计算机语言技术系列丛书(二)

希望

The Waite Group's

MICROSOFT QUICKBASIC BIBLE 经典

Microsoft PRESS

The Complete Reference

Featuring:

Microsoft QuickBASIC through version 4.5

Descriptions, syntax, examples, and tips

学苑出版社

计算机语言技术系列丛书(二)

Microsoft QuickBASIC 经典

Mitchell Waite
Robert Arnson
Christy Gemmell
Harry Henderson
著

秦笃烈 译
燕卫华 校

学苑出版社

(京)新登字 151 号

内 容 简 介

Microsoft Quick BASIC 及其系列是现代化优秀高生产力编程和开发环境。深受专业及非专业计算机工作者青睐。本书是迄今为止有关 QuickBASIC 信息最全面最深刻最权威的大全。全书分为核心、多媒体、设备开发四部分。既有功能的指导性论述，也有各命令和函数的详解和实例。包括混合语言编程、通信及各种 BASIC 兼容性等珍贵信息。内容丰富的附录提供有关 Microsoft 知识库、CompuServe 文件以及第三方开发的高级软件包等深层技术和背景信息。

本书读者对象为广大 BASIC 热心者、计算机初学者、商品化软件专业开发人员以及计算机软件教学和研究人员。它也适于作为大学生及非计算机专业理工类研究生指导教材。

欲购本书的用户，请直接与北京海淀路 82 号希望电脑公司(8721 信箱)书刊部联系，电话 2562329，邮码 100080。

版 权 声 明

本书中文版版权由 Microsoft 公司授权于北京希望电脑公司和学苑出版社独家出版、发行。未经出版者书面许可，本书的任何部分不得以任何形式或任何手段复制或传播。

计算机语言技术系列丛书(二)

Microsoft QuickBASIC 经典

著　　者：Mitchell Waite　Robert Arnson
Christy Gemmell　Harry Henderson
翻　　译：秦笃烈
审　　校：燕卫华
责任编辑：甄国宪
出版发行：学苑出版社　　**邮政编码**：100036
社　　址：北京市海淀区万寿路西街 11 号
印　　刷：兰空印刷厂
开　　本：787×1092　1/16
印　　张：49.25　　**字数**：1149 千字
印　　数：1～5000 册
版　　次：1994 年 5 月北京第 1 版第 1 次
ISBN7－5077－0905－1/TP・29
本册定价：69.00 元

学苑版图书印、装订错误可随时退换

专　序

BASIC 是 Microsoft 1975 年推出的第一个产品，但是它的历史可以进一步追溯到六十年代初期一所名牌大学的数学系。BASIC 发明者创造了一种通用的、容易使用的、低开销程序设计语言，它使计算机能为人文科学大学生使用。自从 BASIC 发明以后，它的配备实际上成为计算机每个重大平台取得成功的关键性战略。

BASIC 如何从大学校园走向 PC 机大世界的呢？1975 年，比尔·盖茨和他中学伙伴经过苦思苦索发现了一种方法，能在 MITS Altair 8800(取得商品化成功声誉的第一台微型计算机)上将 BASIC 压缩成只有 4K 字节的小巧解释器。这一由名不见经传的小人物做出的业绩一下子使每个人都能在世界上首创的微型计算机上编写程序。在为 Altair 开发的这一解释器推出不久以后，Microsoft BASIC 的各种方言促进 Tandy TRS－80、Commodore PET、Apple II 以及 IBM PC 等微机取得成功。不久，每一家个人计算机制造商都配置 Microsoft BASIC，并且用 BASIC 写出了关系新平台成败的实用性很强的应用程序。

正是由于这些开创性的工作，25 年以后全世界将近有 1 亿微机用户了解 Microsoft BASIC。BASIC 不仅以它优美的语法保证了它的广泛流行，而且以它的简单性和威力使它成为广大专业程序设计人员、咨询专家以及用户手中的高生产力实用工具。

但是为什么 Microsoft 现在还如此关注 BASIC 的发展？既然有各种牌号的优良的结构化语言 PASCAL 可供学生教学使用，也有各种用户友好版本的 C 语言供认真的开发者使用，历史久远的 BASIC 还能引起初学者当年那种热情吗？

如果考察一下像 Microsoft QuickBASIC 这样的 BASIC 现代化品种，答案是显然的。由于 Quick BASIC 恪守 BASIC 本来的精神(容易学习、容易使用而且开销低)，QuickBASIC 的各个版本的演变说明 BASIC 在现代语言技术以及计算平台方面的进展。在为 8 位微型计算机设计的面向程序行的 BASIC 语言品种推出不久，BASIC 解释器就被功能强大的新模型取代，该模型把编译器的瞬息响应和解释器的交互性结合起来。现在 BASIC 已兼有现代化编译型语言在速度、容量以及结构化程序设计方面的所有功能。目前有数十万专业水平程序设计人员利用现代化、结构化编译型 BASIC 进行工作，而且 BASIC 仍然是所有学术领域广泛讲授的语言。

但是 Microsoft 的 BASIC 版本远远超出这些已经达到的成就。在不久的将来，我们 Microsoft 公司将利用 BASIC 完成大部分应用程序的可视程序设计而不必编制代码，它也要作为一种中央控制语言使用，即专门设计的 BASIC 程序能将多种应用程序的数据和功能作为库处理；它也将作为跨越各种可编程应用程序的一种共同的宏语言使用。我们认为，BASIC 由于它具有的最高知名度，将仍然是熟知它的专业程序设计人员、咨询专家以及威力用户的首选语言。

我高兴地得知，Waite 出版集团经过深思熟虑和艰苦努力以后将注意力转向 QuickBASIC，他们出版的各种语言经典和 DOS 开发人员指南在程序设计人员的书架上随处可见，我期待本书出版后会出现的重大事件。我手中已有终校清样，我为 Waite 出版集团做了这么多工作而高兴。他们确实为程序设计人员出版了优秀的 QuickBASIC 参考指南。

分类独特的编排方法既适合初学者也适合专业人员的需要。详尽的实例演示每一个 QuickBASIC 关键字、函数和语句的用法。Waite 出版集团利用从 C 语言经典中开始引入的兼

容性框格式向读者一目了然地说明，某个 QuickBASIC 关键字是否能在 QuickBASIC 所有版本、GW－BASIC、BASICA、Microsoft 的 BASIC PDS 等环境中使用。有关第三方产品、QuickBASIC 知识库以及各种面向公众程序的技巧详细说明，这些都使用本书堪称至今所见有关 QuickBASIC 最完备的参考书。

这本杰出的著作应该出现在每一位认真的 BASIC 程序设计人员的案头。

汤姆·勃敦

Microsoft BASIC 程序设计部经理

此为试读，需要完整PDF请访问：www.ertongbook.com

前　言

我们的处理办法

我们在编写本书时遵循一种特殊的处理方法。其想法是先让你得到一般性信息，然后介绍比较特殊的信息。例如，假设你刚刚开始接触 QuickBASIC，需要对整个语言有总的概貌性认识——它和其它语言相比有哪些优越性和局限性，它的威力和潜力。导论"QuickBASIC 面向任务概论"以全面且可读性较好的方式提供这些信息。

假如你了解某种 BASIC 版本，需要了解 QuickBASIC 有哪些新的和不同的领域。为此，可以阅读介绍本书每一章的拓导性教材；这些内容摘要说明对你陌生的命令和关键字的用法并作讨论。

最后，如果你理解了总貌并需要使用某个命令，则应更详细地阅读它。本书每一章分述部分讨论每个语句和函数的用途、语法和用法；它们提供有关每个条目的的权威性信息，包括程序设计技巧和告诫；而它们提供的兼容性信息使你能直观地看到那些 BASIC 版本支持该语句或函数。和多数参考书一样，有关语句和函数的讨论是按专题领域而不是按字母顺序编排的。这种组织方式的目的是为了有助于你理解特定领域中语句之间的关系，从而能迅速掌握它们的关联而不需要从全书去查找散在各处的条目。我们还在首尾页内封中提供两个按字母顺序编排的关键字和概念表，便于迅速查阅特定专题。

我们如何编写本书

本书的基本框架是将 QuickBASIC 全部关键字按它们的相关性分成 22 组。其次，我们透彻研究了每个语句和函数并对设计最佳例子取得一致意见。然后写出每个命令的供读者参考用材料（我们称为参考页面）。然后按趣味性、启发性和实用性原则展示各命令的特点。

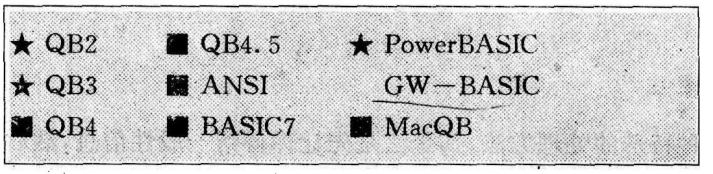
我们使用参考页面作为每一章的拓导教材，将 22 类命令最重要成份浓缩成简洁的教学性摘要，说明如何使这些命令合在一起使用。在此基础上又进一步将这 22 章材料浓缩成导论"QuickBASIC 面向任务概述"。

以上过程中的每一步都对信息进一步提炼、提取最突出的事实并剔除不必要的细节。最后，我们制作在前后内封列出的一览表。前内封一览表是可以用 QuickBASIC 完成的任务的概念性列表并给出用于查找有关信息的页面。后内封一览表按字母顺序列出各类包含的 QuickBASIC 关键字以及它们的所在页面。

精心安排和显微镜比喻

可将本书看作具有不同放大倍数的显微镜。先从前后内封中较低级别开始，考察"QuickBASIC面向任务概述"中的更多细节，研究拓导材料中命令的相互关系，最后，对参考条目以高倍放大观察 QuickBASIC 的每个语言成份。

兼容性框



参考页面中每个关键字都有像上图所示的特殊的兼容性框。该框的用途是很快了解该关键字是否能在 QuickBASIC 从 2.0 到 4.5 各个版本、ANSI BASIC、Microsoft BASIC 7.0 专业性开发系统(PDS)、PowerBASIC、GW－BASIC 以及 Microsoft QuickBASIC for Macintosh(在框内标以 MacQB)下工作,注意,Microsoft BASIC 7.0 专业性开发系统支持用 Microsoft BASIC 6.0 写的全部代码,而 PowerBASIC 是 Turbo BASIC 1.1 的升级版本,该语言以前由 Borland 国际公司发行。目前 PowerBASIC 按照和 TurboBASIC 和 PowerBASIC 的开发者 Robert Zale 的协议由 SPECTRA 出版社出版。

如果关键字在某一版本下完全支持,在兼容性框版本名称旁边出现小方块(■)记号。如果不支持该关键字,则留出一空格。如果在 QuickBASIC 4.5 和所列版本兼容性方面存在差异,则出现标记符★并在标题"兼容性"下参考材料中包含应该了解的解释性注解。这些框使程序设计更富于专业水平:让你知道将程序从一种版本移植到另一种版本的难易程度并告诫你在另一 BASIC 版本下运行程序时必须作出的权衡。

本书未包括的内容

我们有意省略了编译器选项、连接器及库管理程序的内容。我们认为关于这些专题(以及程序开发的更细致问题)的介绍应该构成更为高级的专著,我们希望不久写出这样的书。

我们希望本书对你学习和使用 QuickBASIC 产生实质性帮助。

作者

M·威特,R·安森,C·盖默尔,H·汉德森

目　录

第二部分　多媒体

第三部分　设　备

第四部分 开 发

此为试读，需要完整PDF请访问：www.ertongbook.com

第五部分 附 录

导论　QuickBASIC 面向任务概论

如何使用导论

我们在前言中已解释，Waite 集团出版的 Microsoft QuickBASIC 经典一书按信息访问四个特定级别设计。每个级别对使用 QuickBASIC 编译器出现的不同情况提供帮助。

导论这一章是本书的全景图，它所给出的有关 QuickBASIC 概论来自有远见的专业程序设计人员的观点，他们多次对 QuickBASIC 通盘研究并仍然喜爱它。对于回答诸和"能用这个语句做些什么?"以及"应该留心哪些事情?"这类问题是提供实用信息。如果你从另一种语言(如 Pascal、FORTRAN 或 Cobol)或另一种 BASIC 版本(如 GW－BASIC 或 BASICA)转向 QuickBASIC，应首先阅读这一章以便对 QuickBASIC 能做什么以及不能做什么产生完整认识。

由于本章的组织和全书的组织本身相平行，所以很容易发现特定类别的 QuickBASIC 语句的简介。当阅读本概论时，任何时候都可以翻到相应的拓导材料获取更详细的解释。如果对特定语句或函数的最高技术性解释感兴趣(例如，假设你是仅需要知道语法的高级程序设计人员)，可以翻阅该语句或函数参考条目的第一页。我们对每一章标题都编号，因此很容易知道应参考哪一章。

I. 核心

变量和类型(第一章)

变量

虽然所有程序设计语言都提供各种方法处理信息，但最重要的是它们如何表达这种信息。过去，程序设计语言以相当原始的方式组织数据，它们和高级科学计算器操作数的方式很相似。在 QuickBASIC 中数和其它数据均表示为"类型"。QuickBASIC 提供若干种数据值类型和一种代表 ASCII 字符(字母等等)的类型。QuickBASIC 也允许将这些原始类型结合起来构成更加复杂的用户自义类型。

和所有程序设计语言一样，QuickBASIC 允许使用有说明意义的名称定义变量。变量是符号名称，如 *dollarsOwed*!、*flowerColor* $、*maxRads*%或 *emptyTank*&，可以对它们赋值。QuickBASIC变量的最后一个字符表示该名称代表的变量类型，但是也可以用其它方法定义类型。以下是赋值语句的例子：

```
dollarsOwed! =123.34
flowerColor $ ="Red"
maxRads% =0
emptyTank& =-128.000
```

虽然变量各用法有一些规则，但 QuickBASIC 是非常灵活的，变量名的第一个字符必须是

字母。变量名不能多于 40 个字符，而且不能和诸如语句或函数名这样的保留字有相同名称。在第一个字符以后可以使用任何字符或数字（甚至可以用圆点，不过圆点在某些情况下可能出现问题）。

类型

在 QuickBASIC 中可以表示两种基本数据类型——数和字符。QuickBASIC 有五个标识符说明程序变量可以包含哪一种数据：INTEGER、LONG、SINGLE、DOUBLE 以及 STRING（有关这些类型的更多细节，可参阅第一章“变量和类型”）。

对于整数（无小数部分的数），QuickBASIC 提供 16 位整型（它包括从－32768 至 32768 范围内的整数）以及长整型（它包括的范围是－2147483648 至 2147483647）。长整型对（用分表示的）钱币是有用的，它不会出现用分数表示时会出现的舍入误差。这样它能无误差地操作大到 $21,474,836.48 这样的值而不产生误差。

对于带有十进制小数点的数，QuickBASIC 提供单精度型和双精度型。前者可以表示从－1.401298E－45 至 3.402823E＋38 范围内具有 7 位数字精度的浮点数；而后者可以表示从－4.940656458412465D－324 至 1.797693134862315D＋308 范围内具有 16 位数字精度的浮点数。这种精度对任何应用已经足够。实际上，1.797693134862315D＋308 比宇宙中所有原子数还要大！

QuickBASIC 支持两种类型字符串——变长字符串和定长字符串（STRING＊）。每一种均最多可包含 32767 个字符。定长字符串是最近的发展；而变长字符串是很老的结构（有关定长字符串和变长字符串更详细的内容可参阅本章有关字符串的段落）。

常量

可以将变量设想为读写值容器，变量拥有的值可以是数、字母或字符串。修改变量时是在其中“写入”一个新值。访问变量就是“读取”它的值。有时在程序中需要有不会发生变化的变量，即“只读”变量。例如，可能需要表示决不会改变的重要数学常数，如 π 或阿佛加德罗数；也可能需要表示 EGA 屏幕的宽度并且不希望子程序有修改该值的能力。对这些场合，QuickBASIC 提供 CONST 语句。这个语句能对变量的值施加保护；如果程序试着修改某常量的值，QuickBASIC 显示错误消息——这是一种有用的安全性功能。CONST 语句也使用户能在程序的开始部申明它的位置改变常量的值从而在整个程序中改变它。这样使改变值很容易，例如，在程序的下一版本将 EGA 常量改变为 VGA 宽度常量。

系统默认类型

在有些早期计算机语言中，如 FORTRAN，字母 I 到 N 可作为整型变量使用而其余的字母可作为浮点变量使用。QuickBASIC 增加了 DEFINT 语句，它使 BASIC 能像老式语言那样处理变量。例如，语句 DEFINT I－N 使用每一个以字母 I 到 N 开始的变量为整型变量，而 DEFLNG、DEFSNG、DEFDBL 以及 DEFSTR 可将变量定义为其它类型。

十六进制和八进制数

QuickBASIC 使用十进制（以 10 为底）的记数系统，但有时也需要用十六进制（底 16）和八进制（底 8）记数法显示值，特别是使用汇编语言子程序时更是这样。针对这种用途，QuickBASIC 提供 HEX＄和 OCT＄函数，它们将十进制数转换成由等价的十六进制及八进制值组成的字符串。

置换

假设程序使用变量 A 和 B，你希望交换这些变量拥有的值，即使 A 包含 B 的值，B 包含 A 的值。利用一个临时变量和三个赋值语句就可写出实现这一要求的 BASIC 代码，但更容易的方法是使用 SWAP 语句，它通过一次运算就交换变量的值而不需要设立中间变量。对完成排序的算法，SWAP 特别有用。

用户自定义类型

如果所有数据自然地组织成十进制小数、字符串或长整数，则对程序设计人员是很方便的，但事实是世界极为丰富多彩，数据很少规一化。用索然无味但不会有岐义的方式，可以说你的生日是 693,908.314159（这是典型的 44 岁生日，以公元 1 月 1 日作为起点，整数部分为经历的天数，而小数部分代表出生的时间）。但是按月、日和年表示日期（如 5 月 3 日，1946）更符合人群习惯。

如何在 QuickBASIC 中储存像生日这样的 QuickBASIC 并没有提供日期数据类型的复杂信息呢？回答是利用从 QuickBASIC 4.0 开始引入的 TYPE...END TYPE 语句。这种结构能将不同类型的变量组织成一个单元（经常称为“记录”），其工作方式和通常的变量很相似。通过在名称中加一个圆点就可以访问这样的“用户自定义”类型各个成份，如 *myDay.month* 或 *myDay.year*。

例如，设有一个名叫 customer 的变量，它是用户自定义数据类型，包括若干不同的成份——字符串变量代表客户的姓名和地址、整型数代表拥有的数量、长整形变量拥有由 16 位数字组成的客户帐号等等。记录或用户自定义类型的每个成份都有各自的名称，但如要访问这些成份，必须标有名称 customer。记录型变量的一种方便用法是很容易将它们写入一个随机访问文件；在引入 TYPE...END TYPE 语句之前，这一操作要求利用 FIELD 语句。遗憾的是，用户自定义类型不能包含数组。可以创建用户自定义类型的数组，但用户自定义类型可以将其它用户自定义类型作为自己的成员。但是，QuickBASIC 不允许在用户自定义类型内使用数组。

流程控制（第二章）

程序只有在能够作判别、根据这样的判别改变处理的方向以及重复地完成一项任务时才能作出有威力的工作。和程序操作的方向及路径有关的程序设计领域是所谓“流程控制”。

QuickBASIC 的流程控制语句管理程序的总体方向——决定在何处转移到下一位置以及在何处作分支等等。如果把对字符串、数、文件、声音以及像素操作的语句组看作是 QuickBASIC 程序的“线”，则可以把流程控制语句看作是程序的“织法”。语句是将程序的“线”编织成执行流向“图案”手段。

GOTO

GOTO 语句后面跟有要执行的行号或语句的标号，它可在主程序中以及子程序或函数内部使用。可以利用 GOTO 从一个模块中的任何位置转移到同一模块的主代码中的任何其它位置。但是，不能用 GOTO 从主代码转到模块中子程序或函数中的某个位置，也不能用它转到另一个模块，利用 GOTO 分支到某位置以后不会折回；程序从该位置继续执行而不是像调用子例程、子程序或函数哪样返回。因为包含 GOTO 的程序不是结构化而且难以调试，所以应避免使用 GOTO 语句；使用太多的 GOTO 语句会产生“拉面条”式代码，程序会杂乱无章地从一个位置跳到另一个位置，同时，QuickBASIC 的其它流程控制语句一般说来已使 GOTO 语句没有

使用的必要。

也可以使用 ON...GOTO 语句作为根据指定变量的值分支到若干位置之一的条件语句。它建立要转到的标号或行号表。虽然 ON...GOTO 的这种用法可能导致程序难以解译，但是 GOTO 和 ON...GOTO 是有用处的(参阅第二章"流程控制"中的拓导部分，它提供一种在 ON...GOTO 语句用使用 255 个行号的技巧)。

QuickBASIC 也提供 ON...GOSUB 语句，根据指定变量的值转移到指定的子例程并允许构造转移目标的表。ON...GOTO 和 ON...GOSUB 语句均可用 QuickBASIC 的 SELECT...CASE 语句代替。SELECT CASE 更为灵活而且没有 ON...GOTO 和 ON...GOSUB 具有的局限性。

循环

循环是使程序执行多于一次的语句序列。循环可以包含计算一数的阶乘的语句，在文件中搜索某记录值的语句，也可以包含演奏某种警笛声起伏的语句。最简单的循环语句是 FOR...NEXT 循环，每一种 BASIC 版本均包含这一语句。这种循环使程序按 FOR...NEXT 语句中指定的次数重复执行一组语句。要重复执行的语句从关键字 FOR 以后开始并在 NEXT 关键字处结束。FOR...NEXT 循环也提供 STEP 选项，使用户能控制每一轮循环执行以后 QuickBASIC 使循环指数增加的数量。

在另一种循环中，选代的次数在循环开始时并不知道，究竟循环多少次要在每次轮循环过程中由 QuickBASIC 完成的测试决定。这种结构称为 WHILE...WEND 循环，它是从 BASIC 开始引入的。

WHILE...WEND 是一种"入口一条件"循环，因为决定语句是否执行的测试是从循环开始在 WHILE 语句中完成的。如果测试失败循环不执行；但是如果测试成功，循环执行其中的语句直到达到关键字 WEND 为止。对于如果某个条件不满足就应该跳过去的语句，WHILE...WEND 极为有用，WHILE...WEND 的一种典型用途是读长度未知的顺序文件，利用子句 *WHILE NOT EOF*(1)，可以保证只要达到文件的结尾后循环就停止执行。

WHILE...WEND 不是 ANSI(美国国家标准研究所)制订的标准循环语句。ANSI 设计的是 DO...LOOP 语句，它的功能更强大而且更灵活。可以在 DO...LOOP 中建立入口条件循环也可以建立出口条件循环。除了修饰词 WHILE 以外，它也还提供修饰词 UNTIL，从而使循环的语法简化，例如，*WHILE NOT EOF*(1)的 DO...LOOP 等价语句是 DO UNTIL EOF(1)，这一语句更为清晰而直观。DO...LOOP 可以将测试放在结尾外；例如，可以在关键字 DO 后面跟循环体语句，后面再跟子句 *LOOP NUTIL ready*$ ="P"。对这种情形，在测试完成以前循环总是至少执行一次。

EXIT DO 和 EXIT FOR

有时在循限度达到以前程序必须从循环中退出。例如，你可能运行从一顺序文件的每个记录查找某个值的循环。它使用上一例子中用过的测试 *DO UNTIL EOF*(1)。但是，当你找到正确值以后不希望继续执行循环，而是立即从中退出而执行下一步。EXIT DO 语句使任何 DO...LOOP；不论是入口条件式还是出口条件式，中断并开始执行 LOOP 之后的下一语句。EXIT FOR 以同样方式退出 FOR...NEXT 循环。虽然也可以提前退出用户自定义函数或过程，但 WHILE...WEND 没有相应的 EXIT 语句。

判别和运算符(第三章)

IF...THEN...ELSE

如果程序从头执行到结束,中间没有任何变化、撤离、旁移或驻留,行吗?这样的程序设计工作倒确实容易。但是,程序使命的本质恰恰与此相反;程序经常需要改变方向或基于某种规则变更处理的行为。QuickBASIC 有若干语句用于处置处理上的各种变化,其中最常用的是 IF 语句。IF 可用于建立测试条件;然后根据测试结果执行某种事情。例如,可用它分支到程序中新位置、计算复杂的公式或者执行一子程序或函数。语句 *IF score%=10 THEN PRINT "Game Over"* 规定,如果整型变量 score%的值等于 10,则程序显示"Game Over"。

在 BASICA 中,IF 语句十分简单;它只能是一行,如以下例子所示:

```
IF amountDue# > 10000.00 THEN GOTO creditHold
```

BASIC 中的 IF 语句不能占若干行,以便当条件为真时计算若干运算结果。有些 BASIC 早期版本(如 Applesoft BASIC)也无法处理测试条件为假的情况;换言之,如果条件为假,执行总是跳到下一语句行。为了利用 IF 执行任何种类复杂的判别,必须借助 GOTO 语句使用更聪明的技巧。QuickBASIC 的"块"形式 IF 语句解决了多语句问题,其形式为 IF...END IF 结构。块形式 IF 语句当条件为真时执行一组语句。

至于第二问题,即如果处理测试结果为假,可用 ELSE 及 ELSEIF 子句解决。对单行 IF 语句的情形,可使用语法

```
IF condition THEN statement1 ELSE statement2
```

而对块形式情形,可使用语法

```
IF condition THEN
[statements]

EELSEIF condition THEN
[statements]

ELSE
[statements]
END IF
```

有一些复杂的方法可使块形式 IF 语句嵌套。如果嵌套的块太复杂,可以用较简单的 SELECT CASE 结构代替它。

如果不使用整型或长整型值作为 IF 语句中的测试真或假的条件,应十分懂慎。QuickBASIC很苛刻——如果使用浮点表达式,它对 0.0 的求值结果为假;任何其它值都认为是真(当然,对整型或长整型值情况并非如此,这就是为什么这些类型作为 IF 条件要更好些)。在会产生舍入误差的公式中,很难避免产生虚假的小数 0.0。另一个限制是,表达式求值结果必须属于长整型范围,这意味着大于 10^{10}或小于-10^{10}的科学计算结果将不能返回正确结果。

逻辑运算符

IF 语句中的表达式可以非常简单,像"如果 A 为真,则执行 Z";也可以比较复杂,像"如果 A 为真,而且 B 也是真,但 C 为假而且 D 是除 X 以外的任何值,则执行 Z"。为了处理所有这些

此为试读,需要完整PDF请访问: www.ertongbook.com