SAMPLE REFLECTION #1: WORTHY OF FULL MARKS

Name: Full name here
Student Number: #########
Email: userID@myseneca.ca
Section: XXX
Workshop: 2 (at-home) Reflection

In this workshop I learned how to declare variables of different types and how to apply arithmetic calculations using those variables. I also learned how to accept user input using the standard input output library function "scanf". I found casting one data type to another confusing at first, but now feel confident using this feature. I also had to spend some extra time practicing how to use the modulus operator.

I thought it was very interesting to learn that computers can't store floating-point types accurately. Some numbers can only be approximate at best which can lead to some major inaccuracies in data if not managed properly. Dealing with this was difficult. The printf function will round to the precision you specify so you will think things are fine because it displays correctly, but the actual stored value is not necessarily precise. Now I understand why this workshop forces us to convert to integer using the lowest coin denomination in cents and using integer division with the modulus operator.

It is a best practice to convert floats/doubles to integers before performing arithmetic operations in order to prevent cumulative errors (imprecision) caused when storing floating point values in exponential mode in system memory. To convert float/double types to integer, you need to apply casting (or rely on coercion which does the conversion automatically). Once you have what you need converted to integer types, you can perform calculations with more accuracy and not worry about decimal precision errors.

With this in mind, using the modulus operator to obtain the remainder not only keeps our cents remaining accurate, but it also streamlines the calculation to one line of code rather than 2 (have to divide, store the result, then subtract the result from the original value). The modulus operator however is only applicable to integral data types.

SAMPLE REFLECTION #2: MINIMUM REQUIRMENTS

Name:	Full name here
Student Number:	#########
Email:	userID@myseneca.ca
Section:	XXX
Workshop:	2 (at-home) Reflection

In this workshop I learned how to convert back and forth between numeric variables, float to int, and int to float. I also learned how to use scanf to read user input, and that the scanf function requires the type of the value to be read (format specifier), and the address of the destination variable in order to be able to store the user input.

It is a best practice to convert floats to integers before performing arithmetic operations in order to prevent cumulative errors (imprecision) caused when storing floating point values in exponential mode in system memory.

The modulus operator allows us to extract the remainder of an integer division operation in one step and is faster than subtracting the result of division from the original value.