Faculty of Natural Sciences

School of Computer Science and Mathematics

Assessment Brief

Module	CSC-40044 System Design & Programming
Assessment Component	Coursework
[redacted]	
Deadline	
Module Leader	Dr Alastair Channon
Office Hours/Meeting Booking Link	See the module's KLE entry

What is the task for this assessment?

The Keele University Library Map[1] includes a list of classmarks for each subject, and shows the location of each classmark. For the purpose of this assessment, assume that all classmarks are either one or two letters long, and that a range such as "G to GF" consists of (for this example) G, GA, GB, GC, GD, GE and GF. Consider there to be six locations containing books: Ground Floor, Middle Floor, Top Floor Back Left, Top Floor Back Right, Top Floor Front Left and Top Floor Front Right.

Part I: Develop a text-based application (without a GUI) that 1. prompts the user (with clear instructions displayed) to select one of three options: whether to enter a subject name or part-name (e.g. "English"), a classmark, or a location; 2. prompts (according to the previous response) the user to enter the subject name/part-name, classmark or location (with location options displayed); and 3. outputs each matching classmark together with its location and subject name(s).

Your program should read in two comma-separated values (CSV) files that you write. The first file should contain the list of subjects and classmarks. The second file should contain a list of classmarks and locations; you can design this file either to contain one classmark per row or a range of classmarks per row. Your program should be able to operate correctly if subject names, classmarks or locations are changed within, added to or removed from these files.

Part II: Develop a second application (for the same task) that uses a TKinter-based graphical user interface (GUI) for both input and output, rather than the text-based console. Minimise the amount of repeated code by separating code common to both part I and part II from user-interface-specific code.

For both parts, you may only import modules that ship with a basic Python 3 installation. This can be checked at https://docs.python.org/3/py-modindex.html. Submit a single .zip file, containing your Python .py files and CSV files, to the KLE drop-box for this assessment.

1 A copy of the Keele University Library Map is included on the final page of this document.

2. What is required of me in this assessment?

Guidelines	You should complete your work for this assessment without help or feedback from others, including demonstrators. All work that you submit must be your own individual work, and any other person's work or ideas must be appropriately acknowledged. Further details on this can be found in the University's Student Academic Misconduct Code of Practice. The module co-ordinator will provide feedback on the work that you submit, via the KLE, only once all marking for this assessment has been completed. You can however get help with non-assessment material by making use of the practical sessions timetabled in week 6 (with demonstrators in them) and the weekly virtual help desk sessions. For example, if you are stuck on something to do with the assessment, you might find and try a practical exercise that does a similar thing (e.g. read a CSV file) and get help with that if you need it.
Self- assessment checklist **Make sure that you…**	• address each requirement listed in section 1; • consider your work against the mark scheme; • submit (only) a single .zip file, containing your Python .py files and CSV files, to the KLE drop-box for this assessment.
Three key pieces of advice based on the feedback given to the previous cohort who completed this assignment	• Do not import a module that does not ship with a basic Python 3 installation. • Minimise the amount of repeated code by separating code common to both part I and part II from user-interface-specific code. • Test your code and evaluate the extent to which your applications meet the requirements: each should function as specified in part I.
Assessment Criteria/ Markscheme:	***Part I: Design, manual writing and code for reading in of data (.csv) files (10%)*** ***Part I: Remainder of text-based application (20%)*** ***Part II: GUI design and implementation (20%)*** ***Code quality, including error/exception handling and internal comments (20%)*** ***Overall design of the applications (30%)*** Marks here will be awarded for accordance with the design principles established within the course. In particular, the quality of partitioning and the design goals of Flexibility, Extensibility, Maintainability, Robustness, Performance and Usability. Note that this need not necessarily imply an object-oriented design. Appropriate levels of abstraction, partitioning, decoupling, and maximisation of functional cohesion, should be the driving forces. Within each of the above categories, marking will be according to the University's generic FHEQ level 7 assessment criteria. The pass mark is 50%.

3. What is the purpose of this assessment?

The following table shows which of the module learning outcomes are being assessed in this assignment. Use this table to help you see where and how to transfer feedback from one assignment to another. Note that your feedback may mention some of these outcomes, but that you will not receive a 'mark' against each one.

Module Learning Outcomes assessed
• critically appraise systems design options and select an approach appropriate to a provided set of requirements • develop designs for computer programs on the basis of a set of requirements • develop computer programs using a major programming language and appropriate technology • critically evaluate the extent to which computer programs meet their requirements

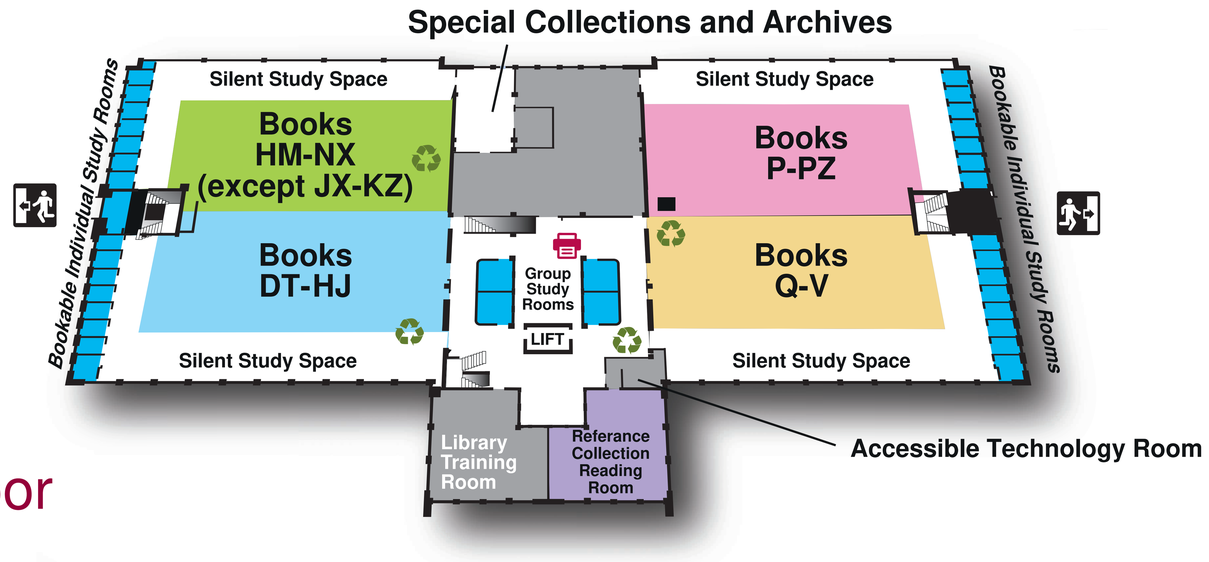
4. What resources might I use to get started?

If you need to study the module material again, to prepare yourself for this assessment, I suggest focussing first on the "Programming in Python" lectures and practicals, which cover:
1: Basic data types in Python, functions, conditions, loops, files
2: List Comprehensions, Dictionaries, Classes and Objects, Inheritance, Modules, Errors and Exceptions, DUNDER Methods, Iterators
3: Generators, Operator Overloading, Recursion, Linked Lists, Stacks, Queues, Sorting Algorithms, Big-O performance
4: Trees, (Priority Queues), Heaps, Graphs, GUI

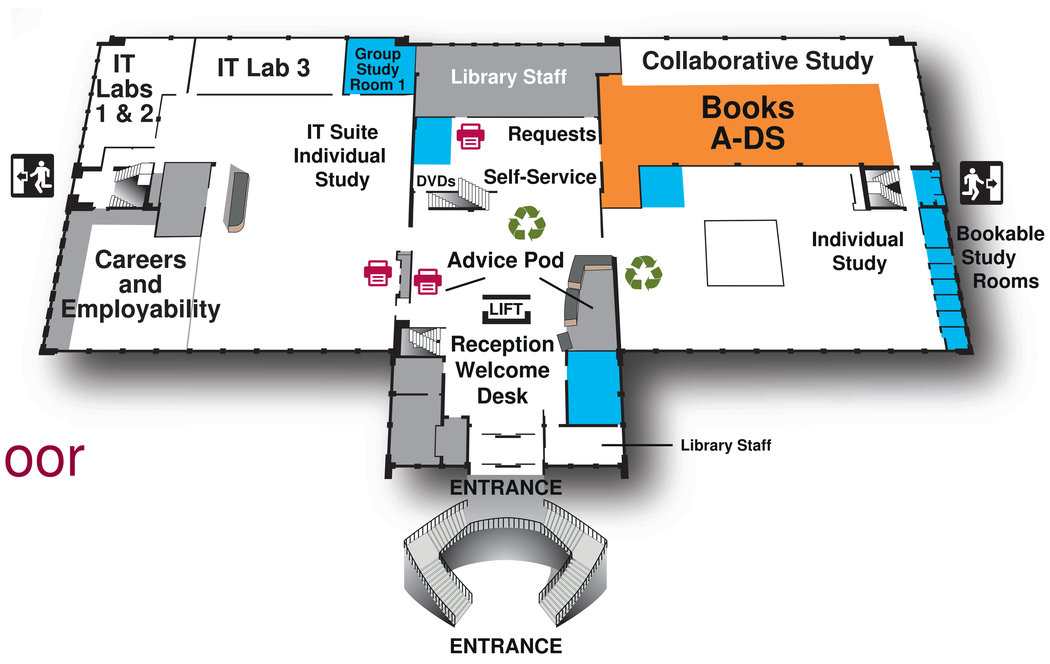
I also suggest that you not distract yourself with other resources. Doing so tends to lead students not familiar with programming into confusion rather than helping you.

Library **Map**

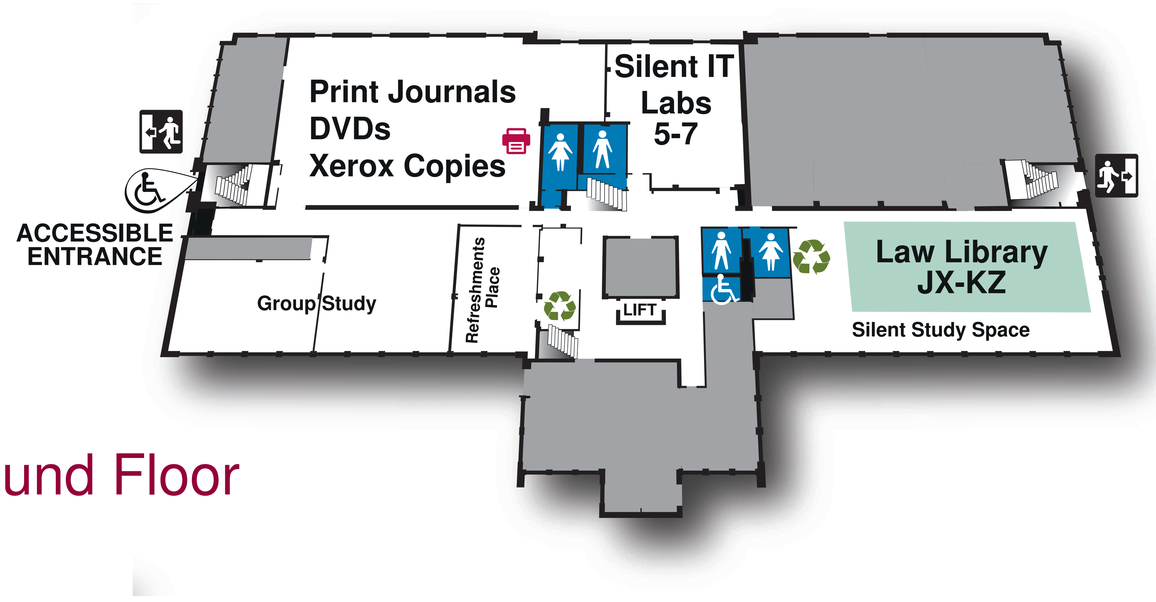
Top Floor



Middle Floor



Ground Floor



Toilets Recycling point Printer

Subject Classmark **References**

Accountancy	HF
American History	E and F
American Literature	PS
Archaeology	CC
Art	N
Astrophysics	QB
Biochemistry	QP and RB
Biology	QH
Biomedical Science	QR and RB
Botany	QK and S
British History	DA
Business	HD to HF
Chemistry	QD
Chinese	PL
Computing	QA and TK
Counselling and Psychotherapy	RC
Criminology	HV
Ecology	QH
Economics	HA to HJ
Education	L
English Language	PE
English Literature	PR
Environmental Studies	GE to GF
Film Studies	PN
Finance	HG to HJ
Forensic Science	HV
French	PC
Geography	G to GF
Geology and Geoscience	QE
German	PF
History	D
Human Biology	QM to QR
Human Resource Management	HD to HF
International Law	JX
International Relations	JZ
Italian	PC
Japanese	PL
Law	K
Management and Marketing	HD to HF
Mathematics	QA
Media Studies	P
Medicinal Chemistry	QP
Medicine	R
Mental Health	RA to RC
Music Technology	M T
Music Technology	ML
Neuroscience	RC
Nursing	RI
Pharmacology	RM
Pharmacy	RS
Philosophy	B to BJ
Physics	QC
Physiology	QP
Physiotherapy	RM
Politics	J to JX
Psychology	BF
Radiography	RC
Religion	BL to BX
Russian	PG
Social Work	HV
Sociology	HM to HT
Spanish	PC
Veterinary Medicine	SF