Books

The programming language landscape, Henry Ledgard and Michael Marcotty, Science Research Associates, Inc., Chicago, 1981. 460 pp. (ISBN 0–574–21340–6, $24.95).

For one who has lived and worked through the entire period of development of every programming language in this linguistic landscape, the experience of reading this book is like that of a child learning to read or an older child confronting formal English grammar for the first time. The same astonishment is expressed by Molière's character in *Le Bourgeois Gentilhomme* when he learns that he has been speaking prose all his life.

To appreciate this book, one should be able to program (speak) in one and preferably more programming languages. The very first exercise asks the reader to choose one of the languages, Ada, Basic, Fortran, Pascal, or PL/I, and two features of that language he would remove and give the reason why.

With this understanding on the part of the reader, the authors guide him in easy stages through the geology of the terrain of programming languages, using the "Mini-language" to illustrate the key language features. Since the authors' philosophy is based on language design principles, Chapters 2 and 3 discuss the principles of syntax (the way of writing a program) and semantics (what a program does, its meaning).

The authors take programming to be the designing of algorithms, as opposed to coding. Therefore, Chapters 4 through 8 present the dominant features of most contemporary languages upon which algorithm design is based—assignment, control structure, data types, procedures, and nesting. These concepts are expanded in Chapters 9 through 12.

Specialized topics are presented in Chapters 12 through 15—input/output, dynamically varying structures, and parallel processing. The use of Mini-language is especially helpful in simplifying and unifying these ideas.

The concluding chapter, 16, is intriguingly called "The Swamp of Complexity." Complexity seems to insinuate itself throughout the landscape, so that the authors give an aerial view, a re-examination of the previous key concepts as an aid to navigation. Although there appears to be no programming language in which complexity has been avoided, the authors give hope in the form of four principles for the provision of programming power and the minimization of complexity.

This book follows the guidelines of Course CS 8, "Organization of Programming Languages," given in Curriculum '78, *Communications of the ACM* **22,** No. 3 (March 1979).

George C. Stierhoff

Note—The Editor assigns reviews of books that might interest our readers. Reviews are signed, and opinions are those of the reviewers.