



# 目 录

## 第一部分 Microsoft GW—BASIC用户指南

### 第一章 欢迎使用GW-BASIC

- 1.1 系统要求..... ( 1 )
- 1.2 预备 ..... ( 1 )
- 1.3 标志的约定 ..... ( 1 )
- 1.4 这本手册的结构 ..... ( 1 )
- 1.5 书目提要 ..... ( 2 )

### 第二章 启动GW-BASIC

- 2.1 安装GW-BASIC..... ( 3 )
- 2.2 操作方式 ..... ( 3 )
- 2.3 GW-BASIC命令行格式..... ( 3 )
- 2.4 GW-BASIC语句, 函数, 命令和变量..... ( 6 )
- 2.5 行格式 ..... ( 7 )
- 2.6 返回到MS-DOS..... ( 8 )

### 第三章 回顾和练习GW-BASIC

- 3.1 直接方式的例子 ..... ( 8 )
- 3.2 间接方式的例子 ..... ( 8 )
- 3.3 功能键..... ( 10 )
- 3.4 编辑行 ..... ( 10 )
- 3.5 保存你的程序文件 ..... ( 11 )

### 第四章 GW-BASIC屏幕编辑

- 4.1 编辑新文件中的行 ..... ( 12 )
- 4.2 编辑保存文件中的行 ..... ( 12 )
- 4.3 特殊键 ..... ( 13 )
- 4.4 功能键 ..... ( 14 )

### 第五章 创建和使用文件

- 5.1 程序文件命令 ..... ( 15 )
- 5.2 数据文件 ..... ( 16 )

5.3 随机访问文件 .....	( 18 )
<b>第六章 常数、变量、表达式和运算符</b>	
6.1 常数 .....	( 23 )
6.2 变量 .....	( 24 )
6.3 类型转换 .....	( 26 )
6.4 表达式和运算符 .....	( 27 )
<b>附录A 错误码和信息</b> .....	( 33 )
<b>附录B 数学函数</b> .....	( 38 )
<b>附录C ASCII 字符码</b> .....	( 38 )
<b>附录D 汇编语言( 机器代码 ) 子程序</b> .....	( 40 )
D.1 存储分配 .....	( 40 )
D.2 CALL语句 .....	( 41 )
D.3 USR函数调用 .....	( 44 )
D.4 调用汇编语言程序的程序 .....	( 45 )
<b>附录E 转换BASIC程序到GW-BASIC</b> .....	( 48 )
E.1 串维数 .....	( 48 )
E.2 多重赋值 .....	( 48 )
E.3 多重语句 .....	( 48 )
E.4 MAT函数 .....	( 48 )
E.5 FOR-NEXT循环 .....	( 49 )
<b>附录F 通讯</b> .....	( 49 )
F.1 打开通讯文件 .....	( 49 )
F.2 通讯 I/O .....	( 49 )
F.3 COM I/O函数 .....	( 49 )
F.4 可能的错误 .....	( 50 )
F.5 INPUT\$函数 .....	( 50 )
F.6 TTY程序实例 .....	( 51 )
F.7 TTY程序实例注释 .....	( 52 )
<b>附录G 十六进制等值表</b> .....	( 54 )
<b>附录H 键扫描码</b> .....	( 55 )
<b>附录I GW-BASIC识别的字符</b> .....	( 57 )
<b>第二部分 Microsoft.GW-BASIC解释程序用户参考手册</b> .....	( 59 )

1. 引言	( 59 )
2. ABS 函数	( 59 )
3. ASC 函数	( 59 )
4. ATN 函数	( 60 )
5. AUT 命令	( 60 )
6. BEEP 语句	( 61 )
7. BLOAD 命令	( 62 )
8. BSAVE 命令	( 62 )
9. CALL 语句	( 63 )
10. CDBL 函数	( 65 )
11. CHAIN 语句	( 66 )
12. CHDIR 命令	( 67 )
13. CHR\$ 函数	( 67 )
14. CINT 函数	( 68 )
15. CIRCLE 语句	( 68 )
16. CLEAR 命令	( 79 )
17. CLOSE 语句	( 70 )
18. CLS 语句	( 71 )
19. COLOR 语句	( 72 )
20. COM ( n ) 语句	( 74 )
21. COMMON 语句	( 74 )
22. CONT 语句	( 75 )
23. COS 函数	( 75 )
24. CSNG 函数	( 76 )
25. CSRLIN 变量	( 76 )
26. CVI, CVS, CVD 函数	( 77 )
27. DATA 语句	( 78 )
28. DATER 语句和变量	( 79 )
29. DEFFN 语句	( 80 )
30. DEFINT/SNG/DBL/STR 语句	( 81 )
31. DEF SEG 语句	( 82 )
32. DEF USR 语句	( 82 )
33. DELETE 命令	( 83 )
34. DIM 语句	( 84 )
35. DRAW 语句	( 85 )
36. EDIT 命令	( 87 )
37. END 语句	( 87 )
38. ENVIRON 语句	( 88 )

39. ENVIRON\$函数	( 89 )
40. EOF 函数	( 90 )
41. ERASE语句	( 91 )
42. ERDEV和ERDEV\$ 变量	( 92 )
43. ERR和E RL变量	( 92 )
44. ERROR语句	( 93 )
45. EXP 函数	( 94 )
46. EXTERR函数	( 95 )
47. FIELD命令	( 95 )
48. FILES命令	( 96 )
49. FIX 函数	( 97 )
50. FOR和NEXT语句	( 67 )
51. FRE 函数	( 99 )
52. GET语句(文件)	( 100 )
53. GET语句(图形)	( 100 )
54. GOTO语句	( 102 )
55. HEX\$ 函数	( 103 )
56. IF语句:	( 104 )
57. INKEY\$变量	( 105 )
58. INP函数	( 106 )
59. INPUT 语句	( 107 )
60. INPUT# 语句	( 108 )
61. INPUT\$函数	( 109 )
62 INSTR 函数	( 110 )
63. INT函数	( 111 )
64. IOCTL 语句	( 111 )
65. IOCTL\$函数	( 112 )
66. KEY 语句	( 112 )
67. KEY ( n ) 语句	( 114 )
68. KILL命令	( 115 )
69. LEFT\$ 函数	( 115 )
70. LEN 函数	( 116 )
71. LET 语句	( 116 )
72. LINE 语句	( 117 )
73. LINE INPUT语句	( 119 )
74. LINE INPUT# 语句	( 120 )

75. LIST 命令	( 121 )
76. LLIST命令	( 121 )
77. LOAD命令	( 122 )
78. LOC函数	( 122 )
79. LOCATE 语句	( 123 )
80. LOCK语句	( 124 )
81. LOF函数	( 125 )
82. LOG函数	( 125 )
83. LPOS函数	( 126 )
84. LPOS ( × )	( 126 )
85. LPRINT和LPRINT USING语句	( 126 )
86. LSET和RSET语句	( 127 )
87. MERGE 命令	( 127 )
88. MID\$函数	( 128 )
89. MID\$ 函数	( 129 )
90. MKDIR 命令	( 129 )
91. NAME命令	( 130 )
92. NEW 命令	( 131 )
93. OAT\$函数	( 131 )
94. ON COM ( n ), ON KEY ( n ), ON PEN, ON PLAY ( n ), ON STRIG ( n ), 和ON TIMER ( n ) 语句	( 132 )
95. ON ERROR GOTO 语句	( 136 )
96. ON.....GOSUB和ON.....GOTO语句	( 136 )
97. OPEN语句	( 137 )
98. OPEN COM ( n ) 语句	( 140 )
99. OPTION BASE语句	( 142 )
100. OUT语句	( 142 )
101. PAINT语句	( 143 )
102. PALETTE, PALETTE USING 语句	( 145 )
103. PCOPY语句	( 147 )
104. PEEK函数	( 148 )
105. PEN语句和函数	( 148 )
106. PLAY语句	( 149 )
107. PLAY ( n ) 函数	( 150 )
108. PMAP函数 ( 图形 )	( 151 )
109. POINT函数	( 152 )
110. POKE 语句	( 153 )
111. POS 函数	( 153 )

112.	PRESET和PSET 语句	( 154 )
113.	PRINT语句	( 155 )
114.	RRINT USING 语句	( 156 )
115.	PRINT #和PRINT # USING 语句	( 159 )
116.	PUT语句(文件)	( 160 )
117.	PUT语句(图形)	( 161 )
118.	RANDOMIZE语句	( 162 )
119.	READ语句	( 164 )
120.	REM 语句	( 165 )
121.	RENUM语句	( 166 )
122.	RESET命令	( 166 )
123.	RESTORE语句	( 167 )
124.	RESUME 语句	( 167 )
125.	RETURN. 语句	( 168 )
126.	RIGHT\$函数	( 168 )
127.	RNDIR 命令	( 169 )
128.	RND 函数	( 169 )
129.	RUN 命令	( 170 )
130.	SAVE命令	( 171 )
131.	SCREEN 函数	( 171 )
132.	SCREEN 语句	( 172 )
133.	SGN 函数	( 177 )
134.	SHELL 语句	( 178 )
135.	SIN函数	( 179 )
136.	SOUND语句	( 179 )
137.	SPACES\$函数	( 181 )
138.	SPC函数	( 182 )
139.	SQR函数	( 182 )
140.	STICK函数	( 183 )
141.	STOP语句	( 183 )
142.	STR\$ 函数	( 184 )
143.	STRIG语句和函数	( 184 )
144.	STRIG ( n ) 语句	( 185 )
145.	STRING\$函数	( 186 )
146.	SWAP 语句	( 186 )
147.	SYSTEM命令	( 187 )
148.	TAB 函数	( 187 )
149.	TAN 函数	( 188 )

150.	TIME\$ 语句和变量	( 188 )
151.	TIMER 函数	( 189 )
152.	TRON/TROFF 命令	( 190 )
153.	UNLOCK 语句	( 191 )
154.	USR 函数	( 192 )
155.	VAL 函数	( 192 )
156.	VARPTR 函数	( 193 )
157.	VARPTR\$ 函数	( 195 )
158.	VIEW 语句	( 196 )
159.	VIEW PRINT 语句	( 197 )
160.	WAIT 语句	( 197 )
161.	WHILE-WEND 语句	( 198 )
162.	WIDTH 语句	( 199 )
163.	WINDOW 语句	( 200 )
164.	WRITE 语句	( 202 )
165.	WRITR # 语句	( 202 )



# 第一部分 Microsoft GW BASIC 用户指南

## 第一章 欢迎使用 GW-BASIC

Microsoft GW-BASIC是简单、易学、易用、类英语的语句和数学记数法的计算机程序设计语言。用GW-BASIC你能编写简单又复杂的程序，使之在你的计算机上运行。你也能修改用GW-BASIC编写的现存的软件。

这本指南是为了帮助你同MS-DOS操作系统一起使用GW-BASIC。节1.5列出了教你如何编程的方法。

### 1.1 系统要求

GW-BASIC这个版本需要MS-DOS版本3.1或更高版本支持。

### 1.2 预备

你的GW-BASIC文件在MS-DOS用户参考手册背面的MS-DOS软盘上。在你开始之前，一定要做一份该盘的工作拷贝。

### 注意

这本手册是为熟悉MS-DOS操作系统用户而写的。有关MS-DOS的更多信息，参阅*Microsoft MS-DOS 3.2*用户指南和用户参考手册。

### 1.3 标志约定

整个手册中，下面约定用来区别文本元素：

**黑体** 用于命令、选择项、开关和文法的直接量部分，直接量部分必须恰如文中显示的形式出现。

**斜体** 用于文件名、变量和用户键入的表示文本类型的位置占有。

**单格** 用命令行实例、程序代码和例子，对话实例。

**小体大写** 用于键、键序列和字首组合词。

方括号包含的是选择命令行的元素。

### 1.4 这本手册的结构

GW-BASIC用户指南分成六章、九个附录和一个词汇表：

第一章，“欢迎使用GW-BASIC”，叙述这本手册。

第二章，“启动GW-BASIC”，是关于如何开始编程的基本准则。

第三章，“回顾和练习GW-BASIC”让你使用在第二章中说明的GW-BASIC准则。

第四章，“GW-BASIC屏幕编辑”，讨论在输入或修改GW-BASIC程序时能用的编辑

命令。也解释了十个功能键、其它键及其打入键的组的独一无二的特性。

第五章, “创建和使用文件”, 告诉你如何创建文件和使用软盘输入/输出(I/O) 的步骤。

第六章, “常数、变量、表达式和运算符”, 定义GW-BASIC的元素, 说明你如何使用它们。

附录A, “错误码和信息” 是在使用GW-BASIC 期间, 你可能遇到的所有错误码 和错误信息的概述。

附录B, “数学函数”, 说明如何计算一些非GW-BASIC内部函数的数学函数。

附录C, “ASCII字符码”, 列出GW-BASIC识别的ASCII字符码。

附录D, “汇编语言( 机器代码) 子程序”, 说明如何用GW-BASIC 包含汇编语言子程序。

附录E, “转换BASIC程序成为GW-BASIC”, 为如何转换用BASIC写的程序成为GW-BASIC程序提供方法。

附录F, “通讯”, 解释支持同其它计算机和外设的RS-232 异步通讯的GW-BASIC 语句。

附录G, “十六进制等值表”, 列出十进制和二进制等价于十六进制的值。

附录H, “键扫描码”, 列出和演示了GW-BASIC中使用的键扫描码的值。

附录I, “GW-BASIC识别的字符”, 叙述GW-BASIC的字符集。

词汇表, 定义常用于GW-BASIC和数据处理中的字和词。

## 1.5 书目提要

这本手册是指导使用GW-BASIC解释程序: 该指导不打算教BASIC程序设计语言。下面课本对想学习BASIC程序设计的人是有益的,

Albrecht, Robert L., LeRoy Finkel, and Jerry Brown. *BASIC*. 2d ed. New York: Wiley Interscience, 1978.

Coan, James. *Basic BASIC*. Rochelle Park, N.J.: Hayden Book Company, 1978.

Dwyer, Thomas A. and Margot Critchfield. *BASIC and the Personal Computer*. Reading, Mass.: Addison-Wesley Publishing Co., 1978.

Ettlin, Walter A. and Gregory Solberg. *The MBASIC Handbook*. Berkeley, Calif.: Osborne/McGraw Hill, 1983.

Knecht, Ken. *Microsoft BASIC*. Portland, Oreg.: Dilithium Press, 1982.

## 第二章 启动GW—BASIC

这一节叙述如何装GW-BASIC到你的系统。也解释两种不同类型的操作方式、行格式、和GW-BASIC的不同元素。

## 2.1 装GW-BASIC

为使用GW-BASIC语言，你必须从你的MS-DOS软盘的工作副本中安装GW-BASIC到计算机内存。用下列步骤：

1. 打开你的计算机，
2. 插MS-DOS盘的工作副本到你的计算机的驱动器A，并且压RETURN。
3. 在A>提示之后打入下面命令，并压RETURN：

```
gwbasic
```

一旦你进入GW-BASIC，GW-BASIC就提示ok，代替MS-DOS提示A>。

在屏幕上，行XXXXX Bytes Free表示在使用GW-BASIC期间内存中有多少字节可以使用。

功能键（F1—F10）的分配出现在屏幕的底行。这些功能键可以减少键的敲入，节省时间。第四章，“GW-BASIC屏幕编辑”包含有关功能键的详细信息。

## 2.2 操作方式

一旦初始化了GW-BASIC(已装入)，它显示Ok提示。OK意指GW-BASIC是处命令级，也就是等待接受命令。在这一点，GW-BASIC可以用两种方式的任一种：直接方式或间接方式

### 2.2.1 直接方式

在直接方式中，GW-BASIC语句和命令打入时就执行。算术和逻辑操作的结果可能立即显示和/或可能存放以便后用，但是它们的指令执行之后就丢失。这种方式对调试和把GW-BASIC用作不要完整程序的快速计算的计算器是非常有用的。

### 2.2.2 间接方式

间接方式用于打入程序。行号总是先于程序行，并且程序行存于内存中。打入RUN命令，存于内存中的程序就执行。

## 2.3 GW-BASIC命令行格式

在使用GW-BASIC过程中，GW-BASIC命令行让你改变使用的环境和条件。

### 注意

当你指定对GW-BASIC的操作环境修改时，一定要保持在语法语句中出现的参数序列。跳过一个参数，就要插入一个逗号。这让计算机知道你没有对特殊参数作任何改变。

GW-BASIC使用下面形式的命令行：

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

filename是GW-BASIC程序文件的名称。如果这个参数存在，那么GW-BASIC如RUN命令已给一样执行。如果文件名没有提供扩展，则假设缺省的文件扩展为.BAS。BAS扩展表示该文件是GW-BASIC文件。一个文件名包含的最大字符是八个，另加一个小数点

和三个扩展字符。

<*stdin*再定向GW-BASIC标准输入来自指定的文件。当已使用再定向时，它必须在任何开关之前出现。

当你有多个文件，你的程序可能使用这些文件，而你希望指定一个特殊的输入文件时，这可能被使用。

>*stdout*再定向GW-BASIC的标准输出到指定的文件或设备。当已使用再定向时，它必须在任何开关之前出现。在*stdout*之前使用》引起附加输出。

通过在命令行上提供输入输出文件名，GW-BASIC能再定向来自标准输入（键盘）和写入标准输出（屏幕），如下所示：

```
gwbasic program name<input file [ > ]>output file
```

文件再定位的解释在GW-BASIC命令行讨论之后。

在命令行中开关经常出现；开关为命令选派了一个指定的行动过程。正好同那种设置的缺省设置相反。一个开关参数之前用斜杠（/）。

/f : n 设置了在 GW-BASIC 程序执行期间可以同时打开的最大文件个数。每个文件需要文件控制块（FCB）194个字节加上数据缓冲区128个字节。用/S : 开关能改变数据缓冲区大小。如果/f : 开关省略，则最大打开文件的个数缺省值是3。如果/i 开关也不在命令行上指定，那么/f : 开关忽略。

/i使GW-BASIC静态地为文件操作分配所需的空间，这是根据/s和/f开关。

/s : n 设置文件允许使用的最大记录长度。在OPEN语句中的记录长度选择不可能超过这个值。如果/s : 开关省略，则记录长度的缺省是128个字节。最大的记录大小是32767。

/c : n 控制RS-232通讯。如果RS-232插件存在，则/c : 0禁止RS-232支持，并且对每一存在的RS-232插件禁止任何后来的I/O企图。如果/c : 开关省略，256字节分配给接收缓冲区和128字节分配给每一存在插件的传送缓冲区。当RS-232插件不存在时，/c : 开关没有任何影响。/c : n开关为接收缓冲区分配n个字节，为每个RS-232存在的插件分配128个字节给传送缓冲区。

/m : n [ , n ] 设置GW-BASIC使用的最高存储器位置（第一个n）和最大块的大小（第二个n）。GW-BASIC试图为数据和线段分配64K字节的存储器。如果机器语言子程序同GW-BASIC程序一起使用，则用/m : 开关设置了GW-BASIC可以使用的最高位置。最大块的大小是16的倍数。它为超过GW-BASIC工作空间的用户程序（汇编语言子程序）保留空间。

最大块的大小的缺省值是最高存储器的位置。最高存储器的位置的缺省值是64 K，除非最大块的大小已指定，在这种情形中缺省值是最大块的大小（16的倍数）。

/d允许某些函数返回双精度结果。当/d1开关指定时，大约3000字节的附加代码空间被使用。受影响的函数是ATN, COS, EXP, LOG, SIN, SQR和TAN。

### 注意

所有开关数值可以用十进制、八进制（前导&O），或十六进制（前导&H）指定。

GW-BASIC命令行实例如下：

下面使用64K字节存储器和三个文件；装入和执行程序文件*payroll.bas*；

A>gwbasic PAYROLL

下面使用64K字节的内存和六个文件，装入和执行程序文件`invent.bas`。

A>gwbasic INVENT /F:6

下面禁止RS-232支持，只使用内存的第一个32K字节。上面的32K字节用于保留用户程序

A>gwbasic /c:0/M:32768,4096

下方使用四个文件和允许512字个的最大记录长度：

A>gwbasic/F:4/S:512

下方使用64k字节的内存和三个文件。分配512个字节给RS-232接收缓冲区冲和128个字节给传递缓冲区；装入并执行程序文件`tty.bas`。

A>gwbasic TTY/C:512

有关RS-232通讯的更多信息，看附录F。

### 标准输入和输出的再定向

当已再定向时，所有INPUT，LINE INPUT，INPUT\$，和INKEY\$语句从指定的输入文件读入而不是键盘。

所有PRINT语句写入指定的输出文件，代替屏幕。

错误信息到标准输出和到屏幕。

来自KYBD的文件输出仍然读自键盘。

文件输出到SCRN仍然输出到屏幕。

当使用ON KEY n语句时，GW-BASIC继续接收键。

当输出再定向时，打入CTRL-BREAK使GW-BASIC关闭任何打开的文件，发出信息“Break in line nnnn”到标准输出，中止GW-BASIC，并返回到MS-DOS。

当输入再定向时，GW-BASIC继续读该源的信息直到检测到CTRL-Z。用EOF函数(文件结束函数)可以测试这个条件。如果不是CTRL-Z中止文件的，或者GW-BASIC文件输入语句读到文件结束，然后所有打开的文件关闭，并GW-BASIC返回到MS-DOS。

关于这本书中提及的这些语句和其它语句的进一步信息，有关函数、命令和变量的进一步信息，参阅GW-BASIC用户参考手册。

一些再定向的例子如下：

```
GWBasic MYPROG>DATA, OUT
```

通过INPUT和LINE INPUT语句读入的数据继续来自键盘。通过PRINT语句输出数据到`data.out`文件。

```
gwbasic MYPROG<DATA, IN
```

通过INPUT和LINE INPUT语句读入的数据来自`data.in`。通过PRINT输出的数据继续到屏幕。

```
gwbasic MYPROG<MYINPUT.DAT>MYOUTPUT.DAT
```

通过INPUT和LINE INPUT语句读入的数据现在来自文件`myinput.dat`，通过PRINT语句输出的数据到`myoutput.dat`。

```
gwbasic MYPROG </SALES/JOHN/TRANS.DAT>>/SALES/  
SALES.DAT
```

通过INPUT和LINE INPUT语句读入的数据现在来自于文件/sales/john/trans.dat, 通过PRINT语句输出的数据附加到文件/sales/sales.dat。

## 2.4 GW-BASIC语句、函数、命令和变量

一个GW-BASIC程序由几个元素组成:关键字、命令、语句、函数和变量

### 2.4.1关键字

GW-BASIC关键字,如print, goto, 和return对GW-BASIC解释程序有特定的意义。GW-BASIC解释关键字作为语句或命令的一部分。

关键字也称保留字。它们不可能用作变量名,否则系统将解释它们为命令。然而,关键字可以嵌入变量名中。

为最有效地利用内存空间,关键字以标记的形式存于系统中(1或2个字节字符)。

### 2.4.2命令

命令和语句两个都是可执行指令。命令和语句之间的差别是命令通常以直接方式执行,或者在解释程序的命令级中执行。它们通常执行一些程序维护类的操作,如编辑,装入或保存程序。当GW-BASIC被调用,并且GW-BASIC的提示Ok出现时,系统假定处于命令级。

### 2.4.3语句

语句,如ON ERROR...GOTO,是一组GW-BASIC关键字,通常用在GW-BASIC程序行中作为一个程序的部分。当程序运行时,随着语句的出现它们执行。

### 2.4.4函数

GW-BASIC解释程序既执行数值函数又执行串函数。

#### 2.4.4.1数值函数

GW-BASIC解释程序能执行某些数字(算术或代数)计算。例如,它计算角度 $x$ 的正弦( $\sin$ ),余弦( $\cos$ ),或正切( $\tan$ )。

如果没有其它说明,那么数值函数只返回整数和单精度结果。

#### 2.4.4.2串函数

串函数对串进行操作。例如,TIME\$和DATE\$返回系统知道的时间和日期。如果在系统启动期间打入当前时间和日期,则给出正确的时间和日期(计算机中的内部时钟保持记录)。

#### 2.4.4.3用户定义的函数

通过DEF FN语句的手段,用户能定义函数。这些函数可能是串或数值的。

### 2.4.5变量

某些字母数字字符串被赋予值,并称之为变量。当变量嵌入GW-BASIC程序中时,当它们执行时提供信息。

例如,ERR定义出现在程序中的最近错误,ERRL给出哪个错误的位置。变量能被定

义，并且/或者由用户或程序内容重新定义。

所有GW-BASIC命令、语句、函数和变量分别在GW-BASIC用户参考手册中叙述。

## 2.5 行格式

GW-BASIC的每个元素能构成一个程序的节，也称为语句。这些语句非常类似于英文中的句子。然后，语句以逻辑的意义放在一起创建程序。GW-BASIC用户参考手册说明所有在GW-BASIC中使用的可得到的语句。

在GW-BASIC程序中，行有下面格式：

```
nnnnn statement ( statements )
```

nnnnn是行号。

statement是GW-BASIC的语句。

一个GW-BASIC程序行总是开始于一个行号并且必须包含至少一个字符，但不多于255个字符。行号表示程序行存于内存中的顺序。当转移操作和编辑操作时，也用作参考。当你压RETURN键时，程序行结束。

依赖你的程序逻辑，一行可以多于一个语句。如果这样，每个语句之间用冒号(；)分开。一个程序中的每一行应该前导一个行号。这个行号可以是0到65529的任一个整数。用诸如10, 20, 30和40的行号是一个习惯，为了替你以后想包括的附加行留下空间。因为计算机以数值的顺序送行语句，所以附加行不必以连续的次出现在屏幕上；例如。如果你在行60之后打入35，那么计算机仍然在行30之后运行行35，在行40之前执行行35。这一技术可以使你包含已忘记的一行而不必重新打入整个程序。

一个屏幕的宽度是80个字符。如果你的程序超过这个宽度，光标将自动地卷到下一屏幕行。只有当你压RETURN键时，计算机才承认一行的终止。当你到了屏幕的边缘(或超过)而没有压RETURN键，计算机自动地卷这一行。你也可以压CTRL-RETURN，使光标移到下一屏幕行的开始而没有实际打入这一行。当你压RETURN时，使行存放于这个程序中，整个逻辑行传送到GW-BASIC。

在GW-BASIC中，开始于一个数值字符的任何文本行认为是程序行，当打入RETURN键之后，以下面三种方式处理：

- 新行加入该程序中。如果行号是合法的(在0到65529以内)，和如果至少一个字母或特殊字符跟在该行的行号后面，则就加入新行。
- 修改一个存在的行。如果这一行号同程序中存在行的行号相匹配，则修改了存在行。新打入行的文本代替存在行。这种处理称为编辑。

### 注意

重新使用一个存在的行号引起包含在原行中的所有信息的丢失。当在间接方式中打入数值时一定要小心，你可能偶然地抹去一些程序行。

- 删除一个存在行。如果行号同存在行的行号相匹配，和打入的行只包含一个行号，则删除就发生。如果企图删除一个不存在的行，那么“Undefind line number”错

误信息就显示。

## 2.6 返回到MS-DOS

在你返回到MS-DOS之前，你必须保存在GW-BASIC下打入的工作，否则，该工作将丢失。

为返回到MS-DOS，在Ok提示之后打入下面信息并压RETURN，

```
system
```

系统返回到MS-DOS，A>提示符出现在你的屏幕上。

## 第三章 回顾和练习GW-BASIC

这一章中的实习会话帮助你回顾所学的东西。如果你没有这样做过，那么，是打开计算机和装入GW-BASIC解释程序的好机会。

### 3.1 直接方式的例子

你现在直接方式中用你的计算机执行基本算术操作。GW-BASIC把下面符号认作算术运算符：

操作	GW-BASIC运算符
加	+
减	-
乘	*
除	/

为开始研讨一个问题，用提问标志(?)响应Ok提示，后跟一个你想解决的问题的语句，再压RETURN键。在GW-BASIC中，提问标志能同关键字PRINT交换使用。问题的答案就显示在屏幕上。

打入下面信息并压RETURN键：

```
? 2+2
```

GW-BASIC在屏幕上显示答案：

```
2+2
```

```
4
```

```
Ok
```

为实习其它算术操作，用期望的运算符替换+符号。

GW-BASIC语言不限制算术函数，你也能打入复杂的代数三角函数。这些函数与格式在第六章“常数、变量、表达式和运算符”中提供。

### 3.2 间接方式例子

GW-BASIC语言能用于除简单代数计算的函数。你能创建一个程序，执行一系列操作，



然后显示答案。为开始编程，建立称为语句的几行指令。记住一行中可能多于一个语句，每一行之前有一个数值。

例如，建立命令PRINT 2 + 3作为一个语句，打入下面信息：

```
10 print 2 + 3
```

当你压RETURN键时，光标移到下一行，但是什么也没有发生。为使计算机执行你的计算，打入下面信息并压RETURN键，

```
run
```

你的屏幕看起来应该象：

```
Ok
```

```
10 print 2 + 3
```

```
run
```

```
5
```

```
Ok
```

你刚刚已用GW-BASIC编写了一个程序。

计算机保留它的计算直到特定地命令继续执行（用RUN命令）。这允许你打入多行指令。当你打入RUN命令时，计算机做加法并显示答案。

下面程序有两行指令，打入：

```
10 x = 3
```

```
20 print 2 + x
```

现在使用RUN命令使计算机计算答案。

你的屏幕上应该看到象这样的显示：

```
Ok
```

```
10 x = 3
```

```
20 print 2 + x
```

```
run
```

```
5
```

```
Ok
```

区别一个程序同一个计算的二个特征是：

1. 已编号的行
2. RUN命令的使用

这些特征让计算机知道所有语句已打入，从开始到结束能执行计算。行的编号首先向计算机发信表示这是一个程序，而不是一个计算。计算机一定不做实际的计算直到打入RUN命令。

换一句话，计算是在直接的方式下做出。程序是在间接的方式下编写。

为再次显示整个程序，打入LIST命令并压RETURN键，

```
list
```

你的屏幕上应该看到象这样的显示：