Chapter 1: - The Role of Algorithm in Computing

Algorithm: is a well defined computational procedure that takes a value or set of value as input, and then produces a value or set of value as output.
It's a sequences of computational steps that transforms the input into the output.

For Ex: we need to sort a sequences of numbers in non-decreasing order.

* Example in formally defining the sorting problem:

Input: n number $\langle a_1, a_2 \dots a_n \rangle$

output: a permutation (re-ordering) $\langle a'_1, a'_2, \dots a'_n \rangle$ where

$$a'_1 \leq a'_2 \leq \dots \leq a'_n$$

Comparing two computers:
One with a better algorithm than another.

Computer A:

* fast prosessor
* Insertion Sort

$c_1 n^2$ — time to sort n items

c_1 is constant that doesn't depend on n

$$\boxed{\text{time Complexity } c_1 n^2}$$

Computer B:

* Slow processor
* Merge sort

$c_2\, n \lg n$ — time to sort n items

* c_1 and c_2 are constants. Not as important as the arithmetic operations done on n. (e.g. n^2, $n \lg n$, etc...).
* Regardless of Input size constant stays the same

* Fast: 10 billion instructions
* Skilled: programmer code requires $2n^2$ instructions to sort n items

* Slow: 10 million instructions.
* Average programmer, code requires $50\, n \lg n$ instructions.

$$\lg n = \lg_2 n$$

$$\lg n = \frac{\log_{10} n}{\log_{10} 2}$$

Computer A to sort 10 million numbers

$$\frac{2(10^7)^2 \text{ instructions}}{10^{10} \text{ instructions/sec}} = 20{,}000 \approx 5.5 \text{ hours}$$

Computer B to sort 10 million numbers:

$$\frac{50 \cdot 10^7 \lg 10^7 \text{ instructions}}{10 \text{ million instructions/sec}} = 1{,}163 \text{ seconds}$$

$$\approx 20 \text{ min}$$