This textbook offers a one-semester introductory course on computer science for undergraduates with no prior experience in programming. It emphasizes computational thinking and tries to guide students to the upper level of Bloom’s taxonomy of educational objectives, namely to produce new original work instead of simply remembering and recalling facts. The material in the book has been used in the University of the Chinese Academy of Sciences (UCAS) since 2014, several sample course schedules are provided in the introduction.

The structure of the book is guided by the idea of characterizing computational thinking by three external and eight internal features, the first external feature being essentially the same as the first internal. The first chapter focuses on these features, exemplifying them with small programs for computing the Fibonacci numbers. Here, the reader with really no prerequisites will also find definitions of fundamental terms from bit to program and a simple explanation of what really happens seen from a user’s perspective when a program is written and executed. Also contained is a section on the impact of computer science on society.

Chapter 2 focuses on the three external features, automatic execution, bit-accuracy, and constructive abstraction, but in effect gives a detailed introduction to the representation of numbers and strings in computer memory, simple programs manipulating numbers and strings, the von Neumann architecture, the transformation of high-level programming languages into assembly code and its execution. The next three chapters follow the grouping of the eight internal features into three “thinking”-perspectives, namely logic thinking, comprising automatic execution, correctness and universality; algorithmic thinking, containing effectiveness and complexity; and systems thinking, with abstraction, modularity and seamless transition. So, Chapter 3 is about propositional and predicate logic, finite automata, Turing machines, the Church-Turing Thesis, and uncomputable problems. It also contains a short section on Kleene algebra. Chapter 4 is on algorithms and different problem-solving strategies, with several standard examples from sorting to matrix multiplication. Time complexity and the classes P and NP are also addressed. In Chapter 5 data types are considered again, going on to long integers, file abstraction and very detailed hardware considerations with combinatorial and sequential logic circuits, flipflops, adders, instruction pipelines, caches and more.

Chapter 6 transcends the thinking perspectives and moves on to network thinking, which has connectivity, Internet and Internet protocols, security (RSA explained) and simple web programming as topics. A short final seventh chapter picks up again the upper levels of Bloom’s taxonomy by describing several design projects from Turing machines to dynamic webpages, which have been implemented in courses at UCAS. The companion website to the book contains necessary resources, further details, and example solutions.

Each chapter comes with lots of exercises, but alas, most of them are at the lower end of Bloom’s taxonomy as they are simple multiple choice questions. But teachers can use them as a rich source when composing exams.

The reviewer did not look purposely for errors, but some which jumped into his eye should be mentioned: On page 121 it is stated that Goodstein’s theorem cannot be proven in a mathematical system that includes the integers, which of course is very well possible. Instead, it cannot be proven using Peano arithmetic alone. On page 196 NaNs within the IEEE 754 floating-point standard are connected with imaginary numbers, but there is no such connection.

The most preeminent characteristic of this book is its “thinking”-perspective, which the reader may or may not like, and which perhaps leads to a suboptimal arrangement of topics. Clearly, the large body of topics cannot be covered extensively everywhere, but the material presented is impressive, and at least the fundamentals are covered in great detail.

Reviewer: Dieter Riebesehl (Lüneburg)