made. In particular, the form of Gödel's Theorem which is given (and proved) claims only the incompleteness of an "adequate, ω-consistent logic," without any indication that in any such logic an undecidable sentence can be effectively found, or that the hypothesis of ω-consistency can be reduced to simple consistency. These features, along with the proof, evidently resulted from a condensation of the discussion in Davis [1], where the creativity of the set of theorems of an adequate logic is discussed, but not emphasized.

In summary, this book can be recommended as an exploration in depth of a selection of the more scientifically respectable attempts at constructing models of the brain, laced together with enough commentary and indication of context to give the reader a reasonably fair notion of what has been done.

J. D. Rutledge

IBM Research Center
Yorktown Heights, New York

1. Martin Davis, *Computability and Unsolvability*, McGraw-Hill Book Co., Inc., New York, 1958.

102[Z].—F. J. Corbato, J. W. Poduska & J. H. Salzer, *Advanced Computer Programming*, M. I. T. Press, Cambridge, Massachusetts, 1963, vi + 192 p., 29 cm. Price $5.00.

In contrast to most of the books on digital computer programming which have been published (in increasing numbers) in the last two or three years, this short text deals exclusively with systems programming. It does this very competently by setting up a prototype system, called Classroom Assembly Program (CAP), and showing how to design an assembler and compiler to translate from CAP language to machine language. The translator program is described in the FAP language of the IBM 7090. The reader is assumed to be sufficiently familiar with FAP so that he can understand the IBM reference manuals on FAP and the 7090 without guidance from the present text. In particular, he is supposed to be familiar with the Binary Symbolic Subroutine (BSS) linkage and relocation techniques used in the IBM Fortran Monitor System, as described in the FAP Reference Manual (IBM Publication C28-6235, September 1962).

The text consists of five chapters and three appendices. The five chapters describe: (1) the CAP language (it is similar to FAP); (2) the CAP assembler (in general terms and using flow charts); (3) the CAP compiler (e.g., what does a compiler do?; precedence of operations; temporary storage); and finally (4) an execution-monitor system which allows the student to perform laboratory exercises on CAP (e.g., modifications and additions suggested in Appendix C).

Appendix A contains a FAP listing of the assembler-compiler program of CAP. Appendix B contains a FAP listing of the execution-monitor program.

Although the authors have based their exposition on a particular programming system, they succeed in explaining many of the general ideas and problems underlying the design of assemblers and compilers. The text is a useful addition to the literature on programming technology.

E. K. Blum

Wesleyan University
Middletown, Connecticut