Algorithms: 2/20

\*Must be Unambiguous & Repeatable

🡪 \*Impossible to interpret it another way; mathematically specific\*

Searching & Sorting:

Sorting a list:

1. Look for lowest number – by comparing a number to every other number in the list

**SWAP** ALGORITHM: done 1 time

Swap(i, j) – takes two parameters

* If List[i] > List[j] – then swap the values

**BUBBLE** ALGORITHM: done N times (however many items are in the list)

* For every value in the list (i) – swap (i, i + 1)

Ensures the largest number is on the right (bubbles all the way to the end of the list)

\*In order to achieve a fully sorted list – must perform bubble the same number of times as the size of the list\* (worst case)

**BUBBLE SORT** Algorithm: done N squared times 🡪 \*Exponential algorithms aren’t fast nor efficient\*

* Bubble N times (N being the number of items in the list)

\*Time Complexity 🡪 Exponential vs Logarithmic Algorithms

Searching Algorithms use Trees 🡪 \*indexing structures – organizing data in a way that is optimized for searching

**Binary Tree**: at every level, each node has only 2 children

* Binary tree w/ sorted list – values to the left is small, values to the right is larger
* Gets rid of half of the possible values (repeatedly dividing by 2 on a calculator)