# Database systems and computer programming

## Prof. Jianyi Lin

***COURSE AIMS AND INTENDED LEARNING OUTCOMES***

The increasing application of computational methods for data analysis requires a thorough understanding of the technical tools to devise effective software for handling data. This course aims to provide students with a solid grounding in the principles of computer programming using the modern Python language, as well as an introduction to the formal and technological framework for modelling, storing and extracting information by means of the well-established relational data model and newer semi-structured data storage systems. Throughout the course, analysis of applications and examples will foster the computational problem solving skills in data representation and processing.

At the end of the course, the students will be expected to achieve the following course learning outcomes:

* understand the fundamental constructs of computer programming under the structured and OOP paradigms, such as flow control structures, functions and objects; demonstrate knowledge of the relational data model and key SQL concepts (*Knowledge and understanding*)
* understand, elaborate and implement programming and database solutions to the given requirement specifications in a particular language; improve problem solving skills that demand computational thinking (*Applying knowledge and understanding*)
* recognise, distinguish and choose good coding practices and more effective implementations in programming and database operations; identify elements in the relational data model that are appropriate for describing facts (*Making judgements*)
* express the purpose of some code using computer science terminology; properly justify the implementation and outline the program or data architecture with a basic engineering approach (*Communication skills*)
* autonomously refer to authoritative documentation of the language and framework adopted; recognise language/formalism independent constructs by abstracting from their concrete usage (*Lifelong learning skills*)

***COURSE CONTENT***

1. Introduction to modern computer architecture and programming
2. Python environment and simple programs
3. Fundamental constructs in structured programming: expressions, variables & data types, input & output, conditional branching, definite (for) loops, conditional (while) loops, loop-based patterns
4. Functions: definition, formal and actual arguments, variable scope, return values
5. Sequences and dictionaries; mutability
6. Elements of object-oriented programming: objects, classes, encapsulation, inheritance
7. Applications to mathematical and textual examples; first libraries for data science
8. Introduction to database systems and relational data model: relations, tuples, attributes, keys
9. Relational algebra: selection, projection, Cartesian product, set operations, join operations, renaming
10. Structured Query Language: basic syntax, set operations, sorting, aggregate functions, join operations, nested queries, further predicates
11. Database management: table and schema creation and modification, constraints, views, transactions, programming interface to DBMS
12. Big data processing: NoSQL and semi-structured data, Pandas library, distributed data storage, MapReduce computing model, data warehouse for analytics

***READING LIST[[1]](#footnote-1)***

J. M. Zelle. *Python programming: an introduction to computer science*, 3rd Edition. Franklin, Beedle, 2016.

C. Horstmann-R. Necaise. *Python for Everyone*, 3rd Edition, Wiley, 2019.

R. Sedgewick-K. Wayne-R. Dondero. *Introduction to Programming in Python: An Interdisciplinary Approach*. Addison-Wesley, 2015.

A. Silberschatz-H.F. Korth, S. Sudarshan. *Database System Concepts*, 7th Edition. McGraw-Hill, 2019.

P. Atzeni-S. Ceri-S. Paraboschi-R. Torlone. *Database Systems: Concepts, Languages, Architectures*, McGraw-Hill, 1999.

Further instructional material, e.g. class notes, source code, handouts, will be posted on Blackboard.

***TEACHING METHOD***

 A blend of lectures (60 hours) and practice sessions with computer-based activities (15 hours).

***ASSESSMENT METHOD AND CRITERIA***

Two written tests (40% weight each) on programming and database topics respectively, with closed-ended, open-ended questions and problems on languages and concepts. The first written test can also be taken as midterm exam.

A practical assessment focusing on developing a small specification-compliant work meant to test the ability in coding and operating with a database at the end of the course (20% weight).

***NOTES AND PREREQUISITES***

Students are supposed to have good skills in using a computer, such as abilities in managing files, navigating the Internet, using text editing and spreadsheet applications, but there are no prerequisites concerning programming languages or database softwares. All students are required to have some prior knowledge of a command-line prompt, such as the Windows command prompt or MacOS/Linux terminal, to browse the file system and perform basic file operations.

1. The texts listed in the bibliography can be purchased from the University bookstores; they can also be purchased from other retailers. [↑](#footnote-ref-1)