Algorithm Analysis

- A **problem** for computer must be defined precisely by its input and desired output.
- Ex: sort an array
  - Input: array and a way to compare the elements
  - Output: Sorted array
- Ex: Compute the factorial of a positive integer
  - Input: a positive integer n
  - Output: n!

Note that we can use the tools of discrete math to define these inputs and outputs precisely

- A solution is some method of taking an arbitrary input and computing the output with the desired properties defined by the problem.
- An **Algorithm**, is a sequence of steps you can perform to get the desired output from the input.

```
Sort(A)
    Let S = the set of all permutations of A
    For x in S:
        if x is sorted:
                return x
```

This is an example of a pseudo code. This is a way of communicating an algorithm to another human. It is written in a way that is easier to understand by a human. It has a mix of precise and unambiguous notations and words.

1. Is this algorithm correct?
2. Does this algorithm work efficiently? (runtime)

How to measure the runtime?

1. Idea 1: Implement the algorithm, run it and then time it.
   - Depends on the software, hardware, operating system, programming language, etc.
   - Implementation takes time and money.
   - How to determine which input to run?
   - How do we know the inputs that we selected to run reflect the runtime?
2. Idea 2: Find a function that reflects the runtime in terms of the input size
   - Runtime: number of primitive operations (arithmetic operations, logical operations, variable retrieval, variable assignment, etc.)

- **Algorithm 1**

Sum = 0 ← variable assignment (1)

For i = 1 to n: ← variable assignment (1), logical operation (1)

    sum = sum + 1 (3 operations) ← arithmetic operation (i++)

For i = 1 to n:

    sum = sum + 1

For i = 1 to n:

    sum = sum + 1

$$f_1(n) = 1 + (1 + 5 \cdot n) \times 3 = 1 + 3 + 15n = 15n + 4$$

- **Algorithm 2**

Sum = 0 ← variable assignment

If n < 100: ← Variable access + logical operation (2)

    For i = 1 to n: ← variable assignment (1), logical operation (2)

        For j = 1 to n: ← variable assignment (1), logical operation (2)

            sum = sum + n/3   5 operations.

Else

    sum = 3*n   3 ops.

$$f_2(n) = \begin{cases} 1+2+(1+(2+(1+(2+5)n)) \cdot n \\ 3+n+2n^2+n^2+2n^2+5n^2 \\ = 9n^2+n+3 & n<100 \\ 1+2+3=6 & n \geq 0 \end{cases}$$

- **Algorithm 1**

```
Sum = 0
For i = 1 to n:
        sum = sum + 1
For i = 1 to n:
        sum = sum + 1
For i = 1 to n:
        sum = sum + 1
```

Variable assignment

Each loop:
Assigns i
Logical operation
Variable access
Arithmetic operation
Variable assignment

$1 + (1 + 4*n)*3 = 4 + 12n$

$f_1(n) = 12n + 4$

Given $f_1(n) = 15n + 4$, $f_2(n) = \begin{cases} 9n^2 + n + 3 & n < 100 \\ 6 & n \geq 100 \end{cases}$

What is best algorithm?