control cards, rules for carrying out computations and gives numerous examples. Instructions are given to the system in a form of English sentences simulating desk computing. Indicative of the popularity of this system, is the fact that subsequent to the issuance of this volume, the system has been rewritten in ASA Fortran and implemented in several other computers such as the Univac 1108, IBM 360-50 and 65 and CDC 6400 and 6600. While the volume under review indicates control cards for the 7094 only, various versions are in use throughout the country and a version has even been unveiled recently in a time-sharing environment.

As an old-fashioned "programming expert", this reviewer has a certain antipathy toward the concept of "instant programs"—I probably feel that one should have enough motivation to write the appropriate subroutine, if one wants to use anything beyond the arithmetic operators and elementary functions. However, the audience is there, and systems such as OMNITAB have served a real need. The user of this or similar systems will be well advised to read the section entitled diagnostic features. What is pointed out there, cannot be repeated too often:

"The concept of a general-purpose program rests in some measure on the assumption that the user, though not a programmer, is familiar with the behavior of the mathematical functions he is using or trying to compute . . . diagnostic features are incorporated . . . however diagnostic statements are no substitute for sound mathematical analysis, which is necessary to avoid the more serious pitfalls of numerical computations."

In this regard, it would have been useful to list the possible diagnostic statements generated by each command. In fact, the numerical methods used in the functions and routines should have been given. Not only would they be useful to users outside of OMNITAB, but they are a must when trying to gauge numerical accuracy. In the final analysis, it probably does not matter whether one learns to program or to use "packages" or both. Of paramount importance is the question of the accuracy of the results. The hope is that all users of computers in future generations, those in the physical as well as those in the social sciences, will learn the elements of numerical analysis.

MAX GOLDSTEIN

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

45[12].—F. R. A. HOPGOOD, *Compiling Techniques*, American Elsevier Publishing Co., Inc., New York, 1969, vi + 123 pp., 23 cm. Price $6.00.

This elegant monograph, suitable both for self-study and for text use in short courses on compiling, manages in its brief (126 pages) compass to discuss many of the most salient issues of compiler writing in an illuminating manner. An introductory section (28 pages) discusses compiler-related data structures and their computer treatment, and includes a thumbnail account of hashing. Backus notation is then introduced. Lexical analysis is discussed in a 10-page chapter which is, unfortunately, less transparent than other passages of the book. Many of the principal parsing methods are then nicely surveyed in a 20-page chapter. Code generation is next discussed, with emphasis on arithmetic expression code generation. This chapter includes a discussion of fundamental code-block optimizations, including redundant store elimination and common subexpression finding, with a following chapter discussing straight-line register allocation and temporary storage minimization in more detail. The book ends with a quick comparative survey of various compiler-writing systems.

J. T. SCHWARTZ

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

46[12].—P. J. KIVIAT, R. VILLANEUVA & H. M. MARKOWITZ, *The SIMSCRIPT* II *Programming Language*, Prentice-Hall, Inc., Englewood Cliffs, N. J., 1969, xiii + 386 pp., 25 cm. Price $10.95 cloth, $6.95 paper.

SIMSCRIPT is a programming language whose primary orientation is towards the programming of computer simulations, but which has the facilities of a general-purpose language. The authors of this book have chosen to emphasize the general-purpose aspects of SIMSCRIPT rather than its simulation capabilities.

Stylistically, SIMSCRIPT is a very "smooth" language, and its design is highly professional. The syntax, like that of COBOL, is intended to make programs read like English sentences, though briefer modes of expression are permitted. With the syntax stripped away, the algebraic part of the language would look like FORTRAN with a few ALGOL features, such as conditional statements and DO loops with variable parameters. There are additional facilities for text manipulation and some rather elaborate report-generating capabilities (quite useful, of course, in simulation experiments). The input-output is well planned and easy to use.

It is doubtful that SIMSCRIPT would attract many users, however, solely on the basis of its general-purpose facilities. The strength of the language lies in its capabilities for handling entities, sets and attributes. An *entity* is a computational object capable of having attributes, of belonging to sets, and of owning sets; an owned set may be thought of as a set-valued attribute. The attribute facility may be used to create PL/I-like structures, though the set operations have no immediate PL/I counterpart. Entities may be removed from or added to sets in a number of different ways, corresponding to various forms of queueing.

The simulation facilities are based upon the notions of a system clock, which keeps track of simulated time, and of events which are computations to be carried out at a certain point in simulated time. Events may arise either endogonously (internally generated) or exogonously (externally generated). After all events associated with the current time are executed, the system clock is simply advanced to the time of the next scheduled event or events. Execution of an event, of course, can cause the creation of new events, which may be scheduled at the current time or at later times.

I disagree with the authors' claim that the general-purpose part of SIMSCRIPT is comparable in power with ALGOL or PL/I. For instance, SIMSCRIPT does not have the ALGOL block structure, though it does permit recursive functions. It also lacks certain conveniences such as the ability to start array subscripts at values other