

APPLE II 资料(五)

APPLESOFT II 程序设计参考手册



目 录

概述	(1)
第一章 基本命令和语句	(3)
§ 1 立即执行命令	(3)
§ 2 延迟执行命令	(3)
§ 3 数字格式	(5)
§ 4 作彩色图形的例子	(6)
§ 5 打印格式	(7)
§ 6 变量名	(8)
§ 7 IF...THEN 语句	(9)
§ 8 其它着色的例子	(10)
§ 9 FOR...NEXT 语句	(12)
§ 10 数 组	(14)
§ 11 GOSUB...RETURN 语句	(15)
§ 12 READ...DATA...RESTORE 语句	(16)
§ 13 实型、整型及字符串变量	(17)
§ 14 字符串	(18)
§ 15 比较完善的彩色作图的例子	(22)
§ 16 高分辨率彩色作图的介绍	(24)
第二章 定 义	(27)
§ 1 语法定义和缩写字	(27)
§ 2 表达式求值的规则	(31)
§ 3 类型转换	(31)
§ 4 执行方式	(31)
第三章 系统命令和实用命令	(32)
§ 1 LOAD 和 SAVE	(32)
§ 2 NEW	(32)
§ 3 RUN	(33)
§ 4 STOP, END, ctrl C, reset, CONT	(33)
§ 5 TRACE 和 NOTRACE	(34)
§ 6 PEEK	(34)
§ 7 POKE	(34)
§ 8 WAIT	(35)
§ 9 CALL	(36)

§ 10	HIMEM :	(36)
§ 11	LOMEM :	(37)
§ 12	USR	(38)
第四章	编辑命令及与格式有关的命令	(39)
§ 1	LIST	(39)
§ 2	DEL	(40)
§ 3	REM	(40)
§ 4	VTAB	(41)
§ 5	HTAB	(41)
§ 6	TAB	(41)
§ 7	POS	(42)
§ 8	SPC	(42)
§ 9	HOME	(43)
§ 10	CLEAR	(43)
§ 11	FRE	(43)
§ 12	FLASH, INVERSE 及 NORMAL	(43)
§ 13	SPEED	(44)
§ 14	esc A, esc B, esc C 和 esc D	(44)
§ 15	repeat	(45)
§ 16	右箭头键和左箭头键	(45)
§ 17	ctrl X	(45)
第五章	数组和字符串	(46)
§ 1	DIM	(46)
§ 2	LEN	(47)
§ 3	STR\$	(47)
§ 4	VAL	(47)
§ 5	CHR\$	(47)
§ 6	ASC	(47)
§ 7	LEFT\$	(48)
§ 8	RIGHT\$	(48)
§ 9	MID\$	(48)
§ 10	STORE 和 RECALL	(49)
第六章	输入/输出命令	(52)
(同时参看第三章中的 LOAD 和 SAVE 命令, 以及第五章中的 STORE 和 RECALL 命令)		
§ 1	INPUT	(52)
§ 2	GET	(53)
§ 3	DATA	(54)
§ 4	READ	(54)

§ 5	RESTORE	(55)
§ 6	PRINT	(55)
§ 7	IN#	(56)
§ 8	PR#	(56)
§ 9	LET	(57)
§ 10	DEF FN	(57)
第七章	与控制流程有关的命令	(59)
§ 1	GOTO	(59)
§ 2	IF...THEN 和 IF...GOTO	(59)
§ 3	FOR...TO...STEP	(60)
§ 4	NEXT	(61)
§ 5	GOSUB	(62)
§ 6	RETURN	(62)
§ 7	POP	(62)
§ 8	ON...GOTO 和 ON...GOSUB	(63)
§ 9	ONERR GOTO	(63)
§ 10	RESUME	(64)
第八章	作图与游戏控制	(66)
§ 1	TEXT (低分辨率图形)	(66)
§ 2	GR	(66)
§ 3	COLOR	(66)
§ 4	PLOT	(67)
§ 5	HLIN	(67)
§ 6	VLIN	(68)
§ 7	SCRN (高分辨率图形)	(68)
§ 8	HGR	(69)
§ 9	HGR2	(69)
§ 10	HCOLOR	(70)
§ 11	HPlot (游戏控制)	(70)
§ 12	PDL	(71)
第九章	高分辨率图形的模式	(73)
§ 1	如何建立一张模式表	(73)
§ 2	模式表的保存	(78)
§ 3	模式表的使用	(78)
§ 4	DRAW	(79)
§ 5	XDRAW	(79)
§ 6	ROT	(80)
§ 7	SCALE	(80)
§ 8	SHLOAD	(80)

第十章 数学函数.....	(82)
§ 1 APPLESOFT 已配置的数学函数.....	(82)
§ 2 派生函数.....	(83)

附 录

附录A: APPLESOFT BASIC 的进入	(84)
附录B: 程序的编辑	(88)
附录C: 错误信息	(92)
附录D: 存贮空间的节省	(94)
附录E: 提高程序运行速度	(96)
附录F: 键字的十进制标记	(97)
附录G: APPLESOFT 中的保留字	(99)
附录H: 将整数 BASIC 程序改换成 APPLESOFT	(101)
附录I: 存贮单元配置图	(102)
附录J: PEEK, POKE 和 CALL 的使用	(104)
附录K: ASCII 字符编码.....	(112)
附录L: APPLESOFT 零页的使用	(115)
附录M: APPLESOFT 和整数 BASIC 之间不同点	(117)
附录N: 按字母顺序排列的语法定义和缩略语	(120)
附录O: APPLESOFT 命令的摘要说明	(126)
附录P: APPLESOFT BASIC 命令参考卡	(136)

概 述

一、引 言

APPLESOFT II BASIC 是在 APPLE BASIC 基础上进行了大量扩充后的语言，由于 APPLE I 计算机具有其它使用 BASIC 语言的计算机所没有的许多特性，所以对 BASIC 进行了扩充，给 BASIC 语言增加了少许新的单词，对使用 APPLESOFT 的任何人来说，这些特性立即可用。APPLESOFT 所提供的这些特性包括：APPLE 的低分辨率彩色作图，高分辨率彩色作图和直接模拟输入（游戏控制器）等。本手册有一个特点：它不是一本自学手册，因为 APPLE 提供了另一本手册（APPLE I BASIC 程序设计手册）。即使你以前从未接触过计算机，那本手册也能帮助你学习程序设计。这本手册假定你已经知道怎样用 BASIC 进行程序设计，只是想学习 APPLESOFT 所具有的附加功能。第一章基本命令和语句对语言必须提供的语言和命令，进行扼要的描述，手册的其它部分则严格而又仔细地描述了语言中的每一个语句，以及每一个语句是怎样工作的。为了帮助你消除某些手册可能引起的失败和麻烦，本手册指出了因程序设计错误可能使你很为难的地方，对于这些地方使用了特殊的符号以引起你的注意。

对于用来描述 APPLESOFT 的方法，其本身几乎就是一种简单的语言，在你刚刚会使用它之后，你将发现这本手册将使你尽快地精确了解在该语言中什么是合法的，什么是非法的，它没有给你留下任何有关句子解释含义不清的地方。（这些地方在使用纯英语进行描述时可能发生含义不清的问题。）

高级程序员将发现本手册是非常有用的，对于初学者来说，不久他们也将不再是个初学者了，而且将了解 APPLE 所提供的一些额外的功能，它是一本独特完整的手册。诚然，一本厚厚的手册看上去似乎令人可怕，但当你需要这些资料的时候，你将高兴地发现，我们为此而花费的代价是很有必要的。

二、本手册的使用

本手册假定你已经有了—些基本的用 BASIC 程序设计语言进行工作的知识，如果你不熟悉 BASIC 语言在《APPLE I BASIC 程序设计手册》中给出了一般性的描述，那本书中描述的是另一版本的 BASIC 语言，它和 APPLESOFT II 相类似，但比 APPLESOFT 更简单。

我们建议你在参考本手册的时候，建立并运行 APPLESOFT II BASIC（通常简称为 APPLESOFT）程序，以便你可以在计算机上试出手册描述或建议的任何东西。如果 APPLESOFT 正在你的系统上运行，则 APPLESOFT 将显示提示字符（），可参见附

录A, 它解释了怎样将 APPLESOFT 装入你的计算机。

当阅读这本手册时, 你必须知道两个术语, 一个是“语法”它涉及到计算机命令的结构以及命令的各个部分的正确形式和次序。另一个是“语法分析”, 它涉及到计算机试图解释你所打入的东西的方法, 并且为了执行计算机命令而区别出计算机命令的各个部分。例如, APPLESOFT 的语法允许你打入:

$$12X=4*3^2$$

当 APPLESOFT 分析这个输入时, 它首先挑选出 12 作为程序行号, 然后把 X5 作为算术变量名, 最后 APPLESOFT 计算出 3^2 等于 9, 再乘 4, 把 36 这个值赋给名为 X5 的变量。

第一章为已经有一点 BASIC 程序设计经验的人提供了许多 APPLESOFT 命令的概述。通过使用你能打入计算机的例子, 来介绍许多初步的概念。附录 B 给出了编辑 APPLESOFT 程序的指点。

第二章的开头所介绍的记号是用来简明无二义性地描述 APPLESOFT 语法的。它将在你了解必须怎样来构造命令的过程中, 使你节省时间, 少走弯路, 你自己不一定要用到这些记号, 但它可以帮助你回答许多在正文中没有特别讨论的问题。例如, 方括号 ([和]) 用以指出命令的选用部分, 花括号 ({和}) 用以指明可以重复的部分。

所以 [LET]C=3 指出: 单词 LET 是选用的, 可以省略

REM[{字符}]指出: REM 命令的组成是单词 REM 后任意地跟有一个或多个字符。

在第二章第一部分中的语法缩写词和定义, 按照逻辑顺序排列, 你将可看到我们是怎样建立我们的符号和定义的。在你遇到正文中的符号和定义前, 你可能不愿过问它, 此时你可以参考附录 N 中按字典次序排列的语法术语汇编表。

第三章到第十章详细地给出了 APPLESOFT 的命令解释, 并且根据内容的不同而进行了分组, 如果你对某条特定的命令感兴趣, 并想找到它时字母顺序的索引将告诉你应上何处去找。附加的参考资料不包括在该附录中的章节里。

在某一段之前的某些地方你将会看到符号。



它指出了不同一般的必须引起特别警惕的特性。

符号



位于某些段之前, 在这些段中描述了这样的场合。APPLESOFT 可能无法从这些场合中恢复, 你将会遗失你的程序, 也可能将不得不重新启动 APPLESOFT。

第一章 基本命令和语句

§ 1 立即执行的命令

试着打入：

```
PRINT 10-4
```

然后按 RETURN 键。

APPLESOFT II 将立刻打印出

6

当你一按 RETURN 键，你打入的 PRINT 语句就立即被执行 APPLESOFT 对 PRINT 后的式子求值，然后打出它的值，这里是 6。

现在试着打入：

```
PRINT 1/2, 3*10
```

(* 表示乘/表示除)

当你按下 RETURN 键后，APPLESOFT 将打印：

.5 30

正如你所看见的，APPLESOFT 不仅能做减法，还能做乘法和除法。注意，在 PRINT 命令中是怎样用逗号(,)来打印两个值，而不是一个值的。PRINT 命令所用的逗号把 40 个字符宽的行分为 3 列，即“制表(tab)域”，可参见第六章 PRINT 命令一节中关于制表域的讨论。

§ 2 延迟执行的命令

某些命令例如你刚才打过的 PRINT 命令，称为“立即执行命令”，还有另一种命令称为“延迟执行”命令，每一条延迟执行命令都以一个“行号”开始，行号是 0 到 63999 之间的一个整数。试着打入下列几行：

```
10 PRINT 2+3
```

```
20 PRINT 2-3
```

(记住：每一行必须按 RETURN 键来结束)

延迟执行命令的序列称为“程序”，APPLESOFT BASIC 把延迟执行命令存贮在 APPLE 的内存里，而不是立即执行它，当你打入 RUN 后，APPLESOFT 首先执行存贮在内存中的具有最小行号的语句，直到整个程序都执行了为止。

假定你现在打入 RUN (要记住，每一行的结尾都要按 RETURN 键)

```
RUN
```

APPLESOFT 现在将在屏幕上显示：

5

-1

在先前的例子中，我们首先打入行 10，然而打入行 20，然而，这与你打入的延迟执行语句的先后顺序没有什么差别，APPLESOFT 总是按照它们相应的行号将它们排列成正确的数值次序。为了要看一看内存中当前的完整程序的列表，看看语句所排列的正确次序，打入

LIST

APPLESOFT 将回答：

10 PRINT 2+3

20 PRINT 2-3

有时可能需要删除程序中的某一行，这时，打入你所希望删除的行号，紧接着按一下 RETURN 键，便可完成这一操作。

打入：

10

LIST

APPLESOFT 将回答：

20 PRINT 2-3

现在，你已从程序中删除了行号为 10 的行，并且没有办法使它恢复，为了要插入新的行号为 10 的行，打入 10，接着打入你要 APPLESOFT 执行的新语句。

打入下列语句：

10 PRINT 2*3

LIST

APPLESOFT 将回答：

10 PRINT 2*3

20 PRINT 2-3

这里采用的是先删除 10 行然后插入新的行。还有一种更方便的更换 10 行的方法：打入新的 10 行（当然要按 RETURN 键），APPLESOFT 将自动把抛弃旧的 10 行，代之以新的 10 行。

打入如下的语句

10 PRINT 3-3

LIST

APPLESOFT 将回答：

10 PRINT 3-3

20 PRINT 2-3

建议你不要程序的行号进行连续编号，因为在以后，在现有的两行之间插入新的一行，可能是必不可少的，在行与行之间，空出 10 作为增量，基本上足够了。

如果你要删去当前存放在内存中的整个程序，则打入

NEW

如果一个程序运行结束，你要运行一个新的程序，则你首先要按 NEW，这是为了防止

新、旧程序混和在一起。

打入：

NEW

APPLESOFT 将用提示符回答；

]

现在打入

LIST

APPLESOFT 将回答

]

表示你的旧程序已不再存贮在内存中了。

§ 3 数 字 格 式

我们暂且离开主题来说明一下 APPLESOFT BASIC 打印的数据格式，数值数据在内部以九位数字以上的精度来存放，每一个数字可有一个指数（以十的幂作为比例因子）

在 APPLESOFT BASIC 中，“实型精确数”（也称为“浮点数”）必须在 -1×10^{38} 至 1×10^{38} 之间。否则，你可能得到一个出错信息。在加、减法时，你可能会产生一个和 1.7×10^{38} 一样大的数，而没有产生错误信息，当一个数的绝对值小于 3×10^{-39} 时，APPLESOFT 将把它作为 0 处理，除了这些限制外，真正整型值的范围必须在 -32767 至 32767 之间。

当一个数字被打印时，下列规则将决定确切的格式：

- (1) 若该数是负的，必须打印负号（-）。
- (2) 若数的绝对值是在 0 至 999999999 之间的一个整数，它将作为整数打印。
- (3) 若数的绝对值大于或等于 .01，小于 999999999.2 则数按照定点表示法打印，不带指数部分。

(4) 如果数不属于(2)或(3)则采用科学计数法。注：（见注）科学计数法用来打印实型精确数，其格式如下：

SX.XXXXXXXE^{TT}

其中，每个 X 是一个 0 至 9 之间的数字。

打头的 S 是数的符号，正数可省略，负数则要个负号（'）。

在小数点之前打印一个非零的数字，接着打印小数点，小数点后面是 8 位尾数，然后打印 E（表示指数），接着是指数的符号（S），接着是指数本身两位数字（TT），打头的零不打印，即小数点前的数不为零，同理，小数后面尾随的零也不打印。

如果在删去所有尾随的零后，只有一个数字要打印，则不打印小数点，指数的符号用（+）表示正，用（-）表示负，指数的两位数字总是要打印的，即在指数域中不删去零。

如上所述的科学计数法表示的数据值为 E 的左边的数乘上 10 的幂次，这个幂次的大小便是 E 的右边的数。

下面的例子中，给出了 APPLESOFT 将用来打印各种数据的输出格式：

注：科学计数法即通常所说的浮点表示法——译者。

数 据	输出格式
+1	1
-1	-1
6523	6523
-23.460	-23.46
45.72E5	4572000
$1 * 10^{20}$	1E+20
$-12.3456789 * 10^{10}$	1.23456789E+11
1000000000	1E+09
9999999999	9999999999

在键盘上打入的一个数或在 APPLESOFT 程序中使用数字常量，可以由最长不超过 38 位的数字组成，然而，只有前 10 位是有意义的，并对第 10 位进行舍入。

例如：如果你打入

```
PRINT 1.23456787654321
```

APPLESOFT 将回答：

```
1.23456788
```

§ 4 彩色作图实例

打入

```
GR
```

屏幕上的正文顶部的 20 行将变为黑色，仅保留底部的 4 行正文，你的 APPLE 目前正处于低分辨“彩色图形”方式。

现在打入

```
COLOR=13
```

APPLESOFT 将仅用提示字符回答：

```
]
```

并闪烁光标，而内部已记下你已选用黄色

现在打入

```
PLCT 20, 20
```

APPLESOFT 的回答是在屏幕中央划出一个小的黄方格，如果这个方格不是黄色，则是你的电视机没有调好，调整色调控制，使它产生清晰的柠檬黄色。

现在打入

```
HLIN 0, 30 AT 20
```

APPLESOFT 将经过屏幕最左边的 3/4，顶端以下的 1/4 处划一条水平线。

现在打入

```
COLOR=6
```

从而改变成一种新的颜色，然后打入

```
VLIN 10, 39 AT 30
```

以后你将学到更多的彩色作图方式，为了要回到文本显示方式。

打入

TEXT

屏幕上所显示的字符是作为正文的 APPLE 彩色信息显示方式。

当打印问题的答案时，为了解释数字的含义，则常常要求把一段正文与答案一起打印出来。

打入：

PRINT "ONE THIRD IS EQUAL TO", 1/3

APPLESOFT 将回答：

ONE THIRD IS EQUAL TO .33333333

§ 5 打 印 格 式

正如前面所解释的 PRINT 语句中的逗号（，）使得在打印接在逗号后的数值之前，留出间隔直到下一个制表域（tabfied）。如果我们用分号（；）代替逗号，则正文值将紧跟在先前的值后打印，可以试试看。

试试下面的例子，

PRINT 1, 2, 3

1 2 3

PRINT 1; 2; 3

123

PRINT -1, 2, -3

-12-13

下面是个程序例子，这个程序是从键盘上读一个值，并用这个值进行计算，打印出结果

10 INPUT R

20 PRINT 3.14159*R*R

RUN

? 10

314.159

其过程是这样的，当 APPLESUFT 遇到 INPUT 语句时，它在屏幕上显示一个问号（？），然后等待你打入一个数，当你打入之后（在上述例子中打入的是 10），把打入的值赋给 INPUT 后的变量，（此处赋 R 为 10）然后继续执行程序中的下一个语句，它在上例中是 20 行，当 PRINT 语句后面的式子被求值时，每当式子中出现变量 R 便代之以 10，因此式子便成为 $3.14159 \times 10 \times 10$ 即 314.159

也许你已猜到，上述程序是计算半径为 R 的圆面积。

如果你要计算不同的圆面积，我们可以对每一个圆，重复运行此程序，有一种更简单的方法是：在程序中简单地增加一行。即：

30 GOTO 10

RUN

? 10

```

314.159
? 3
28.27431
? 4.7
69.3977231
?
BREAK IN 10
]

```

在你的程序的末尾放置 GOTO 语句，你便可以使它在打印了答案之后回到 10 行，这可能会无限地继续上去，但我们决定在计算了三个圆面积之后停止。其做法是：打入一个 CTRL C（在按住 CTRL 键的同时再按 C），再按 RETURN 键，这可以“中断”程序执行，在执行当前语句后用 CTRL—C 可终止任何程序继续执行，不妨请试一试。

§ 6 变 量 名

我们把刚才运行的程序中的字母 R 叫做“变量”，在计算机中，它也是一个存储单元。用名 R 来标识，变量名必须以字母开头，随后跟以字母数字字符，字母数字字符是 A 到 Z 的字母或 0 到 9 的数字。

变量名最多可用 238 字符长，但 APPLESOFT 仅取用前两个字符来区别于其它变量名，因此名 GOOD 4 NOUGHT 和 GOLDRUSH 被看作同一个变量。

在变量名中，头两个字符以后的字母数字字符都被忽略除非它们包含了“保留字”。在 APPLESOFT BASIC 命令中某些字是由于有特殊的作用而被“保留”下来的，你不能将这些字作为变量名使用，或作为任何变量名的一部分。例如 FEND 将是非法的，因为 END 是保留字。APPLESOFT BASIC 中的保留字在附录 F 中列出并加以讨论。

以 \$ 或 % 结尾的变量名，具有特殊的含义。这将在本章的后面 REAL, INTEGR 和 STRING VARIABLES（译注：实型、整型和串型变量）那一节里讨论。下面是一些合法的和非法的变量名的一些例子：

合 法

TP
PSTG\$
COUNT
N1%

非 法

TO（变量名不能是保留字）
RGOTO（变量名不能含有保留字）

除了可用 INPUT 语句给变量赋值以外，你还可使用 LET 赋值语句给变量赋值。

可以试试下面例子：

```

A=5
PRINT A, A * 2
5          10
LET Z=7
PRINT Z, Z-A
7          2

```

从这些例子中可以看出，赋值语句中的 LET 是选用的。

BASIC 将“记住”用此语句赋给变量的值，这个“记忆”的过程就是使用 APPLE 的存储器空间来存储数据。

当有如下四种情况的一种发生时，变量的值将丢失，而释放用存储这些值的存储器空间。

- (1) 一个新行被打入程序或一个旧行被删除。
- (2) 发出了 CLEAR (清除) 命令。
- (3) 发出了 RUN (运行) 命令。
- (4) 打入了 NEW 命令。

这里还有另一个重要的事实，除非给数值变量赋以其它值，否则所有的数值变量将自动地被赋为零，可试一下这个例子：

```
PRINT Q, Q+2, Q*2
```

```
0                2                0
```

另一个语句是 REM 语句，REM 是英语 remark 的简写，这个语句用来把注解或注释项插入到程序中，当 BASIC 遇到 REM 语句时，该行的剩余部分将被掠过，它主要是用来帮助程序员阅读程序用的，它对于解决特定问题的程序操作是不起作用的。

§ 7 IF...THEN 语句

我们写一个程序，来检验打入的数是不是零。到目前为止，我们所讨论的语句还无法做到，我们需要一种这样的语句，这个语句能够有条件地转移到其它语句上去。IF...THEN 语句恰可能做到这点：

打入 NEW，然后打入程序：

```
10 INPUT B
20 IF B=0 THEN GOTO 50
30 PRINT "NON-ZERO"
40 GOTO 10
50 PRINT "ZERO"
60 GOTO 10
```

当运行这个程序时，它将打印一个问号，等待你打入 B 的值。打入你想要打入的值，然后计算机将执行到 IF 语句处，在语句的 IF 和 THEN 之间有一个“断言”，这个“断言”由两个表达式组成，中间由一符号分开，这个符号可以是下列符号之一：

符 号	含 义
=	等于
>	大于
<	小于
<>或><	不等于
<=	小于等于
>=	大于等于

IF 语句要么是真，要么是假，这取决于断言是真还是假。例如：在我们现在的程序中，如果打入的 B 的值为 0，则 $B=0$ 为真，因此 IF 语句为真，程序继续执行该语句的 THEN 部分：GOTO 50。遵照此命令，计算机将跳至 50 行，打印出 ZERO，然后，60 行的 GOTO 语句将使计算机转回到 10 行。

假定打入 B 值为 1，因为现在 $B=0$ 为假，所以 IF 语句为假，程序继续执行下一个行号的语句，而忽略语句的 THEN 部分和该行的其它部分，因此打印 NON—ZERO，40 行的 GOTO 语句将使计算机转回 10 行，现在试一下下列比较两个数大小的程序（记住，要首先打入 NEW，以删除你的上个程序）：

```
10 INPUT A, B
20 IF A<=B THEN GOTO 50
30 PRINT "A IS LARGER"
40 GOTO 10
50 IF A<B THEN GOTO 80
60 PRINT "THEY ARE THE SAME"
70 GOTO 10
80 PRINT "B IS LARGER"
90 GOTO 10
```

当运行这个程序时，10 行将打印一个问号，等待你打入两个数，且用逗号隔开，在 20 行，如果 A 大于 B，则 $A<=B$ 为假，THEN GOTO 50 被忽略，程序转向下一行号后面的语句执行，打印 A IS LARGER。最后，40 行把计算机转回 10 行重新开始。

在 20 行如果 A 和 B 的值相等，则 $A<=B$ 为真，执行 THEN GOTO 50，计算机转向 50 行，在 50 行，因为 A 和 B 的值相同，故 $A<B$ 为假。因此 THEN GOTO 80 被忽略，计算机继续执行下一行，那儿将告诉计算机打印 THEY ARE SAME（它们相等），最后 70 行把计算机转向开始点。

在 20 行，如果 A 小于 B， $A<=B$ 为真，程序继续执行 THEN GOTO 50，在 50 行， $A<B$ 为真，则执行 THEN GOTO 80。最后，打印 B IS LARGER，计算机再返回到记始点。

试着把上次两个程序运行多次。然后用 IF...THEN 语句编写你自己的程序。实际上，要弄懂 APPLESOFT BASIC 怎样工作的最快最容易的办法，便是试写你自己的程序，要记住，要停止这些程序，须按 CTRL C 和 RETURN 键。

§ 8 其它着色的例子

让我们来试一个作图程序，REM 语句是为了表达清楚而引用的。冒号（:）是用来在一个编号的程序行中分隔多条指令的。在打入下面的程序之后，LIST（列出）它，以确认你准确无误地打入了该程序，然后 RUN（运行）它：

```
100 GR:REM SET COLOR GRAPHICS MODE
110 HOME:REM CLEAR TEXT AREA
120 X=0:Y=5:REM SET STARTING POSITION
```

```

130 XV=2:REM SET X VELOCITY
140 YV=1:REM SET Y VELOCITY
150 REM CALCULATE NEW POSITION
160 NX=X+XV:NY=Y+YV
170 REM IF BALL EXCEEDS SCREEN EDGE, THEN BOUNCE
180 IF NX>39 THEN NX=39:XV=-XV
190 IF NX<0 THEN NX=0:XV=-XV
200 IF NY>39 THEN NY=39:YV=-YV
210 IF NY<0 THEN NY=0:YV=-YV
220 REM PLOT NEW POSITION IN YELLOW
230 COLOR=13:PLOT NX, NY
240 REM ERASE OLD POSITION
250 COLOR=0:PLOT X, Y
260 REM SAVE CURRENT POSITION
270 X=NX:Y=NY
280 REM STOP AFTER 250 MOVES
290 I=I+1:IF I<250 THEN GOTO 160
300 PRINT "TO RETURN TO YOUR PROGRAM, TYPE 'TEXT'"

```

GR 命令告诉 APPLE 将开关置成彩色作图方式，同时还将 40×40 的绘图区域清成黑色，将文本输出到屏幕底部所留出的 4 行（每行可显示 40 个字符）文本显示窗口内，将一个要绘点的颜色置成黑色。

HOME 用来清除文本显示区域，把光标置于当前定义的正文窗口的左上角，在彩色作图方式中，这里将从正文的 20 行开始，因为 0 至 19 行是用于彩色作图划定区域的。

在 230 行和 250 行的 COLOR= 命令置下种要画的颜色，这颜色是由跟在 COLOR= 后面的表达式的值所决定的。

230 行的 PLOT NX, NY 在表达式 NX 和 NY 所给定的新位置上，用最近的 COLOR= 命令所给它的黄颜色，画出一个方块。记住，NX 和 NY 都必须是 0 至 39 之间的数，否则方格将超出屏幕，从而产生出错信息。

类似地，250 行的 PLOT X, Y 在表达式 X 和 Y 所给定的位置上画出一个方块，但 X, Y 仅是“旧的”NX 和 NY 的光标，它是在先前黄色方块之后保留的，因此，PLOT X, Y 在“旧的”黄色方块上重画一个颜色，由 COLOR=0 来定义的小方块。COLOR=0 定义的颜色是黑色，它与屏幕的原色相同，所以看上去是删去了旧的黄色。

注意：要从彩色作图返回到文本显示方式，打入

TEXT

然后按 RETURN 键。

正如所指出的，打入 TEXT 是使你退出作图方式，你不必管屏幕上的奇怪符号——，它们是把你的图形显示转换为正文符号的结果。如果你搞不懂 290 行，不必着急，这将在后面的几页中加以解释。

正如你已看见的，APPLE II 除了使用数以外，还能做更多的事，在你学习了更多的

APPLESOFT BASIC 以后, 我们将再次回到彩色作图上。

§ 9 FOR...NEXT 语句

计算机的优点之一是它们具有反复执行任务的能力, 假定我们需要一个从 1 到 10 整数的平方根表, APPLESOFT BASIC 的平方根函数是 SQR,

其形式为

SQR (X)

其中 X 是你要求的平方根的数, 我们可以写出这个程序如下:

```
10 PRINT 1, SQR ( 1 )
20 PRINT 2, SQR ( 2 )
30 PRINT 3, SQR ( 3 )
40 PRINT 4, SQR ( 4 )
50 PRINT 5, SQR ( 5 )
60 PRINT 6, SQR ( 6 )
70 PRINT 7, SQR ( 7 )
80 PRINT 8, SQR ( 8 )
90 PRINT 9, SQR ( 9 )
100 PRINT 10, SQR ( 10 )
```

这个程序将做这一项工作, 然而, 它的效率实在太差, 使用刚才介绍的语句可以大大地改进程序, 如下所示:

```
10 N=1
20 PRINT N, SQR ( N )
30 N=N+1
40 IF N<=10 THEN GOTO 20
```

当运行此程序时, 它的输出看上去将和上面的 10 个语句的程序完全一样, 让我们来看看现在它是怎样工作的。

在 10 行, 有一个 LET 语句, 它给变量 N 置 1, 在 20 行告诉计算机用 N 的当前值打印 N 和 N 的平方根, 因此, 20 行成为

```
20 PRINT 1, SQR ( 1 )
```

打印出这个计算结果。

在 30 行, 有一初看起来好象很奇特的 LET 语句, 在数学上, $N=N+1$ 是不成立的, 然而, 在 LET 语句中, “=” 符号不表示等于, 这是要记住的一件很重要的事情, 在这种情况下, “=” 意味着 “替换”, 这个语句简单地将 N 的当前值加上 1, 因此, 第一次经过 30 行之后, N 的值为 2。

在 40 行因为 N 现在等于 2, 断言 $N \leq 10$ 为真, 所以 THEN 部分将计算机转回至 20 行, N 的当前值为 2。

在 20 行至 40 行之间重复执行, 每经一次, N 就加 1, 就可得到全部结果, 当最后 N 在 20 行的值等于 10 时, 下一行将使它增至 11, 这个结果使得 40 行的断言为假, 因此 THEN 部分