BASIC使用大全与指令详解

标准 BASIC Quick BASIC
→ Turbo BASIC → Visual BASIC
BASIC True BASIC

余海刚 罗齐汉 主编
陈定方 审核

中国铁道出版社

BASIC 使用大全与指令详解

标准 BASIC　QuickBASIC
→TurboBASIC →Visual BASIC
BASIC　True BASIC

余海刚　罗齐汉
罗齐英　王仲君　编写
张景潇　陈　强
陈定方　　审核

中　国　铁　道　出　版　社
1997年·北京

(京)新登字 063 号

图书在版编目(CIP)数据

BASIC 使用大全与指令详解/于海刚等编写．-北京：中国铁道出版社，1997．9
ISBN 7-113-02818-7

Ⅰ．B… Ⅱ．于… Ⅲ．Basic 语言-基本知识 Ⅳ．TP312

中国版本图书馆 CIP 数据核字(97)第 22887 号

BASIC 使用大全与指令详解

标准 BASIC　QuickBASIC
→ TurboBASIC →Visual BASIC
BASIC　True BASIC

余海刚　罗齐汉　主编
陈定方　　审校

*

中国铁道出版社出版发行
(100054，北京市宣武区右安门西街 8 号)
责任编辑　殷小燕　　封面设计　李艳阳
中国铁道出版社印刷厂印　各地新华书店经售
1997 年 12 月第 1 版　第 1 次印刷
开本：787×1092　1/16　印张：22　字数：532 千字
印数：1—3000　定价：39.10 元

内 容 提 要

本书分上、下两篇。上篇简明扼要地介绍了 BASICA、QBasic、TurboBASIC、TrueBASIC、VisualBASIC 的使用方法与操作步骤及编程应用，并比较各版本 BASIC 之间的主要差异。下篇按字母顺序排列逐一介绍各版本 BASIC 的语句、函数及命令的功能、语法和注释，且给出了易于理解的例题，一目了然。

本书既可引导初学者迅速掌握 BASIC 语言，又适用于有 BASIC 基础的程序员对 BASIC 不同版本的理解、转换和灵活运用。

前　言

随着计算机的飞速发展，使得计算机得以迅速走进千家万户。由于各种各样的原因，我国的很多用户对计算机尤其是计算机语言还知之甚少。作者本着从易到难逐步深化的原则，向广大计算机用户和计算机爱好者郑重推荐 BASIC 语言作为实践的基础语言。BASIC 语言由于它的强有力、灵活、易于使用以及更重要的是类似英语而成为一种优秀的语言。

BASIC 是初学者的通用符号指令码(Beginner's All－purpose Symbolic instruction Code)，其最早版本是 1964 年在达特蒙思学院作为一种教学语言开发出来的。在此以后的若干年中，BASIC 有了许多修改和增强。1978 年出现标准 BASIC。BASICA 是 Microsoft 公司为 IBM 个人计算机写的 BASIC 的增强版本。GW－BASIC 是汉化并用于 IBM PC 及兼容机的。BASICA 大约有 190 条指令，超出了标准 BASIC 的其它版本。标准 BASIC/BASICA 都称为解释性语言。当计算机运行标准 BASIC/BASICA 程序时，它实际上是执行一个必须对每一条 BASIC 指令进行读取、理解并按其办事的程序。BASIC 语言使用类似英语的词汇作为指令(命令、语句、函数等)，如：PRINT，WRITE，READ，RESTORE，PLAY，DRAW，IF... THEN... ELSE，CALL，OPEN，CLOSE 等。在任何时候可引入新变量以及几乎没有复杂的结构。BASICA 有许多在其它语言不全都具备的图形、声音和事件陷阱能力。因为对程序解释执行，故可以在任何地方停下来分析变量的值而进行调试。

但是解释性语言也有其缺点。其一，计算机必须读取并理解所遇到的每一条指令。其二，它们需要行号，有时使程序难写以及子程序难以跟踪；其三，因为全部变量都是全程量，从而不能把子程序轻易地从一个程序移到另一个程序以及在两个不同的上下文关系中必须小心避免使用同一变量；其四，不能进行结构化程序设计，如：控制语句 IF/THEN 不具有足够的灵活性和明确性；其五，标准 BASIC/BASICA 都限于 64K 内存，既要作为工作空间也要作为变量值的存储空间。

计算机科学家一致同意，现代程序设计语言不应有行号，但应有广泛的控制结构、局部变量(仅在程序的一部分中使用的变量)以及从过程传送并接受数据的能力。

为此，在标准 BASIC 和 BASICA 的基础上：

Microsoft 公司推出了 QuickBASIC、VisualBASIC；

Borland 公司推出了 TurboBASIC；

BASIC 发明人重新改写和完善了 BASIC，推出了 TrueBASIC。

这四种版本的 BASIC 均具有全屏幕编辑程序的能力，可以取消行号，既像地道的 BASIC 版本那样容易学习，同时具有了像 Pascal、C 这样"严肃"的语言的基本控制结构和能力。支持局部变量、静态变量和全程变量，支持参数传送，支持过程，具有快速高效的编辑或编译功能等。

目前国内还没有一本综合且详细介绍 BASIC 指令的教学或参考书籍，从而使 BASIC 语言的学习学而不畅、学而不深，导致在实践中遇到这样那样的问题和难题无法解决，挫伤了广

大计算机爱好者学习 BASIC 语言的积极性，抑制了他们富有幻想的大脑中的奇妙的构思与非凡的创造。本书以 BASICA 为蓝本介绍多版本 BASIC 语言的使用方法与操作步骤，介绍 BASIC 语言的众多指令的功能、语法和注释，且给出了适当的例题，有些例子是相当有趣而易于理解的。同时，比较了各版本 BASIC 之间的主要差异。书中凡与 BASICA 相同的指令，皆以 BASICA 为基准加以介绍，细微不同之处，通过"特别注释"给出。对于后四种版本新增或独有的指令，则另行列出介绍。个别与 BASICA 原有指令功能或语法相近似的新增指令如 CVL、MKL¥、CLNG 等，则插入原 BASICA 相关指令条，以括号"()"注解说明加以区分，不再另行单列介绍。

本书分为上、下两篇。上篇简明扼要地介绍了 BASICA、QBasic、TurboBASIC、TrueBASIC、VisualBASIC for DOS/for WINDOW 的使用与编程应用，下篇按字母顺序排列逐一介绍各版本 BASIC 的指令，明确指出该指令产生于哪个版本，是作为语句还是作为命令，是作为函数还是作为变量，一目了然。为了便于各版本 BASIC 之间的转换，进行同名指令语法及功能上的比较，以及不同指令相同或相似功能的参照提示。收录尽可能详尽正确，以方便读者参考、查阅和借鉴，有利于全面理解各种 BASIC 语言的指令系统。将编辑键放到附录集中介绍，使本书更加精练。书后附有词汇表，可帮助读者理解和认识计算机常用术语。BASICA 语言从最初的解释型语言发展到目前版本达几十种的编辑、编译型语言，功能已得到极大的增强，语言更加丰富。它们之间既相互交融又有所区别。本书选取目前流行较广的 BASICA、QBasic、TurboBASIC、TrueBASIC、VisualBASIC 进行讲解、分析和比较，以期通过这本书，引导初学者迅速掌握 BASIC 语言并加深 BASIC 的资深用户及具有一定编程经验的 BASIC 程序员对 BASIC 不同版本的理解和灵活运用。

本书由余海刚、罗齐汉主编，罗齐英、王仲君、张景潇、陈强参加编写。余海刚编写了上篇第一章(标准 BASIC/BASICA 使用指南)、第二章(QuickBasic 使用指南)、第六章(BASIC 版本间的差异)的部分内容及下篇 BASIC 指令详解的部分内容；罗齐汉编写了上篇第五章(Visual BASIC 使用指南)及下篇 BASIC 的指令详解的部分内容；罗齐英编写了上篇第六章(BASIC 版本间的差异)的部分内容、下篇 BASIC 指令详解的部分内容及典型范例和附录的部分内容；王仲君编写了典型范例和附录的部分内容；张景潇编写了第三章(TrueBASIC 使用指南)；陈强编写了第四章(TurboBASIC 使用指南)。

陈定方教授为本书的构思及撰写提出了重要的建议，并对全书进行了审校。中国铁道出版社的责任编辑殷小燕为本书的编辑、出版付出了辛勤的劳动。在此表示衷心的谢意。

编　者

1997 年 4 月

目　录

上　篇

下　篇

上　篇

第一章　标准 BASIC/BASICA 使用指南

BASIC 是一种其语句和数学符号类似于英语的简单、易学、易用的计算机程序语言。使用 BASIC 可以编写简单的或复杂的程序。在你的计算机上运行。你也可以修改已编好的 BASIC 软件。BASIC 是世界上最流行的计算机语言。例如，美国波士顿计算机协会所属的程序员中 80%以上使用 BASIC。在美国每年有百万高中和大学生学习 BASIC 课程。在我国 BASIC 曾是几乎所有高校的计算机语言的入门或者说启蒙教程，直至今天 BASIC 已进入千家万户，BASIC 的使用者用千万计亦不为过。程序员喜欢 BASIC 的原因是它容易学，而且能练习所有计算机的功能。专业程序员则欣赏用 BASIC 开发程序的速度和效率。

BASICA 是 Microsoft 公司为 IBM 个人计算机写的 BASIC 的增强版本，兼容标准的 BASIC。下文以 BASICA 为蓝本介绍标准 BASIC/BASICA 使用和编程的方法。

·BASICA 要求运行在 MS－DOS 版本 3.1 及以上。

第一节　启动 BASICA

本节描述怎样把 BASICA 装入系统，也解释了两种不同类型的操作方式及行格式。

一、装入 BASICA

为使用 BASICA 语言，你必须从 MS－DOS 磁盘的工作拷贝中将它装载到计算机内存。步骤如下：

1. 打开计算机；
2. 将 MS－DOS 盘工作拷贝插入计算机驱动器 A：中，并按回车键；
3. 在 A＞提示符后键入下面的命令，然后按回车键(Enter)：

A ＞ BASICA ↓

一旦进入 BASICA，提示符为 OK，将替换 MS－DOS 提示符 A＞。屏幕上 XXXXX Bytes free 行指示出当使用 BASICA 时有多少字节可供内存使用。功能键(F1～F10)赋值出现在屏幕底行。这些功能键可以用来消除击键而节省时间。

当然，你可以将 BASICA.EXE 拷贝到计算机硬盘(例如 C:\BAS)中，然后从 C:\BAS＞子目录键入 BASICA 并按 Enter 键，即把 BASICA 解释程序装载到计算机内存中。至于如何建立子目录及如何将 BASICA 程序拷贝到硬盘中，参考 DOS 的有关资料，也许你已经很熟悉了。

二、操作方式

BASICA 一旦初始化(装载),就显示 OK 提示符。OK 意指 BASICA 在命令级;也就是说,它已准备好接受命令。鉴于此,BASICA 可以用两种操作方式中的任一种:直接方式或间接方式。

1. 直接方式

这是标准 BASIC/BASICA 所独有的方式。在直接方式下,BASICA 语句和命令一当键入即执行。数学或逻辑操作的结果可以立即显示或储存备用,但是指令本身执行后却丢失了。这种方式对于调试和使用 BASICA 进行不需要完整程序的快速计算是十分有用的,对于初学者上机练习和教学指令用法也是非常方便的。

2. 间接方式

间接方式用于输入程序。BASICA 程序行前总是冠以行号,并存储在内存中。内存中存储的程序通过键入 RUN 命令而执行。

三、BASICA 命令行格式

BASICA 命令行让你改变使用 BASICA 时适用的环境和条件。

BASICA 使用下列格式的命令行:

BASICA [filename] [<stdin][[>]>stdout][/f:n][/i][/s:n][/c:n][/m:[n][,n]][/d]

filename 是一个 BASICA 程序文件的名字。如果该参数存在,BASICA 处理时就象给出了 RUN 命令一样。如果没有为文件名提供扩展,则假定为缺省的文件扩展.BAS。.BAS 扩展指示该文件系 BASIC 文件。一个文件名可以包含的最大字符数是 8 个,外带一个小数点和三个扩展字符。

<stdin 重定向 BASICA 的标准输入,而从一个指定的文件中读取。使用时,它必须出现在任何开关之前。当有多个文件可能被你的程序使用且希望指定一个特别的输入文件时,使用此参数。

>stdout 重定向 BASICA 的标准输出到指定的文件或设备。使用时,它必须出现在任何开关之前。stdout 前使用>>引起输出附加。BASICA 可以通过提供如下所示的命令行上的输入与输出文件名而重定向:读自标准输入(键盘)和写到标准输出(屏幕)上:

basica programname <input file [>]>output file

文件重定向的说明见下文 BASICA 命令行的讨论。开关经常出现在命令行;它们指定命令一个特定的动作,与使用缺省设置相对应。开关参数前有斜线(/)。

/f:n 设置 BASICA 程序执行期间可以同时打开的文件的最大数。每个文件要求 194 字节留给文件控制块(FCB)外加 128 字节给数据缓冲区。该数据缓冲区尺寸也可以用/s:开关改变。如果/f:开关省去,则打开文件的最大数缺省为 3。除非/i 开关也指定在命令行上,该开关被忽视。

/i 使 BASICA 静态地为文件操作分配所需空间,基于/s 和/f 开关。

/s:n 设置为文件使用所允许的最大记录长度。OPEN 语句中的记录长度选项不能大于这个值。如果/s:开关省去,记录缺省为 128 字节,最大记录尺寸是 32767。

/c:n 控制 RS－232 通讯。如果 RS－232 卡存在，/c:0 取消 RS－232 支持，以及任何随后的 I/O 企图。如果/c:开关省去，那么 256 字节分配给接收缓冲区而 128 字节给传送缓冲区。当 RS－232 卡不存在时，/c:开关没有影响。当 RS－232 卡存在时，/c:n 分配 n 字节给接收缓冲区，128 字节给传送缓冲区。

/m:n[,n]设置最高内存位置(第一个 n)和最大块尺寸(第二个 n)供 BASICA 使用。BASICA 试图为数据和堆栈段分配 64K 字节的存储空间。如果机器语言子程序为 BASICA 程序调用，用/m:开关设置 BASICA 能用的最高位置。最大块尺寸是 16 的倍数，它被用来在 BASICA 工作空间之外为用户程序(汇编语言子程序)保留空间。缺省的最大块尺寸为最高存储位置。缺省的最高存储位置是 64K 字节，除非最大块尺寸被指定。无论是哪种情况，缺省为最大块尺寸(16 的倍数)。

/d 允许某个函数返回双精度结果。当指定/d 开关时，则使用近 3000 字节的附加码空间。受影响的函数是 ATN、COS、EXP、LOG、SIN、SQR 和 TAN。

值得注意的是：当你对 BASICA 操作环境作指定修改是，确保显示在语法语句中的参数的顺序。跳过一个参数，插入一个逗号，这让计算机知道你对特定参数不做修改。所有的开关数值都可以指定为十进制、八进制(前面加 &O)，或十六进制(前面加 &H)。

BASICA 样板命令行如下：

通常使用最多的是使用 64K 内存和三个文件(即全缺省)：

BASICA

下例是使用 64K 内存和三个文件，装载并执行程序 payroll.bas：

BASICA PAYROLL

下例使用 64K 内存和六个文件，装载并执行程序 invent.bas：

BASICA INVENT /F:6

下例取消 RS－232 支持并仅使用第一个 32K 内存。上面的 32K 保留给用户程序：

BASICA /C:0 /M:32768,4096

下例使用四个文件并允许 512 字节的最大记录长度：

BASICA /F:4 /S:512

下例使用 64K 字节内存和三个文件。分配 512 字节给 RS－232 接收缓冲区和 128 字节给传送缓冲区，装载并执行 tty.bas：

BASICA TTY /C:512

欲了解更多的关于 RS－232 通讯的信息，参见本书附录“BASICA 通讯”。

重定向标准输入和输出

当重定向后，所有 INPUT，LINE INPUT $，INPUT $ 和 INKEY $ 语句皆从指定的输入文件而非键盘上读取。所有的 PRINT 语句皆写到指定的输出文件而非屏幕。错误信息传到标准输出和屏幕上。从 KYBD:输入的文件仍然从键盘上读取。输出到 SCRN:的文件仍然输出到屏幕。当使用 ON KEY n 语句时，BASICA 继续使键自陷。

当输出重定向后，敲 CTRL－BREAK，促使 BASICA 关闭所有打开的文件，发出" Break in line nnnn" 到标准输出，退出 BASICA，并返回 MS－DOS。

当输入重定向后，BASICA 连续不断地从源处读取直至测到一个 CTRL－Z。这个条件可以用文件结束(EOF)函数加以测试。如果文件不是以 CTRL－Z 结束，或者如果一个 BASICA

文件输入语句企图读到文件尾，那么关闭所有打开的文件，而 BASICA 返回到 MS－DOS。

欲了解更多的关于这些语句和其它语句、函数、命令及本书提到的变量，参考本书下篇的 BASIC 指令详解。

一些重定向的例子列于后：

BASICA MYPROG > DATA.OUT

通过 INPUT 和 LINE INPUT 语句连续不断地从键盘上读到数据。由 PRINT 语句将数据输出到 DATA.OUT 文件。

BASICA MYPROG < DATA.IN

通过 INPUT 和 LINE INPUT 语句从 DATA.IN 文件中读取数据。由 PRINT 将数据连续地输出到屏幕上。

BASICA MYPROG < MYINPUT.DAT > MYOUTPUT.DAT

此时通过 INPUT 和 LINE INPUT 语句从文件 MYINPUT.DAT 中读取数据，由 PRINT 语句将数据输出到 MYOUTPUT.DAT 中。

BASICA MYPROG < \SALES\JOHN\TRANS.DAT >>\SALES\SALES.DAT

通过 INPUT 和 LINE INPUT 语句从文件\SALES\JOHN\TRANS.DAT 中读取数据，由 PRINT 语句将数据附加到\SALES\SALES.DAT 中。

四、行 格 式

可以组成一个程序段的 BASICA 的每一个元素称为语句。这些语句很象英语中的句子。语句以一定的逻辑手段合在一起构成程序。本书下篇描述了 BASICA 中能够使用的所有语句。在 BASICA 程序中，有下面的行格式：

nnnn Statement [:statement]

nnnn 为一行号，statement 是 BASICA 的语句。

使用行号是标准 BASIC/BASICA 的一大特色，它要求每个程序行都以一个行号开始并作为该行的标识。行号给程序的调试提供了很大的方便，在行号的支持下，BASIC 允许程序员信手使用 GOTO 型语句把流程从一行转向程序内的任何一行。一行内至少包含一个字符，但不能多于 255 个字符(TrueBASIC 却规定在一行内最多可以有 32767 个字符，在学习 BASIC 语言的过程中，留心不同版本 BASIC 之间的区别，有利于对 BASIC 概念、基本操作以及指令功能和用法等的理解与掌握)。行号指示程序行存储在内存中的顺序，也用作分支和编辑时的参考。当按下回车键时程序行结束。

有赖于你的程序的逻辑，一行上可能有多于一个的语句，这一点与 QBasic 相同(True BASIC 却规定一行只能写一个语句)。如果这样的话，每一个语句必须由冒号(:)隔开。程序中的每一行应冠以行号。这个数字可以是 0 到 65529 之间的全部整数。习惯上使用行号如 10、20、30...以便为任何希望后来加入的附加行留下空间。因为计算机将按数字顺序执行语句，故附加行不必依次出现在屏幕上：例如，如果你在 60 行后键入 35 行，计算机仍将在 30 行后 40 行前运行 35 行。这种技术可以保存再次键入的一个完整的程序，从而包括进你曾忘记了的一行。

屏幕的宽度为 80 个字符。如果你的语句超出了这个宽度，光标则自动地移到屏幕的下一行。仅当按下回车键时，计算机才告知行的结束。当到了屏幕边上(或外边)忍住不按回车键，

计算机将自动地移到下一行。你也可以按 CTRL－RETURN，它使得光标移到下一屏幕行的开始而非实际地进入该行。当按下回车键时，整个逻辑行传送到 BASICA 程序栈。

在 BASICA 中以一个数字字符开始的任何文本行被认为是一个程序行，当回车键按下后按三种方式中的一种进行处理：

. 一个新行加到程序中。如果行号合法（在 0 到 65529 之间），且如果至少一个字母或特殊字符加到本行行号之后的话，那么就会有这种情况。

. 修改一个已存在的行。如果该行号与程序中的已有行号一致，就会有这种情况。已存在行被新键入的行的内容替代。这个过程叫做编辑。再次使用一个已存在的行号将导致该行包含的所有信息的丢失。在间接方式下键入行号须小心谨慎。你可能意外地删掉一些程序行。

. 删除一个存在的行。如果该行号与一个已存在行的行号一致，并且所键的行仅含有一个行号时，那么就会有这种情况。如果企图删除一个不存在的行，则显示出"undefined line number"错误信息。

五、返回 MS－DOS

在你返回到 MS－DOS 之前，你必须保存在 BASICA 下键入的程序，否则程序就丢失了。欲返回到 MS－DOS，在 OK 提示符后敲入下面的内容，并按回车键：

SYSTEM

系统返回到 MS－DOS，A＞提示符（或 C＞提示符）出现在屏幕上。

第二节　BASICA 语句、函数、命令和变量

一个 BASICA 程序由几种元素组成：关键字、命令、语句、函数和变量。

一、关 键 字

BASICA 关键字如 PRINT，GOTO，RETURN... 对 BASICA 解释程序而言有特殊的意义。BASICA 把关键字视作语句和命令的一部分而加以解释。关键字也叫保留字。它们不能用作变量名，否则系统将象命令一样地解释它们。然而，关键字可以嵌进变量名中。关键字作为标记存储在系统中（1 或 2 字节字符）以便最有效地利用内存空间。

二、命　　令

命令和语句都是可执行的指令。命令和语句之间的区别在于命令一般地在直接方式下或解释程序的命令级执行。它们常常执行一些类型的程序诸如编辑、装载或保存程序等。当 BASICA 被请求且 BASICA 提示符 OK 出现时，系统即在命令级。

注：BASICA 的有些命令在 QBasic、TurboBASIC 中被归入语句，详见下卷。

三、语　　句

一个语句，如 ON ERROR...GOTO 是 BASICA 的一组关键字，通常用于 BASICA 程序行以作为程序的一部分。当程序运行时，语句当且仅当它们出现时执行。

四、函　　数

BASICA 解释程序执行数字的和字符串的两种函数

1. 数字函数

BASICA 解释程序可以执行某个数学的(算术的或代数的)运算。例如,可以计算角度 X 的正弦(SIN),余弦(COS)或正切(TAN)。除非另外指出,数字函数仅返回整型或单精度的结果。

2. 字符串函数

字符串函数进行字符串的操作。例如,TIME $ 和 DATE $ 返回系统认可的时间和日期。如果在系统启动时键入当前时间和日期,则给出正确的时间和日期(计算机的内部时钟维持)。

3. 用户自定义函数

由 DEF FN 语句可以定义用户函数。这些函数可以是字符串的也可以是数字的。

五、变　　量

赋予数值的某组字母字符称为变量。当变量在 BASICA 程序中建立时,一旦执行则提供信息。例如,ERR 定义程序中产生的最后的错误;ERL 给出错误的位置。变量也可以由用户或程序内容定义和/或重定义。这里的 ERR、ERL 是机器内部变量。BASICA 共有 8 个内部变量。这些变量在 QBasic 中成为函数。

所有 BASICA 命令、语句、函数和变量被逐一地描述在下篇中。

第三节　常数、变量、表达式和运算符

在你已学习了 BASICA 中的程序原理之后,你会发现你欲编写更为复杂的程序。本节的信息将帮助你学习更多的关于 BASICA 中的常数、变量、表达式和运算符的使用的知识,以及如何用它们演化成更为诡辩的程序。

一、常　　数

常数是程序执行期间 BASICA 解释程序使用的静态的值。有两种类型的常数:字符串和数字型。

一个字符串常数是引有双引号的 0 到 255 个字母字符的一个序列。下面是样板字符串常数:"HELLO","$25,000.00","NUMBER OF EMPLOYEES"。

数字型常数可以是正的或负的。当在 BASICA 中输入一个数字常数时,不能键入逗号。例如,如果数 10,000 作为一个常数输入时,应该键入 10000。有五种类型的数字型常数:整型、定点、浮点、十六进制和八进制。对它们的描述如下:

整型	−32768 和+32767 之间的所有数,不包含小数点
定点	包含小数点的正或负的实数
浮点	以指数形式表示的正数或负数(类似于科学记数法)。一个浮点常数 由一个任意符号的整型数或定点数(尾数),后跟字母 E 和一个任意 符号的整型数(指数)组成。浮点常数允许的范围是 3.0×10^{-39}到 1.7×10^{38}。例如:

235.988E－7＝.0000235988

2359E6＝2359000000

十六进制　十六进制数以 &H 作前缀，例如：&H76、&H1234

八 进 制　八进制数以 &O 或 & 作前缀，例如：&O376、&1234

1. 数字型常数的单精度和双精度形式

数字型常数可以是整型（注：QBabic 等版本的 BASIC 中还有长整型，BASICA 中无，后文中将详细介绍）、单精度或双精度数。整型常数仅作为完整的数存储。单精度数字型常数以 7 位存储（虽然仅有六位是精确的）。双精度数字型常数以 17 位精度存储，而以十六位打印。

一个单精度常数是下列数字型常数中的任一种：

. 七位或更少位

. 使用 E 的指数形式

. 尾随一个感叹号（!）

一个双精度常数是下列数字型常数中的任一种：

. 八位或更多位

. 使用 D 的指数形式

. 尾随一个数字符号（#）

2. 下面是单精度和双精度数字型常数的例子

单精度常数	双精度常数
46.8	345698251
－1.09E－05	－1.094563D－6
4567.5	3489.0#
22.5!	7654321.3455

二、变　量

变量是你所选择的用以表示在 BASICA 程序中使用的值的名字。变量值可以被专门赋给，或是程序计算的结果。如果一个变量没有赋值，BASICA 假定变量的值为零。

1. 变量名和说明

BASICA 变量名可以是任意长度，多达 40 个字符都是有意义的。变量名中允许的字符是字母、数字和小数点。变量名中的第一个字符必须是字母。也允许特殊类型的说明字符。

保留字（BASICA 用作命令、语句、函数运算符的所有字）不能用作变量名。但是，如果保留字嵌入变量名中则是允许的。

变量可以表达数字型值也可以表达字符串值。

2. 类型说明符

类型说明符说明了一个变量描述的内容。类型说明符认知如下：

字　符	变量类型
$	字符串
%	整型
!	单精度
#	双精度

在 QBasic 和 TurboBASIC 中，还有长整型(其类型说明符为 &)，这是标准 BASIC/BASICA 所不具有的变量类型。在 TrueBASIC 中只分字符串和数字两种类型，故不存在单精度、双精度及长整型。

下面是每一种类型的样板变量名：

变量类型	样板名
字符串变量	M$
整型变量	L%
单精度变量	MINIMUM!
双精度变量	PI#

一个数字型变量名的缺省类型是单精度型。双精度型，当需要十分精确时用到，它使用较多的存储空间和较多的计算时间。单精度对大多数的应用已是十分准确。可是，第七有效位(如果打印的话)就不总是精确的。当在整型、单精度型和双精度型之间进行转换时，理当十分谨慎小心。当不指定变量类型时缺省为单精度值，如 ABDC。

以 FN 开头的变量假定为用户自定义函数的调用。

BASICA 语句 DEFtype 可以包含在程序中，用于说明某个变量名的值的类型。

3. 数组变量

数组是具有相同的变量名的一组值。数组中的每一个元素都涉及到数组变量，其下标为整型数或一个整型表达式。下标由圆括号括起。一个数组变量有数组中定义的一样多的下标个数。例如，V(10)是一个一维数组的一个值，而 T(1,4)是一个二维数组的一个值。

BASICA 中一个数组维数的最大值是 255。每个维数的元素的最大数是 32767。

如果你所用的数组下标值大于 10，则应使用 DIM 语句。参见本书下篇以得到更多的信息。如果使用了大于指定的最大值的一个下标，就会收到错误报文"subscript out of range 下标超界"。

多维数组(多于一个下标，以逗号分隔)对于存储表格式数据是很有用的。

例如 A(1,4)可以用来代表一个 2 行 5 列的数组，形如下：

列	0	1	2	3	4
行 0	10	20	30	40	50
行 1	60	70	80	90	100

本例中，元素 A(1,2)=80，而 A(0,2)=30

行与列以 0 开始，而非 1，除非另外说明。欲知更多的信息，参见本书下篇的 OPTION BASE 语句。

4. 变量存储所要求的内存空间

变量的不同类型要求不同的存储量。取决于计算机的存储容量和所编写的程序的大小，这些会是重要的因素。变量及数组的不同类型和它们所需的存储字节数列于下：

整型变量	2
单精度变量	4
双精度变量	8
整型数组	每元素 2
单精度数组	每元素 4

双精度数组　　每元素 8

对于字符串，前面的三个字节加上字符串的内容，字符串中每个字符作一个字节。标注在每个字符串开头和结束处的引号不予计算。

三、类型转换

需要时，BASICA 将数字型常数从一种变量类型转换成另外一种，遵守如下规则：

如果一种类型的数字型常数赋给一个不同类型的数字变量，那么该数以变量名中说明的类型存储。例如：

```
10 a%=23.42
20 print a%
run
23
```

如果一个字符串变量赋以一个数字值或错误的值，那么就会发生“Type mismatch 类型不匹配”错误。

在一个表达式的求值期间，算术的或关系的运算中的所有操作数都转化成相同的精度等级，也就是说，最高等级的操作数的等级。同样，算术运算的结果也以此精度等级返回。例如：

```
10 d#=6#/7
20 print d#
run
.8571428571428571
```

算术运算以双精度执行，返回在 d# 中的结果也是一个双精度的值。

```
10 d=6#/7
20 print d
run
```

算术运算以双精度执行，返回在 d 中的结果（单精度变量）四舍五入并以单精度值打印出来。

逻辑运算符将它们的操作数转化成且返回一个整型结果。操作数必须在－32768 到 32767 之间，否则就会有“overflow 溢出”错误发生。

当一个浮点值转化成一个整数时，小数部分四舍五入。例如：

```
10 c%=55.88
20 print c%
run
56
```

如果一个双精度变量赋以一个单精度值，那么仅转换数的前七位（四舍五入）是有效的。这是因为提供给单精度值只有七位精确度。打印的双精度值与原先的单精度值之间差的绝对值，小于原先单精度值的 6.3E－8 倍。例如：

```
10 a=2.04
20 b#=a
30 print a,b#
```

run

2.04　2.039999961853027

四、表达式和运算符

一个表达式可以简单地是一个字符串或数字型常数、一个变量，或者它可以由常数和变量与运算符组合而成以产生一个单一的值。

运算符执行数学的或逻辑的值的运算。BASICA 提供的运算符可分作四种范畴：

. 算术型

. 关系型

. 逻辑型

. 函数型

1. 算术运算符

下面是 BASICA 识别的算术运算符。它们以执行的先后出现。

运算符	运算
^	指数(乘方)
—	取负
*	乘
/	除
+	加
—	减

括号中的操作优先执行。在括号里面，遵守通常的运算规则。

下面是代数表达式与 BASICA 表达式的样板：

代数表达式	BASICA 表达式
$\frac{X-Y}{Z}$	(X−Y)/Z
$\frac{XY}{Z}$	X * Y/Z
$\frac{X+Y}{Z}$	(X+Y)/Z
$(X^2)^Y$	(X ^ 2) ^ Y
X(−Y)	X * (−Y)

两个连续的运算符必须由括号隔开。

a. 整除和模运算

两个附加的算术运算符是行之有效的：整除和模运算。

整除以反斜杠(\)指示。在进行除之前，操作数四舍五入为整型数(必须在 −32768 到 32767 之间)，而商则截为整数。

下面是整除的例子：

10\4＝2

25.68\6.99＝3

依 BASICA 中的规则，整除仅在浮点除之后执行。

模运算以运算符 MOD 指示。该运算给出整除的余数的整型值。

下面是模数运算的例子：

10.4 MOD 4＝2

(10/4＝2 余数为 2)

25.68 MOD 6.99＝5

(26/7＝3 余数为 5)

依 BASICA 中存在的规则，模运算在整除之后。描述在本书下篇的 INT 和 FIX 函数，对模运算也是有用的。

b. 溢出和被零除

如果在一个表达式计算期间发生了被零除，那么，就会给出错误报文 " Division by zero 被零除"。机器无穷大的数学符号作为除的结果给出，并继续执行。

如果指数运算导致零的负次幂，那么"Division by zero"错误报文就会出现。正的机器无穷大作为指数的结果给出，而继续执行。

如果溢出发生，那么"overflow 溢出"错误报文出现，带有代数的正确符号的机器无穷大作为结果给出，而执行继续。发生在溢出和被零除中的错误，不能由错误陷阱函数捕捉。

2. 关系运算符

关系运算符让你比较两个值。比较的结果或为真(－1)或为假(0)。这个结果可以用来决策后续程序的流动。下表列出了关系运算符：

关系运算符

运算符	关系	表达式
＝	相等	X＝Y
<>	不等	X<>Y
<	小于	X<Y
>	大于	X>Y
<＝	小于或等于	X<＝Y
>＝	大于或等于	X>＝Y

等号也用于把一个值赋给变量。参见本书下篇的 LET 语句。

当算术和关系运算符混杂在一个表达式中时，总是首先执行算术运算：

x＋y<(t－1)/z

如果 x 加 y 的值小于 t－1 除以 z 的值，此表达式为真。

3. 逻辑运算符

逻辑运算符对多项关系、位处理或布尔运算进行测试。逻辑运算符返回一个或为真(非零)或为假(零)的一位结果。在一个表达式中，逻辑运算在算术运算和关系运算之后执行。逻辑运算的输出确定在所列的下表中。运算符以执行的先后列出。

逻辑运算符返回的结果

运算	值	值	结　果
NOT	X		NOT　X
	T		F
	F		T

AND	X	Y	X AND Y
	T	T	T
	T	F	F
	F	T	F
	F	F	F
OR	X	Y	X OR Y
	T	T	T
	T	F	T
	F	T	T
	F	F	F
XOR	X	Y	X XOR Y
	T	T	F
	T	F	T
	F	T	T
	F	F	F
EQV	X	Y	X EQV Y
	T	T	T
	T	F	F
	F	T	F
	F	F	T
IMP	X	Y	X IMP Y
	T	T	T
	T	F	F
	F	T	T
	F	F	T

正如关系运算符可以用作决策程序的流动，逻辑运算符可以将两个及两个以上的关系连接起来，并且返回一个用作决策的真值或假值。例如：

IF D<200 AND F<4 THEN 80

IF I>10 OR K<0 THEN 50

IF NOT P THEN 100

逻辑运算符将它们的操作数转化为16位、有符号、在－32768到＋32767之间的两个整型数。如果操作数不在这个范围内，就会发生错误。如果两个操作数都是0或－1，则逻辑运算也返回0或－1。对这些整型数按位进行操作，也就是说，结果的每一位由两操作数中的相应位确定。

这样，用逻辑运算符测试字节的特定位图案成为可能。例如：AND运算符可以用来屏蔽一个机器I/O端口上的一个状态字节中的每一位。OR运算符可以用来合并两个字节以创建一个特别的二进制值。下面的例子演示了逻辑运算符是如何工作的：

例子	解释
63 AND 16 = 16	63＝二进制111111而16＝二进制10000，于是63 AND 16 = 16
15 AND 14 =14	15＝二进制1111而14＝二进制1110，于是15 AND 14 = 14

−1 AND 8= 8	−1=二进制 1111111111111111 而 8=二进制 1000,于是−1 AND 8 = 8
4 OR 2= 6	4=二进制 100 而 2=二进制 10,于是 4 OR 2 = 6(二进制 110)
10 OR 10=10	10= 1010=二进制 1010,于是 1010 OR 1010 = 1010
−1 OR −2=−1	−1=二进制 1111111111111111 而−2=二进制 1111111111111110,于是−1 OR −2 = −1
NOT X=−(X+1)	任何整型数的补码是位补加 1。

4. 函数运算符

一个函数用于一个表达式中,以调用一个对操作数执行的预定的操作。BASIC 有驻留在系统中的内部函数,如 SQR(平方根)或 SIN(正弦)。

BASICA 也允许由程序员编写用户自定义函数。参见下篇的 DEF FN 语句。

4. 字符串运算符

为比较字符串,使用与数字同样的关系运算符。

BASICA 解释程序通过一次从每一字符串中抽取一个字符比较它们的 ASCII 码而比较字符串。如果每一个字符串的 ASCII 码都相同,则串相等。如果 ASCII 码不同,则低码在前高码在后。如果解释程序在字符串比较期间到了一个字符串的尾部,那么较短的字符较小,所提供的两个字符串到那一点处是相同的。前导和尾随的空格都有意义。例如:

"AA"<"AB"

"FILENAME"="FILENAME"

"x&")"X#"

"CL ">"CL"

"kg">"KG"

"SMTH"<"SMTHEE"

B$<"15/11/1995",当 B$ ="14/11/1995"时。

字符串比较可能用来测试字符串值或按字母排列字符串。比较表达式中使用的所有字符串常数都必须用引号引起来。

字符串可以由加号(+)串联在一起(注:TrueBASIC 则是用 & 连接字符串,详见下文)。例如:

```
10 A$ ="FILE":B$ ="NAME"
20 PRINT A$ +B$
30 PRINT "NEW "+A$ +B$
RUN
FILENAME
NEW FILENAME
```

第四节　回顾与练习 BASICA

本节的练习部分将帮助你复习曾学过的内容。如果你不曾这样做,那么打开计算机并调用 BASICA 解释程序,现在很是时候了。

一、直接方式的例子

你可以用你的计算机在直接方式下完成基本的算术操作。为解决一个问题，以一个问号响应 OK 提示，后跟你要解决的问题的语句，按回车键。BASICA 中，问号可以用关键字 PRINT 替代。接着答案显示了出来。

敲入下面的内容并按回车键：

? 2+2

BASICA 将在你的屏幕上显示答案：

? 2+2

4

OK

练习其它的算术运算，替换＋号，使用你欲取的运算符。BASIC 语言对算术函数没有限制。你也可以键入复合的代数和三角函数。

二、间接方式的例子

BASICA 语言，可以用作实现多于单一代数计算之外的功能。你可以创建一个程序完成一系列的操作，然后把答案显示出来。开始编程，你建立的指令行叫做语句。记住一行上可以有远多于一个的语句，每一行都冠以一个数码(行编号)。

例如，建立命令 print2+3 作为一个语句，敲入：

10 print 2+3

当你按下回车键时，光标移到下一行，但什么也不发生。为使计算机完成计算，敲入下列字符并按回车键：

run

屏幕上显示出：

Ok

10 print 2+3

run

5

OK

你即写了一个 BASIC 程序。

计算机保存其计算值，直到特殊的命令(伴着 run)发出。这允许你敲入更多的指令行。当你键入 run 命令时，计算机则执行加法并显示答案。

下列程序有两个指令行。敲入：

10 x=3

20 print 2+x

现在用 run 命令让计算机计算结果。

那么屏幕上将显示如：

Ok

10 x=3

```
20 print 2+x
run
5
Ok
```

从一个计算中辨别一个程序的两种特征是：

(1)数字行；

(2)run 命令使用行。

这些特征让计算机知道所有语句已键入，计算可以从头至尾地执行。这是一个带数号的行，它首先告计算机该行是一个程序，而非一个计算，它必须不做实际的计算，直至 run 命令键入。

换言之，计算是在直接方式下执行，程序在间接方式下写就。

为再次列出整个程序，键入 list 命令并按回车键：

```
list
```

屏幕上即显示出：

```
Ok
10 x=3
20 print 2+x
run
5
Ok
list
10 X=3
20 PRINT 2+X
Ok
```

你将注意到程序的些微变化。你敲入的小写字母都被转化成大写字母。list 命令使改变自动进行。

三、功 能 键

某些键或键的组合让你以最小的击键次数完成高频使用的命令或功能。这些键称为功能键。这十个功能键(标为 F1～F10)位于键盘的左侧或上方。功能键允许在程序中一次击键便快速输入多达 15 个字符。BASICA 已分配专向功能(最高频使用的命令)给它们中的每一个键。关于这些键的指导和它们被分配的命令出现在 BASICA 屏幕的底部。为节省时间和敲键，你可以按下一个功能键以替代敲一个命令名。

例如，再列你的程序，你不必敲入 LIST 命令，你可以使用指定的功能键替代：

(1)按 F1 键

(2)按回车键

程序则出现在屏幕上。

为运行程序，简单地按下 F2 键，该键被指定为 RUN 命令。

最初地，功能键被赋给下面的特殊功能。

BASICA 功能键赋值

键	功能	键	功能
F1	LIST	F6	,"LPT1:"<—
F2	RUN<—	F7	TRON<—
F3	LOAD"	F8	TROFF<=
F4	SAVE"	F9	KEY
F5	CONT<—	F10	SCREEN 0,0,0<—

注意：功能键尾随的<—指示你在功能键之后不必按回车键。选用的命令立即执行。

如果让你选择的话，你可能改变这些键的安排。10 个功能键中的任一个或全部都可以被重新定义。欲知更多的信息，参见下篇的 KEY 和 ON KEY 语句。

四、编 辑 行

有两种基本的方式改变行，你可以：

· 删除和替换它们

· 使用 EDIT 命令改变它们

为删除一行，仅只敲该行的号码并按回车键。例如，如果你敲 12 并按回车键，12 号行便从程序中被删掉。

为使用 EDIT 命令，键入 EDIT，后跟你想修改的行号。例如，敲入下文并按回车键：

EDIT 10

则可使用下列键进行编辑：

键	功 能
cursor Up	在语句中移动光标，向上
cursor DOWN	在语句中移动光标，向下
cursor LEFT	在语句中移动光标，向左
cursor RIGHT	在语句中移动光标，向右
Backspace	删除光标左边的字符
Delete(Del)	删除当前字符
Insert(Ins)	在光标之左插入字符

例如，为修改 10 号行语句为 X=4，使用右移光标控制键移动光标至 3 下，然后敲入 4。数字 4 替代了语句中的 3。现在，按下回车键，然后按 F2 键。屏幕显示如下：

```
Ok
10 X=4
RUN
6
Ok
```

当然，你可以在输入时编辑 BASICA 程序行，也可以在它们已保存到一个程序文件中之后进行编辑。

1. 新文件中编辑行

当正键入一行时，如果输入了一个错误的字符，则可以用 BACKSPACE 或 DEL 键，或用 CTRLH 删除。当该字符被删除后，你可以在本行继续输入。

ESC 键让你删去正在键入的一行。换句话说，如果你没有按回车键，且你希望删除输入的当前行，按下 ESC 键。

要删除当前驻留内存中的全部程序，键入 NEW 命令。NEW 通常用在输入一个新的程序之前清理内存。

2. 存盘文件中编辑行

当你输入了 BASICA 程序并存盘后，你可能发现需要做修改。为做这些改动，用 LIST 语句显示出要改动的程序行：

a. 重新装载此程序

b. 敲入 LIST 命令，或按下 F1 键

c. 敲入待编辑的行号，或行号的范围

行即出现在屏幕上。

在一个程序行里编辑信息

你可以将光标定位到要修改处以改变一个行的信息，通过下列方法之一：

▲敲键覆盖已存在的字符

▲使用 BACKSPACE 键，删除光标左边的字符

▲使用数字盘上的 DEL 键删除光标所在处字符

▲通过按下数字盘上的 INS 键在光标所在处插入字符。这使得光标后的字符向右移动，以便营造空间接纳新的信息。

▲在程序行尾部加入或截去字符

如果修改多于一行，确保在每个修改了的行按回车键。该修改行将被保存在一个适当的数字序列中，即使该行没有按数字次序而修改。

注意：如果不是当光标在编辑行的某位置时按下回车键的话，BASICA 中的一个程序行不会真正地改变。

你不必在按回车键之前移动光标至行尾。BASICA 解释程序识别每一个行尾，并传送整个一行，即使当光标位于行的中间或开头处时按下回车键。

为了截断或砍掉当前光标位置处的一行，敲 CTRL－END 或 CTRL－E，然后按回车键。

如果你把程序原始地保存到了一个程序文件中，切记保存已作修改的程序版本。如果你不这样做，你的修改则不被纪录。

欲了解有关编辑键的更多的信息，参见附录中的“编辑键一览”。

五、保存你的程序文件

创建一个程序就象建立一个文件。程序是一个包含着特殊计算机指令或语句的一个文件。为了再次使用该程序，你必须保存它，就象你对一个数据文件那样。

BASICA 里保存一个文件，使用下列的步骤：

(1)按 F4 键

命令字"SAVE"出现在屏幕上。

(2)键入程序名，并按回车键。文件保存在你特定的名字下。

为调用一个保存的文件，使用下列步骤：

(1)按 F3 键

命令字 LOAD" 出现在屏幕上

(2)键入程序名

(3)回车

文件便被调入内存,且准备好为你列表、编辑和运行。

第五节　创建和使用文件

MS－DOS 系统中有两种类型的文件:

. 程序文件,该文件包含着计算机程序或指令

. 数据文件,该文件含有为程序文件使用或创建的信息

一、程序文件命令

下面是程序文件高频使用的命令和语句。本书下篇中有关于它们中的每一个的更多的信息。

SAVE filename [,a][,p]

将当前驻留在内存中的程序写到磁盘上。

LOAD filename [,r]

将程序从磁盘装载到内存。LOAD 删去当前内存中的内容并在装载程序之前关闭所有文件。

RUN filename [,r]

将程序从磁盘装载到内存并立即执行。RUN 删除当前内存中的内容并在装载程序之前关闭所有文件。

MERGE filename

将程序从磁盘装载到内存,但是不删除内存中已有的当前程序。

KILL filename

从一个磁盘上删除文件,该命令也可以用于数据文件。

NAME old filename AS new filename

改变一个磁盘文件的名字。仅仅文件名被改变。文件不做修改,且它仍留在磁盘上的同样的空间和位置。这个命令也可以用作数据文件。

二、数据文件

BASICA 程序可以运行两种类型的数据文件:

. 顺序文件. 随机存取文件

顺序文件比随机存取文件易于创建,但是存取数据的灵活性与速度受到限制。写入一个顺序文件中的数据是一系列的 ASCII 字符。数据存储一个单元接着一个单元(连续地),按照一定的顺序传送。数据也以同样的方式读出。

创建和存取随机存取文件比顺序文件要求更多的程序步,但是随机文件要求较少的磁盘空间,因为 BASICA 以一种压缩的格式存储它们。下文讨论了如何创建和使用这两种类型的数据文件。

1. 创建一个顺序文件

下面的语句和函数用于顺序文件：

CLOSE、EOF、INPUT＃、LINE、INPUT＃、LOC、LOCK、LOF、OPEN、PRINR＃、PRINT＃、USING、UNLOCK、WRITE＃

下面是创建一个顺序文件和在文件中存取数据要求的程序步骤：

A. 以输出(O)方式打开文件。当前程序将首先用此文件作为输出：

```
OPEN "O",#1,"filename"
```

B. 使用 PRINT＃或 WRITE＃语句将数据写入文件：

```
PRINT#1,A$
PRINT#1,B$
PRINT#1,C$
```

C. 为在文件中对数据进行存取，必须关闭文件并以输入(I)方式将之重新打开：

```
CLOSE#1
OPEN "I",#1,"filename"
```

D. 使用 INPUT＃或 LINE INPUT＃语句将数据从顺序文件中读入程序：

```
INPUT#1,X$,Y$,Z$
```

例 1 是创建一个顺序文件的短程序，DATA，读自终端上的输入信息。

例 1

```
10 OPEN "O",#1,"DATA"
20 INPUT "NAME";N$
30 IF N$="DONE" THEN END
40 INPUT "DEPARTMENT";D$
50 INPUT "DATE HIRED:;H$
69 PRINT#1,N$;",";D$;",";H$
70 PRINT:GOTO 20
RUN
NAME? MICH MOUSE
DEPARTMENT? AUDIO/VISUAL AIDS
DATE HIRED? 01/12/95
NAME? SHERLOCK
DEPARTMENT? REDEARCH
DATE HIRED? 08/08/94
NAME? DONE
OK
```

2. 存取一个顺序文件

例 2 中程序存取创建于例 1 程序中的文件 DATA，并显示 1994 年雇佣的每一位名字。

例 2

```
10 OPEN "I",#1,"DATA"
20 INPUT#1,N$,D$,H$
```