sophisticated computer support he feels is essential for corporate modelling. After setting out the requirements of a modelling language, he describes the various corparate modelling packages available and not surprisingly devotes nearly twice as much space to his own company's package (COMOS) as to all the others put together. The book ends with some case studies and general conclusions about the scope and the future of corporate modelling.

The book is well structured with helpful sub-headings. The index is rather brief. In many cases the contents list could prove more helpful. The small and stark typeface is rather tiring to the eyes. The incidence of typographical and spelling mistakes was irritating.

Worth inspection by anyone practicing or contemplating corporate modelling.

HOWARD CARTER

Proving Programs Correct

ROBERT B. ANDERSON

Wiley, New York, 1979. 184 *pp.* £5.40

The purpose of this book is to explain and illustrate the use of mathematical induction in proving computer programs correct. The explanations are generally clear and there are examples if you wish to test your understanding. There are excellent signposts as you go along giving you the option of reading the next section or skipping it if you feel that it is not necessary. The last chapter deals with current research on all aspects of the topic. Following that there is a classified bibliography with over 90 entries. The subject index at the end of the book is a useful feature.

However programmers looking for a panacea will be disappointed. On page 116 (unfortunately not before) you are told:

> "All of the example proofs of program correctness in this book have dealt with completed (and small) programs. However, in practice, such proofs should be given by the programmer during the programming process. One should not expect to take a large finished program and prove it correct. The intellectual demands of such a proof are likely to be too great. One must prove the correctness of small sections of code as they are completed."

There follows a discussion of the importance of structured programming in this context. It is therefore evident that this method will not help to get the bugs out of a large complex program. Simpler programs may be improved but straight-forward code inspection would probably be as effective. Therefore, I believe this method will remain a subject for discussion and research amongst mathematicians, but will never be accepted as a useful tool by programmers.

D. EVELYN