LLP109 Coursework - *App development*

Develop your own electronic **'bibliographic information data management system' in either Python or C.**

Assignment: Write a complete, well-structured program that begins by asking the user to enter bibliographic information of library items, such as books, CDs, photographs. The program will produce a database with the input values and display the stored ones to users.

You must use:
when coding in Python: Data Structures, Loop, Conditionals, user defined Functions, and ***Data (keyboard) Input/ (screen) Output***,
or, ***in C: Pointer, Array, Structure, Loop, Conditionals, user-defined Functions,*** and ***Input/ Output***.

Therefore, you should ask yourself how you can incorporate the topics you learnt in the module into this specific project. Be sure to write your **name** and **student number** in the upper right-hand corner of the code. Use **comments** to explain intended meaning, reason, and functions of each part in the program. This will be very helpful for readers to understand your codes.

Directions: Above all, think of the general architecture of the code. Think of a *Data Structure* that includes bibliographic data sets as its elements, such as a title, author, year, identification number, page number, and so on. Your code should be able to display all the data sets on a screen after entering the input data. The code should have an input and output (display) functions that defined by you.

Hint: for example, a sequence of instructions may *loop* endlessly in the code, unless the user chooses terminating condition. Or, you can give the finite appropriate number for the loop. Declare and define 'Insert' and 'Display (Show)' functions. Define a data structure, e.g. *list*, having multiple elements of *lists* in Python. Instead of a *list*, alternatively *class* can also be used in Python (*structure* in C). Think of an idea to append a data structure element to the main data structure. Note, a small data structure can be an element of a bigger main data structure, e.g. *nested list* in Python (refer to the lecture notes). Remember how to access each data at different levels (dimensions), which is explained in the lecture notes. The total number of the nested data structures corresponds to the number of bibliographic items, e.g. books. A few items can be all right in order to demonstrate the functionality of your code.
A user should be asked to choose an operation out of multiple options, such as *(1) Data input, (2) Display the data sets, (3) Finish and exit* and so on. When *(1)* is chosen, the code calls the *'input'* function and the user put bibliographic information for each library item. After entering all the data of a book, the user can make a choice out of the three (or more than three) options again. Therefore, a loop and conditionals are required in the main algorithm. For example, a user can see the data sets stored in the data structure so far when *(2)* has been chosen, finish the job and exit the entire program when *(3)* has been chosen, or carry on with the data input job with the option *(1)*. It is necessary to write a code **preventing (unintended, overflow, memory, wrong input data type) errors. i.e., exception/error routine**, when a user tries to make wrong attempts, e.g. to try putting more data than the storage size of an array.

Start early
Programming is not an activity that can be reliably scheduled to take a predetermined amount of time. The best thing you can do for yourself is to 'get started right away'. Refer to recommended on/offline literature on coding, especially sections about Loop, Data Structure (List, Tuple in Python, Struct in C), Input/ Output, Conditionals, and User-defined Function.