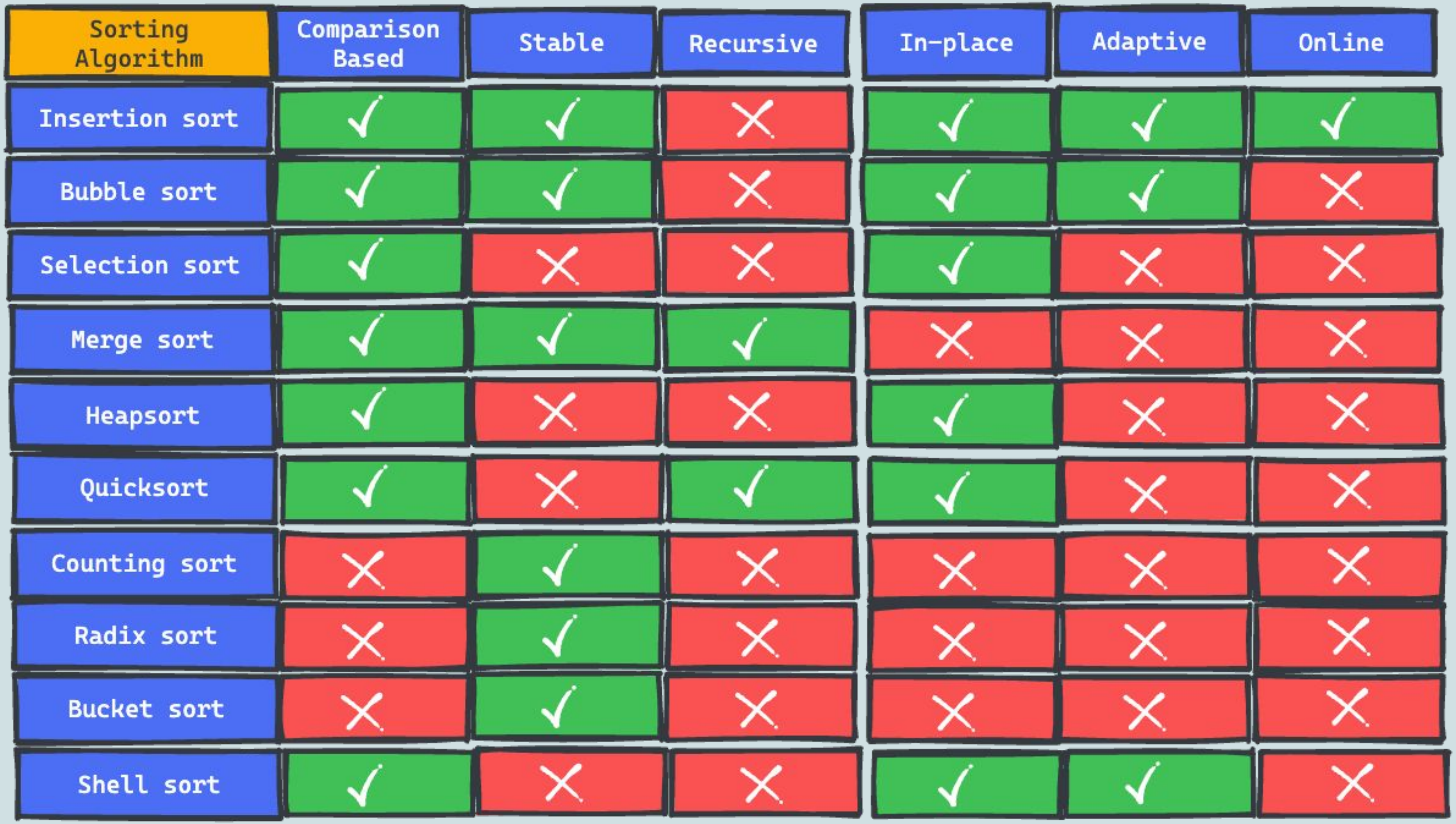
ivanovitch.silva@ufrn.br
@ivanovitchm

Real Python



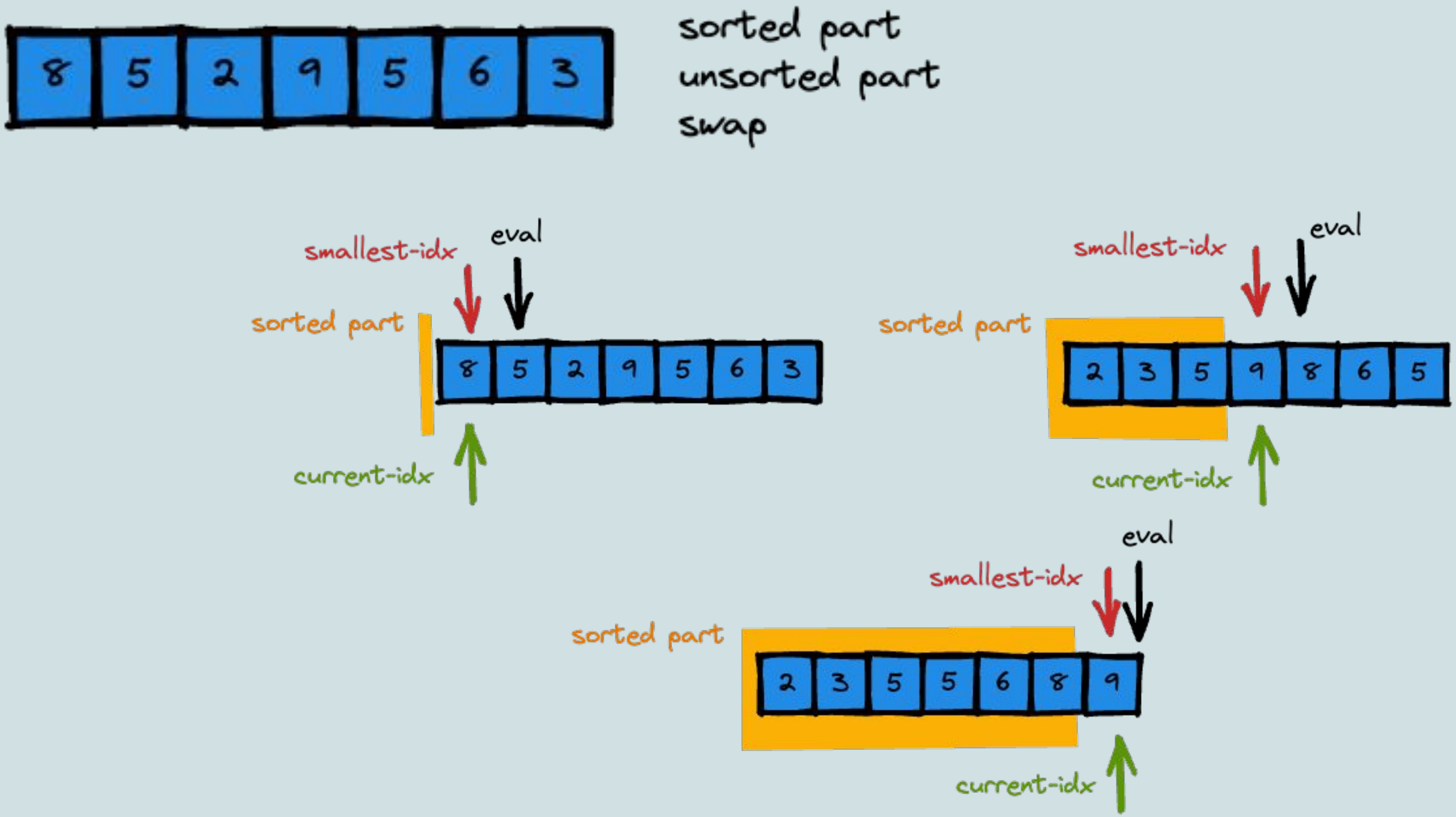
$$O(1) < O(logn) < O(n) < O(nlogn) < O(n^2) < O(2^n) < O(n!)$$

Sorting Algorithm	Time Complexity			Space Complexity
	Best Case	Average Case	Worst Case	Worst Case
Insertion sort	O(n)	O(n^2)	O(n^2)	O(1)
Bubble sort	O(n)	O(n^2)	O(n^2)	O(1)
Selection sort	O(n^2)	O(n^2)	O(n^2)	O(1)
Merge sort	O(nlogn)	O(nlogn)	O(nlogn)	O(n)
Heapsort	O(nlogn)	O(nlogn)	O(nlogn)	O(1)
Quicksort	O(nlogn)	O(nlogn)	O(n^2)	O(logn)
Counting sort	O(n+k)	O(n+k)	O(n+k)	O(n+k)
Radix sort	O(d(n+b))	O(d(n+b))	O(d(n+b))	O(n+b)
Bucket sort	O(n+k)	O(n+k)	O(n^2)	O(n+k)
Shell sort	O(nlogn)	O(nlogn)^2	O(nlogn)^2	O(1)



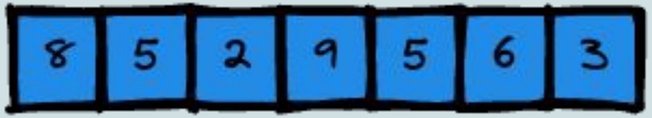
Sorting Algorithm	Comparison Based	Stable	Recursive	In-place	Adaptive	Online
Insertion sort	✓	✓	✗	✓	✓	✓
Bubble sort	✓	✓	✗	✓	✓	✗
Selection sort	✓	✗	✗	✓	✗	✗
Merge sort	✓	✓	✓	✗	✗	✗
Heapsort	✓	✗	✗	✓	✗	✗
Quicksort	✓	✗	✓	✓	✗	✗
Counting sort	✗	✓	✗	✗	✗	✗
Radix sort	✗	✓	✗	✗	✗	✗
Bucket sort	✗	✓	✗	✗	✗	✗
Shell sort	✓	✗	✗	✓	✓	✗

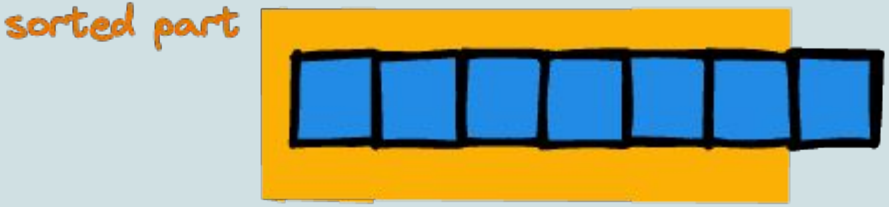
Selection Sort



Selection Sort

Calc of complexity





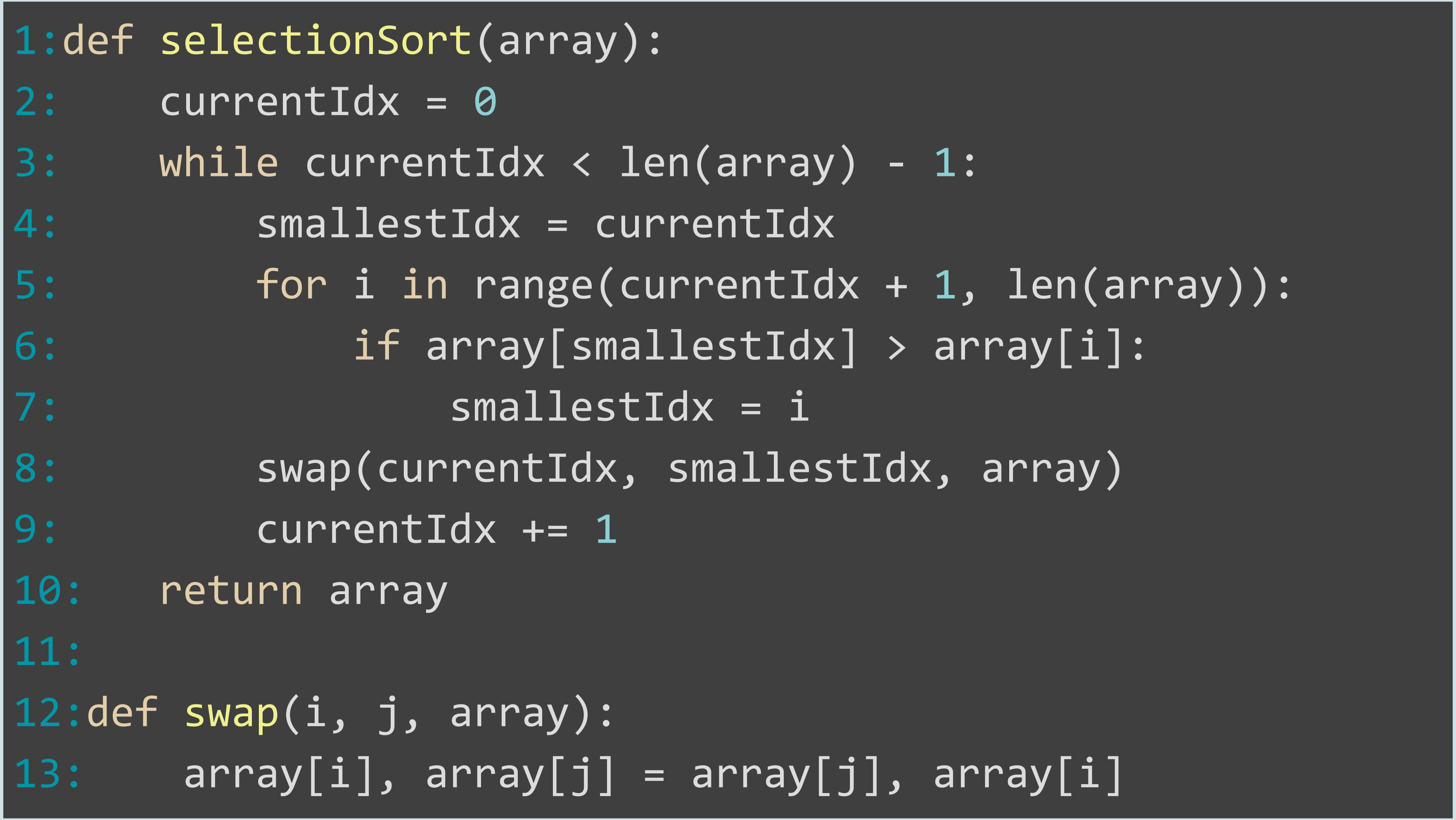
$$T(n) = N$$

$$T(n) = N + (N - 1)$$

$$T(n) = N + (N - 1) + \ldots + 1$$

$$S_n = \frac{N(N+1)}{2}$$

$$O(N^2)$$



```python
def selectionSort(array):
    currentIdx = 0
    while currentIdx < len(array) - 1:
        smallestIdx = currentIdx
        for i in range(currentIdx + 1, len(array)):
            if array[smallestIdx] > array[i]:
                smallestIdx = i
        swap(currentIdx, smallestIdx, array)
        currentIdx += 1
    return array

def swap(i, j, array):
    array[i], array[j] = array[j], array[i]
```