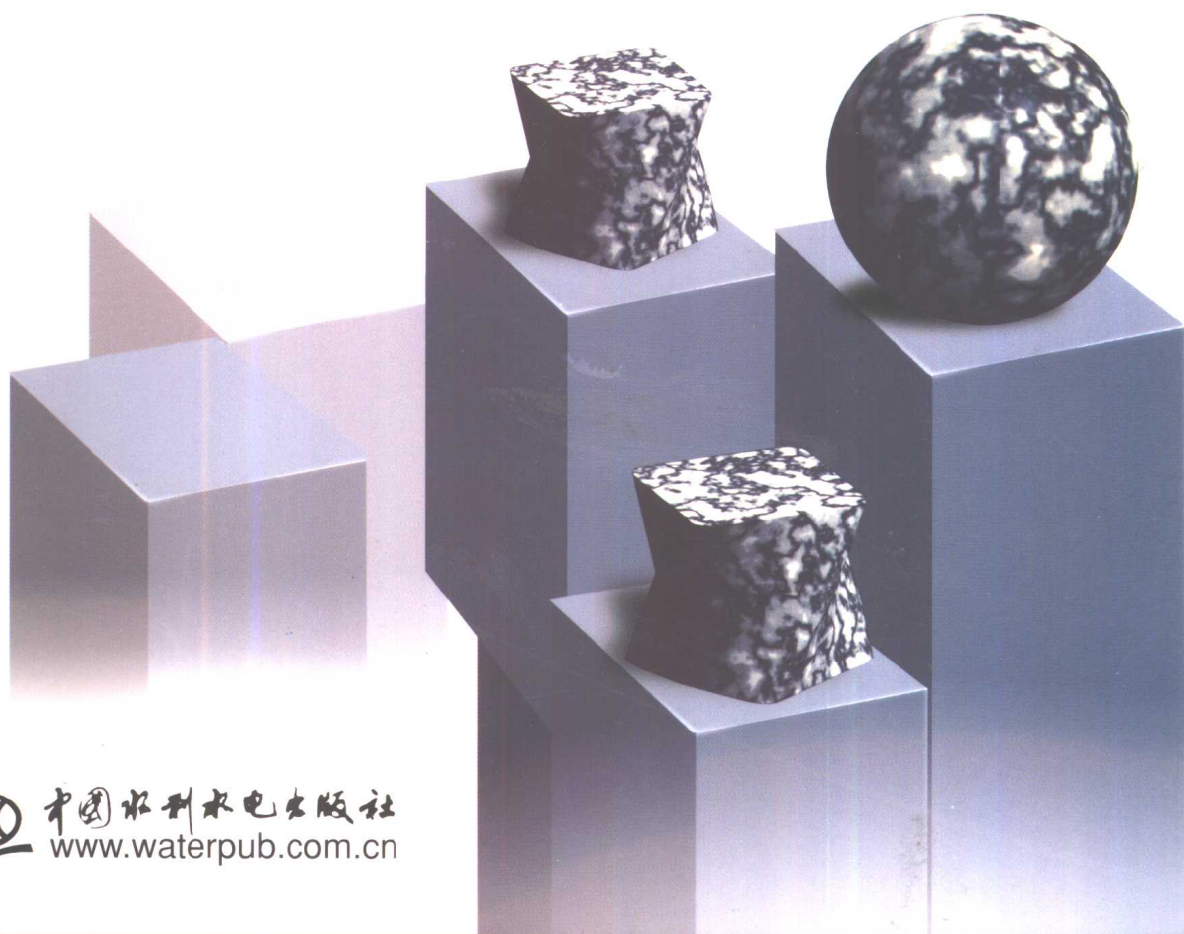


高等学校计算机基础课程系列教材

吴昊 张邦明 张小峰 等 编著

Quick BASIC

结构化程序设计



中国水利水电出版社

www.waterpub.com.cn

内 容 提 要

Quick BASIC 是 Microsoft 公司在 1987 年推出的新的 BASIC 版本。它具有交互性及模块化结构的特点,是初学者首选的入门级程序设计语言。

本书详尽介绍了 Quick BASIC 语言的基本知识、程序结构、语法规则、基本语句、常用算法、上机操作等部分。另附综合实验习题。

本书可作为高等院校计算机基本教育教材使用,同时可运用于多类工程技术人员,并适用于自学或作为参考。

图书在版编目(CIP)数据

Quick BASIC 结构化程序设计/ 吴昊等编著. —北京:中国水利水电出版社, 1999.9

(高等学校计算机基础课程系列教材)

ISBN 7-5084-0105-0

I. Q… II. 吴… III. BASIC 语言-程序设计-高等学校-教材 IV. TP312

中国版本图书馆 CIP 数据核字(1999)第 39570 号

书 名	Quick BASIC 结构化程序设计
作 者	吴昊 张邦明 张小峰 等编著
出版、发行	中国水利水电出版社(北京市三里河路 6 号 100044) 网址: www.waterpub.com.cn E-mail: sale@waterpub.com.cn 电话: (010) 63202266(总机)、68331835(发行部)
经 售	全国各地新华书店
排 版	北京万水电子信息有限公司
印 刷	北京市天竺颖华印刷厂
规 格	787×1092 毫米 16 开本 18.25 印张 408 千字
版 次	1999 年 9 月北京第一版 1999 年 9 月北京第一次印刷
印 数	0001—5000 册
定 价	22.00 元

凡购买我社图书,如有缺页、倒页、脱页者,本社发行部负责调换
版权所有·侵权必究

前 言

近年来,随着计算机科学技术的迅猛发展,计算机已经被越来越多的人所认识、接受。当今社会的各个领域,都在普遍地使用着计算机。掌握计算机技术及应用已成为人们工作、生活中的重要组成部分。

BASIC 自 1963 年出现以来,就以其简单明了、功能强而吸引着广大的计算机爱好者。在 80 年代我国的高校更是将 BASIC 语言作为非计算机专业学生的首选程序设计课程。谭浩强的一本《BASIC 语言》似乎成了一盏将人们引入计算机应用殿堂的明灯,至今有许多人对此仍记忆犹新。

随着计算机技术的发展,软件更新的速度更是迅猛。在程序设计语言方面,也出现了多样化的特点。但不同的工作性质、不同爱好的人们在学习程序设计语言时,都不约而同地选择 BASIC 作为入门语言。

Quick BASIC (QB) 是微软公司于 1987 年推出的。它吸取了 Pascal、C 的优点,集解释型和编译型为一体,具有完善的功能,拥有良好的集成开发环境和用户界面。是一种使程序设计向数据结构和模块化设计方向转化与提高的程序设计语言。

本书编著人员都是在各高校中讲授 QB 多年的教师,具有丰富的教学经验。本书通俗易懂,不仅提供了丰富的例题讲解及习题,还提供了上机指导及实验题,是学习 QB 的理想书籍。

本书由华东交通大学吴昊老师任主编,负责全书编写大纲的编写及组织统稿工作,华东交通大学教师张邦明与南昌航空工业大学教师张小峰担任副主编,华东交通大学教师熊李艳及福建林学院教师陈琼担任编委。其中:吴昊承担前言、第二章、第八章、上机指导及上机实验的编写,张邦明承担第五章及附录 2、3、4、5 的编写,张小峰承担第四章和第七章的编写,熊李艳承担第一章、第三章、第六章的编写,陈琼承担第九章、第十章、第十一章及附录 1 的编写。

本书在编写过程中,得到了华东交通大学计算机基础教研室及计算中心的大力支持,在此表示衷心感谢。为了满足教学需要,教材编写时间紧迫,难免有不当与错误之处,恳请读者指正。

编者

1999 年 6 月 28 日

目 录

第一章 Quick BASIC 的基本知识	1
1.1 Quick BASIC 简介	1
1.1.1 计算机语言	1
1.1.2 BASIC 的发展	2
1.1.3 Quick BASIC 的特点	3
1.2 Quick BASIC 程序结构	3
1.3 语言元素	6
1.3.1 字符集	6
1.3.2 标识符	6
1.4 数据类型	7
1.4.1 基本数据类型	7
1.4.2 用户自定义的数据类型	7
1.5 常量与变量	8
1.5.1 常量	8
1.5.2 变量	10
1.5.3 运算中数值精度的类型转换	11
1.6 运算符与表达式	12
1.6.1 算术运算	12
1.6.2 关系运算	14
1.6.3 逻辑运算	14
1.6.4 字符串运算	15
1.6.5 表达式的混合运算规则	15
1.7 内部函数	16
习题	20
第二章 顺序结构程序设计	22
2.1 数据的输入	23
2.1.1 键盘输入语句	23
2.1.2 读数语句与置数语句	25
2.1.3 恢复数据区的语句	27
2.2 输出	29
2.2.1 屏幕显示语句	30
2.2.2 WRITE 语句	33
2.2.3 PRINT USING 语句	34
2.2.4 赋值语句	37
2.2.5 注释语句	39

2.2.6	暂停语句	40
2.2.7	结束语句	40
2.2.8	交换语句	40
2.2.9	光标定位语句	41
2.2.10	BEEP 语句	41
2.2.11	SLEEP 语句	41
2.2.12	CLEAR 语句	42
2.2.13	COLOR 语句	42
2.2.14	VIEW PRINT 语句	43
2.2.15	清屏语句	43
	习题	44
第三章	字符串	46
3.1	字符串常量与变量	46
3.1.1	字符串常量	46
3.1.2	字符串变量	46
3.2	给字符串变量赋值	47
3.2.1	用赋值语句给字符串变量赋值	47
3.2.2	用 LINE INPUT 语句给字符串变量赋值	48
3.2.3	用 INPUT 语句给字符串变量赋值	48
3.2.4	用 READ/DATA 语句给字符串变量赋值	49
3.3	字符串的比较	50
3.4	字符串函数	50
3.4.1	求字符串长度的函数 (LEN 函数)	50
3.4.2	字符串与数值之间转换的函数	51
3.4.3	求子串的函数	51
3.4.4	删除字符串首尾空格的函数 (LTRIM\$ 函数和 RTRIM\$ 函数)	52
3.4.5	查子串位置的函数 (INSTR 函数)	52
3.4.6	大小写字母间的转换函数 (UCASE\$ 函数和 LCASE\$ 函数)	53
3.4.7	ASCII 码与字符之间的转换函数	53
3.4.8	串生成函数	54
3.4.9	日期和时间函数	55
3.5	举例	55
	习题	56
第四章	算法与流程图	59
4.1	算法的概念	59
4.1.1	程序设计步骤	59
4.1.2	算法	60

4.2 算法的表示	61
4.2.1 传统流程图.....	61
4.2.2 N-S 结构化流程图	62
4.2.3 结构化程序设计.....	68
习题	74
第五章 选择结构程序设计	75
5.1 单行条件语句 (IF 语句)	75
5.1.1 语句形式	75
5.1.2 执行过程	75
5.1.3 语句说明	75
5.1.4 应用举例	76
5.1.5 单行条件语句的嵌套.....	78
5.2 块结构的条件语句 (块 IF 结构)	81
5.2.1 块结构 IF 语句的一般形式.....	81
5.2.2 执行过程	81
5.2.3 结构说明	81
5.2.4 应用举例	82
5.2.5 块 IF 的嵌套.....	84
5.2.6 在块 IF 结构中使用 ELSEIF.....	87
5.3 多分支选择结构 (SELECT CASE)	91
5.3.1 SELECT CASE 结构的一般格式.....	91
5.3.2 执行过程	91
5.3.3 SELECT CASE 结构中的“检测表达式”和“表达式”	92
5.3.4 应用举例	93
5.3.5 在 CASE 子句中使用多个条件	94
5.4 转移控制语句	95
5.4.1 无条件转移语句 (GOTO 语句)	95
5.4.2 多分支选择结构转移语句 (ON—GOTO 语句)	97
习题	98
第六章 循环结构程序设计.....	101
6.1 条件循环	101
6.2 FOR—NEXT 循环	102
6.3 WHILE—WEND 循环.....	107
6.4 DO—LOOP 循环	110
6.4.1 当型循环	110
6.4.2 直到型循环.....	111
6.5 退出循环	114

6.6 多重循环	118
6.7 举例	125
习题	134
第七章 数组	136
7.1 数组的概念与定义	136
7.1.1 一维数组和数组元素	136
7.1.2 二维数组和数组元素	139
7.2 静态数组与动态数组	141
7.2.1 静态数组与动态数组	141
7.2.2 动态数组的重新定义	141
7.3 实例	142
习题	145
第八章 函数与过程	148
8.1 自定义函数	148
8.1.1 单行自定义函数	148
8.1.2 多行形式	151
8.2 子程序	152
8.2.1 转子语句与返回语句	153
8.3 过程的定义	159
8.3.1 SUB 过程的定义	159
8.3.2 FUNCTION (函数) 过程的定义	161
8.4 过程的说明与调用	162
8.4.1 过程的说明	162
8.4.2 过程的调用	163
8.5 参数传递	166
8.6 过程的递归调用	171
8.7 变量的生存期和作用域	173
8.7.1 变量的作用域	173
8.7.2 变量的生存期	176
8.8 多模块调用与程序的连接	178
8.8.1 多模块调用	178
8.8.2 程序的连接	180
习题	183
第九章 文件处理	185
9.1 文件的概念	185
9.1.1 文件的概念和分类	185
9.1.2 数据文件的结构	186

9.2 顺序文件	186
9.2.1 顺序文件的基本操作	187
9.2.2 顺序文件的输出	189
9.2.3 顺序文件的输入	191
9.2.4 文件操作语句和函数	193
9.2.5 顺序文件的简单维护	195
9.3 随机文件	197
9.3.1 随机文件的基本操作	197
9.3.2 随机文件的输出	198
9.3.3 随机文件的输入	202
习题	205
第十章 图形处理	207
10.1 屏幕坐标系统	207
10.1.1 屏幕显示模式	207
10.1.2 屏幕坐标系统	207
10.2 屏幕控制语句	209
10.3 基本绘图语句	213
10.3.1 颜色的设置	213
10.3.2 基本绘图语句	214
10.4 图形着色	219
10.5 图形窗口	223
10.6 实例	226
习题	230
第十一章 库管理	232
11.1 库的概念与使用	232
11.1.1 库的概念	232
11.1.2 库的使用	233
11.2 Quick BASIC 环境下建库	235
11.2.1 建立一个新的 Quick 库	235
11.2.2 扩充原有的 Quick 库	237
习题	238
第十二章 Quick BASIC 4.5 上机实验指导	239
12.1 屏幕窗口	239
12.1.1 窗口介绍	239
12.1.2 屏幕窗口基本名词及基本构成	240
12.2 菜单操作	241
12.2.1 File 菜单	241

12.2.2	Edit 菜单.....	245
12.2.3	View 菜单.....	247
12.2.4	Search 菜单.....	248
12.2.5	Run 菜单.....	250
12.2.6	Debug 菜单.....	252
12.2.7	Call 菜单.....	254
12.2.8	Option 菜单	254
12.2.9	Help 菜单.....	256
第十三章	Quick BASIC 上机实验.....	257
实验 1	Quick BASIC 上机操作入门练习	257
实验 2	基本输入输出的练习.....	259
实验 3	选择结构程序设计.....	262
实验 4	循环结构程序设计.....	262
实验 5	数组的应用	263
实验 6	字符串的应用	264
实验 7	函数的应用	264
实验 8	过程的应用	264
实验 9	数据文件的应用	265
实验 10	图形的应用	266
实验 11	库的应用	266
实验 12	综合设计	266
附录 1	Quick BASIC 保留字	268
附录 2	Quick BASIC 4.5 语句一览表	270
附录 3	Quick BASIC 4.5 函数一览表	275
附录 4	程序运行错误的错误代码及其含义	278
附录 5	字符与 ASCII 代码对照表	280

第一章 Quick BASIC 的基本知识

在这一章中，我们介绍计算机语言的发展和典型的结构化程序设计语言 Quick BASIC，讲述 Quick BASIC 的特点及其程序结构、语言元素、数据类型以及运算量，要求掌握 Quick BASIC 表达式的书写以及标准函数的应用，为今后的学习打下基础。

1.1 Quick BASIC 简介

1.1.1 计算机语言

自从 1946 年美国宾夕法尼亚大学研制出第一台电子计算机 ENIAC 以来，在短短的 50 年里，计算机得到了突飞猛进的发展，经历了电子管、晶体管、集成电路和超大规模集成电路四个阶段。这个时期不管是硬件还是软件每隔数年就有一次重大的发展，这不仅仅使计算机的性能大幅度提高，同时使价格一降再降，给计算机的普及带来了极有利的条件。随着计算机技术的推广，计算机已经广泛地应用于各个领域，遍及社会各个方面，推动了国民经济的发展和科学技术的进步。

人们要利用计算机解决实际问题，首先就要编写程序，而计算机语言就是设计人员用来编写程序的语言，因此也称为程序设计语言。它是人与计算机进行信息交换的工具，也是计算机系统的重要组成部分。计算机语言一般分为三种：机器语言，汇编语言和高级语言。

1. 机器语言

计算机在解决问题时，都是通过依次执行若干简单、基本的操作来完成的。这些操作又是通过指令来描述的。每一条指令都是用二进制表示的，我们把用二进制数表示的指令称作机器语言，把用二进制数表示的指令所编写的程序称作机器语言程序。在指令代码中一般包括操作码和地址码，操作码代表的是计算机执行何种操作，地址码是代表操作对象。如：用 8088CPU 指令系统编写 3+4 机器语言程序为：

1011000000000011 表示将 3 送入累加器 AL 中

1011001100000100 表示将 4 送入寄存器 BL 中

1111011011100011 表示 AL 与 BL 中数据相乘，积放入 AX 中

由于不同的计算机（主要是 CPU）都有他规定的指令系统，而编程就是从指令系统中选择其中若干条指令组成一个文件。因此硬件不同，机器语言也不同，那么针对某种计算机所编写的机器语言程序就不能在另一种计算机上运行，所以说机器语言可移植性差。但是由于机器语言是用二进制进行编写的，计算机可以直接识别，因此执行效率高，能充分发挥计算机的性能。由于机器语言只有 1 和 0，因此难记、难理解，直观性较差，容易

出现错误。

2. 汇编语言

为了克服机器语言中难理解、难记忆等缺点，人们采用了能够帮助记忆的英文缩写符——指令助记符和地址编码来代替机器指令。

如：MOV AL 3 表示将 3 送入累加器 AL 中

MOV BL 4 表示将 4 送入寄存器 BL 中

MUL BL 表示 AL 与 BL 中数据相乘，积放入 AX 中

汇编语言和机器语言相比，直观，容易理解和记忆，因此用汇编语言所写的程序更容易阅读理解和修改，但汇编语言仍是从指令系统中挑选指令，它和机器语言一一对应，因此它还是一种面向机器的语言。

计算机不能直接识别用汇编语言编写的程序，必须由一种专门的翻译程序将汇编源程序翻译成机器语言后，计算机才能识别并执行。这种翻译的过程称为汇编，负责翻译的程序为汇编程序。

3. 高级语言

机器语言和汇编语言都是面向机器的语言，可移植性差，因此称其为低级语言。用它们开发程序要了解这种机器的指令系统，非专业技术人员难以胜任。随着计算机逻辑元件由晶体管代替了电子管，计算机从主要用于科学计算发展到数据处理、事物管理和工业控制等各个方面，从 50 年代中期人们就开始发展面向问题的程序设计语言，即高级语言。和其他语言相比，高级语言更接近自然语言，表达方式更接近被描述的问题，而且它与计算机指令系统无关，易于被人们接受，通用性和可移植性好。和汇编语言相似，高级语言编写的程序（即源程序），计算机不能直接识别，必须通过编译程序或解释程序翻译成机器语言（即目标程序），计算机才能执行。高级语言翻译成机器语言的方式有两种：翻译一句执行一句，这种翻译程序叫解释程序；把高级语言源程序翻译完毕形成目标程序再执行，这种翻译程序叫编译程序。大多数高级语言源程序翻译成目标程序属于编译这一类，在编译时，计算机总是先检查程序中是否有语法错误，检查无误后，再将源程序翻译成二进制代码的目标程序，以便让计算机执行。

计算机高级语言发展至今已有上百种，每一种语言都有其最适用的领域，其中，最常用的如：BASIC 语言，主要用于教学和小型应用程序的开发；FORTRAN 主要用于科学和工程计算程序的开发；C 往往用于中、小型系统程序的开发；目前较流行的 JAVA 语言是面向对象的程序的开发。

1.1.2 BASIC 的发展

BASIC (Beginner's All -Purpose Symbolic Instruction Code) 即初学者通用符号指令代码，自从 1964 年问世以来，就以“通俗易懂”成为人们初学计算机程序设计的首选语言。BASIC 的发展，大致可以划分为三个阶段：

1. 非结构化 BASIC 语言

最初的 BASIC 语言是在小型计算机上使用的，当初才十几条语句，由于简单实用受

到用户的普遍欢迎，很快在各种计算机上实现，被广泛应用到各个领域。由此，各个厂家先后以原有 BASIC 为基础，不断对其功能进行扩充和改善，各种 BASIC 版本应运而生。特别是 70 年代初，美国英特尔公司研制出第一个微处理器，开辟了微型计算机的时代，并且由当今微软（Microsoft）公司总裁比尔·盖茨（Bill Gates）将 BASIC 语言移植到微型计算机上，使 BASIC 功能越来越完善，使其成为人们广泛采用的程序设计语言。这时，比较有代表性的是 Gw BASIC 和 Ms BASIC。

2. 结构化 BASIC 语言

70 年代，随着结构化程序思想的提出，人们限制使用 GOTO 语句，要求程序结构化（程序由三种基本结构：顺序结构、选择结构和循环结构组成），功能模块化（根据功能分成若干模块），因此，出现了一些结构化 BASIC 版本。如 True BASIC、Quick BASIC、Turbo BASIC 等。

3. 面向对象的程序设计语言

80 年代初，Microsoft 公司推出了另一种操作系统——Windows，它的用户界面是图形方式。通过鼠标、窗口、菜单等操作计算机，和原来的磁盘操作系统（DOS）相比，使一些复杂、枯燥的操作变得简单、直观。第一个面向 Windows 的 BASIC 语言为 Visual BASIC，它的意思是可视的 BASIC，即图形界面的 BASIC。这时程序设计思想已从面向过程发展到面向对象。所谓面向对象的编程是指开发者把所要解决的问题分解成名叫对象的模块，这些对象包含数据和指令，能完成一定的功能。目前，面向对象技术主要用于面向对象的程序设计语言和面向对象的数据管理系统的研究和开发。

1.1.3 Quick BASIC 的特点

Quick BASIC 是 Microsoft 公司 1987 年推出的一种结构化程序设计语言，它具有如下特点：

- (1) 和 Gw BASIC 和 BASICA 等兼容，使一些传统的 BASIC 程序不做修改就可在其环境下运行。
- (2) 良好的程序开发的集成环境，全屏幕编辑，下拉式菜单操作。
- (3) 提供解释和编译方式；可产生可执行文件，提高了工作效率。
- (4) 提供全局变量、过程等，增强了语言功能。
- (5) 能自动检查每一行的语法并标识出错误，提供了丰富的联机服务。
- (6) 具有丰富的库管理功能。

同时，Visual BASIC 和 Quick BASIC 完全兼容，Quick BASIC 程序可在 Visual BASIC 环境下直接使用，为学习 Visual BASIC 打下基础。

1.2 Quick BASIC 程序结构

首先我们来看一个例子，从键盘上输入两个数，求两数之和。

【例】

```
REM 求两数之和
INPUT "A, B="; A, B
C=A+B
PRINT A; "+"; B; "="; C
END
```

从题中可知，Quick BASIC 程序是由若干行组成，每行由一个或多个语句构成，程序的结束用 END 来标识。

1. 注释语句

格式：REM <字符串>或 '<字符串>

注释语句中以 REM 开始，REM 是 REMARK 单词的缩写，可单独作为一条语句或放在某条语句的最后，用于解释、说明程序的名称和功能以及语句的作用等，目的是使程序更加容易阅读。REM 语句是一条非执行语句，计算机在编译时，不把它翻译成机器指令。

2. 程序行

格式：[语句 1][：语句 2]…[注释]

Quick BASIC 程序中的每一行，我们称之为程序行，每一行由一个或多个语句构成。程序行的最大行宽为 256 个字符。Quick BASIC 的编辑程序和其他编辑程序不同之处是不能用下划线作续行符。一般是一个语句占一行，如果想多个语句占一行，每个语句之间要用冒号隔开。

每个语句是由行标识符、语句和行结束语句三部分组成。

(1) 行标识符

行标识符可由行号或行标号来标识。

① 行号是范围在[0, 65529]之间的一个整数，在传统的 BASIC 版本中，语句运行的先后是根据其行号的大小执行，因此行号至关重要。但在 Quick BASIC 中程序的运行先后是按语句的书写顺序，而不是按行号大小，因此，行号一般可以省略，仅在较少的情况下才去使用。

【例】

```
20 S=3.14*R^2
10 R=2
PRINT S
40 END
```

这个程序的功能是求半径为 2 的圆的面积，但在 Quick BASIC 中运行的话结果为 0。分析其原因，就是因为在 Quick BASIC 中程序的运行是按书写顺序去执行的，执行第一条语句时，r 没有赋值，自动赋为零，因此 s 为零，再用第二条语句给 r 赋值，已没有意义，由此可见行号根本没有意义。

② 行标号是由 1 到 40 个由字母、数字组成的字符串组成的，它必须以英文字母开头，以冒号表示结束，它的功能和行号相同。在程序设计中，行标号一般用于流程的跳转。

(2) 语句

一个语句一般包括两个部分：语句定义符和语句体。

① 语句定义符：每一个语句都有一个语句定义符，它代表语句的具体操作动作，如 **PRINT** 就是用来表示输出其后所跟的常量、变量或表达式的值。语句定义符是系统定义的，我们不能人为地创造一个，人们不能用它来作为变量名。一般语句定义符都要书写。其中只有赋值语句 **LET** 可以省略。

② 语句体：用于表示语句的操作对象，Quick BASIC 中，大多数语句都含有语句体。如：**PRINT c**，这条语句中，c 代表输出对象，表示输出 c 的值。

Quick BASIC 中，语句分为执行语句和非执行语句两种。

执行语句用来完成某种特定操作，在以后的章节中我们主要介绍一些执行语句的具体应用。非执行语句提供或说明某些信息，如为变量分配存储单元，说明或定义变量类型等。常用的非执行语句有：

REM DATA DIM REDIMSTATIC COMMOM DECLARE OPTION BASE TYPE...END TYPE CONST DEFTYPE SHARED 等。

(3) END

表示程序的结束，程序在运行过程中遇到 **END** 语句就停止，因此往往放在程序的最后。在编译时，计算机总是假定 **END** 语句放在程序的最后，因此，不写 **END** 计算机也会正常结束。

在后面的章节中我们会讲述结构化程序设计，即一个程序是由若干个模块构成的。我们也可以把上面的例题改写成模块形式。程序如下：

```
DECLARE SUB ADD (X, Y)
INPUT "A, B="; A, B
CALL ADD (A, B)
END
SUB ADD (X, Y)
C=X+Y
PRINT X; "+"; Y; "="; C
ENDSUB
```

在这个程序中，包含了一个主模块和一个子模块。主模块中 **DECLARE** 语句是过程说明语句，过程都应该先说明后使用，**CALL** 语句是过程的调用语句，表示把流程转向过程去执行。**SUB.....ENDSUB** 是子程序模块，用于完成两数相加和数据的输出工作。

实际上一个程序往往是由一个或多个模块组成，其中必定包含一个主模块，零个或多个过程模块，采用这种设计，每个模块单独成立，易于修改和验证，增加了可阅读性，并且易于实现程序的共享。有关过程的定义和使用在有关过程的章节中详细讲解。

1.3 语言元素

Quick BASIC 中的语言元素包括两部分：字符集和标识符。

1.3.1 字符集

字符集是构成语言元素的最基本单位。共分英文字母、数字和专用字符三大类，分别是：

- (1) 字母：大写英文字母 A-Z；小写英文字母：a-z。
- (2) 数字：0-9；16 进制中，用英文字母 A-F(或 a-f)代表 10-15。
- (3) 专用字符集共 28 个。

<CR>	回车键	(	左圆括号
<SPACE>	空格	)	右圆括号
!	感叹号（单精度数据类型说明符）	*	星号（乘号）
#	镑号（双精度数据类型说明符）	/	斜杠（除号）
%	百分号（整型数据类型说明符）	+	加号
&	和号（长整型数据类型说明符）	-	减号
\$	美元号（字符串数据型说明符）	,	逗号
.	句号（小数点）	?	问号
:	冒号	@	花 a 号
:	分号	[	左方括号
'	单引号	]	右方括号
<	小于号	\	反斜杠（整除号）
=	等号	^	向上箭头（乘方号）
>	大于号	_	下划线

1.3.2 标识符

标识符是指用来区分常量名、变量名、函数名、类型名、过程文件名等而给它们取的一个名字，它由一系列有效字符序列构成，它分为两种，一种是关键字，另一种是用户自定义的标识符。

1. 关键字

关键字是指系统已经定义并具有某种具体含义的标识符，计算机通过它来执行某种操作，又称保留字。

Quick BASIC 编辑器在编辑时会自动将关键字中的字母全部转换为大写。

2. 用户定义的标识符

除保留字之外，用户用于给变量、符号常量、过程、行标号以及文件等命名的一系列字符为用户自定义标识符。Quick BASIC 语言规定标识符可由字母、数字、下划线三种字

符组成，且第一个字符必须为字母，一个标识符最多不超过 40 个字符。

1.4 数据类型

为准确、有效地处理数据，Quick BASIC 提供五种基本数据类型和一种用户自定义类型。

1.4.1 基本数据类型

1. 字符型数据

字符型数据是指任何可打印的字符，包括任意的 ASCII 字符和汉字。字符型数据又分为定长和不定长两种，其长度不能超过 32767 个字符。

2. 数值型数据

在 Quick BASIC 中数值型数据又分为整型数和浮点数两种。

① 整型数是指不带小数点和指数符号的数，它包括整数和长整数两种：

整数是用 2 个字节所表示的数，其范围为 $[-32768, 32767]$ 。

长整数是用 4 个字节所表示的数，其范围为 $[-2147483648, 2147483647]$ 。

② 浮点数就是平常我们所说的实数，考虑到数据的精确程度，分为单精度数和双精度数两种。

单精度浮点数是用四个字节表示的数，它能精确到 7 位十进制数；双精度浮点数是用八个字节表示的数，它能精确到 15 位或 16 位十进制数。我们来看一看其数据可表示的范围：

单精度正数所能表示数的范围为 $[+1.401298E-45, +3.402823E+38]$ 。

单精度负数所能表示数的范围为 $[-3.402823E+38, -1.401298E-45]$ 。

双精度正数所能表示数的范围为 $[+4.940656D-324, 1.7976931D+308]$ 。

双精度负数所能表示数的范围为 $[-1.7976931D+308, -4.940656D-324]$ 。

除了上述的十进制数以外，Quick BASIC 中还提供十六进制（在数据前面加上 &H 来标识）和八进制数（在数据前面加上 &O 来标识）。在学习关系表达式时，我们会知道其值只有两种，逻辑真和逻辑假，由于计算机识别真假是用零和非零来判断的，因此我们常把逻辑型数据看成数值型。

1.4.2 用户自定义的数据类型

除前面介绍的基本数据类型外，Quick BASIC 为我们提供了用户自定义数据类型，用户可根据自己的需要，按照一定的规定，把若干个基本数据类型组成用户自定义的数据类型。

格式：TYPE <定义的数据类型名>

<数据类型元素名> AS <类型名>

...

...

<数据类型元素名> AS <类型名>

ENDTYPE

功能：定义一个由一个或多个基本数据类型所构成的用户自定义型数据类型。

说明：（定义的数据类型名）是用户给自定义数据类型取的名字，数据类型元素是这个数据类型的组成部分，表示其包含多少数据元素，每个数据元素都要给它命名并定义其数据类型，注意各元素的数据类型是除去可变长度的字符串和数组之外的任何一种数据类型。例如：

TYPE STUDENT

NUM AS INTEGER

NA AS STRING*8

SCORE AS INTEGER

ENDTYPE

定义一个数据类型为 STUDENT，它包括三个数据元素，NUM 表示学号，类型为整型，NA 代表姓名，类型为字符型，SCORE 代表成绩，类型为整型。我们可以用 DIM MEN AS STUDENT 来定义变量 MEN 的数据类型为 STUDENT。

使用用户自定义数据类型应先定义后使用，其具体使用方法我们将在以后的学习中详细说明。

1.5 常量与变量

1.5.1 常量

常量是指在程序运行过程中其值不可改变的量。Quick BASIC 常量又分为两种：文字常量和符号常量。

1. 文字常量

文字常量包括数值常量和字符串常量。

（1）数值常量

数值常量共有四种表示方式：整型数，长整型数，定点数和浮点数。

① 整数是用 2 个字节所表示的数，121，-234 等都是十进制整数。

② 长整数是用 4 个字节所表示的数，3361234，-1234566 等都是十进制长整数。

前面我们讲到数据类型除了十进制外，还有八进制和十六进制数；因此也存在八进制和十六进制常量。八进制和十六进制整数常量需分别用 &O 和 &H 来标识；八进制和十六进制长整数常量除分别用 &O 和 &H 来标识，还需用 & 来标识结束。如：&O13，&H12AA5& 分别表示八进制整数和十六进制长整数。

八进制整数的表示范围是：[&O0，&O177777]

十六进制整数的表示范围是：[&H0，&HFFFF]