the other two scientific languages—Fortran and PL/I—for prominence in the world of programming languages.

Divided into three major parts, the book leaves nothing of importance unsaid. The reader is introduced to the language at a steady pace, one designed for rapid learning, and provides as pleasant a learning experience as one could hope for. One is carried from the mechanics of sitting at an APL terminal through the APL character set, to all the many varieties of functions available to the APL user.

Despite the richness of the language, one is not left in the frustrating position of not being able to see the forest for the trees. Each topic is handled carefully and each description is more than adequate.

In addition to the numerous examples given, each chapter is followed by a series of exercises making the text admirably suitable for classroom use. Chapter 19 contains an interesting discussion of what lies ahead for APL. Though it is perhaps too early to say with any certainty to what degree the APL language will be accepted by the science and engineering community, this text can only help to pave the way for its acceptance.

Henry Mullish

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

47 [12].—Niklaus Wirth, *Systematic Programming: An Introduction,* Prentice-Hall, Inc., Englewood Cliffs, N. J., 1973, 167 pp., 24 cm. Price $10.50.

During the past year or so, the notion of "structured programming" has become quite popular in the computer field. Although the term is still rather ill-defined, it seems to include concepts such as top-down programming, goto-less programming, program verification, and the chief programmer team method of project organization. Wirth's introductory textbook "Systematic Programming" is an attempt to teach some of these ideas to the student when he is first learning about computers. Program verification is presented as being part of programming itself; the *goto* statement is barely mentioned; and an excellent chapter is devoted to the process by which an outline of a program is fleshed out in top-down style until it becomes functional.

This is definitely not a book about "how to code". It requires considerable mathematical aptitude and knowledge on the part of the student (though Wirth disclaims this requirement). Program structure is emphasized at the same time that linguistic details about PASCAL, the language used for the sample programs, are kept to a minimum. A student who intends to continue in the computer field would need to study a programming language in far greater detail, though a mathematician trying to learn about the nature of programming would be quite satisfied with Wirth's presentation.

The first four chapters are a sketchy introduction to the nature of algorithms, computers, and programming systems. Unfortunately, these chapters are too abstract, and a student is unlikely to learn much from them. Chapter 5 introduces flowcharts and the rules of analytic program verification; with each flowchart component, the corresponding verification method is given. The reader may conclude, however, that the emphasis on program verification is a red herring, for in the later chapters when the programs become more complicated, these techniques are hardly used at all. Chapter 6 discusses, very briefly, how one can know that a program will terminate. Chapter 7 (about a quarter of the way through the book) first introduces some details of PASCAL, and again relates each type of statement and each construction to the corresponding flowchart and verification rule. Chapter 8 is devoted to data types.

In Chapter 9 some programs based on recurrence relations are shown; the first two of these are programs for computing inverses and square roots by approximation. The invariant given in the program for computing inverses is incorrect; it should be

$$(1-\epsilon)/x < A < 2/x.$$

Furthermore, the example is confusing in that the range of the variable x should be restricted to $0 < x < 1$ rather than $0 < x < 2$ if the approximation is to be reasonable. (The algorithm given will yield an inverse of 1 for all numbers greater than 1.) The algorithms also have an unpleasant out-of-the-hat quality; a student with any curiosity will be utterly puzzled as to how he could have found them for himself.

Chapters 10 through 14 present a variety of topics: textfiles (a primitive model of input-output), arrays, subroutines and functions, transformation of number representations, and simple text manipulation. The presentation on functions is unnecessarily obscure; the first example of a value-returning function is an integration procedure that itself has a functional parameter. A simpler example of a value-returning function should have been given prior to this complicated one.

The last chapter is devoted to stepwise program development, and presents three case studies: solving a set of linear equations, finding the first n prime numbers, and generating sequences of N characters from an alphabet of 3 characters such that no two immediately adjacent subsequences are identical. Each program is developed as a sketch in which informal statements are expanded to more detailed statements until the program has been fully formalized; the complete programs are then modified to incorporate various improvements. This part of the presentation is very well done.

Overall, "Systematic Programming" is a difficult, frequently obscure, but challenging book. Its approach is heavily mathematical, both in style and in the choice of examples. For students whose primary interest is numerical mathematics, it would be a good choice. For the general computer science student, it would not be, for its view of programming is too abstract and specialized.

PAUL ABRAHAMS

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

48[12, 13.40].—E. S. PAGE & L. B. WILSON, *Information Representation and Manipulation in a Computer,* Cambridge Univ. Press, New York, N. Y., 1973, v + 244 pp., 23 cm. Price $5.95.

I was quite pleased to receive this book for review, since I have been searching for a long time for a satisfactory text for a course in data structures; this is the first text to fill the need. The classic text on the subject is Knuth's "Art of Computer Programming", but the relevant material is spread over two volumes. Furthermore, the explanations are on too high a level for an undergraduate course. Page and Wilson's book might well be entitled, "A Simplified Version of Knuth". Virtually all of their material, except for an early chapter on symbols and codes, is contained in Volumes I and III of Knuth, and their references to Knuth are plentiful.

Chapters 1 and 2 deal with symbols and their encodings and include a treatment of error detection and correction and Huffman coding. These topics seem somewhat remote from the rest of the book. Chapter 3 discusses internal representations of numbers. This would be an appropriate point for a discussion of character strings, but there is none, perhaps because Algol does not have any facilities for manipulating them. Chapter 4 treats arrays and methods of accessing them, and includes a good discussion of sparse arrays (arrays that are mostly zero). Chapter 5 discusses allocation methods for queues, stacks, and deques, considering both arrays and linked lists. There is also a more general discussion of linked lists here. Chapter 6 discusses trees and how to traverse and represent them. The discussion of conversion between binary and general trees seems overblown, however. Chapter 7 discusses searching, including binary and hashcoded searches, and Chapter 8 discusses sorting methods.

There are a number of disconcerting flaws in the book. Explanations are occasionally obscure (e.g., an application of generating functions on p. 158). The justifications given for circular lists (p. 86) are incorrect; a circular (singly-linked) list saves one pointer and nothing else, compared with an ordinary list with front and rear positions. Algol is used throughout as the programming language, and the limitations of Algol (no strings, no pointers) cause these programs, as well as the choice of methods, to be unnecessarily obscure. The book is well supplied with exercises, but many of these are