Date 2025.02.12

Data Structure

- A relation of data that is characterized by the operations used to access & satisfy the data
  - ↳ Data Structures:
    - ↳ Arrays
    - ↳ Dynamic Arrays → Vector, Array Lists, List
    - ↳ Linked List
    - ↳ Stack
    - ↳ Queue

Two Sorting Algorithms

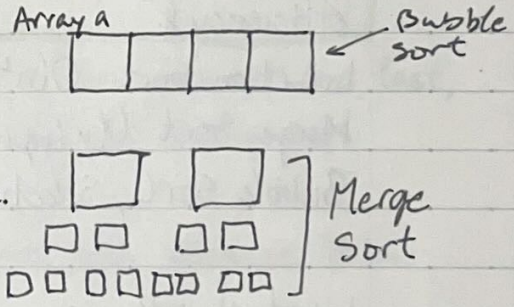
- Insertion Sort (In place)
- Merge Sort ← Not in place
- REQ: General use etc. characteristics ← NOT CODING ☆

Algorithm Efficiency

- Time
- Space (memory)
  - ↳ Will talk about time efficiency 99% of this semester

Space

Either
- In Place
  - ↳ Constant number of temp values
  - ↳ Do not use (really) any extra mem.
- Not In Place
  - ↳ Uses extra space, an example type is linear



Bubble Sort: reorganizes inside 1 array
Merge Sort: Creates multiple arrays to sort 1 array

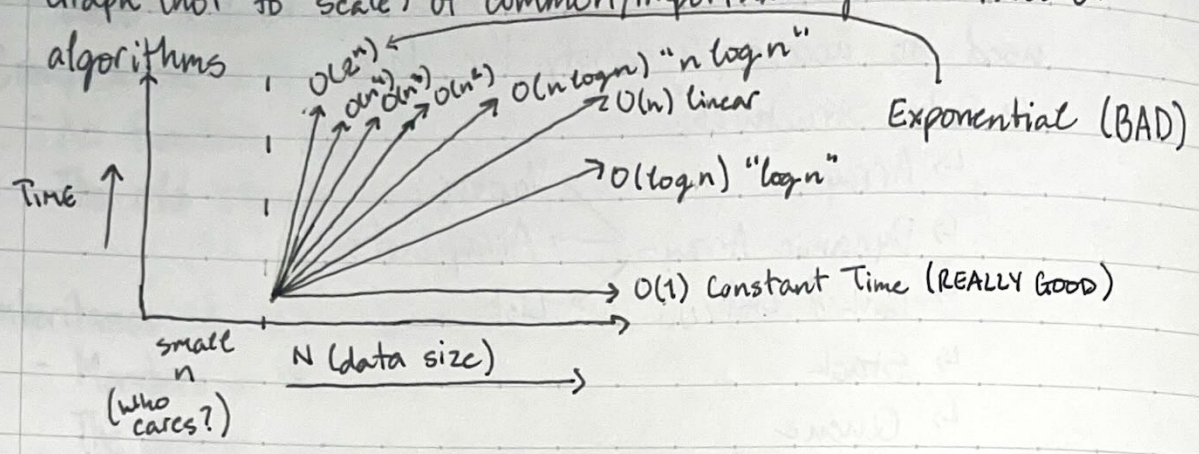
Time Efficiency

- 2 ways of measuring efficiency
  - ↳ "official mathematical way"
  - ↳ in-practice by every coder & cs major ← Will be using only this (O(n))
- big-O (Notation) O(n) ← Either an upper bound or tight bound
  - order ↑ ↑ Factor that relates to size of input

O(n)
["order"
[factor that relates to size or input

Time Efficiency (Cont.)

Graph (not to scale) of common/important growth rates of algorithms



Scenario

Competitor → A fast computer 1 billion ips

Yourself → B Slow computer 10 million ips

:loafmaSad:

2x theoretical ← Codes Insertion Sort

50x Theoretical ← Codes Merge Sort

20 mins of runtime

Takes 2.3 days of runtime

★ Choice of algorithm is essential/important for efficiency ★

Insertion Sort - Exponential

Insertion Sort $O(n^2)$ (Exponential) ← Can still be theoretically useful

Merge Sort $O(n \log n)$

Bubble Sort, Selection Sort, Insertion Sort $O(n^2)$ ← Very Efficient For Small Collections

Looked at code yay

$O(n^2)$	$O(n)$	$O(\log n)$
Bubble Sort	find min, find max	binary search
Selection Sort	linear search	