RMIT University
School of Engineering
EEET2248 – Electrical Engineering Analysis Lectorial Report
Lecturer: Dr. Katrina Neville
Student Name: Aiden Contini Student Numbers: s3780445
Session: 10:30am Monday Submission Due Date: 10/05/2019

Introduction/background:

This assignment required students to make a functional Graphical User Interface (GUI), which was able to act as a unit converter between different imperial and metric measuring units. Students were given various formulae, and the required conversions which should be available to a user for their GUI, and hence had to program an app within app designer to achieve these outputs. This task was a progressive task, which began simply as an app which performed math on hardcoded values, and outputted them as required, and built up to the point of having a fully functional, easy to use GUI. In this way, the purpose of the task was for students to progressively build knowledge on the Matlab computing environment to the point where they could create a GUI.

Design and methodology: (30 marks)

In order to design the user interface, hand drawing was done in order to develop the layout of the interface (figure 1). Various drafts were done, but finally the final draft seen below was settled upon for the GUI layout. Similarly, figure 2 displays the pseudocode which was needed to correctly code the GUI to properly working. It shows the drafted process which I used to convert into the Matlab callback functions and inadvertently create the unit converter program.

Overall, the Matlab tools required for the final GUI creation were Matlab's app designer function. The main techniques needed for this assignment were switch statements in order to adjust specific elements of the GUI dynamically. The conversion formulae also had a very big role in the GUI as they were used to actually perform the math and output the specific converted values.

Figure 1: Final layout of the GUI

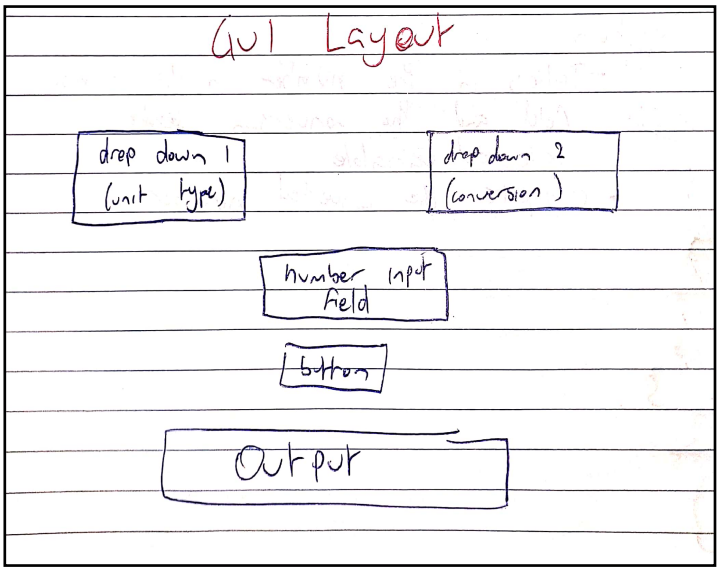
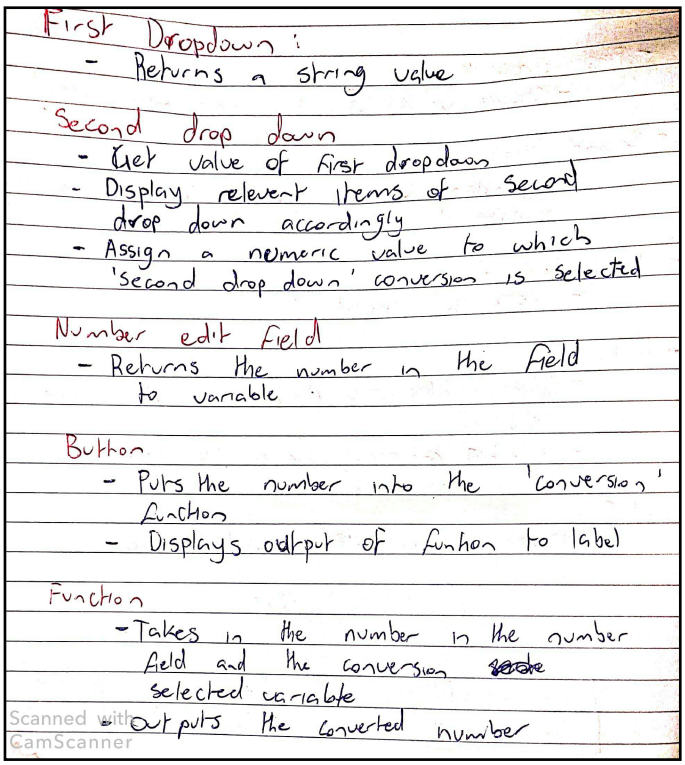


Figure 2: Pseudocode showing the required performance of the application



Below (figure 3) also features a flow diagram showing the structure of the program, and the execution order to complete one full conversion.

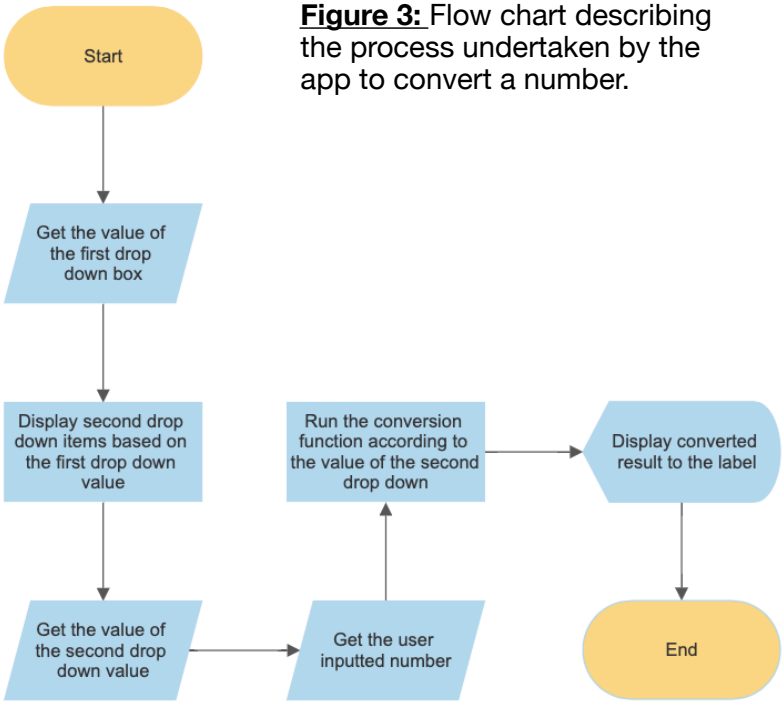


Figure 3: Flow chart describing the process undertaken by the app to convert a number.

Output and testing (50 marks)

Once the app was created, a test was done to ensure all the conversions within the application were outputting accurate values. This, hence, is a test of the switch statement put in place within the app, to ensure the correct conversion is being undertaken when it should be. Figure 4 is a table which shows the various conversions which were undertaken in testing and their outputted values, comparing them to that of the Google online converter for accuracy purposes. Testing was also done to ensure that negative output is not outputted when an invalid negative input is entered (such as negative centimetres). Alongside these, testing for erroneous input, such as entering letters into the input box were tested. This error is handled by the inbuilt feature of app designer, and both this and the negative input test are shown in figures 5 and 6.

Figure 4: Table displaying the tested values and their output

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

Hard Coded Value	MATLAB output value (converted)	Google converted value
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
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

Figure 5: Image showing the handling of invalid negative input

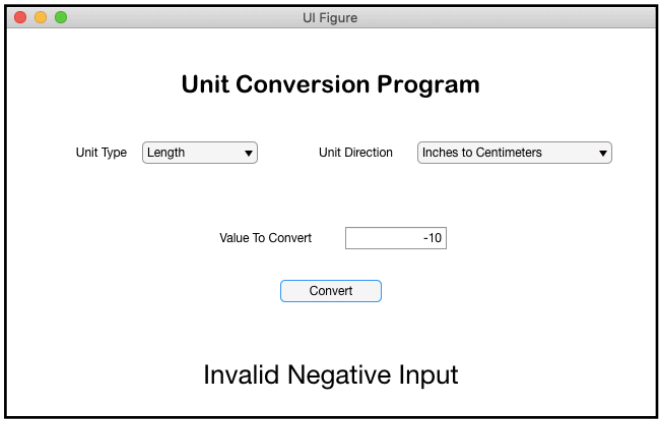
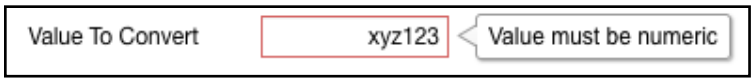


Figure 6: Image showing the handling of invalid non-numeric input



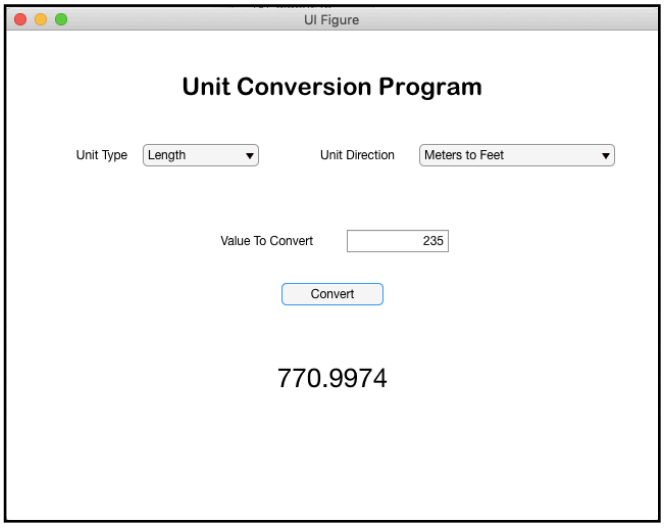


Figure 7: Image showing the evidence for the conversion's accuracy

Figure 7 provides evidence for the programs accuracy, and hence shows the final output of the program once a conversion has been undertaken. In order to show the usability of the program, Figure 8 and 9 provide images of the two dropdown menus which are used to actually use the program. Figure 8 shows the easy to read "type" of conversion, and Figure 9 displays the dropdown for "direction" of conversion (specifically for the "length" units).

Figure 8: Unit Type dropdown

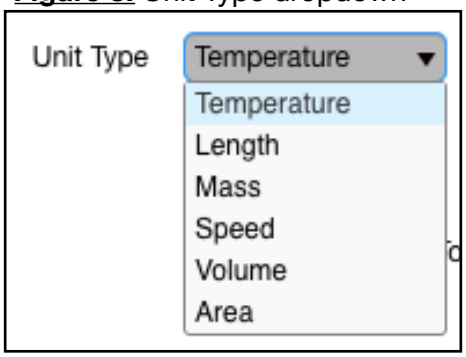
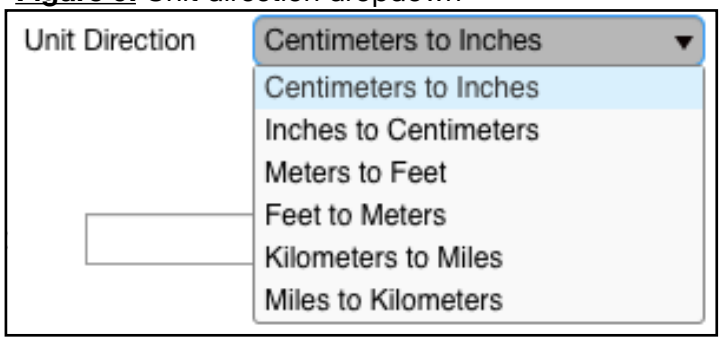


Figure 9: Unit direction dropdown



It can be seen from these images that the program satisfies the design requirements of being easy to use, and the testing above provides a great deal of evidence on the accuracy of the program's performance. Hence this performance fits the requirements of the program to cover units to and from their metric/imperial equivalents.

Conclusion:

In conclusion, the program can effectively perform all the conversions requisition of the electoral sheet, hence satisfying the requirements of the task. The major weakness of the program involves it having a lag of sorts. For example, at times, when the convert button is clicked, it takes a small amount of time to display the output. Similarly, there is also a minor weakness to which sometimes the button callback does not identify that the second drop down item has been changed, hence performing a different conversion unless the second drop down menu is specifically chosen before the button is clicked. When a non-numeric input is entered, the program also throws an error and resets the input box to the last entered numeric input, so a further improvement may be to hold this input in the entry box instead of resetting it.

Finally, an improvement that could be made to the program would be to add more conversion options (such as Kelvin for temperature or nautical miles for length). Similarly, if enough case statements were written, the program could be extended to convert centimetres to meters, or inches to feet, hence converting between any possible unit of the same type. A final improvement which could be made, was to add a currency conversion section to the program, which is able to convert between different types of currency based on the real time conversion rates. This would require the program to access the internet to update the conversion rates in real time.