PREFACE to the First Edition

Good programming cannot be taught by preaching generalities. The way to learn to program well is by seeing, over and over, how real programs can be improved by the application of a few principles of good practice and a little common sense. Practice in critical reading leads to skill in rewriting, which in turn leads to better writing.

This book is a study of a large number of "real" programs, each of which provides one or more lessons in style. We discuss the shortcomings of each example, rewrite it in a better way, then draw a general rule from the specific case. The approach is pragmatic and down-to-earth; we are more interested in improving current programming practice than in setting up an elaborate theory of how programming should be done. Consequently, this book can be used as a supplement in a programming course at any level, or as a refresher for experienced programmers.

The examples we give are all in Fortran and PL/I, since these languages are widely used and are sufficiently similar that a reading knowledge of one means that the other can also be *read* well enough. (We avoid complicated constructions in either language and explain unavoidable idioms as we encounter them.) *The principles of style, however, are applicable in all languages, including assembly codes.*

Our aim is to teach the elements of good style in a small space, so we concentrate on essentials. Rules are laid down throughout the text to emphasize the lessons learned. Each chapter ends with a summary and a set of "points to ponder," which provide exercises and a chance to investigate topics not fully covered in the text itself. Finally we collect our rules in one place for handy reference.

A word on the sources of the examples: *all* of the programs we use are taken from programming textbooks. Thus, we do not set up artificial programs to illustrate our points — we use finished products, written and published by experienced programmers. Since these examples are typically the first code seen by a novice programmer, we would hope that they would be models of good style. Unfortunately, we sometimes find that the opposite is true — textbook examples often demonstrate the state of the art of computer programming all too well. (We have done our best to play fair — we don't think that any of the programs are made to look bad by being quoted out of context.)

Let us state clearly, however, that we intend no criticism of textbook authors, either individually or as a class. Shortcomings show only that we are all human, and that under the pressure of a large, intellectually demanding task like writing a program or a book, it is much too easy to do some things imperfectly. We have no