HULL, T. E., ENRIGHT, W. H., FELLON, B. M. and SEDGWICK, A. E. (1972). Comparing numerical methods for ordinary differential equations, *S.I.A.M. J. Numer. Anal.*, Vol. 9, pp. 603-637.
KAHANER, D. K. (1971). Comparison of Numerical Quadrature Formulas in *Mathematical Software*, ed. J. R. Rice, Academic Press, N.Y., pp. 229-259.
LYNESS, J. N. and KAGANOVE, J. J. (1976). Comments on the nature of automatic quadrature routines, *ACM Trans. Math. Soft*, Vol. 2, pp. 65-81.
LYNESS, J. N. and KAGANOVE, J. J. (1977). A technique for comparing quadrature routines, *The Computer Journal*, Vol. 20, pp. 170-177.
MILLER, G. F. and SYMM, G. T. (1975). Procedure FRED2B, NPL routine, ref. no. DS/06/1/Algol 60/7/75.
THOMAS, K. S. (1976). On the approximate solution of operator equations, Part II, Preprint, University of Oxford.

Book reviews

Information Management Systems/Virtual Storage, by Myles E. Walsh, 1980; 308 pages. (*Prentice-Hall*, £11·00)

How to put a record in a file quickly, allow for a possible future change in its size and make it appropriately cross referenced to permit rapid retrieval, is the problem. This book deals with the specific solution for batch and real time using IMS/VS (which is the most comprehensive of Data Base Management System packages). More restrictively this book is for DP managers of departments with IBM mainframes who need to bridge the gap between the glowing descriptions of the salesmen's glossy flysheets and the volumes of technical manuals. Within this brief the author does very well. The book is readable without being juvenile or condescending, basic terms are defined and there is a good index.

The first chapters build up a clear idea, with the help of analogues, of a data base and its management. Mr Walsh then proceeds to deal with the technical aspects of the system, its batch and telecommunications features, its utilities and other features. A chapter deals with actual managerial problems, like staffing and educating. Another chapter intercompares major DBMS packages: The author's honesty is to be commended when he says some think IMS/VS is the cat's whiskers (Cadillac) and others believe it to be a lemon (Edsel). Likewise he admits IBM was a world follower in so-called Virtual Storage.

The book is understandably peppered with IBM buzz words but unfortunately there is no glossary of mnemonics. IMS/VS may be ideal for many DPDs (Data Processing Departments) but is nvg (not very good) for smaller mainframes DBMSs (Data Base Management Systems) or distributed micro network TP (Teleprocessing) systems, which are now part of the EDP/MIS (Electronic Data Processing/Management Information System) landscape. However having said that the book is well worth selectively reading in order to understand the architecture and concepts of one manufacturer's answer to IMS, especially as other solutions cannot be radically different.

I. R. WILLIAMS (London)

Learning to Program in Structured Cobol, by Edward Yourdon, 1979; 465 pages. (*Prentice-Hall*, £10·35)

Structured Analysis and System Specification, by Tom De Marco, 1979; 348 pages. (*Prentice-Hall*, £16·25)

A fashionable algorithm for the construction of book titles is: (1) devise a title which describes the content in a form suitable for the market; (2) select a noun from the title; (3) insert the word 'structured' before the selected noun. That this algorithm fails on occasion is illustrated by the first of the books reviewed here: it is really about 'Learning to write well-structured programs in ANSI 1974 Cobol'. There are two parts: 'Part 1 [co-authors Gane, Sarson] can be used as a stand-alone introduction to structured programming or it can be used in conjunction with the more advanced concepts and features presented in Part 2 [co-author Lister]'. These 'advanced features' include *inter alia* arithmetic expressions, arrays, and the *call* statement. A punched card environment is assumed throughout. Yet for the intended readership—'people with no previous knowledge of computers'—this book is much better than many which are currently available. It always explains why particular coding techniques are chosen and it makes the reader aware of the virtues and dangers of each language feature.

The best post-ANSI 1974 tyro's book I have seen is McCracken's latest offering (Wiley, 1976) despite its use of flowcharts and its failure to treat indexed and relative files adequately. Yourdon's book, for all its virtues, cannot match McCracken's for price, elegance or, most important, accuracy—the naive reader, of all people, should be spared the confusion and insecurity which result from even a few errors of detail.

De Marco's is another 'structured' book from the Yourdon organisation. Structured analysis turns out to consist of disciplined use of a number of familiar tools: data flow diagrams, a data dictionary, 'Structured English' (resembling pseudo-code) and decision tables. The process of 'deriving a logical file structure' is akin to Codd's normalisation of relations, but there is no indication that the author is aware of the relational model—indeed, he declares himself indebted to two other Yourdon denizens who have remarked that his logical file structure is a 'third normal form set' from 'set theory'. De Marco argues strongly for the use of data flow as a means of partitioning the analysis, and for the use of all the above tools as a means of communicating with users ('something you can't show to a user is totally worthless as an analysis tool'), with much advice on how to make them acceptable to the users. Given this view, it is not surprising that the end-product of the analysis phase, the functional specification (called here, of course, 'structured specification') is itself expressed in terms of the above tools. Data flow diagrams are supported by the data dictionary, the processes being elaborated by Structured English or decision tables as appropriate. The information which the author recommends be included in the specification could well be supplemented (see the paper by S. J. Waters in this journal, vol. 22, no. 3) without affecting the approach. The author is at pains to specify methods which eliminate redundancy from the specification; he dismisses the unrealistic and undesirable notion of a 'frozen' specification and regards maintainability as being of prime importance.

The practising systems analyst will find this book painless to read; the exposition is clear, the pace slow, and the arguments strongly presented.

JIM INGLIS (London)

Digital Networks and Computer Systems 2nd Edition by Taylor L. Booth, 1978; 592 pages. (*John Wiley*, £14·75)

The first edition of this book, published in 1971, pioneered the unified hardware/software approach to the introduction of digital and computer systems which is gaining popularity in universities on both sides of the Atlantic. Indeed, the increasing realisation that this grey area between Mathematics and Engineering is fast becoming a third discipline indicates the need for such a book. The second edition has the same form as the successful 1971 text, but the content has been consolidated and enhanced. It is aimed at the first year undergraduate who wishes to invest his meagre funds in a single textbook covering introductory courses on logic design, digital systems, computer organisation and computer programming.

Taylor Booth adopts a rather formal, but not pedantic, approach to the specification and design of digital information processing systems. Hardware and software implementations of algorithmic processes are given equal weighting. However the means of implementation show the age of the original text. Hardware realisations are mainly at the *SSI* gate level; *MSI* and *LSI* devices are hardly mentioned. A simulated educational computer, called SEDCOM, is used to illustrate the principles of computer organisation and assembly language programming. SEDCOM is a rather thinly disguised version of the PDP 8 which is rather too long-in-the-tooth for many readers.

I recommend this book to the serious student as a good foundation for more advanced study.

R. M. LEA (Uxbridge)