Multiple Good Reasons

This is a **warmup** exercise. It is **not compulsory**, and may be completed **individually or with your lab partner**.

Factorial is a common example of a recursive function.

Factorial is defined on the set of natural numbers, which we can approximate with `long long`, as

$$\begin{aligned} 0! &= 1 \\ n! &= n \times (n-1)! \end{aligned}$$

(The $0!$ is important!)

For this exercise, we'll implement a simple recursive factorial, which has this prototype:

```
long long factorial (long long n);
```

A recursive function should have a *base case*, and a *recursive case*; in the recursive case, the function should call itself, with different arguments.

Beware! Lots of resources will suggest you should just `return`. Our style guide forbids multiple returns. Consider how you define your base and recursive cases to avoid multiple returns.

To run some simple automated tests:

```
$ 1511 autotest factorial
```

To run Styl-o-matic:

```
$ 1511 stylomatic factorial.c
Looks good!
```

You'll get advice if you need to make changes to your code.

Submit your work with the *give* command, like so:

```
$ give cs1511 wk12_factorial
```

Or, if you are working from home, upload the relevant file(s) to the `wk12_factorial` activity on Give Online.