than one. The character string variables are of two types: ALPHA variables, which fit into a single computer word, and TEXT variables, which are arbitrarily long strings. The distinction between the two is thus implementation-based, and could be confusing to the neophyte.

The implementation of SIMSCRIPT uses dynamic storage allocation for entities, and for arrays as well. The result has a great deal of flexibility, though I suspect some of it at the price of efficiency. The SIMSCRIPT "DO" statement, like the ALGOL "FOR" statement, permits the parameters of the "DO" to vary as the loop is executed; thus, unless the compiler is very clever about it, execution of "DO" loops will involve a great deal of unnecessary recomputation.

The book is written in five sections, in order of increasing difficulty. They are described by the authors as:

1. A simple teaching language designed for nonprogrammers.
2. A language comparable in power to FORTRAN.
3. A language comparable in power to ALGOL or PL/I.
4. The entity-attribute-set features of SIMSCRIPT.
5. The simulation-oriented part of SIMSCRIPT.

This arrangement was made for pedagogical reasons, but it does not quite succeed. An experienced programmer will find the book slow reading if he tries to master all the details; a novice would have a great deal of difficulty. However, the information is all there, and a determined reader will assimilate it sooner or later.

I would recommend this book for those interested in the field of programming languages, for those who need to write a computer simulation and are shopping around for a language in which to write it, and for those who are working on problems where the set and property manipulation facilities of SIMSCRIPT would be helpful.

PAUL ABRAHAMS

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

47[12].—JEAN E. SAMMET, *Programming Languages: History and Fundamentals*, Prentice-Hall, Inc., Englewood Cliffs, N. J., 1969, xxx + 785 pp., 24 cm. Price $18.00 ($13.50 student edition).

Programming Languages: History and Fundamentals is a monumental and encyclopedic treatment of its field. The primary aim of the book, as stated by the author, is to provide, ". . . in one place, and in a consistent fashion, fundamental information on programming languages, including history, general characteristics, similarities, and differences." This aim has without doubt been achieved, and achieved well. A second aim is ". . . to provide specific basic information on all the significant, and most of the minor, higher level languages developed in the United States." This aim also has been achieved; there is an impressive collection of about 120 languages described in varying amounts of detail.

The book is organized into three introductory chapters, six chapters on languages grouped more or less according to application areas, and two chapters on unimple-

mented concepts and future long-range developments. The organization is well thought out and suits the author's purposes admirably. The first chapter is devoted to defining what is meant by a programming language, to pros and cons of higher-level languages, to classifying and categorizing languages, and to a presentation of factors influencing the choice of a language. Omitted from consideration are most European programming languages, and certain languages that are not, by the author's definition, higher-level programming languages, e.g., Report Generator, Autocoder, APL and SLIP. The second and third chapters are devoted to describing functional and technical characteristics of programming languages, partly with the aim of developing a format for the later discussions of specific languages. Functional characteristics, roughly, are the historical, political, economic, and pragmatic aspects of a language, i.e., those that are not part of its definition. Technical characteristics have to do with the actual syntax and semantics of the language. This outline of discussion is spelled out in some detail and generally adhered to in the sequel. The fact that approximately equal amounts of space are devoted to functional and technical characteristics is an indicator of the tone of the discussion. The author does not attempt to teach the reader how to program in any of the languages discussed, and indeed, one certainly would not learn programming from *this* book.

As a reference work for background on specific programming languages, the book is a success. As a textbook on programming languages as a field of study, it is not. The tone of the introductory chapters is too platitudinous, and the technical information, both in depth and in organization, is inadequate to the task of imparting a feeling for what programming languages are all about. The difficulty is that this feeling can ultimately be achieved only by actually writing programs in various languages and thus coming to appreciate their fine points. Thus, the uniformity of organization across languages is achieved at the cost of omitting really deep examination of a few of them. The chapter on technical characteristics attempts to raise the issues of what choices a language designer must make, but the question of how, historically, they have been made is scattered among the different language discussions and never really explored in depth. Questions such as scoping rules, extensibility, and the structuring of data aggregates are never treated in a unified way, and the material on minor languages and on functional characteristics tends to deemphasize the importance of the technical characteristics of the major languages. There are no exercises in the book, and, indeed, the material provides no basis for exercises.

In summary: *Programming Languages*: *History and Fundamentals* is a fine source for factual knowledge, but a poor source for gaining understanding.

PAUL ABRAHAMS

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

48[12].—M. V. WILKES, *Time-Sharing Computer Systems*, American Elsevier Publishing Co., Inc., New York, 1968, iv + 102 pp., 22 cm. Price $4.95.

This is an excellent introduction to the design of time-sharing systems, which expresses concisely a number of current ideas. The text is clearly influenced by the