CSCA20 - Lab 4

Functions & Code Cleanup

Learning Objectives

In this lab, we will be practicing using functions and comments to make code cleaner, easier to read and write, and less likely to cause bugs in future. The key takeaway is that there is a difference between 'working' code and 'good' code.

Marks

Your TA will record your marks during the tutorial section. Part of your responsibility is to demonstrate your solutions to your TA accurately.

Arrived with pre-lab completed	/1
Showed up on time & worked through lab	/2
Successfully demonstrated working code	/1
TOTAL	/4

Prelab

- Download and test out the starter file, make sure it's running properly on your machine
- wait... is that it?
- yup... super easy pre-lab this week, because we figured you might be busy with other things over the weekend before this lab

Lab

Brian found this solution to last week's lab. It works... but frankly it's terrible. Whoever wrote it (he blames Nick) was able to get code working, but definitely wasn't thinking about building something that could be added to or maintained over time.

You aren't asked to add any functionality to the code. Instead your job this lab will be to "re-factor" the code: to clean it up and make it better without altering the actual end-user experience.

Your final code should:

- Use properly documented functions to reduce code replication
- Have sensible and helpful internal comments to explain what the code is doing
- Be more efficient, in terms of holding unnecessary data, the number of times it accesses each piece of data, and the complexity of the code

Postlab

(This section will not be marked, but it's good practice if you finish the lab early or want to continue to work at home)

Going back and cleaning up old code is a good habit to get into, and it's also a great way to practice. Plus, it feels good to go back to something you struggled with at the beginning of the semester and now find that you can complete it with ease.

- Re-factor your submissions from previous labs. Try to break code down into functions where appropriate, see if you can re-use functions from one lab in another, use your new-found skills to solve old problems in new ways.
- Try to generalize your functions, so that they might be more useful in a wider array of situations. A function that converts a grade into UofT GPAs is nice, but a function that can work for other institutions with other GPA scales is better. Checking whether a number is a valid grade is good, but checking whether a general string represents a valid grade is probably useful to even more people. Since you're only ever writing the function once, it's worth your while to add more functionality.