# Book Reviews

*The following reviews were selected from those recently published in various IEEE* Transactions, *Magazines, and Newsletters. They are reprinted here to make them conveniently available to the many readers who otherwise might not have ready access to them. Each review is followed by an identification of its original source.*

**Principles of Programming Languages: Design, Evaluation, and Implementation**—Bruce J. Maclennan. (New York: Holt, Rinehart and Winston, 1983, 544 pp., $31.95.) *Reviewed by Grace C. N. Yeung and Henderson C. Yeung, Department of Mathematics, California Institute of Technology, Fresno CA.*

A standard computer science course given at the sophomore or junior level, generally entitled *Programming Languages*, is a comparative study of several languages—such as Simula, Snobol, APL, and Lisp—in which orthogonal properties of languages and implementation issues are stressed. This type of course is especially important since most curricula today are based on a single language, often Pascal.

This book, written for such a course, is organized slightly differently from any previous texts that we are familiar with. The focus is on block-structured languages (such as Algol) and functional languages (such as Lisp). The historical development of languages and their related software products are covered, including Pseudo-code Interpreters, Fortran, Algol-60, Pascal, block-structured languages, Ada, Lisp, Smalltalk, and Prolog. Although this list represents a narrow choice of languages, we found the concentration on them —since they are "mainstream" developments—resulted in one of the most coherent texts written for this type of course.

With the validation of several Ada compilers (such as NYU Ada/Ed, Data General) by the Department of Defense earlier this year and with many more compilers on the verge of validation, there is now an obvious need for academic programs to address language and compiler development, perhaps shifting from a Pascal-based curriculum to one based on Ada. An equally important development is The Fifth Generation Computer Project of Japan and the ascendancy of artificial intelligence. The chosen language is Prolog, which clearly indicates a trend towards very high-level declarative languages. Both trends will, to a great extent, shape the languages of the next 10 to 15 years. Consequently, we think that this text is currently the best of its kind, and will certainly use it for a course in programming languages.

In general, students should have a year of introductory computer science prior to taking a course based on this text. In fact, some familiarity with assembly language will be needed in order to tackle the exercises in the first chapter. The second chapter on Fortran is very well written, and here we take note of an outstanding feature of this text, namely the enunciation of principles in the design of languages, which are helpful in crystalizing the ideas behind many design decisions. In the chapter on Fortran, good examples are used to illustrate some of the many principles, such as the information-hiding principle and the abstraction principle.

Following the historical order, the text then goes into Algol-60 before proceeding to Pascal. In this portion of text, the original design philosophy of Wirth is well-presented. However, we have some technical reservation regarding the examples used to illustrate type declaration. The choice to explain dynamic scoping before static scoping is a bit dubious pedagogically, given the fact that only block-structured languages have been covered up to this point in the text. Moreover, the discussion on language specification and Backus-Naur Form could be expanded.

The two chapters on Ada are very well done, however. The key features of the language and especially their positions relative to the earlier discussion are clearly delineated. Three chapters (over 100 pages) are devoted to Lisp. Having taught Lisp extensively, we can say—without reservation—that this constitutes one of the very best introductions to Lisp. The pace and level of instruction are just about right. Lisp, being the first functional language and predominant in the artificial intelligence area, certainly warrants the substantial treatment. In addition, interesting implementation issues abound. For programs which do not offer full-semester courses on Lisp, we strongly endorse this text.

The chapter on Smalltalk is adequate. However, when it comes to Prolog, the text is deficient. The significance of ideas in the development of any science is of major importance. Prolog can serve to demonstrate this development better than any other language. Therefore, we are disappointed in the sketchy history that is given —with no mention of the contributions of Robinson, Green, and Kowalski. Binding through the application of the unification algorithm and automatic backtracking with its implementation in terms of stacks are also omitted. Furthermore, better examples to illustrate Prolog could be given. Nevertheless, we are pleased to see Prolog in the text.

Finally, a list of references either at the end of each chapter or at the end of the text would be helpful, and certain chapters could be expanded and improved.

Although we have been quite harsh on this text at times, we are very fond of it. Since we are in agreement with the author in how such a course should be organized, we highly recommend this book.

Reprinted from *IEEE Computer Magazine*, vol. 17, no. 7, pp. 141–142, July 1984.

**Design of High-Performance Negative-Feedback Amplifiers**—Ernst H. Nordholt. (Amsterdam, The Netherlands: Elsevier, 1983, 234 pp.) *Reviewed by T. R. Cuthbert, Rockwell International, Richardson, TX 75081.*

This book is a revised and reviewed version of the author's 1980 Ph.D. dissertation. It is addressed to amplifier design engineers for use prior to the design phase in which computing aids are employed. The author states that this work is "a systematic and consistent arrangement of design considerations regarding various quality aspects of information transfer." He emphasizes that amplifiers often must provide more than simple power gain, so that the model of the information source and load must be carefully formulated. The book deals exclusively with negative-feedback amplifiers having one terminal in common with source and load. DC