UNIVERSITY of York

Department of Electronic Engineering

ELE00107M C Programming for MSc Coursework Assessment 2021/22

SUMMARY DETAILS

This coursework **(Code and Report)** contributes **100%** of the assessment for this module.

Clearly indicate your **Exam Number** on every separate piece of work submitted.

Submission is via the VLE module submission point. **The deadline is 12:00 noon on 26 January 2022, Spring Term, Week 3, Wednesday.** Please try and submit early as any late submissions will be penalised. Assessment information including on late penalties is given in the Statement of Assessment.

ACADEMIC INTEGRITY

It is your responsibility to ensure that you understand and comply with the University's policy on academic integrity. If this is your first year of study at the University then you also need to complete the mandatory Academic Integrity Tutorial. Further information is available at http://www.york.ac.uk/integrity/.

In particular please note:

- Unless the coursework specifies a group submission, you should assume that all submissions are individual and should therefore be your own work.
- All assessment submissions are subject to the Department's policy on plagiarism and, wherever possible, will be checked by the Department using Turnitin software.

University of York

Department of Electronic Engineering

C Programming for MSc (ELE00107M)

Assignment 2021/22

Introduction

It is important for engineers to follow a structured process when developing software. A structured approach to software engineering helps to ensure that the software produced is of high quality. An important part of this approach is the careful documentation of each stage of the process in the form of a report. This assignment gives you practice in software development and report writing for software engineering. Treat the task you have been given as a set of requirements; you could imagine that they have been given to you by a client. If you feel that the requirements are unclear, you may make realistic assumptions as necessary; however, you must make sure that you document these assumptions carefully.

You are required to complete the task using the C Programming Language for a PC running the Microsoft Windows operating system (i.e. the language and type of computer that you have been using in the laboratories).

Assignment Structure

IMPORTANT: We have a policy of marking anonymously. Do not identify yourself in your report or your programs. Use only your student Examination number. Make sure that your examination number is printed on your report (first page).

Your report should document the development process that you have been through, leading up to the creation of a piece of software. This should correspond to the development model that you have covered in the lectures and is detailed in the labscripts. Your report should consist of no more than 12 pages (minimum font size 10 point), including diagrams and figures.

Your submission should contain:

1. The **Requirements** that were given to you;
2. Your **Analysis** of the problem, and any assumptions you feel you will have to make;
3. A **Specification** for your software that you have derived from your analysis of the requirements;
4. Documentation of the **Design** process you have used.
5. Documentation covering how your program is structured into source files and any changes you made from the design as originally stated in your specification. You should justify any changes to your design that you made during the implementation. This should take the form of an **Implementation Report**.

6. Documentation of the **Testing and Verification** process that your went through to ensure that your program executes as you required;
7. A simple **User Manual**.
8. Source **programs**.

Submitting your work

You are expected to submit your work electronically using the web links provided on the course web page. You must use the assignment project available from the course web page (see the section SUBMITTING YOUR ASSIGNMENT).

You will be expected to link your source code files (files ending in .c and .h) with the assignment project. An explanation of how to do this is given on the course web page under the heading BLANK ASSIGNMENT PROJECT. It is **your** responsibility to ensure that you have submitted all the files your program needs to be successfully compiled into an executable program. If your program cannot be compiled you may receive no marks for program execution.

Your assignment project MUST work in Windows.

Your assignment report must be a .pdf file. Please place this in the folder containing your assignment project (i.e. it should be located in the same directory as the file assignment.cbp). Once you have tested your program and have finished your report make sure everything is in your assignment folder. Please zip up the folder containing your project and report. Your zipped file should be called assignment.zip.

Please submit your work in good time before the deadline to ensure that it is uploaded before the deadline (note the server might be busy very close to the deadline).

If you submit multiple versions of the assignment project, the markers will only take the last submitted version. If this version is submitted after the deadline then marks will be deducted for lateness.

For your guidance, marks will be awarded as follows:

- Requirements, Analysis and Specification — *15%*;
- Design — *20%*;
- Implementation of Program — *25%*;
- Testing and Verification and User Manual — *12%*;
- Maturity, consistency, presentation and innovation — *13%*;
- Execution of program — *15%*.

(Total 100%)

Important

You should be aware of the University regulations on academic misconduct before completing this assignment — any failure to correctly acknowledge work of others will be regarded as academic misconduct. This includes the use of solutions provided by module/lab leaders. Thus, if you use these, you must make clear which parts of your programs were copied or adapted from other sources and where these sources can be found (i.e. the web address). Comments should be made in your C code about this.

Introduction

The laboratories introduced you to some simple two-dimensional graphical output based on basic physics. In this assignment you will take the principles you have met in the labs and develop them to create a playable computer game.

The Task

You are required to design and implement a computer game which is based on the javelin throw track and field event (see: https://en.wikipedia.org/wiki/Javelin_throw). As the player, you control the force and elevation with which the javelin is projected across the field, after gaining momentum by running within a predetermined runway area. The aim is to throw the javelin as far as possible, accounting for other factors, such as the effects of wind. The game must take account of the following:

- the flight of the javelin must be subject to the effect of gravity;
- the distance the javelin is thrown is used to calculate a score;
- that the javelin was thrown whilst the thrower was still within the predetermined runway area – otherwise the throw is invalid;
- that the user can control:
    - the speed at which the thrower runs
    - the angle at which the javelin is thrown
    - when to release (throw) the javelin
- the game should keep a score of play over 6 throws;
- incorporating sound into the game;
- making use of both the mouse and keyboard to control the game.

The game must be enjoyable and playable for someone who is not experienced with computers but rewarding and challenging.

You might like to think about:

- adding a multi-player option;
- adding a simulated (computer) competitor.
- ensuring that the game is not too frustrating to play — for example, give the player a practice session and give hints on how to improve the throw.

Think about how to incorporate sound into your game. To achieve this, you can use the functionality provided by the amio lib. In particular, you could use a call back design to supply real-time audio to your application (e.g. sound of the wind, the flight of the javelin and cheers of the crowd). Some good examples on how to achieve something like this can be found in the audio example files (audio01.c and other examples).

