The Year 2000 Computing Crisis A Date Conversion Plan

review by Jeff Loeb

The Year 2000 Computing Crisis
A Date Conversion Plan
By Jerome T. Murray and Marilyn J. Murray
ISBN 0-07-912945-5-5
Published by Osborne McGraw-Hill
Cost $55.00; book includes a 3-1/2" diskette

"With the year 2000 coming, you can't put off date conversion any longer. This guide is the solution. With it, you'll have all the tools necessary to avoid a systems meltdown."

"At the dawn of the 21st century, legacy systems in mainframes worldwide will start generating bad data because existing software interprets years as 2 digits rather than as 4 digits."

Plainly stated, computer systems can't distinguish 1900 from 2000. With uncommon clarity, Jerome and Marilyn Murray explain the problem and map out a strategy to neutralize the problem. Their solution includes source code written in COBOL, RPG, and IBM Basic Assembler Language.

The intended audience for this book is managerial type people. The language used makes for easy reading and is meant for the intended audience. The introduction of the book states that in 1968, the National Bureau of Standards issued Federal Information Processing Standards Publication 4, which mandated the use of 6-digit dates for all information exchange among federal agencies. The standard became effective on January 1, 1970.

Typically, a Fortune 500 company has a library of 50,000 COBOL programs. At an average of 750 lines of code per program, that comes to 37,500,000 lines of code in the program library. If you apply industry standards cited by Yourdon in a 1975 study, mainframe productivity level is about 15 debugged lines of source code a day. Therefore you have 37,500,000/15 work days, or 2,500,000 work days. Allowing 10 days for vacation, 10 days for holidays, and 10 days as sick days means that only 230 work days remain in a year. Consequently, 2,500,000/230 work years, or 10,870 work years are required to rewrite the program library. Assuming that there are 500 programmers available means 10,870/500, or 21.7 calendar years are needed.

The book is highly technical and details a foolproof method for solving the year 2000 problem using subroutines written in COBOL, RPG, and IBM Basic Assembler Language. The book is rather useful for someone wanting to learn coding in any of those languages. With the book comes a 3-1/2" diskette containing these subroutines.

Under DOS, the 3-1/2" diskette drive is made the default drive, and the tour of algorithms is activated by typing 2000 and pressing ENTER. The tour has its greatest meaning if all the algorithms have been read. The contents are very technical and reading this book can be a formidable task if one is not familiar with the algorithms contained therein. As far as indexing goes, it is hard to find topics without rereading the entire book. I would not purchase this book again, but the book might make for good reading if it were skimmed at a library.

The book does not have any content that could apply for finding a solution to the Y2K problem in M. **M**

Jeff Loeb is a Volunteer at the Veterans Administration Hospital in San Diego, California and has been programming in M for about 2 to 3 years. He can be reached at jeffloeb@delphi.com

---