# Mission Sidequest 3.1: Force Efficiency

Start date: 29 August 2017
**Due: 3 September 2017, 23:59**

Readings:

- Textbook Sections 1.1 to 1.2

This side quest consists of **three** tasks. **Click here for the link to the template**. Use the template provided in the link to complete the tasks.

## Task 1:

Consider the functions $f(n)$ and $g(n)$ . Decide whether $f(n)$ has order of growth $O(g(n))$ or $g(n)$ has order of growth $O(f(n))$ .

1. $f(n) = 4^n n^2$ or $g(n) = n3^n$
2. $f(n) = 1000000000n^2$ or $g(n) = 2^n/1000000000$
3. $f(n) = n^n + n^2 + 1$ or $g(n) = 4^n + 2^n$
4. $f(n) = 1^n$ or $g(n) = n^2$

## Task 2:

Consider the following function `foo`. Give a precise mathematical function $m(n)$ (not just the order of growth) that calculates the number of arithmetic operations (`===`, `+`, `-`, and `*`) that occur during the computation of `foo(n)`. Give the simplest function $h(n)$ such that $m(n)$ has order growth $\Theta(h(n))$.

```
function foo(n) {
    function bar(n) {
        if (n === 0) {
            return 0;
        } else {
            return 1 + bar(n - 1);
        }
    }
    return n * bar(n);
}
```

## Task 3:

Consider the following functions `foo` and `bar`. Implement a function `improved_foo` that gives rise to an iterative process and whose runtime has an order of growth that is better than that of `foo`. What do we mean by better here?

```
function bar(n) {
    if (n === 0){
        return 0;
    } else {
        return n + bar(n - 1);
    }
}
function foo(n) {
    if (n === 0) {
        return 0;
    } else {
        return bar(n) + foo(n - 1);
    }
}
```

## Submission

To submit your work for this mission, copy the url on your browser and email it to your respective Avengers. Strictly follow to the deadlines set at the start of this file.

IMPORTANT: Make sure that everything for your programs to work is on the left hand side (source code) and not in the interpreter! This is because only that code is preserved when opening the url you have emailed to us.