The routines have been found to be easy and 'natural' to use, although some tailoring was necessary on the input/output routines MAKEST and GETSTR before their present form was decided upon.

It is appropriate at this point to consider the possibility of adding variable length string facilities to the FORTRAN standard. As has been pointed out, this has already been done on a few compilers, and there are clear advantages to the idea. The arguments against such a course are philosophical, but probably valid.

FORTRAN has survived so long, despite the spirited attacks of academics who would like to see it stamped out, largely because it is not very machine independent and it is still possible to see the machine through the language. This concept of machine visibility allows programs to be tuned to high efficiency and is probably the reason that FORTRAN and COBOL are still so much more widely used than their more powerful modern equivalents such as ALGOL and PL/1. Fixed length strings fit in with this philosophy, but variable length strings with the necessity of indexed addressing, fluctuating use of run time storage and random garbage collection phases do not.

There does not appear to be a clear answer to this question: FORTRAN has existed until now with virtually no string facilities, although admittedly under pressure in recent years; and COBOL users appear to be quite happy with only fixed length strings.

On the other hand, where text lengths vary widely and arbitrarily it becomes difficult, wasteful and inflexible to decide on a fixed upper limit for the text. Probably the best solution is to live with fixed length strings as in the new standard, but to have in the system library such a variable length string package as has been described for use in difficult cases.

References

ACM SIGPLAN Notices, Vol. 11, No. 3, March 1976

ANS X3 COMMITTEE (1966). *ANS FORTRAN* (X3.9-1966 and X3.10-1966)

ANS X3J3 COMMITTEE (1976). *Draft Proposed ANS FORTRAN* (X3J3/76)

DAY, A. C. (1972). *FORTRAN Techniques*, Cambridge University Press

HANSON, D. R. (1974). A Simple Technique for Representing Strings in FORTRAN IV, *CACM*, Vol. 17, pp. 646-647

HERTWECK, F. (1970). A Proposal for String-Handling in FORTRAN, *Proceedings of SEAS*, Vol. XV, pp. 197-206

GENERAL ELECTRIC COMPANY (1972). Series 6000 FORTRAN Compiler Reference Manual (2200.01)

GENERAL ELECTRIC COMPANY (1973). FORTRAN IV Reference Manual (3102.01)

SHELL, D. L. (1959). A High-speed Sorting Procedure, *CACM*, Vol. 2, pp. 30-32, *CACM*, Vol. 6, p. 445 and *CACM*, Vol. 7, p. 349

Book reviews

Computer Control and Audit, Second Edition, by W. C. Mair, D. R. Wood and K. W. Davis, 1976; 489 pages. (*The Institute of Internal Auditors*, US $20)

This considerable tome with attached control evaluation tables bears witness to the significant developments in computer control and audit since the appearance of the first slim volumes in the early sixties followed by Pinkney in this country in 1968. This book (in its second and enlarged edition) is the product of a task force set up by The Institute of Internal Auditors with three partners of Touche Ross & Co (USA) behind whom is an accumulation of many years practical experience.

The book does not set out to be a complete cure to all computer control ills. Neither is it meant to be a reference manual since full comprehension of the later chapters is dependent upon reading certain earlier chapters. It does, however, offer a very practical and sound approach to the analysis and audit of all varieties of controls, both computerised and manual. It also includes a twelve page glossary and a four page list of recommended reading.

The book first covers basic areas including the need for controls to prevent what the authors call 'Exposures and causes'. Various evaluation procedures and charts are introduced. The following sections explain in detail how the approach can be applied to the different elements of a computer system—applications, systems development, and computer operating (processing). There are additional chapters on advanced topics including online systems, minicomputers and data bases. There is also a section on audit management which advises on the problem of recruiting and training computer audit staff. The section on computer abuses is interesting, not the least because it places the Equity Funding fraud computer in a truer perspective than did television and the popular press.

The objective of the IIA task force was to extend an existing limited distribution external audit manual by incorporating the viewpoints of internal auditors, data processing personnel, and top management and, at the same time, to update the book to reflect current technology. In this they have succeeded. *Computer Control and Audit* is well written and readable. The book is intended equally for all concerned but it is interesting that the authors place systems developers at the head of their list, whilst confirming that the prime responsibility for controls is with the user.

J. A. SHACKLETON (London)

Associons and the closure statement, by M. Rem, 1977; 115 pages. (*Mathematical Centre Tracts* 76, Dfl. 14)

Based on an initial concept of Dijkstra, the book expands on the two papers on associons written by the author, Feijen and Dijkstra. The author takes as starting point the opportunities for a store with a high level of concurrency that modern techniques, such as associative addressing and large scale integration, have presented to the programming language designer.

Starting from the definition of an associon as 'an ordered *n*-tuple of names' the book introduces the concept and the properties of the associon and in particular the closure statement. After the formal definition of the closure statement, the following chapters deal with such topics as 'the repetitive construct', 'dynamically created names' and 'the cliques of an undirected graph'. The final chapter is interesting, dealing as it does with the various ideas and concepts that have been rejected in the research.

The book, another in the Mathematical Centre Tracts series, is well written and particularly well documented, containing a number of examples, an ample bibliography and cross references in the text.

M. P. COATES (Norwich)

Errata

There were three mistakes in the published version of the ALGOL program appearing on p. 184 of the paper by D. J. Evans, 'On the use of fast methods for solving boundary value problems' (*The Computer Journal*, Vol. 20, No. 2, pp. 181-184). In the second column the first line should be terminated by a semicolon whilst in lines five and six the symbol **and** should be in bold type.