INTRODUCTION TO COMPUTER SCIENCE I
LAB 3
Basic loops

This lab will require you to practice using *iteration* (a.k.a., `while` loop statements).

1 Factorials

Consider the *factorial function*:

$$fact(n) = \begin{cases} 1 & \text{if } n = 0 \\ n \times fact(n-1) & \text{if } n > 0 \end{cases}$$

This function forms a *sequence*, where the 0^{th} entry is 1, and the n^{th} entry is the product of all numbers from 1 to n. The sequence begins: $1, 1, 2, 6, 24, 120, 720, \ldots$

You will be writing code that works with this sequence. Let's see exactly what that will entail...

2 Getting started

In order to get started with this assignment, do the following:

1. **Login:** As always, connect with Remote Desktop to `remus` or `romulus`.
2. **Open terminal:** Right-click on the desktop (within Remote Desktop!) and use the drop-down menu that appears to open a Terminal window.
3. **Make a directory:** Create a new `lab-3` directory with the `mkdir` command, and then change into that new directory with the `cd` command.
4. **Copy the initial source code:** Take a copy of the initial source code, which you must complete, like so:

```
$ cp ~sfkaplan/public/COSC-111/lab-3/Factorial.java .
```

5. **Open the source code for editing:** Use `emacs` to open this new program's code.

3 Your assignment

Complete the source code provided in `Factorial.java`. Specifically, the comments in the code guide you write the following critical loops:

1. **Get a valid input:** Prompt the user to enter a number (which we'll call n) that is a *non-negative integer*. If the user enters three invalid values, then the program should print a message that it is "giving up", and it should not move on to the following two steps.

2. **Calculate the factorial:** Via repeated multiplications, calculate $n!$. Use a `long` integer to calculate your answer (since factorial values quickly get big as n increases).

3. **Find the maximum factorial:** Even a `long` integer has a limited range. Any number larger than about 8 quintillion cannot be stored in such a variable. Remember that if you take the largest positive value that can be stored in a `long` integer and then add `1` to it, the value will *wrap around* into the negative numbers. Consequently, if we try increasing values for n, eventually we will find a value (let's call it n_{max}) that yields the **largest** factorial number that can be correctly contained in a `long` integer variable—let's call that one f_{max}. **Write a loop** that calculates `n_max` and `f_max`.

4 How to submit your work

Use the CS submission systems to submit your work, as usual. Recall that you may submit via a web browser or the command line (with the `~lamcgeoch/submit` command).

This assignment is due on Thursday, Feb-18, 11:59 pm, before it becomes Friday, Feb-19.