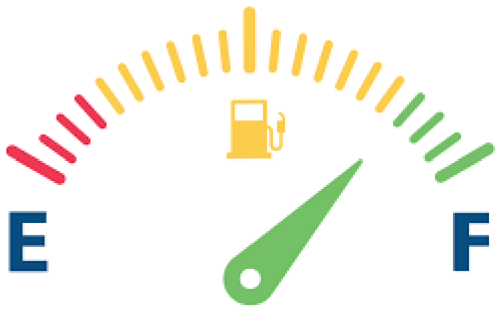
Programming Principles - JIT121 June 2019

First Assignment – Problem Solving Task
Fuel Consumption Program

Weighting 20% - Due date: 14 June 2019

Specifications



You are required to write a simple Console application to calculate fuel consumption of a vehicle based on the amount of fuel used and the distance travelled.

The amount of fuel input by the user will be in **litres** with a minimum value allowable by the program of 20 litres. The distance travelled input will be in **kilometres** with a minimum value that is at least eight (8) times greater than the absolute value of fuel used. For example if 30 litres is entered than the value entered as distance travelled will need to be at least 240km.

The actual fuel consumption is then expressed as **litres per 100 kilometres** (l/100km).

For example if 45 litres of fuel are used to cover a distance of 490 kilometres, the fuel consumption is approximately 9.18 l/100km:

litres per kilometre:	45 / 490 = 0.0918…
litres per 100 kilometres:	0.091 * 100 = 9.18...

Your program will need to also express the fuel consumption in the British Imperial measurement of **miles per gallon (mpg)**.

To convert from l/100km to mpg use the formula:

mpg = 282.48 divided by (l/100km)

For example using this formula, 9.18 l/100km is equivalent to 30.76 mpg:

282.48 / 9.18… = 30.76 (rounded to two decimal places)

Your program should consist of one class only containing *at least* 4 methods (not including **`Main`** or trivial **`void`** methods). The additional methods should be a mixture of value-returning and **`void`** methods. There is no upper limit on the number of methods you can use.

Design before Implementation

"The sooner you start coding … the longer it will take". As demonstrated in class, design your algorithm first. Refine each step of the algorithm until it is "almost", but not quite, code. Retain the algorithm in your solution file as comments. Once your project is complete, revisit the comments and remove any that add no value to your solution. What you should be left with is a well-documented, readable solution.

Functionality

Your program will prompt the user to enter the amount of fuel consumed (in litres). If a valid amount is entered (must be at least 20), the user will then be prompted to enter the distance travelled (in kilometres). Only distances of at least eight(8) times the absolute value of the fuel consumed should be accepted by the program. If an amount out of this range is entered, the user must be prompted to enter a valid distance. Once a valid distance is entered, the program should display a message including the fuel consumption in l/100km as well as mpg. The user will then be asked if another calculation is required, and if so, the whole process is repeated until no further calculations are required, at which point the program should exit.

Note that your program must also handle the user entering non-numeric values when numeric values are expected. You should make good use of the **TryParse** methods as demonstrated in the lectures.

See the Appendix for example screen shots. You are not required to follow exactly the same format for your console application. However, the functionality must conform to specifications described in this document.

Assignment Goals

The assignment problem is straightforward. The solution of the problem will be a C# project consisting of a single class which will use programming constructs covered in Lectures 1 & 2.

This assignment can be completed using only the concepts covered in Lectures 1 and 2. However, you are free to use any valid C# programming construct, even if it has not been covered in lectures - except for the `goto` statement. You must also avoid global variables (i.e. those declared at the class level). See the CRA document for more information.

Ensure that your code conforms to the Coding Style Guide discussed in class.

Though the code could be written as *straight line code* (ie entirely contained within a single method), the sheer number of lines would make the solution difficult to read, hard to understand, hard to modify and maintain, and much more likely to contain logical errors. **A straight line code solution will not receive full marks even if it fulfils the required functionality.**

The body of the `Main` method should simply resemble a high-level algorithm with minimum low level C# statements – just as you have been shown in Lectures, and have been asked to practice in Tutorials. There should be no input/output statements (i.e. writing to the screen or reading from the keyboard) in `Main`.

You are required to use methods (`void` and **value returning**) to provide a structure to your solution. Your program should consist of only one class containing at least four methods (excluding `Main` and trivial methods such as `ExitProgram` and `Welcome` as seen in Lecture 2). The additional methods should be a mixture of value-returning and `void` methods. There is no upper limit on the number of methods used.

As your code **must not** use global variables (variables declared outside methods), you will need to pass values between various methods (see Lecture 2).

Documentation

The class must be preceded by a comment containing your full name, student id and a short overall description of the program using block style comments. It should be placed between the namespace and class statements.

Each method must be preceded by a comment describing its purpose.

Any non-trivial code should also be commented – but only where it adds value to the readability of your solution. These comments should initiate from the design phase of this assignment (see section above regarding design before implementation).

Learning Goals

Sub-goals of the assignment are to experience:

- Top-down design & development
- Incremental Development & Incremental Implementation
- Translating simple algorithms into C# code
- Writing user defined methods
- Parameter passing
- Using `while` loops
- The mechanics of editing, compiling (building) and running your program
- Testing your program
- Becoming confident and comfortable with programming in the small

Academic Integrity

This assignment is for individual assessment only. That means the work you submit must be your own work and not the work of anyone else. You are not permitted to collaborate with your peers on this assignment, other than to discuss high-level strategies. You are not permitted to ask or commission someone else to write the assignment for you, or help you write the assignment.

If you are in any doubt as to what is permitted and what is considered a breach of academic integrity, please talk to one of the teaching staff as soon as possible.

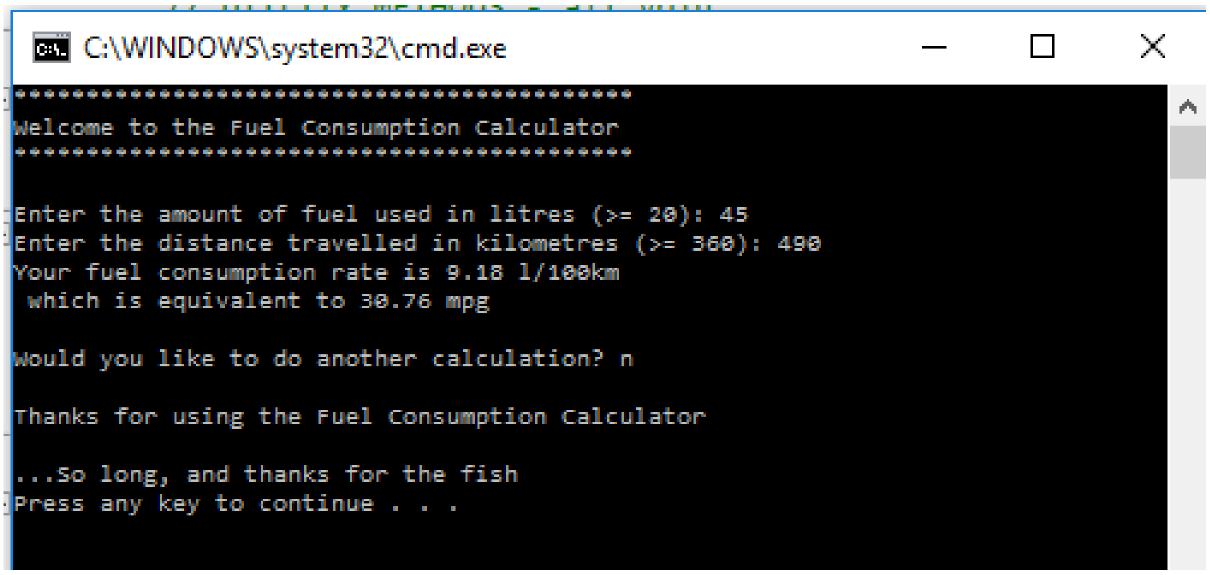
Final Comments

This assignment is not about screen design so do not waste time attempting to produce the fanciest looking screen output. Though you are free to alter and enrich the output statements it will not gain any additional marks. The basic console interaction as seen in lectures and in workshops is sufficient, as long as it is readable. There are no marks for “pretty” or ‘clever” screen interactions.

Plan to be finished well before the afternoon of the due date! You should avoid writing all of the code in one sitting and then attempt a build (compilation). Remember ***Incremental Development and Incremental Implementation.***

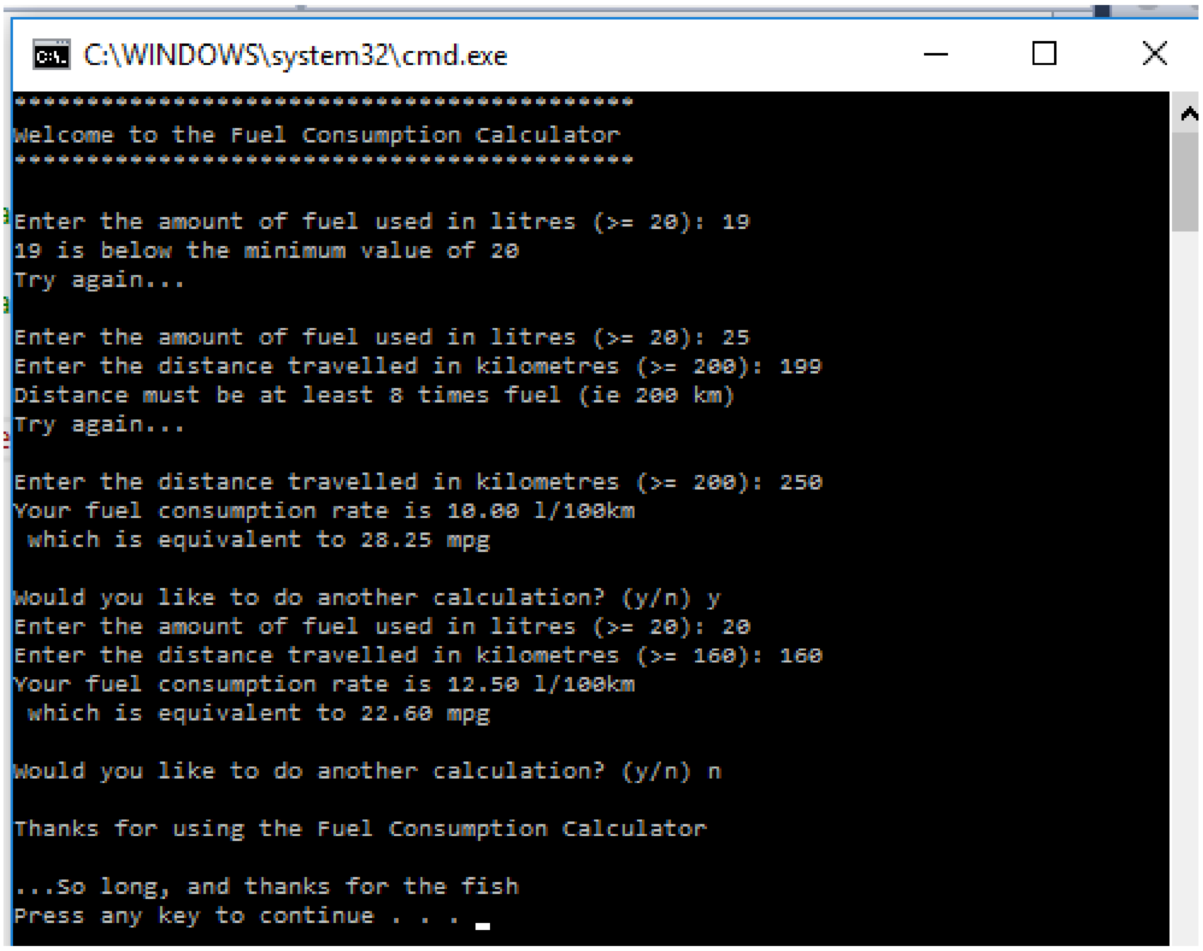
Enjoy the experience of this assignment!

APPENDIX



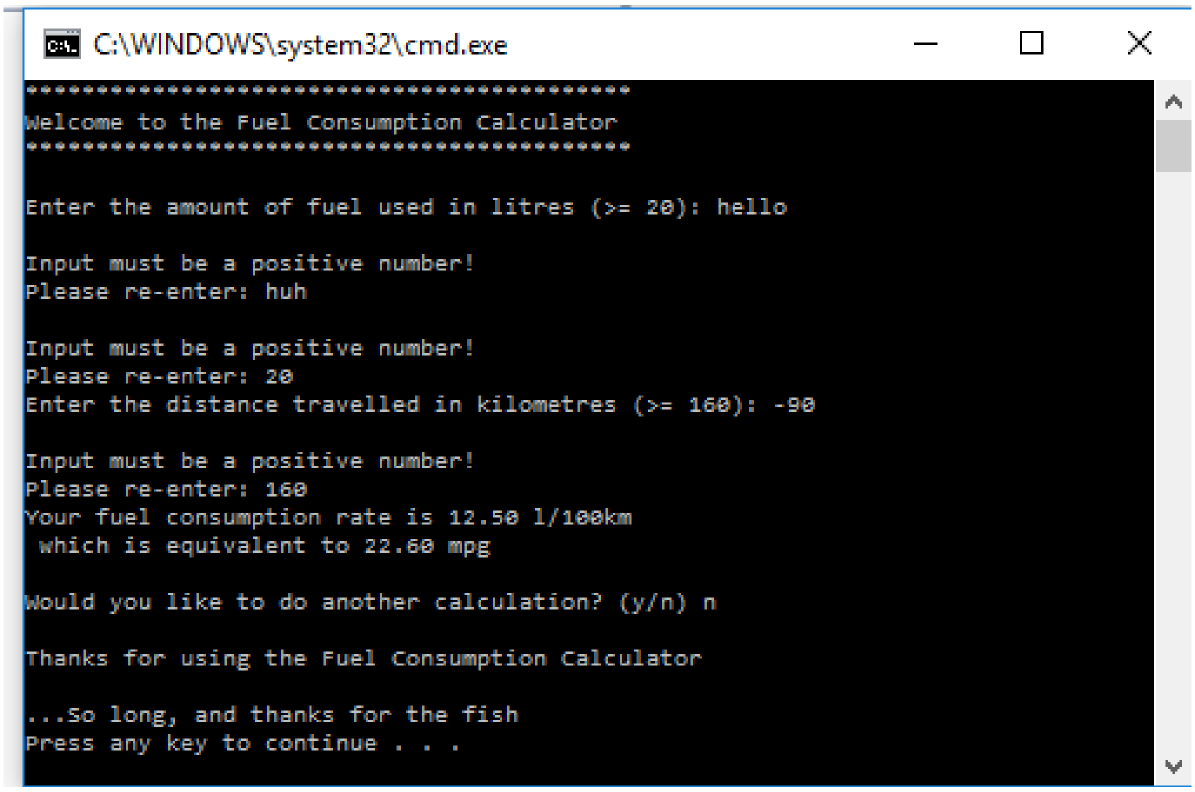
```
C:\WINDOWS\system32\cmd.exe
***********************************************
Welcome to the Fuel Consumption Calculator
***********************************************

Enter the amount of fuel used in litres (>= 20): 45
Enter the distance travelled in kilometres (>= 360): 490
Your fuel consumption rate is 9.18 l/100km
 which is equivalent to 30.76 mpg

Would you like to do another calculation? n

Thanks for using the Fuel Consumption Calculator

...So long, and thanks for the fish
Press any key to continue . . .
```

This screenshot shows a simple interaction: one calculation only, with valid input values.



```
C:\WINDOWS\system32\cmd.exe
***********************************************
Welcome to the Fuel Consumption Calculator
***********************************************

Enter the amount of fuel used in litres (>= 20): 19
19 is below the minimum value of 20
Try again...

Enter the amount of fuel used in litres (>= 20): 25
Enter the distance travelled in kilometres (>= 200): 199
Distance must be at least 8 times fuel (ie 200 km)
Try again...

Enter the distance travelled in kilometres (>= 200): 250
Your fuel consumption rate is 10.00 l/100km
 which is equivalent to 28.25 mpg

Would you like to do another calculation? (y/n) y
Enter the amount of fuel used in litres (>= 20): 20
Enter the distance travelled in kilometres (>= 160): 160
Your fuel consumption rate is 12.50 l/100km
 which is equivalent to 22.60 mpg

Would you like to do another calculation? (y/n) n

Thanks for using the Fuel Consumption Calculator

...So long, and thanks for the fish
Press any key to continue . . .
```

This screenshot shows handling numeric values below the specified minimums.


C:\WINDOWS\system32\cmd.exe

```
*******************************************
Welcome to the Fuel Consumption Calculator
*******************************************

Enter the amount of fuel used in litres (>= 20): hello

Input must be a positive number!
Please re-enter: huh

Input must be a positive number!
Please re-enter: 20
Enter the distance travelled in kilometres (>= 160): -90

Input must be a positive number!
Please re-enter: 160
Your fuel consumption rate is 12.50 l/100km
 which is equivalent to 22.60 mpg

Would you like to do another calculation? (y/n) n

Thanks for using the Fuel Consumption Calculator

...So long, and thanks for the fish
Press any key to continue . . .
```

This screenshot shows handling non-numeric input.