MODULA-2
程序设计教程

艾德才 译
翁瑞琪 校

天津大学出版社

MODULA-2

程序设计教程

艾德才　译

翁瑞琪　校

天津大学出版社

内容提要

MODULA 2程序设计语言是由N. Wirth在PASCAL语言的基础上发展起来的一种结构化程序设计语言，是PASCAL的直接后继语言。本书共21章，全面系统地介绍了MODULA-2 的数据类型、各种语句、各种模块、程序结构、程序设计的基本原理及程序设计方法学等，尤其对MODULA-2所扩充的模块概念、进程概念、MODULA-2 的低级设施、过程类型等都列专题进行了详细论述。

本书内容安排由浅入深，循序渐进，简单易懂，且论证严谨。

本书是一本MODULA-2 程序设计语言的教科书，可作为高等院校计算机软件、硬件专业教材或参考书，也可供计算机系统软件、应用软件及其他工程技术人员参考。

MODULA-2 程序设计教程

艾德才　译

翁瑞琪　校

*

天津大学出版社出版

（天津大学内）

河北省永清县印刷厂印刷

新华书店天津发行所发行

*

开本：850×1168　毫米$^{1}/_{32}$　印张：$11^{3}/_{4}$字数：300千字

1988年11月第一版　1988年11月第一次印刷

印数：1—4000

ISBN 7-5618-0116-5

TP·17

定价：2.80元

译者前言

MODULA-2程序设计语言是从PASCAL派生出来的新语言。MODULA-2语言中的许多概念和构造，直接反映了十年来Niklaus Wirth在软件工程中产生和形成的主要概念与成果。MODULA-2不仅继承了PASCAL语言的所有优点，象典型的控制结构、数据类型和过程等概念，而且对PASCAL加以改进，并在如下几个方面作了扩充。

（1）MODULA-2引进了重要的模块概念。模块是获得模块性、可靠性、可读性、可补充性、可重复使用性和不依赖于机器等性能的重要手段。模块化是当今程序设计发展的必然趋势，可以使系统功能保持局部化，可以使系统比较方便地分解成可管理的一个个模块。模块也反映了数据防护的需要，可确保模块中数据的安全性。由于模块的实现，才能实现分块编译。

（2）引入了进程概念。进程是实现多道程序设计功能的关键。

（3）使语法更加系统，从而便利了学习过程。特别是，每一结构都以一个关键字开始，又以一个关键字结束。也就是说，每一结构都被适当地括了起来。

（4）MODULA-2扩充了所谓的低级设施。虽然用高级语言以结构化方式可设计出结构非常清晰的程序，但对于用绝对地址访问存储器或进行设备管理等，高级语言却无能为力。MODULA-2很巧妙地将与机器和实现有关的低级设施放在伪模块SYSTEM中，用户可以象使用其他模块那样使用它。这就为编制系统软件和并发软件提供了极大方便。

（5）MODULA-2所扩充的过程类型，允许将过程动态地赋给变量。

MODULA-2是一种结构化的程序设计语言，它不仅可用作通用程序设计语言，还可用作系统设计语言，可以说是80年代的程序设计语言。由于MODULA-2是PASCAL的直接后继语言，所以对已熟悉PASCAL语言的程序设计者来说，掌握MODULA-2程序设计语言便是轻而易举的事。

在欧洲，MODULA-2语言已相当普及，英国的许多大学已把MODULA-2作为计算机科学系本科生的第一门程序设计语言课。据信当今世界上已有几十种MODULA-2编译系统在各种不同的计算机系统上运行。

译者认为，鉴于MODULA-2具有其他高级语言无法比拟的众多优点，随着人们对MODULA-2更深入的理解，象PASCAL最终取代ALGOL一样，MODULA-2替代PASCAL也将成为必然趋势。MODULA-2热在我国会很快到来。

本书是根据Daniel Thalman著《MODULA-2：an introduction》1985年版翻译的。书中对模块、过程和数据结构等重要概念进行了详尽解释，并且还特别强调了程序设计方法学的有关内容。书中的每一概念都经过了严格的证明。语言的语法都用语法图作了解释。每一章都有许多程序实例，每章末附有摘要。书中的程序作者已在AppleⅡ、IBMPC和VAX11/780等计算机上运行过。

本书深入浅出、论证严谨，是一本较好的MODULA-2程序设计语言的教科书。它可作为高等院校计算机软件、硬件以及其他有关专业的教材和教学参考书，也可供从事计算机及计算机应用工作的科研人员、工程技术人员学习参考。

由于译者水平有限，错误在所难免，欢迎读者批评指正。

译者

1987年4月30日于天津大学

目 录

第一章 绪 论

§1 程序设计的作用和起源

计算机硬件本身的能力是有限的。

为增强其能力，必须编制程序。硬件必须接受指令和理解指令。此外，必须把许多基本指令预先储存在计算机内。这些基本指令称为计算机的基本软件。这些指挥计算机的指令不能随机地、无逻辑地或无规则地送入计算机。给计算机编制程序，需要使用程序设计语言。

早在1801年，法国人Jacc rd就发明了一种织布机。它借助卡片上的穿孔代码控制机器，织出图案。这种织布机用一组指令编程，不用人工干预就可工作。

在19世纪初期，英国人Charles Babbage设计了一台机器，通常被认为是第一台计算机。这台机器称之为分析机，其存储容量为1000个数。程序用穿孔卡片输入。Babbage的计划由于缺乏资金而失败。然而他的思想却是计算机发展的起源。

在计算机发展过程中最关键的一步无疑是存储程序的概念。其主要思想是：可把程序编成数字代码，也可把它象数据一样地储存在存储器中。甚至当今的计算机仍然以存储程序概念为基础。

§2 程序设计语言的演变

第一批计算机用二进制语句编制程序。显然，与这样的计算机对话颇为麻烦，也常见错误。为了改善这种状况，John Backus

和他的小组在1954年设计了一种符号程序设计语言，名为FORTRAN（FORmul TRANsl tion）。该语言主要是为科学计算开发的。由于计算机只理解（二进制的）机器语言，因此需要研制把FORTRAN语言翻译成机器语言的程序。这些程序就是编译程序。虽然FORTRAN经过若干次重大修改，但至今仍是非常通用语言。1959年，在商业界的压力下，美国政府把计算机用户组织和制造厂家汇集一起，设计出商用程序设计语言。该语言在1961年推出，称为COBOL（COmmon Business Qrienled L ngu ge）。COBOL虽没有FORTRAN那样的数学本领，但它除了具有FORTRAN中已有的数组概念外，还提供分层记录的概念。控制语句更加结构化（IF…ELSE，PERFORM…UNTIL），并大大发展了文件处理功能。

1960年设计的ALGOL语言是现代块结构语言的先辈。该语言引进若干重要的概念，例如递归、参数传送方式和局部变量。不幸的是，ALGOL没能得到广泛的应用。

在另一方面，MIT的John Mc Carthy设计了一种以表处理为基础的语言，叫LISP。它是面向人工智能的语言，应用很普遍，在专家系统领域内尤其如此。它对于PROLOG语言有很大的影响，PROLOG是日本人为他们的第五代计算机选中的语言。

Dartmouth学院的John Kemeny和Thomas Kurtz引进了一种非常简单但十分有限的语言，叫BASIC。该语言没有提供任何新的语言概念。然而，它是第一个真正的交互式语言。

60年代后期结构化程序设计思想变得非常普遍。这主要是因为出现了软件开发危机，程序越来越大，越来越不易读，而且充满错误。结构化程序设计的主要思想着重于数据结构、自顶向下的程序设计和较好的控制结构的设计。特别是，Dijkstr 强调了GOTO语句所引起的一些问题。这些思想导致一些新语言的设计，其中之一就是PASAL语言。

PASCAL是由Niklaus Wirth设计的。它提供了卓越的控

制结构、用户定义的数据类型和正交性。由于该语言简单，所以编译程序很快研制出来。目前，在大学里PASCAL语言是非常普遍的。大多数微型计算机和个人计算机上都可使用PASCAL。

MODULA- 2语言是1977年由Wirth设计的。它包括PASCAL的所有方面，且对其作了某些改进。此外，它还提供了重要的模块概念，多道程序设计能力和以优美的方式实现低级软件的手段。因此，MODULA- 2可用作通用程序设计语言，也可用作系统实现语言。

§3 “模块”概念

任何程序设计语言至少有以下三个重要性质：模块性、独立于机器和可扩充性。

模块性是能把一个大程序分成若干个可由几个程序员独立编程部分的性质。这就要适当地定义不同部分之间的接口。

MODULA- 2给程序员提供了用来获得模块性的有力工具。它可把对象和过程封闭在一个模块中。一个模块分成两部分：一个定义部分和一个实现部分。定义部分对应于与其他模块进行通信所要求的接口。实现部分可以且应该由程序员独自负责。

独立于机器是可移植性的先决条件。用于一台机器或一个操作系统的专门程序很难转移到另一台机器上去。在MODULA- 2中把依赖于机器的概念集中在模块的实现部分，因而依赖于机器的性质可显著改善。

MODULA- 2是一种结构清晰、界限分明的语言。然而，程序员往往喜欢对语言进行扩充。通过编写一些模块，就可容易地扩充MODULA- 2语言，而其基本结构并不因此改变。

§4 词法、语法和语义分析

虽然MODULA-2提供了一些低级特征，但它是一种高级程序设计语言，且和机器语言完全不同。也就是说任何MODULA-2程序都必须通过编译程序翻译成机器语言。

编译程序完成如下任务：

1.读程序并且检出词法错误。例如，经识别PROCEDURE这个字属于MODULA-2词汇表，而PROZEDURE不属于MODULA-2词汇表。

2.检查程序语法是否正确。为此，必须核对是否违反了该语言的语法规则。例如，前置一个左括号的表达式后面是否跟着一个右括号。

3.保存程序所用的符号表，并且检查所用操作是否与操作对象的类型相容。例如，一个整数与一个字符串相加是无意义的。这叫语义分析。

4.产生对应于源程序（源代码）的机器代码，并且应优化机器代码。

为进行词法、语法和语义分析，编译程序必须知道该语言的所有规则。为避免错误，用户也必须学习这些规则。这意味着，程序设计语言的任何描述都应强调它的词法、语法、语义规则和定义。下面将非正式地给出MODULA-2的语义规则，因为对语言的语义进行描述尚无简单的正式方法。词法规则非常有限，这将在下一节中讨论。描述语法的方法在第6节中给出。

§5 MODULA-2词汇表

MODULA-2词汇表中有六类符号：

1）标识符

标识符是英文字母和数字的序列。不含其它字符（连字符、空格等）。第一个字符必须是英文字母，可以使用大写字母和小写字母。大写字母和小写字母被看作是有区别的。

合法的例子：V.ness. MyDaughter JamesBondoo 7

非法的例子： 7DW rfs NEW YorK

一个标识符可由另一个标识符限定。

在这种情况下，标识符间必须用一个圆点（句号）隔开。例如：

Person·.ge　　Cow·Tail

2）数

可以有两种数，在第二章§1中给出。

3）字符串

字符串是任何用双引号或单引号括起来的字符序列。例如：“Hello” ‘Peace or War?’

用双引号括起来的字符串内不能再包含有其他双引号。同样，在用单引号括起来的字符串内也不允许有单引号。例如，以下是合法的字符串：

“It's You” ‘He Said: “You are Stupid”’

4）特殊字符

特殊字符用来表示操作符和定界符，可以由单个字符构成，例如：+、-、*、(,)；也可以由两个其间不带空格的字符构成，例如：<= : =。

5）关键字

MODOLA- 2包括若干关键字。它们不能作标识符用。关键字用大写字母书写，在本语言中有特定的作用。图 1.1 中给出MODULA- 2的关键字表。

AND	ELSIF	LOOP	REPEAT
ARRAY	END	MOD	RETURN
BEGIN	EXIT	MODULE	SET
BY	EXPORT	NOT	THEN
CASE	FOR	OF	TO
CONST	FROM	OR	TYPE
DEFINITION	IF	POINTER	UNTIL
DIV	IMPLEMENTATION	PROCEDURE	VAR
DO	IMPORT	QUALIFIED	WHILE
ELSE	IN	RECORD	WITH

图1.1 保留字表

6) 注释

注释是括在一对注释号（*和*）内的字符序列。注释用来帮助程序员（或他人）理解自己的程序。

例如：

```
(*This comment is especially useful for you*)
```

§6 语法图

语法图用来描述MODOLA-2的语法。

语法图是一些含有正文的框的组合。这些框用箭头连接起来。框有如下两种：

1) 矩形框，例如，|语句|

2) 椭圆框，例如，|BEGIN|

矩形框表示非终止符号，椭圆框表示终止符号。一个终止符号被定义成一个基本符号，终止符号对应于前一节描述的特殊字符和关键字。非终止符号是可用语法规则描述的符号。任一非终止符号都有一个用以描述它的语法图。

箭头的作用是表示符号如何组合的。例如，符号可以任选，也可以重复。为了弄清语法图是怎样构成的，现在看下面的小例

子。假设有一种语言，它只允许使用下列表中一个或多个句子：

Melanie is nice

Vanessa is nice

Melanie is funny

Vanessa is funny

但这些句子可以重复任意次。这个简短语言的语法规则可以用图 1.2语法图描述。

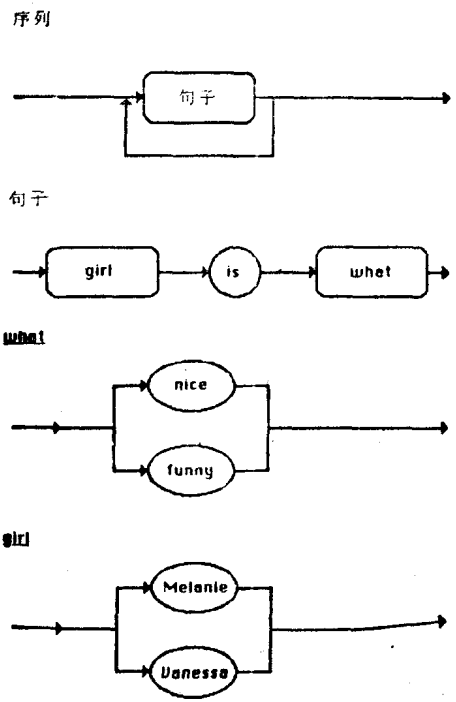

图1.2　简短语言的语法图

可以看到，在这几个语法图中，有一个顺序性语法图(句子)、一个重复性语法图（序列）和两个选择语法图（girl、what）。当然，也还有另外一些可能的组合，如图 1.3所示。

在本书中，MODOLA- 2的全部语法将用语法图给出。

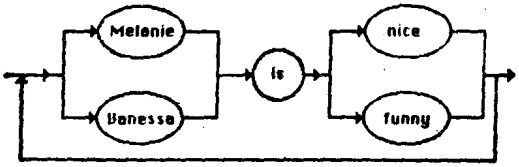

图1.3 较复杂的语法图

§7 MODULA-2程序结构

事实上，MODULA- 2是一个模块（module）。它是一个主模块，并且要用一个名字(name)或标识符（identifier）表征。图 1.4示出模块的语法图。

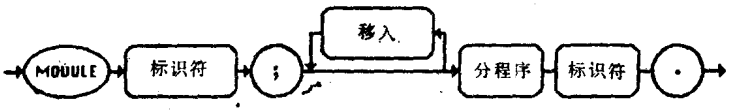

图1.4 模块的语法图

模块的名字必须在模块末尾予以重复。而且主模块必须用一句号终止。下一节将会看到，可对其他模块进行访问。方法是把一些对象从其他模块移入（import）。实际上，模块常包括一个分程序（block）。此分程序由说明（declaration）和语句(statement）序列构成，语句序列用BEGIN和END这两个关键字括起来，图1.5中的说明和语句将在后几章讨论。

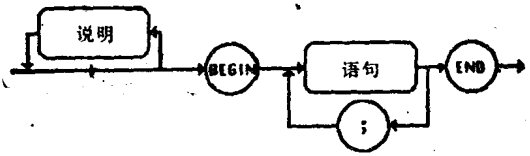

图1.5 分程序语法图