A book review for the IBM Systems Journal.

Structured Programming, Theory and Practice, R.C.Linger, H.D.Mills, and B.I.Witt, Addison-Wesley Publishing Company, Inc., Reading, MA, 1979. 402 pp. (ISBN 0-201-14461 ABCDEFGHIJ-HA-79, $ 18.95)

Though published by Addison-Wesley, this book should be regarded as an IBM product, rather than as a general textbook or a scientific monograph. According to its title page, all its authors are from IBM Corporation, and it appeared in "The Systems Programming Series", which is not only sponsored by IBM but, according to page IV of the book, has an "IBM Editorial Board". (Its members have been listed in alphabetical order, with the exception of mr.Gruener from Addison-Wesley, who is mentioned last!) Addison-Wesley did a very neat job of binding and printing, with the exception of eight pages at the end of section 7.5 which are a (reduced) facsimile of (poor) lineprinter output.

The book has been divided into seven chapters with the following titles: 1. Precision Programming, 2. Elements of Logical Expression, 3. Elements of Program Expression, 4. Structured Programs, 5. Reading Structured Programs, 6. The Correctness of Structured Programs, and 7. Writing Structured Programs. I found the majority of them quite illuminating and/or instructive.

Chapter 1 (14 pp.) sets the scene quite nicely and contains many remarks that deserve to be quoted, such as:

"There is a simple reason why you should commit yourself to writing programs that are free from errors from the very start. It is that you will never be able to establish that a program has no errors in it by testing. Since there is no way to be certain that you have found the last error, your real opportunity to gain confidence in a program is to never introduce the first error." (Actually, the quotation ends with "to never find the first error", but that seemed one of the very rare misprints I detected.)

The authors' healthy attitude regarding the proper role of mathematical proofs is shown by:

"A mathematical proof is an agenda for a repeatable experiment, just as an experiment in a physics or chemistry laboratory. But the main subject in each experiment is another person instead of physical objects or material. The intended result of the experimenter is a subjective conviction on the part of the other person that a given logical hypothesis leads to a given logical conclusion. [...] As noted, any human fallibility may be present because reasoning is a human activity; that is, an agreement that a proof is correct and the actual correctness of the proof are two quite independent things."

Chapter 2 (30 pp.) --ending with a one-page bibliography of "Related Reading"-- introduces the necessary concepts and notations from mathematics in general, while Chapter 3 (46 pp.) does the same for programs in particular.

Chapter 4 (56 pp.) pays to control flow an amount of attention which I thought a bit too much until I read Chapter 5 (66 pp.: you see, the chapters are getting longer and longer!), which ends with a 25-page "Case

Study in Program Reading", one of the absolute highlights of the book! The authors have undertaken the horrendous task of unravelling a contorted PL/I program --taken from the open literature!--, introducing this job with the very true words: "as it stands, it is a formidable object for human understanding!". This case study is impressive in more than one way: not only does it drive home the message how terribly (and needlessly) contorted programs can be, but it also illustrates the power of the methods derived in the preceding chapter.

Chapter 6 (84 pp.: yes, again longer!) appealed less to me. As the authors remark themselves (p.237): "Even though the above derivation is a little tedious,[...]". Being used to what I am used to, I found the qualification "a little tedious" applicable to most of that chapter, as only a modest amount of mathematical sophistication could have shortened many an argument. Under the assumption that I was not representative of their target audience I decided not to blame the authors.

Chapter 7 (94 pp.) is largely devoted to the systematic development of three programs, and the authors should be recommended on their choice of problems, as the three are very different from each other.

One detailed comment on line 6 of the program on page 176, where, in the functional description of the succeeding program fragment, they have written

$$(\exists k(key = table(k),\ lo \leq k \leq hi) \rightarrow i := k\ [...]) \quad .$$

The truth of the boolean expression to the left of the arrow does, however,

not define the value of k that is used at the right-hand side of the arrow: at the left-hand side k is a bound variable.

One general comment. The authors have clearly tried to reach as large an audience as possible, a goal that evidently implied for them that they could permit themselves only a modest departure from the PL/I subculture. In a more radical attitude a shorter (and simpler) book could have been written, but then they would probably have failed to reach their target audience. In short: for an IBM product, the book is excellent.

Edsger W.Dijkstra

The Netherlands