ACS130 Introduction to Systems Engineering and Software
C Programming Project

Tara Baldacchino (t.baldacchino@sheffield.ac.uk, Room D13, AJB)

Assignment weighting: 25% of module mark

Assignment released: Friday 22 March (Semester 2, week 7)

Assignment due: 11.59pm Wednesday 1 May (Semester 2, Week 10)

Submission (all dropboxes are in the 'Assessment' tab):

1. Submit your .c file to the submission dropbox entitled "C Program .c file". Do not copy your .c file into any other format. Do not submit .exe files to the dropbox.

2. Submit any text files that are needed to run your code to "Supplementary material".

Please make sure your code runs and it must run on Codeblocks or XCode if using a Mac.

Marking and Feedback: You will be required to attend a 1-1 viva to demonstrate your code. **We will download your code from Blackboard to run.** You will get oral feedback during this marking session. A day and time will be provided for everyone closer to the time (in weeks 11 and 12). Provisional marks may change as a result of unfair means, moderation from the module leader and/or late submissions. **If you don't turn up to the marking session, then your code will not be marked.**

You will be given a preliminary mark during the viva session. All code and marks are then reviewed by me and your mark may change.

Assignment briefing: This assignment will assess your ability to solve a problem using C as a tool. Your program must be well laid out and commented, including a meaningful header comment. Your program **must** provide meaningful interactions with the user. You must also use all of the following C programming elements:

- selection (if or switch case),
- repetition (for or/and while or/and do while)
- at least one function with **an array of struct** in the input parameter list; the array of struct is declared in main (or another function) and this function must set values in the array of struct, and the calling function (main or other) must make use of the array of struct on return (you choose whether to pass by value or pass by address)
- at least one function with a pointer to a variable of type char, int, float or double in the parameter list; the function must set the contents of the variable, and the calling function (main or other) must make use of the variable on return
- your program needs several instances where the user must input commands from the keyboard in order to interact with the program
- defensive programming: guarding against incorrect entries and incorrect types
- your program must loop to the start allowing the user to run through the program time after time (this is an independent requirement from the second point to use repetition)

Optional:

- you can create other functions and/or structures so as to modularise your code (but you must have the 2 functions mentioned above).
- you can use anything else we have covered in this module, eg, arrays/strings, an input and/or output file (and checking to see if the file exists)
- you can use any pre-defined header files in your code

You need to decide on what problem to solve in your program. Do NOT use one of the problems from the ACS130 lectures, labs, previous assignments or past papers.

The 2-3pm sessions in weeks 8&9: I will be available if you want to discuss your problem with me. During this session I will not be looking at individual code or helping you with debugging, but I will be available to answer general questions or clarify certain concepts with respect to lecture slides and tutorial questions.

Instructions to the screen need to be clear so that I can successfully understand how your code runs!

Component	Marks	
Does the program use any **global variables**? Does the program use **while(1) loop/s**? Does the program use break (other than in switch case), go to, jump, continue etc	Yes/no (If yes, 50% penalty)	
Program layout and commenting: • Meaningful header comment (including a **detailed description of code**) • sufficient comments interspersed within code • Good indentation • Meaningful variable names	-2 marks if **any** missing	
Are function prototypes listed at start of the program? It allows me to check you have satisfied the requirements, as well as efficiency of the code	-2 marks if missing	
Only one .c file must be submitted- no user defined .h files (you can submit .txt files in supplementary materials)	-5 marks if code is over several files	
Marks available for executing program only:		
Is there sufficient interaction between user and program? Yes/No **Interaction with the user** **The user needs to be able to interact several times with the program.** • Is the user given prompts to enter any information that is needed from the keyboard? • Is the information displayed to the user easy to understand, including any required ranges? • Does the program loop to start?	If No, 0 for this section /2	
Use of function(s), with array of struct in parameter list: Does the program use at least one function with an **array of struct** included in the parameter list? Eg **struct Books B[3]** Yes/no • Declaring the array of struct in main or other calling function (**not** in the called function). • Setting some values within the array of struct from inside the function. • Using (on return from the function) values in the array of struct in main or other calling function. **-> notation as an alternative to dereference and dot notation is allowed.**	If No, 0 for this section. /3	
Use of function(s), with pointer variable in parameter list: Does the program use at least one function with a pointer to a variable (not an array) of type char, int, float, double, **struct (not the same as previous function)** in the parameter list? Yes/no • Declaring the variable in main/calling function (not in the called fn) • Setting the contents of the variable from inside the function. • Using (on return from the function) the variable in main or other calling function.	If No, 0 for this section. /2	
Does the program work with a variety of disparate inputs, without malfunctioning? (i.e. is code debugged and tested thoroughly) **Please test the following:**	If No, then 0 for this section.	

• Inputting out of range numbers (including boundary values, massively too small/large, zero, negative/positive, entering a char instead of an int) at every instance • Inputting the wrong type, eg, it asks for an int and you supply a char • If using an input file: Program must inform user and exit if file not found. In each case, the program should allow re-input, or a message should be given to alert the user to problems such as missing files.	/1	
Complexity of code* (marks only awarded if repetition, selection and array of structs with more than one member used)	/10	
Efficiency of code**	/5	
Explanation of code	/2	
Is IDE Codeblocks, DevC++ or Xcode? If not, -5 marks.		
Do we need to use an alternate file (ie, not the .c file submitted on Blackboard) for marking, Or make any changes to your program to get it to run?	If yes, -5 marks.	
Total mark	/25	

*Complexity of code: Out of 10 marks.

To get complexity marks you need to have used:

- **Repetition**
- **Selection**
- **Array of structs with more than one member**

in your code, otherwise you will get 0 for this part of the marking scheme.

0 marks: Your code is very simple/basic and mainly consists of scanf and printf statements.

Up to 3 marks: **Basic algorithms/control sequences:** Eg some basic algorithms eg least common multiple, greatest common divisor, calculating number of combinations and permutations, or multiple choice quiz

Up to 6 marks: **Medium complexity algorithms/control sequences:** Eg shortest path from source to destination in a 2D maze , root to leaf path for a variable tree (please note that the example shown is for a static tree), extensive analysis of 2nd order systems (inc TF, pole-zero, variable gain and plots), tic-tac-toe (also known as noughts and crosses), snake game, a simple game using lots of graphics.

Up to 10 marks: **Complex algorithms/control sequences:** Eg complex simulated games with a few rules eg Pacman, battleships, or make one up! Complex control paths eg A* and Djikstra. Complex algorithms eg training a neural network with multiple layers.

**Efficiency of code: Out of 5 marks

This is looking at things like: using loops, arrays/strings if appropriate, modularity- adding other functions to the code other than the ones mentioned in the briefing. Is the code compact? Adding graphics to files.

Penalties for late submission: 5% reduction in the mark for every day or part thereof that the assignment is late, and a mark of zero for submission more than five days late.

How you should work: **This is an individual assignment**, and it must be wholly your own work. You must not copy algorithms or code for this assignment from any other source. You must not show anyone else any part of your code (small or large), nor must you look at any part of anyone else's code, because if you do this is likely to result in similarity in your algorithms and code. Do not post any question relating to this assignment on any forums eg stack exchange or CSDN.

Unfair means: Any suspicions of the use of unfair means, alerted to the module leader either by the marker or by the plagiarism checker, will be investigated and may lead to penalties. See unfair means for more information.

Extenuating circumstances: If you have medical or personal circumstances which cause you to be unable to submit this assignment on time, please complete an extenuating circumstances form.

Help: This assignment briefing, the lectures and labs and the C online resources on your ACS130 reading list provide all the information that is required to complete this assignment. You need to decide on the algorithm to solve this problem, subject to the instructions in the assignment briefing. You also need to create and debug the C code to solve this problem. However if you need clarifications on the assignment then please **post a question on the ACS130 Blackboard discussion board**. **I will not reply to assignment related emails unless the query is of a personal nature. The discussion board is an opportunity for everyone to see the Q&A so that no one is disadvantaged.**