Exercises:

Recursion BOOOOOOOOOOOOY!

I'm pretty sure you know very well what's recursion is. If not, make sure everyone knows.

1. Look at the code below. Transform it in a recursive function:

```
for(i = 0; i < sizeof(array); i++) {
  array[i] = array[i] * 2;
}
```

Observation: remember a function can have multiple clauses with different parameters.

2. Refactor the code you did in question `01` using guarded clauses.

If you forget what is a guard, look the example below:

```
def print_multiple_times(msg, n) when n <= 1 do
```

So, recursion has some powerful utility. Look the code below:

```
defmodule Math do
  def sum_list([head | tail], accumulator) do
    sum_list(tail, head + accumulator)
  end

  def sum_list([], accumulator) do
    accumulator
  end
end
```

3. Show me what this function got. What do i mean? Given the initial parameters, [1,2,3,4,5] and 7, show all iterations functions call!

But this doesn't represent the real world. What do i mean? Functions like above are very common so Elixir created a module to handle this, called `Enum`.

4. You're (mostly) ruby developers, so the next two snippets should not be strange to you guys. Explain what each one of them does:

(a)

```
iex> Enum.reduce([1, 2, 3], 0, fn(x, acc) -> x + acc end)
6
```

(b)

```
iex> Enum.map([1, 2, 3], fn(x) -> x * 2 end)
[2, 4, 6]
```

5. What's the name of the structure `fn(x) -> x * 2 end`.

Tip: Javascript also defines it with the same name.

6. Use `Enum.reduce` to calculate sum of the list `[1,2,3,4]` starting with 7.