RMIT University
School of Engineering
EEET2248 – Electrical Engineering Analysis

Lectorial Progress Report

Lecturer: Dr. Katrina Neville
Student Name: Aiden Contini Student Number: s3780445

Submission Due Date: 22nd March 2019

Abstract:

This section of the assignment required individuals to write a code in MATLAB which enables hard coded, programmer chosen numeric values to be converted to and from their imperial units and metric units. For example, it was required that certain values for Fahrenheit were to be chosen by the programmer, converted into their metric value (Celsius), and then outputted by the program into the command window. This process was repeated for numerous other metric and imperial units, such as inches to centimetres, and litres to gallons.

Up until this point, I have implemented the defining of hardcoded values into variables, and using these variables within specific formula to output specific values into the command window. This enables the values selected for each input unit to be placed into a formula which can convert it to its equivalent metric/imperial unit. In this milestone I have also implemented sections into the side to separate each conversion. This is to ensure ease of access when using the code in order to ensure the different conversions can be easily run, managed and understood by a user.

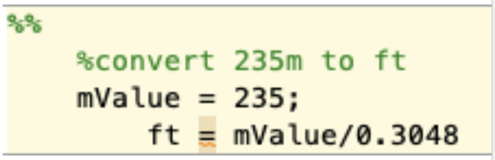
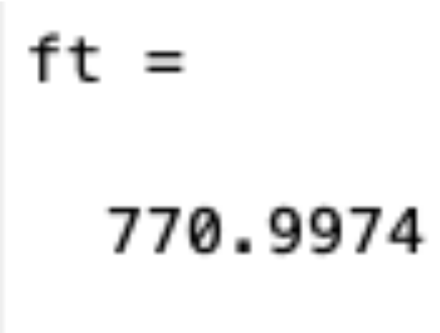
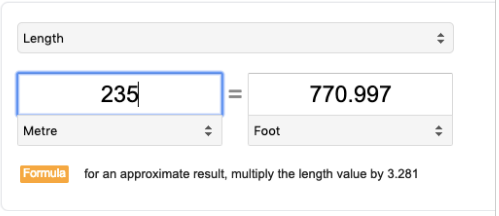
As of this stage, the program can convert with accuracy hard coded values into their metric/imperial equivalent. With the primitive nature of this program however, in order to change what values are converted, the values within the code itself must be changed, as no user input functionality has been introduced yet. This said however, the proof of the correct outputs being displayed by the program is evidence that the program is in fact accurate and does work the way in which it should.

Software solution and testing

Hard Coded Value	MATLAB output value (converted)	Google converted value
0° Celsius	32° Fahrenheit	32° Fahrenheit
100° Fahrenheit	37.7778° Celsius	37.7778° Celsius
33cm	12.9921 inches	12.9921 inches
25 inches	63.5 inches	63.5 inches
235m	770.9974 ft	770.997 ft
654 ft	199.3392m	199.339m
3.65km	2.268 miles	2.268005 miles
7.79 miles	12.5368km	12.53679km
156 grams	5.5027 ounces	5.50274 ounces
15 ounces	425.243 grams	425.2425 grams
3.4 kg	7.4956 pounds	7.49572 pounds
9.35 pounds	4.2412 kg	4.241089 kg
0.657 km/h	0.4082 mph	0.40824087 mph

Hard Coded Value	MATLAB output value (converted)	Google converted value
1567.92 mph	2.5233e+03 km/h	2523.322644 km/h
1.56 litres	0.4121 gallons	0.4121084 gallons
57.4 gallons	217.2843 litres	217.2826 litres
16.4 hectares	40.5260 acres	40.52528 acres
19.54 acres	7.9074 ha	7.9075574 ha

<u>Screenshot Proof:</u>

Code	Output	Google Output
 %% %convert 235m to ft mValue = 235; ft = mValue/0.3048	 ft = 770.9974	 Length 235 = 770.997 Metre Foot Formula for an approximate result, multiply the length value by 3.281

This data shows the testing of the program that was done to ensure the program was outputting the correct converted values for each input. Google converter was used as a point of reference as this converter can be widely taken to be very near accurate, and the data shows the close proximity each MATLAB outputted answer to the google answer. Up until this point, there is no user inputs into the program, and hence the program does not have to deal with erroneous inputs or usability. Up until this point, the program simply requires the conversion of hard coded values, and hence due to the accuracy proven with the data above, the requirements of the assignment have been fulfilled to this point.