reading for people without a strong mathematical orientation. This main development is supplemented by a number of self-contained and more expository papers which approach the semigroups from a variety of more machine-oriented points of view. This includes Zeiger's independent proof of the decomposition theorem.

The final two papers show two approaches to context-free and other languages: a grammatical approach and a power series approach.

The somewhat specialized subject matter of the book and its diversity of notation, level, and style make it an unlikely choice as a textbook, but it could prove valuable as a reference book and most people with interest in automata theory are likely to find some of the material of interest. Its strongest virtue is in its diversity of approaches and viewpoint.

R. E. Stearns

General Electric Research and Development Center
Schenectady, New York 12301

54[12].—D. W. Barron, *Recursive Techniques in Programming,* American Elsevier Publishing Co., Inc., New York, 1968, 64 pp., 22 cm. Price $5.25.

This monograph deals with the use of recursive techniques in programming. It considers very briefly and sketchily the ideas of recursion, the mechanisms for implementation and the formal relationship between recursion and iteration. It also contains some examples and applications. It is the sort of material that should be covered in a lecture or so in an introductory course in computer science.

M. Goldstein

Courant Institute of Mathematical Sciences
New York University
New York, New York 10012

55[12].—Peter Wegner, *Programming Languages, Information Structures and Machine Organization,* McGraw-Hill Book Co., New York, 1968, xx + 401 pp., 23 cm. Price $10.95.

This book is very much a mixed bag. On the one hand, it is a useful collection of information on many of the most modish topics in computer science today; on the other hand, it suffers from a general lack of organization, occasional narrowness of viewpoint, and sometimes obscure explanations. The author states in his introduction that he plans ". . . to classify programming techniques and to develop a framework for the characterization of programming languages, programs, and computations. In the present text such a framework is developed, starting from the notions that a program with its data constitutes an *information structure,* and that a computation results in a sequence of information structures generated from an *initial representation* by the execution of a sequence of *instructions.*" In fact, it appears that these notions are too abstract to be applied in any meaningful way to the subject matter of this book.

The first chapter of the book is on machine language and machine organization. The author treats the instruction set of the IBM 7094 as a paradigm for computer instruction sets, but unfortunately does not point out that he is doing so. The result leaves the impression to the naive reader that all machines are constructed in more or less this way, when in fact current machine architecture is far more diverse than this book would lead one to believe. The material on virtual address spaces, paging, and segmentation is not quite so narrow in its view, but it is plagued by repetitions of the same material in slightly altered form, with no cross-referencing among these repetitions. For instance, the term *activation record* is defined three different times in the first chapter (and several more times in succeeding chapters) as though the earlier definition had never existed. Also, the author defines access modes to information in a virtual memory, and later defines the same access modes for information in segments, but no relationship is given between these two sets of definitions.

The second chapter is a discussion of assembly programs, and is somewhat better than the first chapter. However, this chapter also suffers from a certain vagueness and lack of cohesiveness.

The third chapter is on macro generators and the lambda calculus. The discussions of Strachey's general-purpose macro-generator and of Mooers' TRAC system are well written, and there is some effort made to relate the two. The material on the lambda calculus is unfortunately more obscure than it needs to be, since the author fails to introduce appropriate abbreviative devices and thus is saddled with awkward notations for much of the exposition. However, there is much useful material here, particularly the section on Landin's SECD machine for evaluating lambda-expressions.

The fourth chapter is on procedure-oriented languages, and is probably the best chapter in the book. In this chapter, the author discusses ALGOL and several strategies for its implementation. He also describes a number of aspects of PL/I, with emphasis on those parts of the language, such as controlled storage allocation and structure definition, that are not in ALGOL. The chapter concludes with a discussion of simulation languages. There are also apprendices on syntactic specification and analysis and on the syntax of ALGOL 60.

Paul Abrahams

Courant Institute of Mathematical Sciences
New York University
New York, New York 10012

56[13.15, 13.20].—Julian D. Cole, *Perturbation Methods in Applied Mathematics*, Blaisdell Publishing Co., Waltham, Mass., 1968, vii + 260 pp., 23 cm. Price $9.50.

The literature of applied mathematics contains many ordinary or partial differential equations solved by various kinds of asymptotic expansions and connection procedures. It is difficult for a graduate student or research worker, not active in the field, to find a good reference from which to start learning the techniques. An earlier book devoted to this aim, *Perturbation Methods in Fluid Mechanics* by Milton Van Dyke, Academic Press, New York, 1964, fulfills the goal for those interested in fluid dynamics, but would be difficult reading for others. The book under review is similar in approach and style to Van Dyke's book, but the newer