UNIVERSITY OF LIVERPOOL

Department of Electrical Engineering and Electronics

Introduction to Programming in C

(ELEC129)

Assignment 1

Objectives

To design, implement and document simple modular programs that use functions and flow control statements.

Assessment

This assignment is an **assessed component** and the mark for this work will contribute towards the **overall module mark**. **The weight of Assignment 1 is 12%**. The marking criteria can be found in the *Exam Resources* section of ELEC129 in VITAL.

This assignment is composed of a number of exercises. The relative weight of each exercise on the overall assignment mark is indicated between brackets.

Instructions

Students are required to **do all exercises** and **submit a single Word file** in the Assessment section of ELEC129 in VITAL (https://vital.liv.ac.uk) by **Tuesday 12/11/2019 at 17:00 (5pm) UK local time** (week 8 of semester 1). Delay penalties apply. Please double check your report and make sure your work is in its final form before submission as **the online application will not allow resubmissions**. **Email submissions and/or resubmissions will not be accepted.**

Important: grading of all coursework is anonymous, therefore do not include your student name, student ID number or any other personal information in the submitted report (reports containing personal details will be penalised). Submissions must be a single Word file containing the work done by the student. The file must contain **two parts for each exercise** proposed in this assignment as detailed below.

Part I: The first part must contain the source code. Please use font `Courier New` with a size of 8 points, use **indentation** to make the code readable, and observe the following requirements:

- The source code must be the result of your own original and individual work.
- The source code must be entirely written in the C programming language. Source code written in any other programming languages (e.g., C++) will receive a mark of zero.
- The use of global variables (variables declared outside the body of a function) is not needed in ELEC129 assignments and its use is not allowed unless otherwise stated. All variables should be local to a function (declared within the body of a function). The use of global variables will be penalised. If you are not sure about what a global variable is, please ask a lab demonstrator or the course instructor for clarifications.
- All exercises can be solved based on concepts explained in previous lectures. Students are allowed to use concepts from other (future) chapters but this is not expected. Feel free to discuss your approach to solving the exercises with a lab demonstrator.

Part II: The second part must contain a detailed explanation of the software development process followed by the student (i.e., the first five steps of the software development method):

1. *Problem specification*: Formulation of the problem and its objectives.
2. *Analysis*: Identification of: i) inputs, ii) outputs, and iii) other relevant aspects, requirements or constraints (e.g., relevant formulas, etc.).

3. *Design*: Formulation of the algorithm (list of steps) needed to solve the problem. At this stage, the problem should be divided into a number of sub-problems that can be solved using functions.
4. *Implementation*: List of functions used in the program (function names), indicating for each function which step(s) of the algorithm is/are implemented by the function.
5. *Testing and verification*: Explanation of how the program was tested and verified. Snapshots of the program's output window can be obtained by using [Alt] + [PrtScr(or PrintScreen)] and pasting into the report.

Please indicate clearly if in your opinion the program works correctly. If you do not think the program works correctly or does not compile, indicate what the problems are. You will then be able to get points for that programming task. If in the testing section of your design document you have indicated that the program works correctly but it turns out that your code does not even compile (because of syntax errors, for example) you will not receive adequate points.

Academic integrity

Students should familiarise themselves with Section 6 of the University's Code of Practice on Assessment, which provides important information regarding the submission of assessed coursework (https://www.liverpool.ac.uk/aqsd/academic-codes-of-practice/code-of-practice-on-assessment).

Students should also familiarise themselves with Section 9 (Academic Integrity) and Appendix L (Academic Integrity Policy) of the University's Code of Practice on Assessment which provide the definitions of academic malpractice and the policies and procedures that apply to the investigation of alleged incidents (including collusion, plagiarism, fabrication of data, etc.).

Students found to have committed academic malpractice are liable to receive a mark of zero for the assessment or the module concerned. Unfair and dishonest academic practice will attract more severe penalties, including possible suspension or termination of studies.

By electronically submitting this coursework you confirm that:

- You have read and understood the University's Academic Integrity Policy.
- You have acted honestly, ethically and professionally in conduct leading to assessment for the programme of study.
- You have not copied material from another source nor committed plagiarism nor fabricated data when completing the attached piece of work.
- You have not previously presented the work or part thereof for assessment for another University of Liverpool module.
- You have not copied material from another source, nor colluded with any other student in the preparation and production of this work.
- You have not incorporated into this assignment material that has been submitted by you or any other person in support of a successful application for a degree of this or any other University or degree awarding body.

Students are encouraged to contact the module instructor if any clarifications are needed.

Exercise 1 (40% of assignment mark)

Write a program that reads three characters from the keyboard and prints on the screen the approximate shape of a pyramid using those characters as shown below. For example if the characters are '%', '#' and '@', then the program should display the following:

```
     %
    %#%
   %#@#%
  %#@@@#%
 %#@@@@@#%
%#@@@@@@@#%
```

Your program should include:

- A function `info` that displays instructions to the user, reads the three characters from the keyboard and returns those values to the calling (`main`) function.
- A function `pyramid` that takes three parameters of type `char` and prints on the screen the approximate shape of a pyramid using the characters provided as the actual arguments. This function should also be called from within the `main` function.
- The `main` function should call first the `info` function to instruct the user to input three characters and read those characters from the keyboard. The `main` function should receive the three characters returned by the `info` function and then pass these characters to the second function `pyramid` to print the pyramid shape.

In test part of your report, include at least three different test cases (with different characters).

Exercise 2 (60% of assignment mark)

You have been hired by the IT department of a freight company and your first task is to create a program to compute the total cost of a journey, which includes the cost of fuel consumption plus the driver’s salary, based on the following considerations:

- The company has 2 types of vehicles:
  - Type A: Car with the following fuel consumption in miles per gallon (mpg):
    - In built-up area: 40 mpg
    - In single carriageway: 50 mpg
    - In dual carriageway/Motorway: 60 mpg
  - Type B: Car with the following fuel consumption in miles per gallon (mpg):
    - In built-up area: 45 mpg
    - In single carriageway: 55 mpg
    - In dual carriageway/Motorway: 65 mpg
- Costs are calculated using the following fuel prices:
  - Unleaded (for Type A cars): 150 pence per litre (£1.50/litre)
  - Diesel (for Type B cars): 130 pence per litre (£1.30/litre)
- The following conversion factor applies: 1 gallon = 4.54609 litres (imperial gallon).

- The driver's remuneration for a journey is computed based on fuel cost as follows:
    - For the first £10 of fuel, the driver receives £1 per each pound spent in fuel.
    - For the next £20 of fuel, the driver receives £2 per each pound spent in fuel.
    - For the next £30 of fuel, the driver receives £3 per each pound spent in fuel.
    - For the rest of fuel in excess of £10+£20+£30=£60, the driver receives £5 per each pound spent in fuel.

Write a program that computes the **fuel cost**, **salary cost**, and **total cost** of a journey based on the employed car type and the distances travelled in built-up areas, single carriageways and double carriageways/motorways. All costs should be displayed with 2 decimal digits.

In the test part of your report, indicate the results provided by your program for these cases:

- Case I:

Car type:	A
Distance in built-up areas:	10 miles
Distance in single carriageways:	15 miles
Distance in double carriageways/motorways:	25 miles

- Case II:

Car type:	A
Distance in built-up areas:	80 miles
Distance in single carriageways:	30 miles
Distance in double carriageways/motorways:	20 miles

- Case III:

Car type:	B
Distance in built-up areas:	20 miles
Distance in single carriageways:	50 miles
Distance in double carriageways/motorways:	500 miles

- Case IV:

Car type:	B
Distance in built-up areas:	50 miles
Distance in single carriageways:	100 miles
Distance in double carriageways/motorways:	1200 miles