Texts and Monographs in Computer Science

Niklaus Wirth

Programming in Modula-2

Fourth Edition

Springer-Verlag
Berlin Heidelberg NewYork
London Paris Tokyo

Author

Professor Niklaus Wirth
Institut für Informatik, ETH, CH-8092 Zürich

Series Editor

Professor David Gries
Department of Computer Science, Cornell University
Ithaca, NY 14853, USA

The 1st edition was published 1982 as a monograph

ISBN-13:978-3-642-83567-4 e-ISBN-13:978-3-642-83565-0
DOI: 10.1007/978-3-642-83565-0

2145/3140-543210 Printed on acid-free paper

Preface

This text is an introduction to programming in general, and a manual for programming with the language Modula-2 in particular. It is oriented primarily towards people who have already acquired some basic knowledge of programming and would like to deepen their understanding in a more structured way. Nevertheless, an introductory chapter is included for the benefit of the beginner, displaying in a concise form some of the fundamental concepts of computers and their programming. The text is therefore also suitable as a self-contained tutorial. The notation used is Modula-2, which lends itself well for a structured approach and leads the student to a working style that has generally become known under the title of *structured programming.*

As a manual for programming in Modula-2, the text covers practically all facilities of that language. Part 1 covers the basic notions of the variable, expression, assignment, conditional and repetitive statement, and array data structure. Together with Part 2 which introduces the important concept of the procedure or subroutine, it contains essentially the material commonly discussed in introductory programming courses. Part 3 concerns data types and structures and constitutes the essence of an advanced course on programming. Part 4 introduces the notion of the module, a concept that is fundamental to the design of larger programmed systems and to programming as team work. The most commonly used utility programs for input and output are presented as examples of modules. And finally, Part 5 covers facilities for system programming, device handling, and multiprogramming. Practical hints on how and when to use particular facilities are included and are intended as guidelines for acquiring a sound style of programming and system structuring.

The language Modula-2 is a descendant of its direct ancestors Pascal [1] and Modula [2]. Whereas Pascal had been designed as a general purpose language and after implementation in 1970 has gained wide usage, Modula had emerged from experiments in multiprogramming and therefore concentrated on relevant aspects pertinent to that field of application. It had been defined and implemented experimentally by 1975.

In 1977, a research project with the goal to design a computer system (hardware *and* software) in an integrated approach, was launched at the Institut für Informatik of ETH Zürich. This system (later to be called Lilith) was to be programmed in a single high-level language, which therefore had to satisfy requirements of high-level system design as well as those of low-level programming of parts that closely interact with the given hardware. Modula-2 emerged from careful design deliberations as a language that includes all aspects of Pascal and extends them with the important module concept and those of multiprogramming. Since its syntax was more in line with that of Modula than with Pascal's, the chosen name was Modula-2. We shall subsequently use *Modula* as synonym for Modula-2.

The language's main additions with regard to Pascal are:

1. The *module* concept, and in particular the facility to split a module into a *definition part* and an *implementation part.*

2. A more systematic syntax which facilitates the learning process. In particular, every structure starting with a keyword also ends with a keyword, i.e. is properly bracketed.
3. The concept of the *process* as the key to multiprogramming facilities.
4. So-called *low-level facilities* which make it possible to breach the rigid type consistency rules and allow to map data with Modula-2 structure onto a store without inherent structure.
5. The *procedure type* which allows procedures to be dynamically assigned to variables.

A first implementation of Modula-2 became operational on the PDP-11 computer in 1979, and the language's definition was published as a Technical Report in March 1980. Since then the language has been in daily use in our institute. After a year's use and testing in applications, the compiler was released for outside users in March 1981. Interest in this compiler has grown rapidly, because it incorporates a powerful system design tool implemented on widely installed minicomputers. This interest had given rise to the need for this manual and tutorial. The defining report is included at the end of the manual, primarily for reference purposes. It has been left unchanged, with the exception that the chapters on standard utility modules and on the use of the compiler have been omitted.

This text has been produced in camera-ready form by a Lilith minicomputer connected to a Canon LBP-10 laser printer. Concurrently with the writing of the book, the author designed the programs necessary for automatic text formatting (and controlling the printer) and designed the interface connecting the printer. Naturally, all these programs have been written in Modula (for Lilith).

It is impossible to properly acknowledge all the influences that contributed to the writing of this text or the design of Modula. However, I am particularly grateful for the inspiring influence of a sabbatical year (1976) at the research laboratory of Xerox Corporation (PARC), and for the ideas concerning modules presented by the language Mesa [3]. Perhaps the most important insight gained was the feasibility of implementing a high-level language effectively on minicomputers. My thanks are also due to the implementors of Modula, notably L. Geissmann, A. Gorrengourt, Ch. Jacobi and S.E. Knudsen, who not only have turned Modula into an effective, reliable tool, but have often wisely consulted against the inclusion of further fancy facilities.

Zürich, February 1982 N. W.

Preface to the 3rd Edition

This 3rd Edition has been adapted to the few amendments and revisions of Modula-2 introduced in late 1983. The one essential change concerns definition modules, which no longer contain an export list, but are rather regarded as constituting an export list themselves (s. Chapter 24). A few standard modules which have proven to be widely useful are included in an additional appendix. They mainly concern the subject of input and output, i.e. the use of a keyboard, the display, and a file system.

Zürich, September 1984 N. W.

Preface to the 4th Edition

The principal improvement in this 4th Edition lies in the layout and the printing font. I am most grateful to H. E. Meier, who designed the *Syntax* font and transformed it into electronic form using our Lilith workstation and its font design system.

The main change in the contents is the use of the data type INTEGER instead of CARDINAL in most program examples. It reflects the fact that many new implementations of Modula treat CARDINAL as a subrange of INTEGER, thereby avoiding nasty type incompatibilities between INTEGER and CARDINAL operands in expressions. Accordingly, the definition of the language has also undergone a few minor adaptations in the area of Standard Functions:

FLOAT(i) and CHR(i) accept an argument of type INTEGER
TRUNC(x), HIGH(a), ORD(ch), and SIZE(T) are of type INTEGER

And finally, the language rule about assigning a string s to an array a of characters is made more restrictive. The assignment a := s is acceptable, if the length of s is strictly less (rather than less or equal) to the number of elements of a. This ensures that a string is always terminated by a 0C character, simplifying tests for the end of a string.

Zürich, July 1988 N. W.

References

1. N.Wirth: The programming language PASCAL. *Acta Informatica 1*, 35-63 (1971).
2. N.Wirth: Modula: A language for modular multiprogramming. *Software - Practice and Experience 7*, 3-35 (1977).
3. J.G.Mitchell, W. Maybury, R.Sweet: Mesa Language Manual. Xerox PARC, CSL-78-1 (1978).

Contents