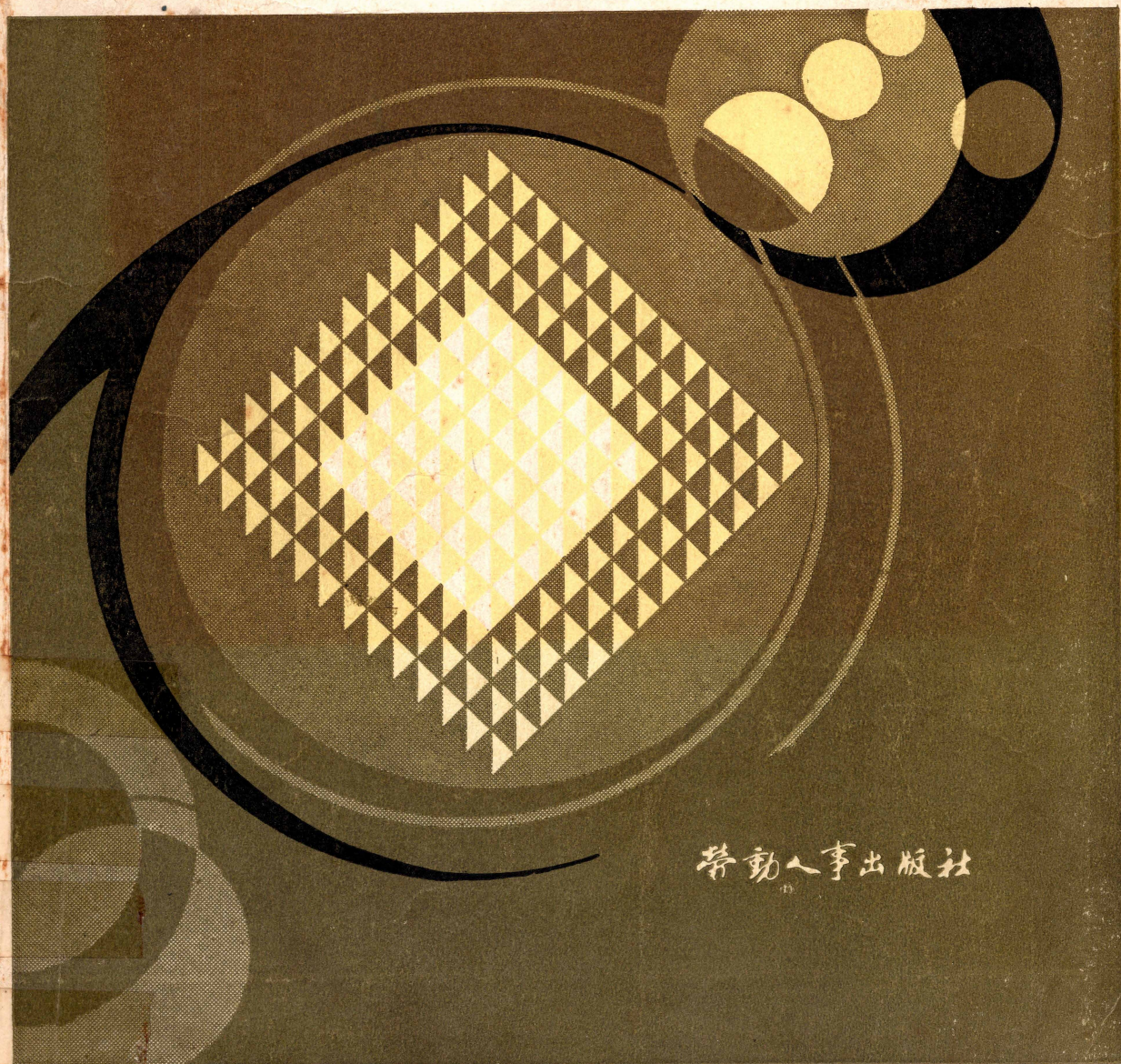


技工学校机械类通用教材

微电脑基础与应用



劳动人事出版社

封面设计：张美芝

书号：7238 · 0157

定价：1.65元

00109

微 电 脑 基 础 与 应 用

劳动人事部培训就业局 编

劳动人事出版社

前 言

本书是我局为适应普及微电脑技术的需要，委托江门市技工学校编写的技工学校试用教材。本书也可供在职职工、微电脑操作人员自学和参考。

由于编写时间比较紧促，缺点和错误在所难免，希望使用教材的同志提出宝贵意见，以便在重印时修订。

劳动人事部培训就业局

一九八五年十二月

微 电 脑 基 础 与 应 用

劳动人事部培训就业局 编

劳动人事出版社出版

(北京市和平里中街12号)

新华书店北京发行所发行

天津新华印刷一厂印刷

787×1092 16开本 10.5印张 259千字

1986年9月北京第1版 1986年9月天津第1次印刷

印数：1—70,000册

书号：7238·0157

定价：1.65元

目 录

第一讲 计算机的发展与作用	(1)
习 题	(2)
第二讲 微电脑的基本构造	(3)
1. 微处理器或中央处理器	(3)
2. 存贮器	(3)
3. 输入/输出部件	(4)
4. 连接总线	(4)
习 题	(4)
第三讲 微电脑的内部工作过程	(5)
1. 微电脑工作流程框图	(5)
2. 工作过程描述	(5)
习 题	(6)
第四讲 电脑中数的表示方法	(7)
1. 二进制	(7)
2. 十六进制	(7)
3. 十进制数转换为二进制数	(8)
4. 二进制数转换为十进制数	(10)
5. 二进制数转换为十六进制数	(11)
6. 十六进制数转换为二进制数	(11)
7. 十六进制数转换为十进制数	(12)
8. 十进制数转换为十六进制数	(13)
习 题	(15)
第五讲 电脑语言	(16)
1. 机器语言	(16)
2. 汇编语言	(16)
3. 高级语言	(17)
习 题	(18)
第六讲 BASIC 及其编程知识	(19)
1. BASIC 概要	(19)
2. BASIC 程序的构成和基本规则	(22)
3. BASIC 的特点	(24)
习 题	(25)
第七讲 程序设计知识 (一)	(26)

1. 程序框图(流程图).....	(26)
2. 程序设计方法.....	(27)
3. 程序中信息的输出.....	(27)
4. 程序中数据的提供.....	(31)
第八讲 程序设计知识 (二)	(37)
1. 基本数学函数.....	(37)
2. 导出函数.....	(41)
3. 自定义函数 (DEF 语句)	(43)
第九讲 程序设计 (一) —— 转移及循环	(45)
1. 转移.....	(45)
2. 循环.....	(51)
习 题.....	(59)
第十讲 程序设计 (二) —— 主程序与子程序	(60)
1. 无条件转子程序与返回主程序.....	(60)
2. 选择转子程序与返回主程序.....	(64)
3. 出错转移和返回重做.....	(64)
4. 程序设计举例.....	(65)
习 题.....	(67)
第十一讲 程序设计 (三) —— 数组	(69)
1. 数组的意义.....	(69)
2. 数组的建立.....	(70)
3. 数组的运用.....	(71)
习 题.....	(75)
第十二讲 程序设计 (四) —— 字符处理和光标 (显示) 定位	(76)
1. 字符处理.....	(76)
2. 光标 (显示) 定位.....	(81)
习 题.....	(84)
第十三讲 华宇—200 (LASER—200、LASER—310) 的使用与保养	(85)
1. 华宇—200 (LASER—200) 微电脑的简介	(85)
2. 微电脑系统的启动与保养.....	(85)
3. 打键操作注意事项.....	(87)
习 题.....	(87)
第十四讲 华宇—200 (LASER—200、LASER—310) 键盘操作 (一)	(88)
1. 普通键.....	(88)
2. 号符键的作用.....	(88)
3. 比较符 (关系符) 号键的作用.....	(89)
4. 函数符键的作用.....	(89)
习 题.....	(90)
第十五讲 华宇—200 (LASER—200、LASER—310) 键盘操作 (二)	(92)

1. 直接按(打)键	(92)
2. 特殊键及其联用	(92)
习 题	(95)
第十六讲 磁盘系统的操作	(96)
1. 磁盘系统的组成	(96)
2. 磁盘机的连接方法	(96)
3. 磁盘机及磁盘的保养	(97)
4. 资料或程序存入、调出磁盘	(97)
第十七讲 华宇—200微电脑系统使用(一)	(99)
1. 打印机的控制	(99)
2. 几个语句的用法	(100)
第十八讲 华宇—200微电脑系统使用(二)	(103)
1. 荧幕字体的两种选择	(103)
2. 图象	(103)
3. 颜色命令(COLOR I,J)	(105)
4. 发声命令(SOUND……)	(106)
第十九讲 程序和资料的外贮存与调用	(108)
1. 将资料或程序存入外贮存器	(108)
2. 从外贮存器中查找或调用程序	(108)
3. 核对文件	(109)
4. 把磁带上的程序载入内存并运行的命令	(109)
5. 向磁带文件传送资料的命令	(109)
6. 从磁带文件读取数据并赋予指定的变量命令	(110)
第二十讲 华宇—3000(LASER—3000)微电脑系统操作	(111)
1. 华宇—3000的功能键操作	(111)
2. 打印机的使用	(112)
第二十一讲 LASER—3000及APPLE II微电脑系统对图形的处理	(114)
1. 低分辨图形	(114)
2. 高分辨图形	(116)
第二十二讲 微电脑系统对汉字处理	(121)
1. 英/汉接口板	(121)
2. 汉字数据处理与实数 BASIC 综合程序设计	(127)
3. 汉字的组构	(128)
4. 微电脑汉字功能的运用	(135)
习题参考答案	(138)
第三讲 微电脑的内部工作过程	(138)
第四讲 电脑中数的表示方法	(138)
第六讲 BASIC 及其编程知识	(138)
第九讲 程序设计(一)——转移及循环	(139)

第十讲	程序设计(二)——主程序与子程序	(139)
第十一讲	程序设计(三)——数组	(139)
第十四讲	华宇—200(LASER—200、LASER—310)键盘操作(一)	(141)
第十五讲	华宇—200(LASER—200、LASER—310)键盘操作(二)	(141)

附 录

附录 I	华宇—3000(兼容LASER—3000、APPLE II)实数 BASIC 命令汇集	(142)
附录 II	微电脑 LASER—310 BASIC 摘要	(154)
附录 III	LASER—3000、APPLE SOFT 实数 BASIC 的错误信息(ERROR MESSAGE) 表	(157)
附录 IV	微电脑键盘图	(158)
	华宇—200 (LASER—200) 键盘图	(158)
	LASER—310 键盘图	(158)
	APPLE 键盘图	(159)
	LASER—2001 键盘图	(159)
	华宇—3000 (LASER—3000) 键盘图	(160)
	ZD—065 键盘图	(161)

第一讲 计算机的发展与作用

计算机是人类在同大自然的斗争中创造并逐步发展起来的武器，是现代科学技术发展的产物。

我们的祖先，早在春秋时代就发明了用竹筹计数的“筹算法”，在唐代末期制造出有十三档的算盘。随着生产力的发展，需要做开方、对数、三角函数等计算，世界上又出现了比较先进的计算工具，如计算尺、计数器等。

由于近代科学技术飞跃发展，生产规模越来越大，需要运算处理的问题更加复杂，要求计算和处理问题的速度更快，因而又大大促进了计算科学技术的发展。二十世纪四十年代中，美国就制造出世界上第一台电子计算机ENIAC (Electronic Numerical Ind Computer的简称)。电子计算机一出现就飞速地发展起来。近代的数字电子计算机不但有很强的计算能力，还有贮存信息、处理资料、管理事务、过程控制等一系列功能，因此人们称其为“电脑”。

电脑的发展经历了使用电子管、晶体管、集成电路、大规模集成电路、超大规模集成电路装配的几个阶段。1946年世界上第一台电子数字计算机的出现，开创了电子计算机的第一个年代。约从1958年开始，计算机进入第二代，运算速度由第一代的每秒几千次提高到每秒几百万次。六十年代初进入第三代，运算速度每秒达千万次。七十年代进入第四代，除了运算速度更高以外，还研制出微处理器，从而使微电脑迅猛发展起来。现在世界上已开始研制第五代超级计算机（或叫有人工智能的第五代电脑）。

计算机发展如此之快，是由于它适应了高速发展的社会生产力的需要。例如，计算机能快速地作复杂而又精确的计算，控制各种仪器、机械；使宇宙飞船进入轨道并按计划运行；使导弹准确地击中目标；使实验过程、样品测试分析得以自动进行；使机床能够加工非常复杂的零件。它还能存贮大量资料并作数据处理而代替名医诊断疾病，自动开处方、配方；代替人们管理城市交通；能编辑稿件、排字、制版；能实现火车的行车调度、编组、售票自动化；能选择最佳的参数进行生产和实施最佳的自动控制。企业管理、统计、成本核算、人事档案和财务管理、工资发放等工作都可以使用计算机。此外，计算机对人们的娱乐、教学、自学等活动也可以提供直接的帮助。

电子计算机不仅能代替人完成一般的工作，而且还能完成人们极难胜任的工作。例如，电脑可代替人不能进入的禁区去完成取样、处理等工作；导弹轨道的运算往往需要几十万乃至几百万个数据，且运算公式复杂，用人力是无法在很短时间内完成的，只有计算机能够胜任；象Ⅲ型仪表差压变送器的不锈钢膜盒，要求互换性能好，精度高（0.1ms），用冲压方法不能制造，普通车床也不能生产，只有用“电脑”控制的车床方可胜任；要作出及时、准确的气象预报需要收集各地资料进行数据处理，工作量极大，只有用“电脑”才能做到。1948年美国原子能研究中心有一项计划所需要的数据，需作900万次运算。这一工作若用人工来做，大约要1500名工程师运算一年才能完成。当时用一台计算机进行运算，只用了150

个小时就完成了。有人估计1983年美国现有电脑完成的工作量，需要4000亿人工作一年才能完成。

电脑是二十世纪最重大的科技成果之一。它不仅在高能物理、空间科学技术、量子化学、遗传工程等尖端科学中是必不可少的，而且在信息社会中占有极为重要的位置。它将是当代面临着的信息革命的中枢。

在信息社会中，电脑将在信息采集、交流、贮存、处理等方面起到重要的作用。到那时，文件资料可以存入超微电脑随身携带；人们可以在家里或者在旅途中，用电脑跟世界各地联系，迅速交换信息资料；科学技术人员进行研究设计，可以通过电脑取得所需要的各种数据和最新资料；文学、法律工作者可使用电脑查找资料；商业管理人员可用电脑随时获得市场信息，作出市场预测，经营计划。人们也可以把自己工作得出的实验数据和资料存入电脑，变成信息社会的资源。

这样一来，各行各业、各个科学领域的信息，就成为一种资源存在于社会而为世人共享了。

我国对于发展计算机科学事业是很重视的，也已经取得了一定的成绩。早在1958年就研制出第一代电子计算机。此后大约每六年就往前赶一代。至七十年代末我国能制造大型数据处理机，使电子计算机事业跨入了第四代。巨型电子计算机——“银河”的问世及各种微型电子计算机的生产，标志着我国电子计算机技术已达到新的水平。然而，我国的电子计算机工业同发达国家相比，差距还是相当大的。

为了迎接新的技术革命，加强我国四个现代化的建设，要更加重视发展电子计算机科学技术。一方面要加强研制硬件和软件，另一方面要加强人才的培养和应用技术的推广、普及。

习 题

简述微电脑的用途。

第二讲 微电脑的基本构造

电子计算机以工作模式来分，可分为模拟式、数字式和混合式；以性能（主要是运算速度和主存贮量）来分，有巨（大）型、中型、小型和微型的。下面主要是介绍微型数字计算机——微电脑的基本构造。

微电脑包括硬件和软件两大部分。

硬件是指机器设备部分，分为主机及外围设备。主机主要由微处理器（中央处理器），存贮器，输入、输出部件，连接总线路等组件构成。外围设备一般的有键盘、显示器（专用的监视器或家用的电视机，有单色和彩色两种）、磁带机（盒式磁带录音机）、磁盘机（又叫磁盘驱动器，有软磁盘机和硬磁盘机两种）、打印机、绘图机、各种接口板。

软件是指电脑系统运转的管理及应用开发方面的规约和程序。如指令体系，操作体系，语言体系，监控程序，实用程序等。它可分为系统软件和应用软件。系统软件是电脑系统的重要组成部分，在设计制造时就确定下来了。应用软件是用户为使用电脑去完成某种任务而编制的程序。

下面再进一步介绍微电脑系统的主体——主机的内部结构。尽管电脑的种类繁多，但是主机所包含的部件和结构却是大同小异的。通常主机内有：

1. 微处理器（Micro Processor Unit——MPU UP）或中央处理器（Central Processing Unit——CPU）

CPU	机 种
6502	APPLE, LASER-3000, ZD-065
Z-80	TRS-80
6800	DG-6800, M-6800, MCP-68
8080	MCS-80, SDK-80

这是主机的核心，是微电脑中最最重要的控制和处理部分，它基本上决定了微电脑的性能。这个好象“心脏”一样的部件内部主要有运算器、寄存器、控制器等。

看一部微电脑的性能如何，首先是看CPU是什么样的。

目前比较常见的8位微电脑所用的CPU如上表。

2. 存贮器（Memory）

用来存放资料的。由一系列编定号码（地址）的单元组成。每一个单元可以“写”一个“字”——存入资料，又可以从某个单元“读”一个“字”——取出资料。按照其存取资料的性能可分为几种。

（1）主存贮器 配置在主机内的存贮器。

①只“读”存贮器（Read Only Memory—ROM）预先在芯片上存入资料，装进主机后，只能从其中读出资料（取资料，却是取之不尽的）。这种存贮器中的资料（主要是管理系统运转使之正常工作所必须的），在装入机内以前就已“写”（存）进去的。操作者在开机后就可使用，关机后资料仍保留着。

②随机存取存贮器（Random Access Memory—RAM），也叫随意“读”、“写”

存贮器。

操作者在开机以后，可根据需要随意在其上存（“写”）入或取（“读”）出资料。存入时是以“后入为主”的，即后面存入的内容自动冲掉原有的而取代之。在只“取”的时候，是取之不尽的，但是关机后，原来存入的内容则会消失。

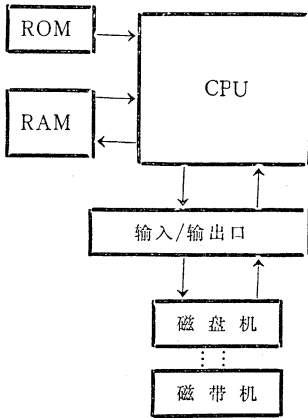


图 2.1

(2) 辅助存贮器（又称为外存贮器）在主机之外另加配备的存贮设备，用于弥补主存贮器的容量不足。它的容量大，价格便宜，存取数据的速度不如主存贮器快，如磁带机及磁带，磁盘机及磁盘等。它们属于外围设备，经过一定的连接后可以跟主机连通。既可将主存贮器内的资料转到磁带或磁盘上，通过更换磁带或磁盘，能够存贮大量的资料。又可以将录存在磁带或磁盘上的资料调入主存贮器内。

上述的存贮器与CPU的连接如图2.1所示。

示。

3. 输入/输出部件 (Input/Output—I/O)

输入/输出部件包括串行输入/输出电路和并行输入/输出电路。通过这些部件可以把各种输入/输出设备，如键盘，显示器，打印机，磁盘机等与主机连接起来，成为一个有机的整体。

4. 连接总线 (Bus)

用来把主机内的各个部件连接成有机的整体。包括地址、数据和控制三类总线。

地址总线：传送关于存贮器地址信息。

数据总线：传送资料（数据）等信息。

控制总线：传送操纵系的信息。

习 题

1. 电脑软件是指什么？包括哪些内容？
2. 微电脑的主机有哪些设备？

第三讲 微电脑的內部工作过程

现以数字运算为例，来看微电脑的工作过程。

1. 微电脑工作流程框图

微电脑工作流程的框图见图3·1。

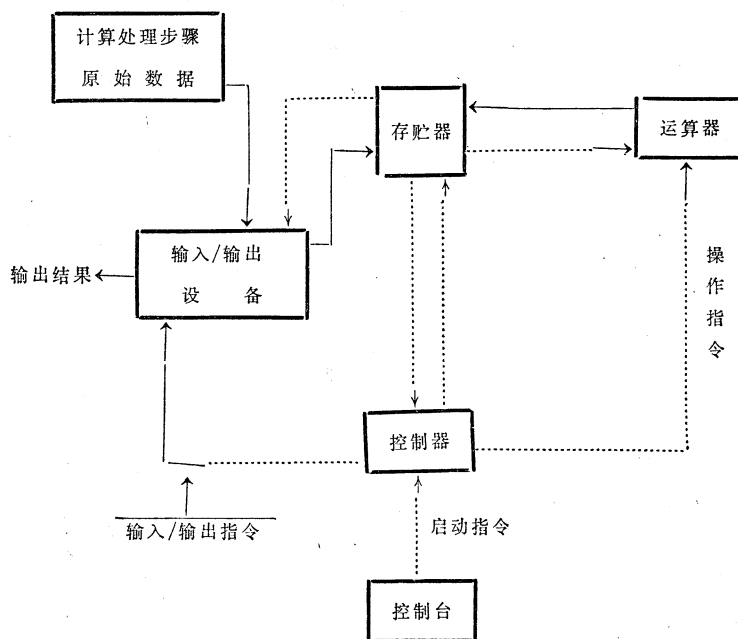


图 3·1

2. 工作过程描述

以数值运算为例。设方程 $X = 6Y^3 + 2Y + 4$ 中，当 $Y = 3$ 时，求 X 的值。

(1) 由键盘将事先编好的计算程序和原始数据 (6、2、4、 $Y = 3$) 输入到“电脑”存储器中存放起来。

假设把 6 存入存贮单元 A 中，2 存入存贮单元 B 中，4 存入存贮单元 C 中， $Y = 3$ 存入存贮单元 D 中。

计算程序的一系列指令从标号 10 开始存入。

(2) 启动计算机，在控制器的控制下，计算机按计算程序自动进行如下操作。

- ① 从存储器中取出 Y 的数值 3 送到运算器，进行乘法运算，得出乘积数；
- ② 乘积数存入存储器，以备调用；
- ③ 从存储器取出别的数作其他运算，最后得出结果。

(3) 把存储器中的最后结果送出，送到监视器荧光屏上显示或在打印机上印出。到此，解题过程结束。

其过程用表3·1表示：

表3·1

指令所在 地址单元码	操作码	地址码	执行内容
10	取数	D	从单元D中把Y值3取出，送到运算器内。
11	乘法	D	从单元D中把Y值取出，送运算器，并与原已在运算器中的Y值相乘得 $Y^2 = 9$ ，仍保留在运算器内。
12	乘法	D	从单元D中把Y值取出，送运算器并与原已在运算器内的 Y^2 值相乘得 $Y^3 = 27$ ，仍保留在运算器内。
13	乘法	A	从单元A中，把数6取出，送运算器和原已在运算器内的 Y^3 值相乘，得 $6Y^3 = 162$ 。
14	存数	E	将中间结果 $6Y^3 = 162$ 存入存贮单元E中，把运算器腾出空来以便进行下一步的运算。
15	取数	D	从单元D中把Y值3取出送运算器内。
16	乘法	B	从单元B中把数2取出与运算器内的Y值相乘，积为6，仍保留在运算器内。
17	加法	E	从单元E中把162取出，送运算器，并与运算器内的6相加，即 $6Y^3 + 2Y = 168$ ，将结果168保留在运算器内。
18	加法	C	从单元C中把数4取出送运算器，并与运算器内的168相加，其结果： $6Y^3 + 2Y + 4 = 172$ ，仍保留在运算器内。
19	打印显示		把运算内的运算结果 $X = 172$ ，送往打印机或显示器进行输出。

实际上这些过程人们是看不见的。但是这些过程又是人们通过程序去指挥电脑进行的。譬如，上述的过程用BASIC语言（一种编程语言）编出的一种程序如下（具体语句意义后面介绍）：

```

10 READ A, B, C
20 DATA 6, 2, 4
30 LET Y = 3
40 LET X = A * Y ^ 3 + B * Y + C
50 PRINT "X=" ; X
60 END

```

习 题

1. 电脑作数字运算时大致可分几步？
2. 设 $X = 4$ ， $Y = 2X^2 + 1$ ，编一程序求Y的值。（练习）

第四讲 电脑中数的表示方法

0、1、2、……9 这些人们日常通用的十进制数字，电脑内部的机器是不认识的，它也不能直接进行 $3 + 4 = 7$ 这样的运算。不过，电脑内部的电路对于电的接通与断开的反应是非常敏感的。所以，在设计制造电脑时，就利用这个特点，把电的导通（或高电位）规定与二进制数字“1”对应，而电的断开（或低电位）则与二进制数字“0”对应。这样就可以将电脑内部的电性状态跟二进制数学直接联系起来。再应用某些信息交换代码，又能进一步将人们日常中使用的数字（如0、1、2、……9），符号（如+、-、×、÷、?、<、=、……），文字（如A、B、C……）等与电脑内部的电性状态建立起对应的关系，从而实现人与电脑的“对话”（联络）。所以，对二进制数学应该有所了解。此外，在使用微电脑时，还往往要碰到十六进制数。下面介绍二进制、十六进制以及它们与十进制之间的转换。

1. 二进制

二进制是用1和0两个数字来表示数值，逢二进一位。日常生活中，如筷子、鞋、手套等等，就是采用二进制的。其运算以加法为例：

$$0 + 0 = 0$$

$$1 + 0 = 1$$

$$1 + 1 = 10$$

用它来表示数据时，可以用若干个数字的组合来表示，每一个数字称为一位，如1001为四位数，01101001为八位数。最右边的数字为最低位，由右至左，位次逐渐增高，最左边的数字称为最高位。每一个数位所代表的十进制数值可用 2^n （ n 为整数）乘以该位上的数字来表示，每向左移一位，指数加1。（个位上 n 为0，在个位左边 n 为正数，个位右边 n 为负数。）

如

1	1	1	1	1	1	1	1	1	1
2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0	2^{-1}	2^{-2}
128	64	32	16	8	4	2	1	$\frac{1}{2}$	$\frac{1}{4}$

2. 十六进制

十六进制是用0、1、2、3、4、5、6、7、8、9、A、B、C、D、E、F十六个数字来表示数值，逢十六进一位。通常在十六进制数的前面冠以H，以与十进制数相区别。如加法：

$$HF + 1 = H10$$

$$HF0 + 10 = H100$$

$$HF00 + 100 = H1000$$

它是用十六个数字的选择组合来表示数值的，每个数有一至若干个位，有高、低位之

分。右边为低位，左边为高位，每一个数位所代表的十进制数值可用 16^n 乘以该位数字来表示。在个位上指数 n 为0，每向左移一位指数加1，反之减1。

如 7F3D
 D在第0位，为 $16^0 \times 13$ ；
 3在第1位，为 $16^1 \times 3$ ；
 F在第2位，为 $16^2 \times 15$ ；
 7在第3位，为 $16^3 \times 7$ 。

十进制(D)、二进制(B)、十六进制(H)基本转换关系见表4.1。

表4.1 十进制(D)、二进制(B)、十六进制(H)基本转换表

D	B	H
0	0 0 0 0	0
1	0 0 0 1	1
2	0 0 1 0	2
3	0 0 1 1	3
4	0 1 0 0	4
5	0 1 0 1	5
6	0 1 1 0	6
7	0 1 1 1	7
8	1 0 0 0	8
9	1 0 0 1	9
10	1 0 1 0	A
11	1 0 1 1	B
12	1 1 0 0	C
13	1 1 0 1	D
14	1 1 1 0	E
15	1 1 1 1	F
16	1 0 0 0 0	10

3. 十进制数转换为二进制数

对十进制整数，依次用2去除。第一次除得的余数就是二进制数的第0位（最低位），再用2去除第一次除所得的商，得出的余数就是二进制数的第1位。如此不断进行，直至所得的商小于2为止，而这最后一个商则是二进制数的最高位。

例1：将十进制数24转换成二进制数。

解：

$$\begin{array}{r}
 2 \overline{) 24} \quad \text{余数} \\
 \underline{2 12} \quad \dots\dots 0 \\
 2 \overline{) 6} \quad \dots\dots 0 \\
 \underline{2 3} \quad \dots\dots 0 \\
 2 \overline{) 3} \quad \dots\dots 0 \\
 \underline{1} \quad \dots\dots 1 \\
 \downarrow \\
 \text{商小于 2}
 \end{array}$$

$$\therefore (24)_{10} \Rightarrow (11000)_2$$

例 2：将十进制数 167 转换成二进制数。

解：

2	167	余数
2	83 1
2	41 1
2	20 1
2	10 0
2	5 0
2	2 1
	1 0
	↓	
	商小于 2	

$$\therefore (167)_{10} \Rightarrow (10100111)_2$$

对十进制小数部分，依次用 2 去乘。第一次用 2 乘所得乘积中的整数（包括 0），就是二进小数部分的最高位（小数点后第一位）。若乘积中小数部分不为 0，再用 2 去乘该小数部分，所得乘积中的整数，则是二进制小数的下一位。直至所得乘积中小数部分等于 0 为止（或足够了所要求的进制小数的位数为止）。

例： $(0.375)_{10}$ 。

	整数	小数
×) 2	0.750 0
×) 2	1.500 1
×) 2	1.000 1
整数	0

$$\therefore (0.375)_{10} \Rightarrow (.011)_2$$

对既有整数又有小数的十进制数，分别用上述方法将整数部分及小数部分化为二进制数，然后合起来便是带小数的二进制数。

例： $(84.285)_{10} \Rightarrow (\quad)_2$ 。

$$84 \Rightarrow (1010100)_2$$

$$.285 \Rightarrow (.010010)_2 \quad \text{只取六位}$$

$$\therefore (84.285)_{10} \Rightarrow (1010100.010010)_2$$

$$\begin{array}{r}
2 \overline{) 84} \\
\underline{2 42} \dots 0 \\
2 \overline{) 21} \dots 0 \\
\underline{2 10} \dots 1 \\
2 \overline{) 5} \dots 0 \\
\underline{2 2} \dots 1 \\
\underline{1} 0 \\
\text{小于 2 的商}
\end{array}$$

整数	小数
	.285
	2
0	.570
	2
1	.140
	2
0	.280
	2
0	.560
	2
1	.120
	2
0	.240
	⋮
	⋮

故整数部分为1010100，小数部分为.010010合起来是1010100.010010。

4. 二进制数转换成十进制数

对二进制整数部分，用“倍位相加法”，即从二进制数的最低位（最右边的位）起，各个位的基数依次为 2^0 、 2^1 、 2^2 、 2^3 、 2^4 、 2^5 、 2^6 、 2^7 、 2^8 、……。只要将二进制数的各个位上的数字乘以各个对应位的基数，然后再相加起来，便得出十进制整数。

例 3：将 $(111001)_2$ 转换为十进制数。

解： $1 \times 2^0 + 0 \times 2^1 + 0 \times 2^2 + 1 \times 2^3 + 1 \times 2^4 + 1 \times 2^5 = 57$

∴ $(111001)_2 \Rightarrow (57)_{10}$

例 4：将 $(10110110)_2$ 转换为十进制数。

解： $0 \times 2^0 + 1 \times 2^1 + 1 \times 2^2 + 0 \times 2^3 + 1 \times 2^4 + 1 \times 2^5 + 0 \times 2^6 + 1 \times 2^7 = 182$

∴ $(10110110)_2 \Rightarrow (182)_{10}$

将二进制小数部分转换为十进制数，方法同上，还是“倍位相加法”只是从小数点后起，各个位上的基数依次为 2^{-1} 、 2^{-2} 、 2^{-3} ……。

例： $(.011)_2$ 。

$$\begin{aligned}
& 0 \times 2^{-1} + 1 \times 2^{-2} + 1 \times 2^{-3} \\
& = 0 + 0.25 + 0.125 \\
& = 0.375
\end{aligned}$$

$$\therefore (.011)_2 \Rightarrow (0.375)_{10}$$

也可以把上述方法归结为另一种方法，如下所述：

①把二进制小数部分的小数点去掉，看作“整数”。

②把该“整数”转换为十进制数。

③再将该十进制数除以 2^n （或乘以 2^{-n} ），其中 n 是原二进制小数部分所含的位数。所得出的数就是该二进制小数转换成的十进制数。

如上面的例子 $(.011)_2$ 改用另一种做法：

$$\textcircled{1} (.011)_2 \rightarrow (011) \text{ 含有 } 3 \text{ 位}$$

$$\textcircled{2} (011)_2 \Rightarrow (3)_{10}$$

$$\textcircled{3} 3 \div 2^3 = 3 \times 2^{-3} = 0.375$$

$$\therefore (.011)_2 \Rightarrow (.375)_{10}$$

5. 二进制数转换为十六进制数

根据 $2^4 = 16$ 及 $1\ 1\ 1\ 1 \Rightarrow F$ ，可见用一个十六进制数字，就足以表示四个位的二进制数。故把二进制数转换为十六进制数的做法可归纳为：将二进制数的整数部分，由右至左每四个位划成一组，小数部分则由左至右每四个位划成一组。再分别将各个组转换为十六进制数字，然后依次串起来便是所转换成的十六进制数。

例5: $(01100111)_2$ 。

$$\begin{array}{cc} \text{解:} & 0110 \quad 0111 \\ & \downarrow \quad \downarrow \\ & 6 \quad 7 \end{array}$$

$$(0111 \Rightarrow 7, 0110 \Rightarrow 6)$$

$$\therefore (01100111)_2 \Rightarrow (67)_{16}$$

例6: $(0.010110011)_2$ 。

$$\begin{array}{ccc} \text{解:} & 0.0101, & 1001, & 1 \\ & \downarrow & \downarrow & \downarrow \\ & 5 & 9 & 8 \end{array}$$

$$(0101 \Rightarrow 5, 1001 \Rightarrow 9, 1 \text{ 是四个位中的最高位可看作 } 1000)$$

$$\therefore (0.010110011)_2 \Rightarrow (0.598)_{16}$$

例7: $(101011.010011)_2$ 。

$$\begin{array}{cccc} \text{解:} & 10, & 1011.0100, & 11 \\ & \downarrow & \downarrow & \downarrow \\ & 2 & B & 4 \quad C \end{array}$$

$$\therefore (101011.010011)_2 \Rightarrow (2B.4C)_{16}$$

6. 十六进制数转换为二进制数

把十六进制数的每一个数字先分别换成二进制数字，（若有小数部分，则小数点的位置不动）然后再依次串起来，便是所换成的二进制数。

例8: 将 $(5B)_{16}$ 转换为二进制数。

$$\begin{array}{cc} \text{解:} & 5 & B \\ & \downarrow & \downarrow \end{array}$$

$$0101 \quad 1011$$

$$\therefore (5B)_{16} \Rightarrow (01011011)_2$$

例9: $(FD.7EC)_{16}$ 。

解:

$$\begin{array}{ccccc} F & D & 7 & E & C \\ \Downarrow & \Downarrow & \Downarrow & \Downarrow & \Downarrow \\ 1111 & 1101 & 0111 & 1110 & 1100 \end{array}$$

$$\therefore (FD.7EC)_{16} \Rightarrow (11111101.011111101100)_2$$

7. 十六进制数转换为十进制数

整数部分, 用“倍位相加法”。即把十六进制数的各个数字(凡遇A、B、C、D、E、F则转换为10、11、12、13、14、15)分别乘以各个数字所对应的位上的基数(由最低位起, 依次为 16^0 、 16^1 、 16^2 、 16^3 ……), 然后再加起来, 便是所换成的十进制数。

例10: 将 $(4C)_{16}$ 转换为十进制数。

解:

$$\begin{aligned} (4)_{16} &\Rightarrow 16 \times 4 = 64 \\ (C)_{16} &\Rightarrow 12, \quad 16^0 \times 12 = 12 \\ &64 + 12 = 76 \\ \therefore (4C)_{16} &\Rightarrow (76)_{10} \end{aligned}$$

例11: $(E6)_{16}$ 。

解:

$$\begin{aligned} (E)_{16} &\Rightarrow 14, \\ \therefore 6 \times 16^0 + 14 \times 16 &= 230 \\ \therefore (E6)_{16} &\Rightarrow (230)_{10} \end{aligned}$$

小数部分, 还是用“倍位相加法”。注意, 从小数点后起, 各个位上的基数是 16^{-1} 、 16^{-2} 、 16^{-3} ……。把十六进制小数部分的各个数字(遇A、B、C、D、E、F换为10、11、12、13、14、15)分别乘以各个数字所对应的位上的基数, 然后再相加起来, 就是所换成的十进制数。

例: $(.CA9BF3)_{16}$ 。

$$\begin{aligned} C \rightarrow 12, A \rightarrow 10, B \rightarrow 11, F \rightarrow 15 \\ 12 \times 16^{-1} + 10 \times 16^{-2} + 9 \times 16^{-3} + 11 \times 16^{-4} + 15 \times 16^{-5} + 3 \times 16^{-6} \\ = (12 \times 16^5 + 10 \times 16^4 + 9 \times 16^3 + 11 \times 16^2 + 15 \times 16^1 + 3 \times 16^0) \times 16^{-6} \\ = 13278195 \times 16^{-6} \\ = (.791442096) \end{aligned}$$

$$\therefore (.CA9BF3)_{16} \Rightarrow (.791442096)_{10}$$

由上述方法可见, 也可以归结成下述的做法:

把十六进制的小数部分单独分出, 去掉小数点而看作“整数”。把该“整数”化为十进制数, 再乘以 16^{-n} (或除以 16^n , 其中 n 是原十六进制数中小数部分所含的位数), 便转换成十进制数。

如① $.CA9BF3 \Rightarrow CA9BF3$, 六位($n = 6$)

$$\begin{aligned} \text{② } CA9BF3 &\Rightarrow 12 \times 16^5 + 10 \times 16^4 + 9 \times 16^3 + 11 \times 16^2 + 15 \times 16 + 3 \\ &= 13278195 \end{aligned}$$

$$\text{③ } 13278195 \times 16^{-6} = 0.791442096$$

$$\therefore (.CA9BF3)_{16} \Rightarrow (.791442096)_{10}$$

若同时也包含有整数和小数，则同样用“倍位相加法”。

例： $(2B.48)_{16}$ 。

$$B \rightarrow 11$$

$$\begin{aligned} & 2 \times 16^1 + 11 \times 16^0 + 4 \times 16^{-1} + 8 \times 16^{-2} \\ &= (2 \times 16^1 + 11 \times 16^0) + (4 \times 16^{-1} + 8 \times 16^{-2}) \\ &= 43 + (4 \times 16^1 + 8 \times 16^0) \times 16^{-2} \\ &= 43 + 72 \times 16^{-2} \\ &= 43 + 0.28125 \\ &= 43.28125 \end{aligned}$$

$$\therefore (2B.48)_{16} \Rightarrow (43.28125)_{10}$$

8. 十进制数转换为十六进制数

对整数部分：不断用16去除十进制整数。第一次除得的余数，就是十六进制整数的最低位（整数部分的最右边位）。所得的商若等于或大于16，则再除以16，得出的余数为十六进制整数的上一位。直至所得的商小于16，则这最后一个商便是十六进制整数的最高位（整数部分的最左边位）。上述的最后一个余数及商，若是10、11、12、13、14、15则要换成A、B、C、D、E、F。

例12： 将 $(186)_{10}$ 转换为十六进制数。

解：

$$\begin{array}{r} 16 \overline{)186} \\ \underline{11} \quad \dots\dots 10 \Rightarrow A \\ \downarrow \\ B \end{array}$$

$$\therefore (186)_{10} \Rightarrow (BA)_{16}$$

例13： $(24317)_{10}$ 。

解：

$$\begin{array}{r} 16 \overline{)24317} \\ \underline{16} \quad \dots\dots 13 \Rightarrow D \\ 16 \overline{)1519} \\ \underline{16} \quad \dots\dots 15 \Rightarrow F \\ 16 \overline{)94} \\ \underline{5} \quad \dots\dots 14 \Rightarrow E \end{array}$$

$$\therefore (24317)_{10} \Rightarrow (5EFD)_{16}$$

对小数部分：不断用16去乘十进制小数。第一次乘所得乘积中的整数（包括0），就是十六进制小数部分的最高位。若乘积中的小数部分不为0，再乘以16，所得乘积中的整数便是十六进制小数的下一位。直至所得乘积中的小数部分等于0为止（或取够所要求的十六进制小数的位数为止）。上述乘积中的整数，若遇10、11、12、13、14、15则要换成A、B、C、D、E、F。

例14： 将 $(0.71875)_{10}$ 换为十六进制数。

解:

$$\begin{array}{r|l} \text{整数} & .71875 \\ \hline & \times) 16 \\ \hline 11 & .50000 \\ & \quad 16 \\ \hline 8 & 00000 \end{array}$$

11 → B

$$\therefore (0.71875)_{10} \Rightarrow (.B8)_{16}$$

例15: 将 $(0.895)_{10}$ 换为十六进制数, 取五位小数。

解:

$$\begin{array}{r|l} \text{整数} & 0.895 \\ \hline & \times) 16 \\ \hline 14 & .320 \\ \hline & \times) 16 \\ \hline 5 & .120 \\ \hline & \times) 16 \\ \hline 1 & .920 \\ \hline & \times) 16 \\ \hline 14 & .720 \\ \hline & \times) 16 \\ \hline 11 & .520 \end{array}$$

14 → E, 11 → B

$$\therefore (0.895)_{10} \Rightarrow (0.E51EB)_{16}$$

对既有整数又有小数的十进制数, 要转换为十六进制数, 可分别将整数和小数转换后合起来。

例: 将 $(1789.3642)_{10}$ 换成十六进制数, 并只取四位小数。

整数部分:

$$\begin{array}{r} 16 \overline{) 1789} \quad \text{余数} \\ 16 \overline{) 111} \quad \dots\dots 13 \Rightarrow D \\ \quad \quad 6 \quad \dots\dots 15 \Rightarrow F \end{array}$$

$$\therefore 1789 \Rightarrow 6FD$$

小数部分:

整数	.3642
	×) 16
5	.8272
	×) 16
13	.2352
	×) 16
3	.7632
	×) 16
12	.2112

$13 \Rightarrow D, 12 \Rightarrow C$

$\therefore (.3642)_{10} \Rightarrow (.5D3C)_{16}$

$\therefore (1789.3642)_{10} \Rightarrow (6FD.5D3C)_{16}$

另外，也可以先将十进制数化为二进制数，然后再转换为十六进制数。

例： $(40539)_{10}$ 。

$(40539)_{10} \Rightarrow (1001111001011011)_2$

$\Rightarrow (9E5B)_{16}$

习 题

1. 为什么微电脑要采用二进制？
2. $(0110)_2$ 这组数中，哪个数是高位，哪个是低位？
3. 请将十六进制数A、D、F转换成二进制表示？
4. 分别将二进制数1100、1110、1011用十六进制数表示出来？

第五讲 电 脑 语 言

人与人之间交流信息需要用到语言（文字）。而人们为了使用电脑来办事，总得将人的意图——做什么事、怎样去做、怎样回复……等等，以机器能够接受的方式传送给电脑。电脑又要以使用电脑的人能够明白的方式给予回复。这种人与电脑之间的联络，可以比作“人机对话”，所以需要有电脑语言。“人机对话”及电脑系统的运转需要人设计出程序，按程序进行。人们为了设计程序，要有包括文字、符号、编程规则等一整套的规约，这种规约是人们与电脑打交道所必需的，在电脑行业中通用的，故称为电脑语言。

在计算机科学技术发展过程中，为了适应各种需要，让人们更好地使用电脑，出现了许多电脑语言，但就其类型来说，可归纳为三种。

1. 机器语言(MACHINE LANGUAGE)

这是一种直接根据电脑内部的工作原理和电路状态而制订的编程规约，基本字符用二进制数字“位”，用直接代表机器操作功能的“位”组合式来作指令，再由指令构成程序。编程时，每一条指令必须安排好存放它的地址，以区分执行的先后次序及各指令之间的转接关系；而指令本身则包含着表示机器做什么操作的代码、被操作的数据或者关系到数据的地址，所有这些都是用二进制数字表示。现在，大部分微电脑在实际使用方面已改用十六进制数字了，电脑会自动地转换为“位”组合式。

如：地址（存贮单元）167，应表示为10100111或A7；

加法，操作代码可表示为10110110或B6（注意，各种操作的代码，对不同的CPU会有所不同）；

数据84，应表示为01010100或54。

用机器语言编制的程序，是面向机器的，它直接同电脑内部的工作状态相联系，无需经过译编就可直接地变成机器的动作，执行起来最快，故称为目的程序（或目标程序）。

用机器语言设计程序，既有优点，也有缺点。优点是程序运行最迅速，所需的系统配置简便，适用于电脑内部的软件设计或应用方面的过程控制编程。缺点是编程的工作繁琐，且不同CPU的电脑的指令系统不同（操作代码往往不同），程序通用性差，编出的程序不直观，故难于交流和推广。

2. 汇编语言(ASSEMBLER LANGUAGE)

为了克服机器语言中操作代码是特定的“位”组合式，不直观和难记难用，编程很繁琐等缺点，同时，又尽可能保持其优点，制订出另一种电脑语言——汇编语言（组合语言）。汇编语言主要是将机器操作（功能）用英语词意的简写——助记符号，数据及地址用十六进制数来表示，再加上一些特定的符号。用它们汇编（组合）成简写的语句形式作为命令，再由命令构成程序。编程输入时，只需要指定存放整个程序的一个起始地址（存贮单元号），便可逐条编入命令（不同的电脑系统可能有些差异）。

用这种语言编制的程序，会由电脑内所配的译编系统翻译及汇编成机器语言程序而运

行。这样，使用电脑的人就可以避免许多繁琐的工作而比较简便地设计程序。这就是汇编语言相对机器语言来说的一个优点。但是，汇编语言也有一些缺点，如一是程序运行比较慢，二是电脑系统的配置要求增加，三是不同的电脑系统的汇编语言不同，通用性差。

汇编语言也是面向机器的。适合于专业程序设计人员用来设计电脑的系统软件。对于一般的用户是难于掌握和使用的。

3. 高级语言（算法语言ALGORITHMIC LANGUAGE）

机器语言和汇编语言都是面向机器的，主要是为机器“着想”。它们跟人们通常使用的语言差别较大，使一般的人难于掌握，不利于电脑的推广应用，对于一般人来说是“低级语言”。为了使人们更方便地使用电脑，根据人们计算或处理问题的常规以及各种领域的需要，设计师们设计出许多为人们着想的、与人们的通常语言很接近的电脑语言。这种类型的电脑语言，比较通用，对于一般人来说容易掌握，故统称为“高级语言”。

目前国内、外在微电脑方面比较普遍使用的高级语言有如下几种：

(1) BASIC (Beginner's All-purpose Symbolic Instruction Code. 初学者通用的符号指令编码，小型会话语言)

(2) FORTRAN (FORmula TRANslation 的缩写)

(3) ALGOL (ALGOrithmic Language, 算法语言)

(4) COBOL (COmmon Business Oriented Language的缩写，面向公共行业的语言)

(5) PASCAL (PASCAL——法国的著名哲学家的名字，结构程序设计语言)

(6) PL/1 (Programming Language one程序语言之一，大型通用语言)

高级语言是从人们利用电脑来计算或处理问题的过程和方法概括出来的。其优点很多，如设计程序的流程跟人们通常计算或处理事情的思路相近；整个程序用语句的形式构成，好象一篇文章；所用的语句（表达式）与常用英语的词句或算式相近；所涉及的数值用通常的十进制数；所用的字符及语法规则等也是简明易懂的，故容易为一般的用户熟悉和使用。二是设计程序时，不必理会电脑内部的具体结构和指令系统，故程序的通用性好。当然，不同的电脑系统所配置的同一种高级语言，可能会有些差异（大同小异），但在一种机型中通行的程序，只要稍加改动便可以应用到另一种机型。

用高级语言编制的程序，总是先由电脑自动地翻译并改编成机器语言程序而执行。因此程序的运行较慢，并要求电脑配置有相应的译编系统及有关设备。

使用高级语言设计的程序，主要是从人们计算和处理问题的常规出发的，故又称为源程序。要把它变成机器能够识别的目的程序，有两种方式，即编译方式和解释方式。

编译方式：在电脑系统的配置中，预先加入一种称为编译系统的软件设备。当相应的高级语言编制的源程序输入电脑后，编译系统便把源程序整个地翻译出来并编排成用机器语言表示的目的程序，然后执行。图5·1为编方式示意图。

解释方式：事先将一种称为解释系统的软件设备加入到电脑系统内。当高级语言编写的源程序输入电脑时，电脑将逐句地翻译和检查，译出一句检查无误即执行一句，若有错误即告知操作者，改正后才继续执行。是一种边翻译边执行的会话方式。这种方式比编译方式费时间，但是，可少占电脑的内存地址。图5·2为解释方式示意图。

FORTRAN ALGOL COBOL 等高级语言采用编译方式，BASIC 则基本上采用解

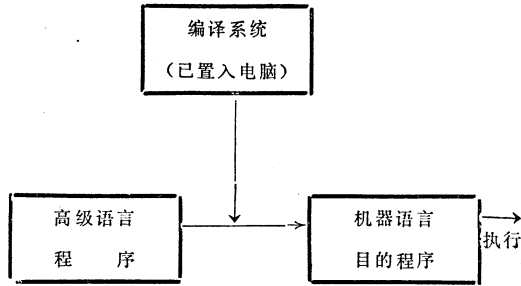


图 5.1

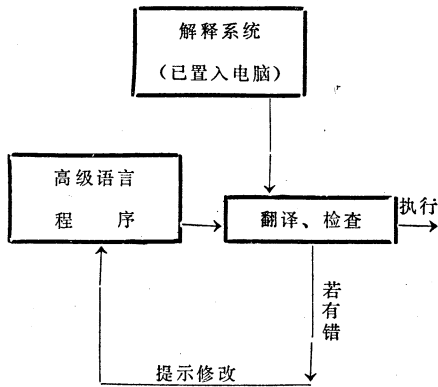


图 5.2

释方式。

不同型号的电脑，译编系统中的软件（程序）不同，出厂时一般把它们装入 ROM 内，或以外存贮（磁带、磁盘）方式提供。

习 题

1. 为什么要建立电脑语言？
2. 什么是电脑的机器语言，汇编语言，高级语言？试比较各种语言的优缺点。
3. 什么是编译方式，解释方式？

第六讲 BASIC及其编程知识

BASIC是英语 Beginner's All-purpose Symbolic Instruction Code——初学者通用的符号指令码的简写。对于不同的机型有不同版本的 BASIC,但其基本部分相同,只是扩展部分有些差异,若掌握了基本BASIC,又熟悉了一种机型所配用的扩展 BASIC,那么其它版本的BASIC,只要看看随机附送的有关资料,就能使用。下面就基本 BASIC 部分先作简要介绍。

1. BASIC概 要

BASIC既然是一种用于微电脑程序设计的高级语言,它就象通常的某一种语言(文字)一样,有它的字、词句、符号、语法规则等。它是从通常的英语中抽取出来的某些成份再构成的一个完整体系。它的基本内容有如下几个方面:

(1) 基本字符

字母: A……Z及a……z

数字: 0……9

符号: + 加号或正数号

- 减号或负数号

* 乘号

/ 除号

^ 或↑ 乘幂

E 10的幂次符或字母E

· 小数点或圆点号

(左括号

) 右括号

= 等号

< 小于号

> 大于号

<= 小于等于号

>= 大于等于号

<> 或> < 不等于号

AND 逻辑运算“与”

OR 逻辑运算“或”

NOT 逻辑运算“非”

, 数据分隔号或逗号

: 语句分隔号或冒号

" 引号

␣ 书写程序时的空格号

特别专用符号: !、@、#、\$、%、&、?、→、←

上述符号,除了按定义使用外,都可当作字符使用。不同机型的符号规范可能会有些不同,具体见键盘及随机说明。

(2) 保留字(专用词) 是专门用于代表电脑内部的各种功能或操作的英语词(有些简写),可用来作程序的命令、语句。每一种版本的BASIC都有一套保留字,具体可参阅有关版本的BASIC资料说明。

(3) 数据的表示

①常量 表示一个固定的具体数据,在程序运行中保持不变。分为数值型和字符型。

1) 数值型常量 一个固定的具体数值, 直接用十进制数给出, 可用定点或浮点表示。

例如:

定点表示某一常量: 99999999.1

浮点表示某一常量: $-8.34567891E-31$

2) 字符型常量 一个按照特定形式组合的字符串。它必须在双引号 (“ ”) 内直接给出。

如: “BOOK”、 “GOOD BYE!”、 “NO-AOOI-1”

②变量 用来代表一个数据的符号。它所代表的数据在同一个程序中可以变更。同一个符号在不同的地方可代表不同的数据。可分为简单变量和注标变量。

简单变量, 是用来代表一个独自的数据的符号, 其中又有数值型和字符型两种。

1) 数值型简单变量, 通常要以大写的英语字母开头, 用一个或两个甚至多个字符作为一个变量的符号。如A, AB, A1……, 但保留字不能作变量使用。

2) 字符型简单变量, 通常是在数值型简单变量的末尾加用符号\$来表示。

如 A\$, AB\$, A1\$……

关于注标变量见后面的详述。

(4) 规则 要编制某种电脑上的BASIC程序, 必须按照该电脑所配的 BASIC版本的规范。在程序中不允许出现该版本没有定义的字符、命令、语句等等, 并且必须按照该版本所规定的编程规则、语句格式(可称为“语法”)来编制程序。

进行数学运算, 必须使用BASIC的字符, 按规定的格式给出表达式。其运算的次序规定为:

①括号((())), 先里后外进行运算

②函数(如SIN(X))

③幂次(∧或↑)

④乘或除(*或/), 排在前者先做

⑤加或减(+或-), 排在前者先做

⑥关系运算(>、<、>=、<=、<>), 排在前者先做

⑦逻辑非(NOT)

⑧逻辑与(AND)

⑨逻辑或(OR)

如: 代数式 $\frac{2(\sin 3x - 1)^5}{2} - 3 + 8$

写成表达式: $\frac{2 * (\text{SIN}(3 * X) - 1) \wedge 5}{(5) (2) (1)} / 2 - 3 + 8$
(3) (4) (6) (7) (8)

电脑的运算次序按横线及横线下数字的顺序进行。

(5) 命令及语句

①BASIC的命令是用来指挥电脑工作的, 它分为立即执行和延缓执行两种方式。

1) 立即执行 这种方式是指命令发出后, 电脑便立即执行该命令, 执行过后, 命令不再保留在内存中。

2) 延缓执行 这种方式是把命令编入存贮式的程序中, 输入电脑后先在内存中存贮起

来，只是在发出执行命令（一般用RUN—运行）后才按照程序所编定的次序依次执行，执行过后仍保留在内存中。

在BASIC的命令中，有些命令既可作立即执行又可作延缓执行，而有一些命令只能作延缓执行，要注意区分。

②从命令的输入方式来说，也可以分为单键命令和多键命令两种。

1) 单键命令 在电脑的键盘上配有一些命令键（功能键），每按一次某个命令键，就输入一个对应的命令。具体的作用参看各种电脑的键盘说明。

2) 多键命令 要由操作者按照特定的格式选择键盘上的字符，逐个打键而构成命令输入。

目前在微电脑中实际使用的BASIC几乎都是扩展BASIC，不同版本的扩展BASIC虽有差异，但都是在基本BASIC的基础上加以扩展的，所以就其基本部分来说，都是相同的。为了使读者在学习分述命令之前对BASIC有个概括了解，这里先对基本BASIC的命令及语句作简要的介绍。

③单用命令和联用命令 BASIC 的命令，代表着电脑系统的操作或功能，一般的是用保留字（专用词）来表示的。其中有些命令是单独使用的，称为单用命令，有些是要与别的命令或字符组合成语句来使用的，称为联用命令。

1) 单用命令 一般是用来控制系统直接实现某种功能，也可以作程序中的语句。

如：清去荧屏上的全部字符（简称为清屏）	HOME
列出内存中的程序（列表）	LIST
程序运行中暂停（暂停）	STOP
继续运行程序（继续）	CONT
程序的结尾（结束）	END

2) 联用命令 联用命令是用来组成语句然后再构成程序的。

④设置输出（印出） PRINT……

⑤求数学函数值或某些特殊数值的命令，统称为函数，不能单独使用。

求 x 的绝对值	ABS (X)
求 e 的 x 次方	EXP (X)
求 x 的平方根	SQR (X)
求 x 的自然对数	LOG (X)
取 x 的整数部分	INT (X)
取 0 至 1 之间的某一个随机值	RND (X)
取一个符号	SGN (X) 当 $X = 0$ SGN (X) = 0
	$X > 0$ SGN (X) = 1, 取正号
	$X < 0$ SGN (X) = -1, 取负号

求 x 的正弦值	SIN (X)
求 x 的余弦值	COS (X)
求 x 的正切值	TAN (X)
求 x 的反正切	ATN (X)

⑥自定义函数 用户自己定义一个数学函数的命令（详见后面分述）。

	DEF FN.....
②提供数据	
单式赋值 (还可作运算)	LET..... =
序列式赋值	READ
{ 读赋数据	DATA
{ 置存数据	RESTORE
重新恢复数据区作用	INPUT
程序运行中才输入数据	DIM (.....)
③设置数组	
④转移	
无条件转移	GOTO
条件转移	IF THEN
选择转移	ONGOTO
⑤转子程序与返主程序	
转子程序	GOSUB
返主程序	RETURN
⑥循环	FOR = TO STEP
	:
	:
	NEXT
⑦注解	REM

(6) 出错提示 在BASIC的解译系统中, 有一套查错处理体系。当操作者使用电脑当中出现违反该语言规范时, 会有信息(荧屏显示或打印机上印出)提示操作者。操作者可对照在BASIC的解说资料中的出错提示信息表, 很快找出错误。因此, 使用者必须熟悉这个表。不同版本的BASIC, 出错信息表不同, 可参阅本书的附录。

2. BASIC程序的构成和基本规则

按BASIC语言规约编制并能在有相应配置的电脑上运行的程序称BASIC程序, 也称为BASIC源程序。这种程序的基本构成和规则如下:

(1) 一个全整的程序, 由若干个“行”组成。一般一个“行”只含一个语句, 但对扩展BASIC程序大多数则允许一个“行”内含有几个语句, 各语句之间以冒号(:)分隔开。每一个语句规定电脑执行某一方面的功能。所以, 实际上程序是由若干语句按一定顺序组合而成。

(2) 一个“行”, 一般分为三部分。

①语句标号(行号) “行”的开头必须是十进制数字, 这个数字称为语言标号或叫“行”的标号, 简称标号(行号)。一个标号代表一个程序行。标号必须是整数。一般情况下, 电脑按标号的大小顺序执行, 或按标号进行改编。不同电脑对标号数值范围有不同的规定。如华宇 200规定1—9999, APPLE的实数BASIC规定0—63999。

在编程输入时, 可以不按标号大小的顺序输入电脑, 解释系统会把源程序中所有的语句, 按标号的大小顺序排列好, 电脑运行程序时就依此顺序。标号不一定连续, 一般是标号间隔为10, 以便修改程序时增插一些“行”(如上述间隔, 可以插入9个“行”)。

②语句 语句包括语句定义符和语句体。其中语句定义符，是规定电脑执行电脑内部已约定的功能命令。它是一些特定的字符组合，又叫做电脑的保留字。

例： 10 LET A = 15

10是标号。LET A = 15 是语句，其中LET A = 是语句定义符（命令），它表示对变量A赋值。

语句体（内容），是跟在语句定义符（命令）后面的需要执行的具体内容。如上例中的15，表示值是15。整个语句就表示：将数值15赋予（存入）变量A或以变量A代表数值15。

又如 20 PRINT 3 + 4

20是标号，表示在运行程序时该行的次序。PRINT是命令，规定电脑要在荧屏上显示（或在打印机上打印）。3 + 4是内容，表示将3与4相加。整个语句就表示将3与4相加的和在荧屏（或打印机）上输出。

有些简单语句只有语句定义符（命令）。如 20 PRINT 3 + 4 : STOP中的STOP表示暂停的语句。

③“行结束”标志（↵） 当把书写的程序（一般省去↵）输入电脑时，每输入一个“行”的最后，必须给电脑一个“行结束”信号（一般是按RETURN键或ENTER键或↵键……），电脑才会把从标号开始的所有内容，作为以该标号为标志的一个“行”（与其他“行”区分）而存入内存RAM中。

如：10 LET X = 3 + 4 ↵
20 PRINT X ↵

（3）每个程序的末尾要用语句END作为程序的结束。运行程序时，遇到END语句便结束该程序的运行。若一个程序的末尾无结束语句，电脑就会接着执行内存中的别个程序，也有的电脑作为程序出错处理。

下面举一个求某车间某月人平均奖金的例子，编写一个完整的BASIC程序。

例： 设某车间共计54名职工。某月奖金分配情况如下：一等奖9元，有14名职工获得；二等奖6元，有29名职工获得；三等奖4元，有9名职工获得；四等奖2元，有2名职工获得。若获得一、二、三、四等奖的人数分别以A、B、C、D表示，求该月人平均奖。

程序如下：

标号	语 句
10	INPUT "A = " ; A
20	INPUT "B = " ; B
30	INPUT "C = " ; C
40	INPUT "D = " ; D
50	LET Y = A + B + C + D
60	LET X = (9 * A + 6 * B + 4 * C + 2 * D) / Y
70	PRINT X
80	END

（4）当一个程序通过输入装置（键盘或纸带）输入后，就先存入电脑的内存RAM

中，只是当操作员发出（键输入）运行命令（RUN↵）后，电脑才开始执行该程序。程序运行过后，该程序仍保留在电脑的内存RAM中而不会消失。如上述例子的程序开始运行后，电脑会提示操作者输入获各等奖的人数。当操作者依次将14、29、9、2输入后，电脑便显示或打印出X的值6.29629。如再次键入运行命令，电脑就再运行一遍。并且再次显示或打印出X的值6.29629。

（5）BASIC规定一个语句，必须在一行内输入完。一行内能输入多少个字符，要按各电脑的规定。这在随机说明书上有说明。如果一个语句太长，一行或二行输入不完，必须把这一语句拆成两个或多个语句来输入。

例如：

```
10 LET A = 3.14159 + 2 * 1.23456 / 3 * COS (X + Y)
```

可改写为：

```
12 LET B = COS (X + Y)
```

```
15 LET C = 2 * 1.23456 / 3
```

```
20 LET A = 3.14159 + C * B
```

这里，12号、15号、20号三行与10号一行等效。

3. BASIC的特点

从上述可见，BASIC有如下一些特点：

（1）比较简单、容易使用。基本语句只有十几种，所使用的词与英语中的词句（简写）相近。数学运算的表达方式跟通常的算式、算法相似。程序比较直观，易于理解记忆。不同机型的BASIC大同小异，通用性好。

（2）是一种会话式的语言，便于修改。人们可以通过电脑的键盘和显示器或打印机进行人机对话。当源程序输入电脑运行时，电脑会自动检查出程序中的错误并提示操作人员，操作员可以用键盘进行改正。也可以随时调出已输入过的程序，边运行边修改，直至满意为止。

例如，一个计算平均值的程序。若输入为：

```
10 READ C, M, F, K
```

```
20 DAT 90, 98, 100
```

```
30 LXT V = (C + M + F + K) / 4
```

```
40 PRINT "V = ", V
```

```
50 END
```

当发出运行程序的命令（RUN↵）后，会显示或打印出：

```
? SYNTAX ERROR IN 20
```

其意思是提示操作员，20号语句有语法错误。这时可以通过键盘把20号语句中的DAT改正为DATA。再运行程序，又会显示出。

```
? OUT OF DATA
```

这是电脑提示操作员20号语句已部分改正，但程序仍有错误——数据用完了，却还要去读取。这时，可调出程序来检查，原来20号语句中少给了一个数据。可再通过键盘在20号语句中加上一个数据。变成

```
20 DATA 90, 98, 100, 105
```


再运行程序，又会显示出

? SYNTAX ERROR IN 30

这是电脑提示操作员，30号语句有语法错误。可再用上述方法将30号语句中的LXT改正为：

30 LET V = (C + M + F + K) / 4

再运行程序，电脑检查无错误了，便一直运行下去并把结果显示或打印出来。

(3) BASIC是一种小型的算法语言，允许使用的数值有一定范围（一般为 -10^{38} 至 10^{38} ），数据处理或科学计算方面的功能不够完备，且运行速度慢。不同版本的扩展BASIC，扩展功能也有较大的差别。

习 题

1. BASIC的特点是什么？
2. BASIC程序的基本结构如何？编程时应注意什么？
3. 试将下列各题用BASIC编出程序。

(1) $y = 5x^2 + 2x + 2$ ，其中 $x = 4$ ，求 y 值。

(2) $x = 6 \times 5 + 2 \times 2 + 4 \div 2$

(3) $x = (6 \times 5 + 2 \times 2 + 4) \div 2$

(4) $x = 6 + 4 \div 2$

(5) $x = (6 + 4) \div 2$

第七讲 程序设计知识(一)


电脑是要人去指挥和控制的。要求电脑计算一个数据或处理一个问题，必须由人制定出一套指挥电脑工作的方案来，这就叫做程序设计。

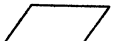
为了编制出能够指挥电脑实现人的意图，达到预期目的的程序，必须熟悉有关程序设计的知识。


1. 程序框图 (流程图)

程序框图也称流程图 (简称框图)，是用来粗略地描述电脑解决问题的步骤。为通用起见，一般规定使用统一的符号画框图。目前，在微电脑的应用中尚未完全统一。通常使用的符号有：

(1) 桶形框  表示开始或结束。

(2) 矩形框 (叙述框)  代表一个步骤或表示一个执行内容。

(3) 平行四边形框  表示输入或输出。

(4) 菱形框  表示一个判断或比较。

(5) 箭符 (连接线) \rightarrow 表示程序运行中各步骤的次序及转接关系。

上述的框中，第2、第3二个都是一个入口、一个出口的，第4个却有一个入口、二个或三个出口，如图7.1所示。

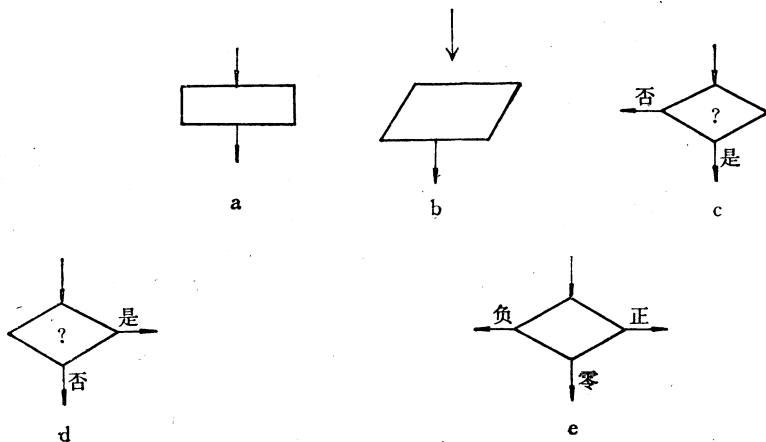


图 7.1

使用上述符号，将解决问题的步骤用文字 (或代号) 写在框图内，再用连接线联起来，便是流程图。

在流程图中，可运用判断 (比较) 框图及连接线的箭头，将处理问题过程中的分支或循

环直观地表现出来。

2. 程序设计方法

(1) 制订方案 首先分析任务, 归纳类型(数值计算型或数据处理型或其它)。若是属于数值计算, 须确定计算方法, 及对结果的要求, 再写出数学表达式。若是非数值计算的问题, 需找出数学模型, 确定处理办法, 再订出处理步骤及结果要求。

不管属于哪一种类型, 作为一个完整的程序, 总是包含输入、处理、输出三个部分。

输入: 主要是将程序中所要计算或处理的具体数据输进, 提供给程序计算或处理。

处理: 是指具体的计算或处理步骤, 以及各步骤的安排。

输出: 主要是把所需要取得的结果或资料显示或打印出来。

把上述考虑及已知条件、必要的数, 程序中要用到的各个代表符号等列出清单。

(2) 画出流程图 根据订出的方案, 选用框图, 把解决问题的叙述用文字或符号填进框内, 用箭符连接成一个整体。

(3) 编写程序 按照所用的语言规范, 把流程图变换成按一定顺序编排的命令或语句。

(4) 调试程序 把书面编写的程序输入电脑运行。在运行中校验、修改, 直至满意后, 再把程序录存起来。

下面以微电脑APPLE II、LASER-3000所配用的BASIC语言, 来介绍程序设计的知识。

3. 程序中信息的输出

电脑系统中的显示器(Monitor或TV), 是一种输出装置, 用来显示操作者从键盘上输入的字符(从键盘上输入的字符, 立即在荧屏上显示出来, 以便校对); 也用来显示操作者所要知道的信息, 告诉人们内存的资料, 程序运行的情况和结果、系统所处的状态等等。总之, 电脑系统通过输出装置给予操作者回复。

操作者为了获得程序运行的情况和结果, 必须在程序中设置有关输出的命令或语句。至于要以何种方式, 怎样的格式输出, 都可以用程序语句去设定。

设置输出字符的基本命令为:

PRINT 内容 (在打键时, 可用?号代替PRINT)

在PRINT后面的“内容”是用来设定输出的内容及输出的格式。格式可以有几种:

(1) 空出一行

PRINT

在命令PRINT后面不给出任何内容。执行此命令后, 光标跳过一行, 在荧屏上或打印纸上给出一行空白。

命令格式如:

·50 PRINT

(2) 照原样输出

PRINT “内容”

执行此语句后, 将程序输入时在“ ”内所设定的内容, 按原样显示(印出)。

如: ① PRINT “APPLE” ↵

APPLE

```

PRINT "*LASER-3000*" ↵
*LASER-3000*
② PRINT "1 + 1 = " ; 1 + 1 ↵
1 + 1 = 2
PRINT 1 + 1 ↵
2
③ 10 PRINT "PRINT FORMAT=NULL:"
20 PRINT
30 PRINT "GOOD!"
40 END
RUN ↵
PRINT FORMAT=NULL:

```

GOOD!

(3) 紧凑输出——；号的使用 在命令PRINT后面，将要输出的几项内容，用紧凑号(；)分隔开构成语句。执行此语句后，各项内容便紧挨着（不留空格）显示（印出）。

① ；号用在各项内容之间

```
1) PRINT "A" ; "P" ; "P" ; "L" ; "E" ↵
```

APPLE

```
2) 10 PRINT "7 * 5 =" ; 7 * 5
20 PRINT "S=" ; 3 + 4 ; "CM"
```

RUN ↵

7 * 5 = 35

S = 7 CM

② ；用在PRINT语句中最后一项内容之后

PRINT 内容；内容；内容；

执行此语句，会在所设定的几项内容按设定的格式显示(印出)后，光标(打印头)不换行，而只后移一格，紧挨着显示出(印出)下一个要输出的程序行所设定的显示(印出)内容。

```

如 10 PRINT "*" ;
20 PRINT "A" ;
30 PRINT "P" ;
40 PRINT "P" ;
50 PRINT "L" ;
60 PRINT "E" ;
70 PRINT "*" ;
80 PRINT "*"
90 PRINT "A"
100 PRINT "P"
110 PRINT "P"

```

```

120 PRINT "L"
130 PRINT "E"
140 PRINT "*"
150 END

```

上述程序，从10行至70行，由于输出语句的末尾，用了紧凑号（；），因而使各个行输出的字符紧挨着显示（印出）。第80行至140行，则每显示（印出）一个字符后，光标换行，故各个字符是在不同行上显示（印出）的。其运行的结果为

```

RUN
*APPLE*
*
A
P
P
L
E
*

```

（4）分区输出——，号的使用 在荧屏上显示字符时，每行可显示40个，分成三个区。

第一区：第1至16个位置（格）。

第二区：第17至32个位置（格）。

第三区：第33至40个位置（格）。

划分如：

第一区	第二区	第三区
┌───┐	┌───┐	┌───┐
1 16	17 32	33 40

第一区有16个格，第二区有16格，第三区只有8个格。

若第一区显示的字符 ≤ 15 个，第二区显示的字符 ≤ 7 个，第三区显示的字符 ≤ 8 个，则三个区正常自然分区显示。

若第一区显示的字符 ≥ 16 个，则第二区不显示字符（全区空出）；若第二区显示的字符 ≥ 8 个，则第三区不显示字符（全区空出）；若第三区要显示的字符 ≥ 9 个，则接到下一行的第一区接着显示。

上述是微电脑APPLE特殊的设定，在使用分区显示时要特别留意。其它的电脑所配的BASIC对分区的设定，请查阅有关资料。

①分区号，用在各项显示（印出）内容之间

```
PRINT 内容,内容,内容
```

使用上述语句，可使要输出的内容分区显示（印出），即显示（印出）的各项内容之间有一些空格隔开，显得清晰。若有二至三项需要对比时，可采用这种输出语句。但要注意各区的显示字符的个数是否合适。

为了说明三个自然区的显示情况，举例如下：

```

10 PRINT "123456789012345", "1234567", "12345678"
20 PRINT "1234567890123456", "1234567", "12345678"
30 PRINT "123456789012345", "12345678", "12345678"
40 PRINT "123456789012345", "1234567", "123456789"

```

运行上述程序时，第10行输出字符的格式为：

```
123456789012345  1234567  12345678
```

第20行的输出为：

```
1234567890123456  12345678  123456789
```

整个第二区空出

```
12345678
```

第30行的输出为：

```
123456789012345  12345678  123456789
```

空八格 整个第三区空出

```
12345678
```

第40行的输出为：

```
123456789012345  1234567  123456789
```

空九格 只能显示八个

9 (第9个换行)

```

例1: 10 PRINT "ORDER", "NAME", "RESULT"
      20 PRINT 1, "LI", 100
      30 PRINT 2, "ZHOU", 90

```

运行后，输出为：

```

ORDER  NAME  RESULT
   1    LI    100
   2    ZHOU   90

```

上例中的输出格式，常用于列表式的结果输出，其中第10行为设计“表头”的格式。

```

例2: 10 PRINT "ONE", "TWO", "THREE"
      20 PRINT "(1)", "(2)", "(3)"

```

②分区号，用在PRINT语句的末尾

PRINT 内容，

这种输出语句的作用是，在显示（打印）完该语句的内容后，光标（打印头）不换行而是跳到后一个显示区（打印区），显示（印出）下一个输出程序行的输出内容。

```

例3: 10 PRINT "A",
      20 PRINT "B",
      30 PRINT "C"
      RUN

```

```
A      B      C
```

```
例4: 10 PRINT 1, 2,
```

```

20 PRINT 3, 4, 5, 6, 7,
30 PRINT 8
40 PRINT 9, 0,
50 PRINT "END"
RUN ↵
1 2 3
4 5 6
7 8
9 0 END

```

综合练习（初步编程）：

```

10 PRINT
20 PRINT "DISPLAY TYPE"
30 PRINT "TEXT (NORMAL):",
40 PRINT "**APPLE I PLUS**"
50 PRINT:PRINT
60 PRINT "INVERSE:",
70 PRINT "**APPLE I PLUS**"
80 PRINT:PRINT
90 PRINT "FLASH:",
100 PRINT "**APPLE I PLUS**"
110 NORMAL
120 LIST

```

4. 程序中数据的提供

BASIC语言不能直接进行变量的运算，而只能对具体的数据做运算。因此在运行程序中，必须对各个变量给出具体的数据，否则运算的结果总是0或根本无法进行运算。

试做下面的简单程序：

```

10 A = 3 * X
20 PRINT X, A, B$
30 END
RUN ↵
0 0

```

由此可见，在一个程序中的变量，必须给定数值才有意义。

BASIC对变量提供数据的方式有：

- (1) 赋值 LET 变量 = 数据
或省去LET 变量 = 数据

其中的=号是赋值符号，其意义是将右边的数据赋予左边的变量。

①对数值型变量的赋值，直接在=号的右边给定，可有几种形式。

1) 直接给出具体数值

如 A = 5

A3=10.5

2) 用表达式给定 (式中也可以含有已赋值的变量)

如 $A2 = 2 * 5 - 0.1$
 $Y = 3 \wedge 2 + 4 * 5$

或 10 X=1
20 X1=2*X+1+X^2

3) 传递式赋值 (原变量的值不会消失)

如 10 A=10
20 B=A
30 C=B
40 PRINT A, B, C
RUN ↵
10 10 10

4) 取代赋值, 可在程序运行中更换变量的值, 以新代旧 (后赋的值冲除先赋的值)。

如 10 X=0.5
20 X=1
30 PRINT X

还可以利用原变量的值作新的赋值。

如 10 X=10
20 X=X+0.1
30 PRINT X

②对字符型变量的赋值 直接在=号的右边的双引号“ ”内给出具体字符串。

如 G\$="GOOD!"
GB\$="GOOD BYE!"

没有赋值的字符型变量, 无任何字符 (即空白)。

例: 10 NAME\$="LI"
20 NO\$="S-001-A3"
30 F\$=" "
40 PRINT NAME\$, NO\$, F\$, A\$

(2) 序列式赋值 上述的赋值语句, 在一个语句中只能把一个数据赋予一个变量。若要把许多数据分别赋予各个变量, 按对应的顺序排列构成专门的赋值语句, 可作成批赋值。有关的命令如下:

置存数据 DATA
读赋数据 READ
读数指针复始 RESTORE

①有关语句

1) 置存语句

行号 DATA 常量, 常量, 常量, ……
如 100 DATA 3, BOOK, 5.7, 16

语句的作用是把一批数据（常量），按顺序置入内存数据区。

①这个语句中的常量就是所要输入并赋予各个对应变量的直接具体数据，可以是数值型的，也可以是字符型的。但不能用变量，也不能用表达式。

②语句中的各个数据之间，要用逗号（，）分隔开，但最后一个数据后面不能加逗号（，）。

③若数据太多，一个语句放不完，可分为两个或多个语句，紧接着下行再写（注意数据的顺序）。

④此语句可以排在程序中的任何位置，通常排在最后。

⑤此语句要配合读赋语句才有意义（见下述）。

2) 读赋语句

行号 READ 变量, 变量, 变量, ……

如 20 READ N, B\$, N1

语句的作用：在置存数据区内，从读数指针所指示的位置开始读出数据，并赋予语句中第一个变量，又把读数指针往下移动一个数据位置。接着读出下一个数据并赋予语句的下一个变量，再把读数指针下移，直至本语句中的最后一个变量赋予数据，把读数指针下推一个位置为止。如果下面还有另一个读赋语句，则继续读赋数据。

①读赋语句必须配有相应的置存数据语句。此语句中有多少个变量，置存语句中就应有多少数据（不能少，可以多）。否则，出现OUT OF DATA（数据用完）的错误。

②语句中的变量可以是简单变量或注标变量，可以是数值型的或字符型的。但各个变量的类型及顺序编排（序列）与置存语句中的数据序列必须一致。否则，出现TYPE MISMATCH（类型不匹配）的错误，或者不能按要求把某个数据对应地赋予某个变量。

③语句中的各个变量之间必须用逗号（，）分开，但最后一个变量之后不能加逗号（，）。

3) 读数指针复始语句

行号 RESTORE

语句作用：把置存数据区的读数指针恢复到起始位置（即指向第一个数据），使已被读过的数据恢复作用，可供读赋语句再从头开始使用。

①此语句要在用含有置存语句及读赋语句的程序中才有意义。

②此语句要放在两个读赋语句之间，即在已执行过的读赋语句之后及将要读赋数据的语句之前。

③编程联用 上述的三个语句要在程序中联用才有意义。适用于在编程输入（键入）时，已经确定了要输入的具体数据，而且数据较多的情形。运用上述三个语句可以实现成批输入数据。在编程输入时必须把数据及变量按对应关系编排好。

1) 要输入一个数据，必须有一个变量对应，且类型要一致。

2) 有读赋语句，必须有相应的置存数据语句。可供读赋的数据个数不得少于读赋语句中的变量个数。

3) 在置存语句中的数据，若是属于数值型的，要直接给出具体的数值（常量）。若是属于字符型的，可直接给出字符串（不能用算式），但在字符串的第一个或最后一个空隔无效，也可采用在“ ”内给出字符串。

4) 当程序中有几个读赋语句时, 要注意第一个读赋语句总是从第一个置存数据语句中的第一个数据开始读赋, 再依次连接下去。读完一个置存语句中的数据后, 该读赋语句中的变量赋值还不全, 电脑会自动从下一个置存语句中的第一个数据继续读赋。要记住每执行完一个读赋语句后, 读数指针的位置, 以便确定下一次读赋语句从哪一个数据开始读赋。

例 5: 10 READ A, B, C\$, D, E\$, F
 20 PRINT A, B, C\$, D, E\$, F
 30 READ A1, B1, D\$, N1\$
 40 PRINT A1, B1, D\$, N1\$, "!"
 50 DATA 1, 0, OK, 5
 60 DATA "END", 15.5, 101, 103, "COME"
 70 DATA IN

例 6: 10 READ S1, S2, S3\$,
 20 RESTORE
 30 READ A1, A2, A\$,
 40 RESTORE
 50 READ B1, B2, B\$:RESTORE
 55 READ C1, C2, C\$
 60 PRINT S1, S2, S3\$
 70 PRINT A1, A2, A\$
 80 PRINT B1, B2, B\$
 90 PRINT C1, C2, C\$
 100 DATA 10, 55, "GOOD!"

例 7: 计算图7.2所示并联电阻。

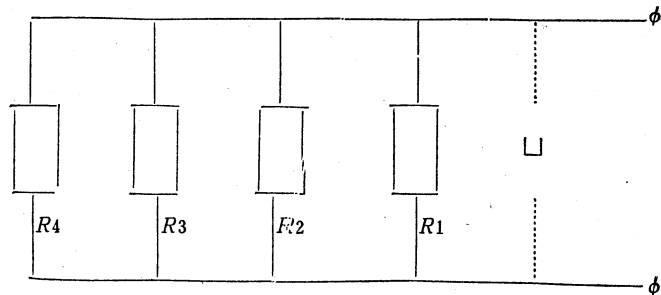


图 7.2

10 READ R1, R2, R3, R4, U
 20 $R = 1 / (1/R1 + 1/R2 + 1/R3 + 1/R4)$
 30 $I = U/R$
 40 PRINT "R=" ; R ; "(OM)", "I=" ; I ; "(A)"
 50 DATA 100, 150, 240, 360, 24

③运行中输入 在输入程序时, 可以暂不输入数据, 而是编入特定的命令, 当程序运行到此命令时, 出现特定信号, 这时才由使用者由键盘输入数据, 或由别的设备 (如磁盘机)

输入。下面主要介绍由键盘输入的操作：

1) 输入单个字符——GET

语句格式： 行号 GET 变量名

例： 50 GET D
100 GET N\$

使用此语句应注意以下几点：

①当程序运行到此语句时，暂停运行，出现光标，等待使用者由键盘输入一个合乎要求的字符后，荧屏上无显示，不用按RETURN键，程序会继续执行。

②若编程输入时，在GET后面是数值型变量名，则运行到此语句时，输入数字有效。如果输入+，-，*，/，E等运算符号，则当作0，显示出EXTRA IGNORED（额外的不理）。如果输入字母等非数值型的数据，或只按RETURN等，则显示出？SYNTAX ERROR（语法错误）。

③若编程输入时，GET后面是字符型变量名，则运行到此命令时，输入单个字母或符号有效。如果按CTRL-@，CTRL-H，CTRL-C，→，←等键，则当作空字符串而无效。

④由于在程序运行中，这种命令本身无法使操作者明白需要如何操作，故最好是在此命令前加上一句说明，以提示操作者应该做什么。

如 PRINT "ENTER A CHARACTER"

或 PRINT "ENTER A NUMBER"

例： 10 GET D
20 ? D,B
30 ? "ENTER A NUMBER:" :GET D
40 ? D,B
50 GET A\$
60 ? A\$,NA\$
70 ? "ENTER A CHARACTER:" :GET NA\$
80 ? A\$,NA\$

2) 输入一组数据——INPUT

语句格式： 行号 INPUT 变量名，变量名，变量名或 行号 INPUT "提示字符"；变量名，变量名，变量名

例： 10 INPUT A
10 INPUT A,D,B,E,N\$
10 INPUT "ENTER YOUR NAME AND NUMBER";N\$,N1,N2
10 INPUT "CHARACTER,N1,N2,N3=";C\$,N1,N2,N3

运用这种语句，在编程输入时，对某些变量（包括数值型和字符型）可不给定数据，而是在运行程序的过程中才由键盘或磁盘机输入数据。这样，可以根据需要，随时变更程序中的某些参量，临时输入数据。但这种语句要使运行停顿，降低了执行命令的速度，最好用在程序的开头，而且不宜多用。

若在程序中编有INPUT命令，当运行到此命令时，电脑会暂时停止运行，而在荧屏上

显示出信号（没有使用提示字符时，显示出光标和问号。使用提示字符时，则将字符照印并加光标），要求操作者对INPUT后面的变量名依次输入对应的数据。每输完一个数据，按一下RETURN（也可以按逗号（，）键，直到不出现提示符为止。再按RETURN，电脑将自动地执行下一个命令。

在回答时应注意：

①输入（即对INPUT语句的回答）的必须是具体的数据，且类型要一一对应（数值或字符）。不能输入表达式、函数、字符串内含问号（？）、引号（"）、CTRL -X、CTRL -M等。

对数值型数据，可包括0~9、空格、+（正号）、-（负号）、小数点、及10的幂次符E。

若输入的不合要求，按RETURN后会显示：

? REENTER (再输入)

②逗号（，）可用来分隔开各个输入的数据。但不要用作最后的一个字符（否则其后的空格无效）。若用作第一个键入的数据的起头字符，则该数据当作0或空字符串。

③冒号（:）在未加引号（"）的情况下，当键入冒号（:）后，以后所键入的数据均无效（EXTRA IGNORED——额外的不理）。若用作键入的第一个数据的起头字符，则该数据当作0或空字符串，以后键入的数据无效。

④CTRL-C用在第一个键入的数据之始，按下RETURN后便中断程序的运行，即中断了INPUT命令的执行，不能用CONT继续运行。用在键入的第一个数据之后，则当作字符。

```
例 8: 10 INPUT "A=" ; A
       20 INPUT "B=" ; B
       30 PRINT "A+B=" ; A+B
```

```
例 9: 5 INPUT "R1,R2,R3,R4=" ; R1,R2,R3,R4
       10 INPUT "U=" ; U
       20 R=1/(1/R1+1/R2+1/R3+1/R4)
       30 I=U/R
       40 PRINT "R=" ; R, "I=" ; I
       50 END
```

RUN ↵

R1,R2,R3,R4=100,200,300,400

U=220

R=48 I=4.58333334

再运行:

RUN ↵

R1,R2,R3,R4=1210,1936,806.6,484

U=220

R=215.10637 I=1.02274981

第八讲 程序设计知识 (二)

为了使电脑能够方便地处理数学问题,作数值计算,BASIC提供了一套用于求基本数学函数(定义)值的命令。运用这些命令,又可以导出求解其它函数(定义式)值的表达式——导出函数。

这一类的命令,专门称为函数,不能单独使用,否则作SYNTAX ERROR(句法错误)处理。因此,必须与其它的命令构成语句使用。

1. 基本数学函数

(1) 求三角函数值的命令

①SIN(X) 表示求 $\sin x$ (x 的正弦函数)值。括号()内的 X 代表要给定的数值,以弧度为单位。

例: 求 30° 的正弦值

```
PRINT SIN (30/180*3.14159265)
```

②COS(X) 表示求 $\cos x$ (x 的余弦函数)值。括号内 X 代表要给定的数值,以弧度为单位。

例: 求0.5弧度的余弦值

```
PRINT COS (0.5)
```

③TAN(X) 表示求 $\tan x$ (x 的正切函数)值。 X 代表要给定的数值, X 的绝对值不能等于或接近于 $\frac{\pi}{2}$,否则作OVERFLOW ERROR(数值太大,当数值 $>10^{38}$ 溢出)处理。

例: 求 $7 \times 3 - 50 \div 2.5$ 的正切值

```
PRINT TAN (7*3-50/2.5)
```

④ATN(X) 表示求 $\tan^{-1}x$ (x 的反正切函数,或 $\arctan x$)值。 X 代表要给定的数值。所求得的函数值的单位是弧度,在 $-\frac{\pi}{2} \sim \frac{\pi}{2}$ 之间。

例: 求正切函数值为28.7所对应的角度

```
PRINT ATN (28.7)
```

(2) 求算术函数(定义)值

①SQR(X) 表示求 \sqrt{x} (x 的平方根) X 代表要给定的数值,应 ≥ 0 (实数)

例: 求2的平方根

```
PRINT SQR (2)
```

②EXP(X) 表示求 e^x (e 的幂次)值, $e=2.71828183$ 。 X 代表要给定的数值,必须 $X \leq 88$,否则作为数值超界(溢出)错误处理。

例: 求 e^9 的值

```
PRINT EXP (9)
```

③LOG(X) 表示求 $\ln x$ (以 e 为底的自然对数)值。 X 代表要给定的数值,必须 $X > 0$ 。

例：求24的自然对数值

```
PRINT LOG (24)
```

若求常用对数（以10为底）或其它对数的值，则要根据换底公式 $\log_a x = \frac{\log_e x}{\log_e a} = \frac{\ln x}{\ln a}$ ，

换成以 e 为底的自然对数式，如：

```
lgx $\Rightarrow$ LOG(X)/LOG (10)
```

例：求1000的常用对数（以10为底）的值

```
PRINT LOG (1000) /LOG (10)
```

④ABS (X) 表示求 $|x|$ (x 的绝对值)。X代表要给定的数值。

例：求算式 $3 \times \sqrt{2} - 5$ lg500的绝对值

```
PRINT ABS (3 * SQR (2) - 5 * LOG (500) )
```

⑤INT (X) 表示取整数值。X代表要给定的数值，此命令的作用是舍去X中的小数部分，取不大于原数X的最大整数。

例1：对7.99999999取整数

```
PRINT INT (7.99999999)
```

```
RUN ↵
```

7

例2：对-7.99999999取整数

```
PRINT INT (-7.99999999)
```

```
RUN ↵
```

- 8

如果要求保留N位小数，对N+1位后的小数部分要作四舍五入的处理，可运用如下形式的命令：

```
INT (X * 10 ^ N + 0.5) / 10 ^ N
```

```
或 INT (X EN + 0.5) / 1 EN
```

式中N要先给定具体数值。

例3：对5.4342取三位数字，即要求保留2位小数，对第3位以后的小数作四舍五入。

```
PRINT INT (5.4342 * 10 ^ 2 + 0.5) / 10 ^ 2 ↵
```

```
5.43
```

例4：对5.4354作四舍五入，保留2位小数

```
PRINT INT (5.4354E 2 + 0.5) / 1 E2 ↵
```

```
5.44
```

⑥SGN (X) 称为符号函数。其作用是根据 () 内所给定X的数值如何而得出一个1 (正号) 或-1 (负号) 或0值。

即

$$\text{SGN}(X) = \begin{cases} 1 & (\text{当 } X > 0) \\ 0 & (\text{当 } X = 0) \\ -1 & (\text{当 } X < 0) \end{cases}$$

例：PRINT SGN (3.457) * 5 ↵

```

5
PRINT SGN (- 3) * 5 ↵
- 5
PRINT SGN (0) * 5 ↵
0

```

练习程序：运行此程序后，任意键入不同数值，看显示情况。

```

10 INPUT "X=" ; X
20 PRINT "X=" ; X, "SGN (X) =" ; SGN (X)
RUN ↵

```

⑦RND (X) 称为随机函数，此命令的作用是根据括号内所给定的数值如何，可以取得一个在 $0 \sim 1$ 之间的（可以为 0，而不能为 1）随机值。每使用一次此命令，都可以取得一个随机值，所取到的随机值也是可以利用的。

1) 当括号内的数值为 0 时，即 RND (0)，能取得一个随机值，这个值可能是机内（有专门机构）刚产生的，或是已用命令取出过的随机值。

2) 当括号内的数值是一个任意的正数时，即 $X > 0$ ，RND (X) 能取得一个随机值。每使用一次（不管括号内所用的正数是否和前面相同），就取得一个不同的随机值。此命令使用多少次，就可取得多少个不同的数值。

如 PRINT RND (3)，PRINT RND (3)，PRINT RND (7) 可显示出三个不同的数值。

3) 当括号内的数值是任意一个负数时，即 $X < 0$ ，RND (X) 的作用除了取得一个随机值外，还可以“记下”或“取回”某一个随机值序列。括号内的负数不同，所取到的随机值及所“记下”或“取回”的随机值序列也不同。

例：

```

10 PRINT RND(- 7)
20 PRINT RND( 6)
30 PRINT RND( 1)
40 PRINT RND( 4)
50 PRINT RND(- 3)
60 PRINT RND( 9)
70 PRINT RND( 1)
80 PRINT RND( 6)
90 PRINT RND( 7)
100 PRINT RND(- 3)
110 PRINT RND(15)
120 PRINT RND(17)
130 PRINT RND( 8)
140 PRINT RND( 2)
150 PRINT RND(- 7)
160 PRINT RND(20)
170 PRINT RND(50)

```

```
180 PRINT RND(9)
```

运行后会显示出18个在0~1之间的数值，且其中第1至第4与15至第18的数值相同，排列顺序也相同，第5至第9与第10至第14两组的数值及顺序相同。

练习程序：

```
@10 INPUT "X="; X
20 PRINT RND(X)
30 PRINT RND(X)
40 PRINT RND(X)
50 END
RUN
```

运行后，键入0或不同的正数。

```
⑥10 INPUT "X="; X
20 PRINT RND(X)
30 X1 = ABS(X)
40 PRINT RND(X1)
50 PRINT RND(X1+1)
60 PRINT RND(X1+3)
70 END
RUN
```

运行后，键入不同的负数。

```
例：10 PRINT RND(9)
20 PRINT RND(0)
30 PRINT RND(3)
40 PRINT RND(1)
50 PRINT RND(5)
60 PRINT RND(-4)
70 PRINT RND(2)
80 PRINT RND(6)
90 PRINT RND(8)
100 PRINT RND(-35)
110 PRINT RND(1)
120 PRINT RND(7)
130 PRINT RND(9)
140 PRINT RND(-4)
150 PRINT RND(11)
160 PRINT RND(15)
170 PRINT RND(7)
180 PRINT RND(-35)
190 PRINT RND(17)
```



```
200 PRINT RND(8)
```

```
210 PRINT RND(20)
```

```
RUN
```

```
.741013586
```

```
.741013586
```

```
.282506557
```

```
.0110821604
```

```
.498302609
```

```
2.99214662E-08
```

```
.808563215
```

```
.236586843
```

```
.533367257
```

```
3.27181624E-08
```

```
.886728929
```

```
.46380911
```

```
.69979308
```

```
2.99214662E-08
```

```
.808563215
```

```
.236586843
```

```
.533367257
```

```
3.27181624E-08
```

```
.886728929
```

```
.46380911
```

```
.69979308
```

2. 导出函数

为了节省内存，BASIC只直接给出四个求三角函数值的命令及一些求其它函数（数学定义）值的指令。如果在处理数学问题时，要用到别的函数（数学定义式）时，可在编制程序中，运用BASIC中已有的函数及其它命令，编出有关的算式来。这种算式称为导出函数。只要在程序中已有设定的导出函数的语句，那么在后面的程序行中，导出函数的作用就跟基本函数的作用一样了。

关于导出函数的设定，主要是根据数学公式及运用基本函数来编。

如根据

$$\sec x = \frac{1}{\cos x}$$

可设定

$$\text{SEC}(X) = 1/\text{COS}(X)$$

下面列出的是一些可以利用基本函数组合出用以求其它函数值的算式——导出函数。对某些函数来说，有些X值是不能用的（例如，当 $\text{COS}(X) = 0$ 时就不能计算 $\sec x$ ，所以在使用时必须注意。

$$\text{ARCCOS}(X) = -\text{ATN}(X/\text{SQR}(-X * X + 1)) + 1.5707633$$

计算X的反余弦值， $\text{ABS}(X) < 1$ 。

$$\text{ARCCOT}(X) = -\text{ATN}(X) + 1.5707633$$

计算X的反正切值。

$$\text{ARCCOSH}(X) = \text{LOG}(X + \text{SQR}(X * X - 1))$$

计算双曲函数中，X的反余弦值($\text{ABS}(X) \geq 1$)。

$$\text{ARCCOTH}(X) = \text{LOG}(((X + 1)/(X - 1))/2)$$

计算双曲函数中，X的反余切值($\text{ABS}(X) > 1$)。

$$\text{ARCCSCH}(X) = \text{ATN}(1/\text{SQR}(X * X - 1)) + (\text{SGN}(X) - 1) * 1.5707633$$

计算X的反余割函数值($\text{ABS}(X) > 1$)。

$$\text{ARCCSCH}(X) = \text{LOG}((\text{SGN}(X) * \text{SQR}(X * X + 1) + 1)/X)$$

计算双曲函数中，反余割函数值($X > 0$)。

$$\text{ARCSEC}(X) = \text{ATN}(\text{SQR}(X * X - 1)) + (\text{SGN}(X) - 1) * 1.5707633$$

计算反正割函数值($\text{ABS}(X) \geq 1$)。

$$\text{ARCSECH}(X) = \text{LOG}((\text{SQR}(-X * X + 1) + 1)/X)$$

计算双曲函数中，反正割函数值($0 < X \leq 1$)。

$$\text{ARCSIN}(X) = \text{ATN}(X/\text{SQR}(-X * X + 1))$$

计算反正弦函数值($\text{ABS}(X) < 1$)。

$$\text{ARCSINH}(X) = \text{LOG}(X + \text{SQR}(X * X + 1))$$

计算双曲正弦函数值。

$$\text{ARCTANH}(X) = \text{LOG}((1 + X)/(1 - X))/2$$

计算双曲函数中，反正切函数值($\text{ABS}(X) < 1$)。

$$\text{COSH}(X) = (\text{EXP}(X) + \text{EXP}(-X))/2$$

计算双曲余弦函数值。

$$\text{COT}(X) = 1/\text{TAN}(X)$$

计算余切函数值($X < > 0$)。

$$\text{COTH}(X) = (\text{EXP}(-X) / (\text{EXP}(X) - \text{EXP}(-X))) * 2 + 1$$

计算双曲余切函数值($X < > 0$)。

$$\text{CSC}(X) = 1/\text{SIN}(X)$$

计算余割函数值($X < > 0$)。

$$\text{CSCH}(X) = 2/(\text{EXP}(X) - \text{EXP}(-X))$$

计算双曲余割函数值($X < > 0$)。

$$\text{LOG}_a(X) = \text{LOG}(X)/\text{LOG}(a)$$

计算以a为底的对数值(a > 0是先给定的，X > 0)。

$$\text{LOG}_{10}(X) = \text{LOG}(X)/2.30258509$$

计算常用对数(10为底)值(X > 0)。

$$\text{MOD}_a(X) = \text{INT}(X/a - \text{INT}(X/a) * a + .05) * \text{SGN}(X/a)$$

计算X被a除之后的余数(a < > 0的已知数。)

$$\text{SEC}(X) = 1/\text{COS}(X)$$

计算正割函数值($X < > \pi/2$)。

$$\text{SECH}(X) = 2/(\text{EXP}(X) + \text{EXP}(-X))$$

计算双曲正割函数值。

$$\text{SINH}(X) = (\text{EXP}(X) - \text{EXP}(-X))/2$$

计算双曲正弦函数值。

$$\text{TANH}(X) = -\text{EXP}(-X)/(\text{EXP}(X) + \text{EXP}(-X) * 2 + 1)$$

计算双曲正切函数值。

③ 自定义函数(DEF语句)

用户可以根据自己的需要,在编制程序时,把某些计算式设定为函数的形式。这样,程序中若多次用该计算式算出参数不同时算式的结果,就象求函数值那样方便了。

(1)自定义函数的设定 DEF是设定自定义函数的命令, FN是自定义函数的标记,自定义函数的名称是 变量名(变量),如S(R)。

语句格式: 行号 DEF FN 变量名(变量) = 含变量的表达式

如: 10 DEF FN S(R) = 3.1416 * R ^ 2

上述是用户在程序中设定自定义函数的语句。其中命令DEF及标记FN是固定的形式。变量名及变量可由编程人员按照BASIC中变量使用的规范,任意选用。括号内的变量必须与表达式(计算式)中的参数(变量)一致才有意义。表达式中除一个变量与左边括号内变量相对应外,还可以有其它的变量(由其它程序行赋值)。语句中所用的变量,是个象征性的,可任意选用,即使是在别的程序中出现过的变量,也可以用。如下述情形是允许的:

10 XA = 5

20 Y = 5 * XA + 2

30 Z = 2 * Y - 5

40 DEF FN YAZ(XA) = 3.14 * XA + 10 * Y + Z / 2

(2)自定义函数的作用 用设定自定义函数的语句,可以将一个含有可变参数的复杂计算式,变为求基本函数值一样方便的命令形式。只要在程序中把某一计算式设定过某一自定义函数,在后面的程序行要用到该计算式时,就可以只写出该自定义函数(包括FN 变量名(数值))。其作用是把括号内的数值代入计算式中相对应的参数变量,同时还会将前面已有赋值的各个变量所对应的值代入计算式中,一齐计算而给出整个算式的结果。

如: 10 X = 5

20 Y = 2 * X

30 Z = 2 * Y

40 DEF FN Q(X) = 10 * X + 20 * Y + Z / 2

50 T = FN Q(20)

运行后T值为410。

注意: 式中变量X代入的是20,而不是5; Y代入的是2 * 5; Z代入的是2 * 2 * 5,即 $T = 10 * 20 + 20 * 2 * 5 + 2 * 2 * 5 / 2$

(3)自定义函数的调用

①必须是在本程序中已经设定过的。

②自定义函数本身不能单独使用,必须与其它命令(如PRINT)或语句(如赋值于变量)联用。

③必须给出所要调用的自定义函数全名(包括FN 变量名(数值))。其中括号内

给定的值，可以直接给出具体的常数值，也可以用前面已经有赋值的变量或可算出具体数值的表达式。

例 5: 10 DEF FN SI(R) = 3.14159 * R ^ 2
 20 PRINT FN S(2)
 RUN
 12.56636

例 6: 运行下述程序并分析各个自定义函数的作用。

```
5  N = 9
10 DEF FN Y(X) = 2 * X + X - N
15 PRINT FN Y(20)
20 DEF FN P(B) = 6 + 4
25 M = FN P(10)
30 PRINT M
35 DEF FN Z(L) = L ^ 2 * 6
40 DEF FN C(V3) = V3 / 2 + FN Y(X) + FN Z(L)
45 PRINT FN C(M)
50 END
```

第九讲 程序设计(一)

——转移及循环

1. 转移

电脑在运行程序时，一般情况下是按行号的大小，由小到大顺序执行，直至程序完结为止。但也可以改变其运行的次序和方向，以致产生分支或形成循环。方法是在程序中编入某些转移（跳越）语句。属于这一类的命令或语句有如下几个。

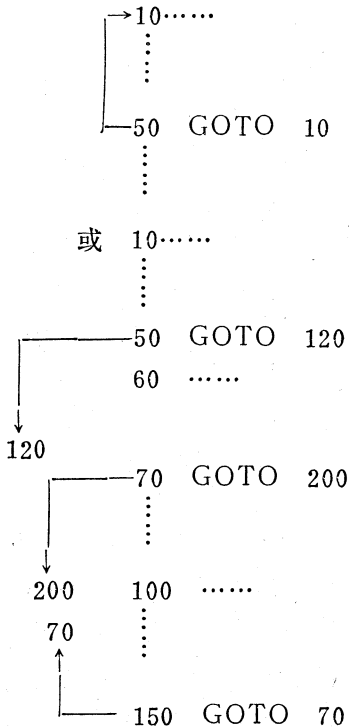
(1) 无条件转移（跳越）命令——GOTO

语句格式：行号1 GOTO 行号2

例： 50 GOTO 10
 50 GOTO 100

上述是无条件转移（跳越）语句。GOTO是转移（跳越）命令。当程序运行到此语句时，无条件地转移到GOTO后面的行号所指定的程序行去执行。这个程序行可以在无条件转移语句的前面（倒转去运行）也可以在无条件转移语句的后面（跳过一些程序行去运行），但必须在本程序中能找到的行号，否则会当作出错处理。（若转移到一个非执行性的REM语句行也是允许的。）

这种语句在运行中的作用是改变运行方向、形成重复或出现跳越式地运行，产生分支等等。其类型为：



例1： 加法（累加）器程序（设每次给出的数为X，累计数为C）。

```
10 C = 0
20 INPUT "X="; X
30 C = C + X
40 PRINT "X="; X, "C="; C
50 GOTO 20
```

例2： 求正方体的总表面积，设某边长L，每次增加0.5。

```
10 INPUT "L="; L
20 A = L^2 * 6
30 PRINT "L="; L, "A="; A
40 L = L + 0.5
50 GOTO 20
```

(2) 条件转移（跳越）命令——IF THEN

语句格式：行号 IF 条件关系 THEN

使用这种语句，是在程序运行过程中，根据事先所设置的条件是否达到（条件关系是否满足或是否成立），而决定下一步执行什么命令或如何运行。若是条件满足，则执行 THEN 后面所指定的内容；若是条件未满足，便继续接着下一个程序行往下运行。如图9·1所示。

①在IF后面所设置的条件关系式，可为下述的几种。

- 1) 代数关系式，如 $3 * X + B \leq 4 * \text{SIN}(X) - 5$
- 2) 逻辑关系式，如 $K > 7 \text{ AND } Y = 3$
- 3) 字符串等式，如 $N \$ = \text{"END"}$

总之；要求所设置的关系式，能够给出明确的判断结果。

②在代数关系式中，进行比较的项目有下述一些类型

- 1) 变量与数值，所用的变量在比较之前已赋值；
- 2) 变量与变量，所用的变量在比较之前已赋值；
- 3) 变量与表达式，所用的变量在比较之前已赋值；
- 4) 表达式与表达式，所用的变量在比较之前已赋值。

③在THEN后面，设置条件达到时所要执行的内容，可以是需转去执行的程序行号，但必须是在本程序中找到，也可以是有具体内容的语句、命令，而且可编入几个语句、命令，各语句或命令之间要用：号分隔开（注意：一行内不得超过255个字符）。

（注：IF THEN 30与IF GOTO 30等效。）

若IF后面的条件没有达到，THEN后面属于本程序行的全部内容无效。

④在IF后面的关系式中，有两种特殊情形。

1) 字符串计算式无效。遇到这种情况，总是当作条件成立；

2) 空字符串（" "），当作条件不成立。若一个程序运行中，遇到这种情形三次以后，当作出错而给出如下信息：

? FORMULA TOO COMPLEX ERROR

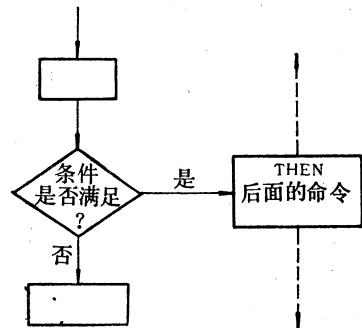


图 9·1

⑤在IF与THEN之间使用的字符，要避免保留字的出现。如IF B=AT HEN的语中句，A与T组成保留字AT，将致使原语句内容改变，造成语法错误。下面是一些供参考的语句类型：

```
40 IF A = 3 THEN 80
40 IF X > Y GOTO 100
40 IF <= A ^ 3 - 5 THEN GOTO 70
40 IF 3 * A * X <> 5 * LOG(A) THEN 90
40 IF X * Y <> 0 THEN 80
```

或

```
40 IF X * Y THEN 80 (不等于 0, 整个内容可以省去不写出)
40 IF A = B + 5 THEN PRINT NA $
40 IF C4 $ = "M" THEN IN = 0
40 IF Q < 14 AND M < M1 THEN A = 5 : GOTO 60
40 IF V > 100 THEN PRINT "GOOD!" : END
40 IF Q < 145 OR N > 65 THEN B = N + 1 : GOTO 80
```

例 3：求 50 以前 (含 50) 的自然数总和
流程图如图 9·2 所示：

参考程序：

```
10 LET S = 0
20 LET A = 0
30 LET A = A + 1
40 IF A > 50 THEN 70
50 LET S = S + A
60 GOTO 30
70 PRINT "S =", S
80 END
```

例 4：某工厂产值增长率为 $x\%$ ，求 x 分别为 4、5、7、9、15 时，该厂产值过几年后，可以翻两翻 (100:400)。

设原来产值 $P = 100$ ， N 为年数。

流程图如图 9·3 所示。

程序是：

```
10 LET P = 100
20 LET N = 0
30 READ X
40 IF X < 0 THEN 100
50 LET P = P * (1 + X/100)
55 LET N = N + 1
60 IF P >= 400 THEN 80
70 GOTO 50
```

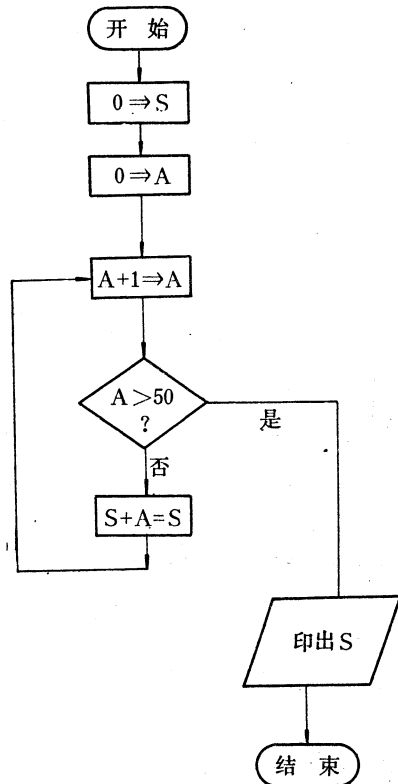


图 9·2

```

80 PRINT "P="; P, "N="; N
90 GOTO 10
95 DATA 4, 5, 7, 9, 15, -1
100 END

```

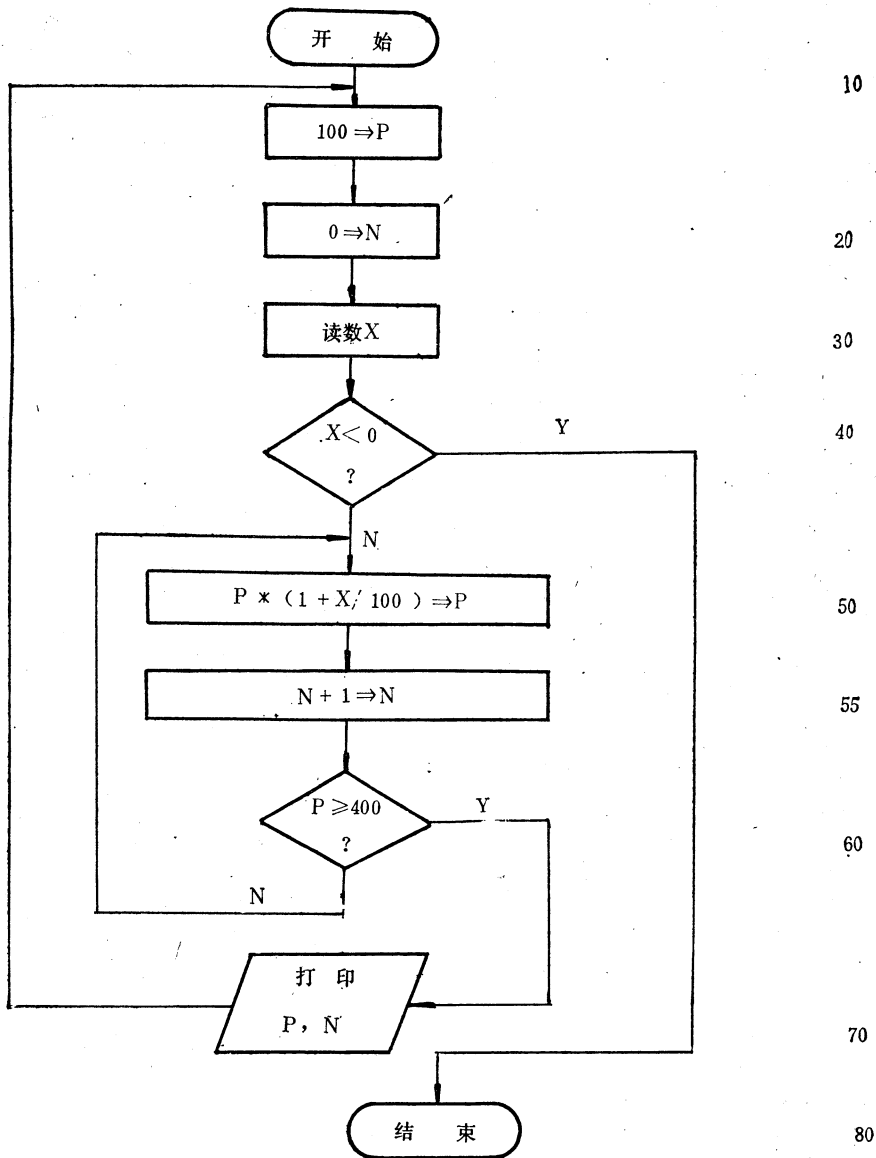


图 9·3

40号语句是设置终止点的，当增长率X为负数时，转到100号即结束，否则，接下一个语句，即50号继续执行；50号语句等号左边的P表示经过一年后的产值；55号语句表示每执行50号语句一次后，增加一年；60号语句当满足 $P \geq 400$ 时就转到80号语句打印输出，否则，执行下一个语句，即70号语句。70号语句转到50号语句再算产值。

例5： 计算行李费。若行李重量 $W \leq 50$ 公斤，为0.13元/公斤，超过50公斤的部分，为0.2元/公斤。

```
10 INPUT "W="; W
20 IF W = 0 THEN 70
30 IF W > 50 THEN X = 0.13 * 50 + 0.2 * (W - 50) : GOTO 50
40 X = 0.13 * W
50 PRINT "W="; W, "X="; X, "YUAN"
60 GOTO 10
70 END
```

例6： 在任意N个数中，找出其中的最大值和最小值，设MA代表最大值，MI代表最小值。

```
10 INPUT "N="; N
20 READ X1
30 MA = X1
40 MI = X1
50 FOR I = 1 TO (N - 1)
60 READ X
70 IF X <= MA THEN 90
80 MA = X
90 IF X >= MI THEN 110
100 MI = X
110 NEXT I
120 PRINT "MAX:"; MA, "MIN:"; MI
130 DATA 3.5, 4.6, -201, -3.1, 1987, 4321, 41.6, 815, 253.4
140 DATA 5.3, -6.4, .8
```

(3) 选择转移(控制转移, 开关转移) 命令——ON.....GOTO.....

语句格式: 行号 ON 数值或表达式 GOTO 行号1, 行号2, 行号3

例: 50 ON 4 GOTO 20, 70, 10, 100, 50
50 ON X/4-8 GOTO 20, 70, 10, 100, 50
50 ON A GOTO 20, 70, 30, 100, 10, 50, 800

说明: ON后面正整数(0~255)有效, 可给出具体的数值或已赋值的变量或可算出数值的表达式。遇小数会自动取为整数(按INT命令取)。

GOTO后面应给出本程序中有的行号, 各行号间用逗号(,)分隔开, 最后不加任何符号。

语句作用是: 当运行到此语句时, 根据ON后面的数值而选择GOTO后面的某个行号为下一个执行的程序行。其规则是:

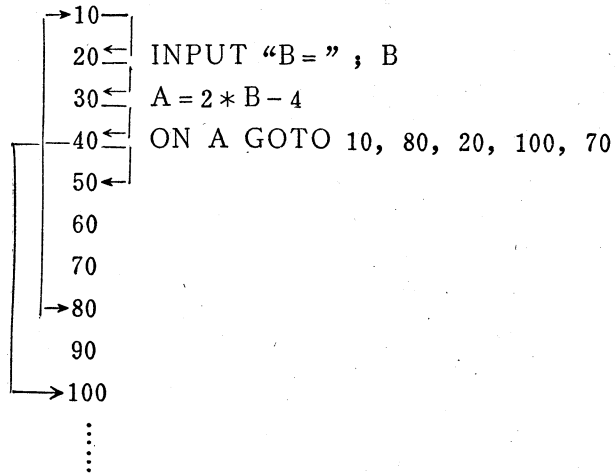
ON后面的数值	下一个执行的行号
1	GOTO后面的第一个
2	GOTO后面的第二个

```

3          GOTO后面的第三个
:
:
:
5          GOTO后面的第五个

```

或大于GOTO后面的行号个数 接此语句的下一行继续执行
 以下面的程序中有关程序行来说明此语句在程序运行中的作用。



应注意以下两点：①ON后面的数值在0~255之外（如负数），作ILLEGAL QUANTITY（数值超界）错误。

②若GOTO后面所给的行号在本程序中不存在，作UNDEF'D STATEMENT（程序行不存在）错误。

```

例7:  10 PRINT  "OK"
      20 PRINT  "NEXT"
      30 INPUT  "B = " ; B
      40 A = B - 1
      50 ON A GOTO 10, 20, 30, 40, 70, 80, 90, 100, 110, 120
      60 PRINT  "6"
      70 PRINT  "7"
      80 PRINT  "8"
      90 PRINT  "9"
     100 PRINT  "10"
     110 PRINT  "11"
     120 PRINT  "12"
     130 PRINT  "13"
     140 PRINT  "14"

```

在运行后，试依次键入B=0~15看结果如何，并作解释。

```

例8:  分段统计学生成绩。
      10 INPUT  "N = " ; N

```

```

20 A0=0:A6=0:A7=0:A8=0:A9=0
30 FOR I=1 TO N
40 READ S
50 IF S<60 THEN 80
60 IF S=100 THEN 160
70 ON S/10-5 GOTO 100, 120, 140, 160
80 A0=A0+1
90 GOTO 170
100 A6=A6+1
110 GOTO 170
120 A7=A7+1
130 GOTO 170
140 A8=A8+1
150 GOTO 170
160 A9=A9+1
170 NEXT I
180 PRINT "0--59:" ; A0
190 PRINT "60--69:" ; A6
200 PRINT "70--79:" ; A7
210 PRINT "80--89:" ; A8
220 PRINT "90--100:" ; A9
230 DATA 76, 58, 84, 32, 91, 95, 94, 88, 78, 83, 82, 67, 63
240 DATA 69, 74, 77, 100, 78, 81, 92, 95, 67, 49, 90
300 END
RUN↵
N=24
0--59: 3
60--69: 4
70--79: 5
80--89: 5
90--100: 7

```

2. 循环

```

FOR ..... = ..... TO ..... STEP .....
    ⋮
NEXT .....

```

在编制程序去处理问题时，往往会有某些操作或功能重复地用到，而每次用的时候功能或操作是同样的，只是操作（处理）的数据不同，且数据的变更又是有规律的。对于这一类的问题，使用循环的程序段去处理是很方便的。

语句格式: 行号 FOR X=X1 TO X2 STEP X3

⋮

行号 NEXT X

例: 一个包含有循环语句的循环程序段

```
10 FOR X=1 TO 13 STEP 2
20 N=X^2
30 PRINT X, N
40 NEXT X
50 END
```

其中第10行和40行是循环语句, 整个循环语句包括两个语句, 构成比较复杂。

第一个语句, 称为循环的入口语句(循环起点)。FOR、=、TO、STEP是命令, 用来说明循环的, 但不能各自单独使用, 必须按上述格式构成语句来用。

X 为循环变量, 代表循环段中的替换性的数值型数据(不断变更对它的赋值)。是控制循环段进行的关键, 对它每赋给一个值就使循环段运行一次, 直至对它赋的值超过指定值, 循环段就结束。

X1 为初值, 是刚开始运行循环段就赋给循环变量的第一个值(首值)。

X2 为终值, 是控制循环段结束运行的值, 当循环变量所赋的值等于终值时, 作最后一次运行循环段。当循环变量所赋的值大于终值时, 结束循环段的运行而转出。

X3 为步长(间隔), 它是赋给循环变量的各个值之间(各替换值间)的间隔。当其数值设定为1(正1)时, 可连同STEP一起省略掉。

所以, 入口语句可归纳为: FOR变量名=初值 TO 终值 STEP 间隔值

上述的X1初值, X2终值, X3步长各值可为正数, 负数、整数、小数。可以直接给出具体数值, 也可以用已有赋值的变量或可算出具体数值的表达式。

从上述可知, 这个语句规定了循环变量的赋值情况及循环操作的次数。

末尾的语句, 称为循环的出口语句(循环终点)。NEXT是专用的命令, X是与入口语句中的循环变量名相同的(对应的)变量。即:

ENXT 变量名

当执行此语句后, 将刚才的循环变量值增加(或减少)一个间隔值。再将这新的循环变量的值与终值比较, 若 \leq 终值, 转回入口语句的下一个语句, 继续循环操作。若循环变量的值已超过终值, 则转出到此语句的下一个语句去接着运行。

运用上述的两个语句可以构成各种各样的循环。

(1) 简单循环 包含只有一个入口语句和一个对应的出口语句的循环段。要循环地执行的语句编在入口语句与出口语句之间, 这一段程序行称为循环体。对简单循环, 出口语句中可不用变量名而只有NEXT

①空循环 只有入口语句及出口语句而无循环体的循环段。这种循环段可以起“延时”作用。如:

```
行号 FOR X=X1 TO X2 STEP X3
```

⋮

```
行号 NEXT X
```

也可以改为 行号 FOR X=X1 TO X2 STEP X3:NEXT X

例: 50 FOR I=1 TO 1000:NEXT I

②操作循环 有一个入口语句、循环体及一个出口语句的循环段。如:

```
行号 FOR X=X1 TO X2 STEP X3
      :           :
      :           :
      :           :
行号 NEXT X
```

其中可设定正循环(循环变量的赋值递增: $X1 < X2$, $X3 > 0$;

或设定负循环(循环变量的赋值递减): $X1 > X2$, $X3 < 0$

例9: 求顺序数列 $1 + 3 + 5 + 7 + 9$ 的总和。

```
10 S=0
20 FOR X=9 TO 1 STEP -2
30 S=S+X
40 NEXT X
50 PRINT S
60 END
```

例10: 求阶乘 $10! = ?$

```
10 N=1
20 FOR Y=1 TO 10
30 N=N*Y
40 NEXT Y
50 PRINT N
60 END
```

例11: 求任意边长的正方体的总表面积。

设边长 $X1$ 到 $X2$, 间隔值 $X3$

```
10 INPUT "X1=" ; X1:INPUT "X2" ; X2:INPUT "X3=" ; X3
20 FOR L=X1 TO X2 STEP X3
30 S=L^2*6
40 PRINT "L=" ; L: "S=" ; S
50 NEXT L
60 END
```

运行后再输入 $X1=10$, $X2=1$, $X3=-0.5$ 。

练习如下程序, 运行后, 试对 $Y1$, $Y2$, $Y3$ 键入各种数值。

```
10 INPUT "Y1=" ; Y1:INPUT "Y2=" ; Y2:INPUT "Y3=" ; Y3
20 IF Y1=0 THEN END
30 FOR M=Y1 TO Y2 STEP Y3
40 PRINT "M=" ; M
50 NEXT
```

(2) 复合循环(循环的嵌套, 多重循环) 由若干个简单循环相套而构成的程序段。在该段程序中, 含有若干对循环语句。各种版本的BASIC允许复合的层数不同。

如: APPLE II的实数BASIC可允许10重复合循环, LASER—3000相同。

例: 10 FOR I=0 TO 15 STEP 3
 :
 50 FOR J=1 TO 88
 :
 90 FOR K=45 TO 5 STEP -5
 :
 150 FOR X=.4 TO .9 STEP .1
 :
 180 FOR Y=.5 TO .001 STEP -.1
 :
 220 NEXT Y
 :
 250 NEXT X
 :
 300 NEXT K
 :
 350 NEXT J
 :
 420 NEXT I

①每一重循环, 必须有一个入口及一个对应的出口, 入口与出口的变量名要一致。

②一重循环套一重循环, 构成一层层的, 绝对不能交叉。

③如遇几重循环的出口语句紧挨着相接的情形, 可合用一个出口语句, 将变量名由内而外(由先而后)编排在NEXT后面。

运行到复合循环段时, 从第一个循环语句进入后, 按程序运行规则执行, 依次进入各个循环, 至进入最里层的循环后, 执行完最里层的循环段才转出, 再执行完次里层的循环段后再转出……。总之, 在整个复合循环段中, 对非出口语句, 仍按程序运行规则及语句内容执行, 只是运行到出口语句时, 才由其决定转到那一个语句。对于循环段是先执行完里层, 由里至外逐层完成。

例12: 做“九九表”的程序。

```
10 FOR X=1 TO 9
20 FOR Y=1 TO 9
30 M=X*Y
40 PRINT X; " * "; Y; " = "; M
50 NEXT Y, X
60 END
```

例13: 求图9·4所示AB两端的等效电阻 $R = ?$

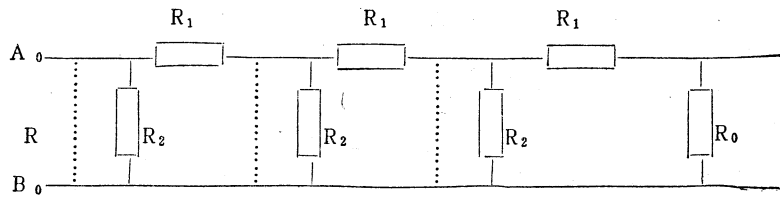


图 9·4

把整个电路用虚线分为若干段级。在第一级中串联电阻值为 $R_1 + R_0$ ，把这个值用 R 来表示（即此时 $R = R_1 + R_0$ ），再将 R 与 R_2 并联，并联值为 $\frac{R * R_2}{R + R_2}$ 再以 R 代表。

则

$$\frac{R_{(原)} \times R_2}{R_{(原)} + R_2} \Rightarrow R_{(新)}$$

$R_{(原)}$ 是 $R_1 + R_0$ 的和；这时求出的 R 是第一级的等效电阻。再将这个 $R_{(新)}$ 与第二级中的串联电阻 R_1 相加，然后再与电阻 R_2 并联，求出第二级等效电阻值，如此一直求出总的等效电阻 R 值。

参考程序如下：

```

10 INPUT R0, R1, R2
20 PRINT
30 PRINT "N", "R"
40 LET R = R0
50 FOR N = 1 TO 7
60 LET R = R + R1
70 LET R = R * R2 / (R + R2)
80 PRINT N, R
90 NEXT N
100 END

```

设 $R_0 = 100$ 、 $R_1 = 200$ 、 $R_2 = 300$ ，执行后，印出各级对应的电阻 R 值：

N	R
1	150
2	161.538
3	163.953
4	164.448
5	164.569
7	164.574

若设 $R_0 = 100$ 、 $R_1 = 100$ 、 $R_2 = 100$ ，则：

N	R
1	66.6666
2	62.5

3	61.9047
4	61.8181
5	61.8055
6	61.8037
7	61.8034

例14: 求长、宽、高取不同数值时, 长方体的总表面积。

```

10 INPUT "A1="; A1:INPUT "A2="; A2:INPUT "A3="; A3
20 INPUT "B1="; B1:INPUT "B2="; B2:INPUT "B3="; B3
30 INPUT "H1="; H1:INPUT "H2="; H2:INPUT "H3="; H3
40 FOR A=A1 TO A2 STEP A3
50 FOR B=B1 TO B2 STEP B3
60 FOR H=H1 TO H2 STEP H3
70 S=(A*B+B*H+H*A)*2
80 PRINT "A="; A
90 PRINT "B="; B
100 PRINT "H="; H
110 PRINT
120 PRINT "S="; S
130 PRINT
140 PRINT
150 NEXT H, B, A
160 END

```

(3) 转移构成循环 除上述几个专门循环语句可实现重复操作外, 也可以运用转移语句来构成循环。

当转移语句所指定转移的行号, 小于此语句的行号时, 就可以重复地执行这两个行号之间的程序段(各有关语句), 形成局部循环。若要从这段循环中转出来, 可在该程序段中编入条件转移语句或选择转移语句。这样就可以构成一种有“入口”, 也有“出口”的另外一种循环。有如下几种类型:

```

①  :
   :
20  :
   :
80  IF.....THEN 130
   :
   :
120 GOTO 20
130
   :
   :

```


例15: 计数器程序

```
10 Q = 0
20 Q = Q + 1
30 PRINT "Q="; Q
40 IF Q = 100 THEN 60
50 GOTO 20
60 PRINT "Q="; Q
70 PRINT "END"
80 END
```

例16: 累加器 (计算总和) 程序

```
10 T = 0
20 INPUT "X="; X
30 T = T + X
40 PRINT "X="; X, "T="; T
50 IF X = 0 THEN 70
60 GOTO 20
70 PRINT "CUMULATION:"; T
80 END
```

②

```
⋮
⋮
50
⋮
⋮
100 IF.....THEN 200
⋮
⋮
150 IF.....THEN 50
⋮
⋮
200
⋮
⋮
或
⋮
⋮
40
⋮
⋮
100 IF.....THEN 40
```

③

```

:
:
110
:
:
:
:
20
:
:
50 ON.....GOTO.....110.....
:
:
100 GOTO 20
110

```

④

```

10
:
:
100 ON.....GOTO.....250.....200.....
:
:
150 IF.....THEN 10
:
:
200
:
:
250
:
:

```

例17: 求正方体总表面积 (边长的起点值L1与终点值L2及间隔值L3不定, 运行时才确定并输入)

```

10 INPUT "L1="; L1
20 INPUT "L2="; L2
30 INPUT "L3="; L3
40 L=L1
50 A=L^2*6

```

```
60 PRINT "L="; L, "A="; A
70 L=L+L3
80 IF L>L2 THEN 100
90 GOTO 50
100 END
```

习 题

1. 设 $Y = \sin(2X)$, 当给出任何一个 X 值后, 电脑能立即指出 Y 是正值、负值、还是零。要求用条件语句编程序, 并且先画出框图, 然后按框图编出程序。
2. 编出求不同半径, 不同高度时圆柱体体积的程序。
3. 试把例17的程序改为只用一个条件转移语句, 不用无条件转移语句而构成循环。

第十讲 程序设计 (二)

——主程序与子程序

在一个处理问题的程序中，可能要重复地使用到某些操作，可以把这些操作单独地编成一段程序，这种具有专门功能的一段程序，称为子程序。而整个处理问题的程序称为主程序（也可以把调用子程序的程序看作主程序）。在运行程序中，可用语句调用子程序，用完后，再用语句返回到主程序，接下去继续运行。

1. 无条件转子程序与返回主程序

若需要无条件地转到子程序去执行，执行完后自动返回主程序继续运行，可用如下命令。

转子程序命令 GOSUB
返主程序命令 RETURN

(1) 基本用法

①必须把主程序及所需调用的子程序一齐存入内存中，要清楚各个子程序的入口地址。子程序的入口地址，往往是第一个语句的程序行号。

②在主程序中必须编入调用子程序的语句——转子语句，语句格式如下：

行号 GOSUB 子程序入口行号

此外，在主程序的末尾必须编入程序结束语句END。

③在子程序段的最后一个程序行（语句），必须有返回语句——返主语句，格式如下：

行号 RETURN

如：主程序 { 10
 :
 :
 40 GOSUB 1000 (转子语句：调用入口行号
 为1000的子程序)
 50
 :
 :
 100 END

子程序 { 1000
 :
 :
 1050 RETURN (返主语句：返回到主程序50行继
 续执行)
 :
 : }

转子语句与返主语句必须在同一个程序中成对使用，缺一不可。

在转子语句中，GOSUB是GO SUBROUTINE（转到子程序去）的简写，是转子程序的命令。其后面必须给出具体的程序行号，此行号是立即要调用的语句在子程序段中的行号，叫入口行号，不一定是子程序中的第一个行号。执行转子语句后，在堆栈式的寄存器“顶部”记下刚才执行的转子语句的下一个主程序行号。

在返主语句中，只有一个命令 RETURN（返回）。这是一个逐字键入的命令（不是按↵键!）。此语句在被调用的子程序中，是在用完子程序段而将要返回主程序时的一个语句，但不一定是整个子程序的最后一个语句。执行此语句后，将从堆栈寄存器的“顶部”取得一个即将接下去执行的程序行号。

上述语句的执行情况，如图10·1所示。

```

例：  10 HOME
      20 PRINT  " *—*—*—*—*—*—*—*—*—* "
      30 GOSUB  100
      40 PRINT  "$—$—$—$—$—$—$—$—$—$ "
      50 END
      100 REM  SUB—1
      110 PRINT
      120 PRINT  " □□□□#####□□□□ "
      130 PRINT  " □□□□#"
      140 PRINT  " □□□□#"
      150 PRINT  " □□□□#"
      160 PRINT  " □□□□##### "
      170 PRINT  " □□□□#"
      180 PRINT  " □□□□#"
      190 PRINT  " □□□□#"
      200 PRINT  " □□□□##### "
      210 PRINT
      220 RETURN
  
```

例1： 求 5! + 10! 的和。

```

10 N = 5
20 GOSUB 1000
30 X = K
40 N = 10
  
```

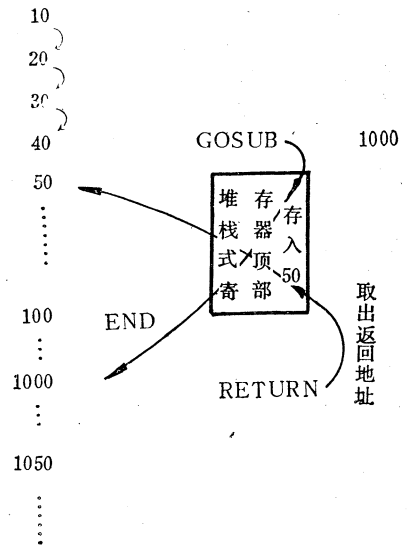


图 10·1

```

50 GOSUB 1000
60 PRINT X + K
70 END
1000 K = 1
1010 FOR I = 1 TO N
1020 K = K * I
1030 NEXT
1040 RETURN

```

(2) 子程序嵌套 在被调用的子程序中，可以再调用另一个子程序（如APPLE实数BASIC允许接递着调用24次子程序，即可以嵌套24个子程序）。在使用子程序嵌套时，必须注意层次分明，有转出就要有返回。即用一次GOSUB，就要有一个RETURN与其对应。

如：主程序

```

      ⋮
50 GOSUB 200
60      ⋮
70      240 GOSUB 300
      ⋮
      250      ⋮
99 END      360 GOSUB 400
      ⋮
      280 RETURN 370
      380 RETURN 480 GOSUB 600
      ⋮
      490      ⋮
      500 RETURN      650 RETURN

```

若使用了GOSUB而无相应的RETURN，则不能返回主程序去运行。若没有GOSUB命令，却多出了RETURN，语句则会出现出错信息？RETURN WITHOUT GOSUB ERROB。

主程序与子程序嵌套如图10.2所示。

```

例： 10 PRINT "1—$"
      20 GOSUB 500
      30 PRINT "2—$"
      40 GOSUB 600
      50 PRINT "3—$"
      60 GOSUB 700
      70 PRINT "井—5"
      80 PRINT "*END*"
      90 END

```

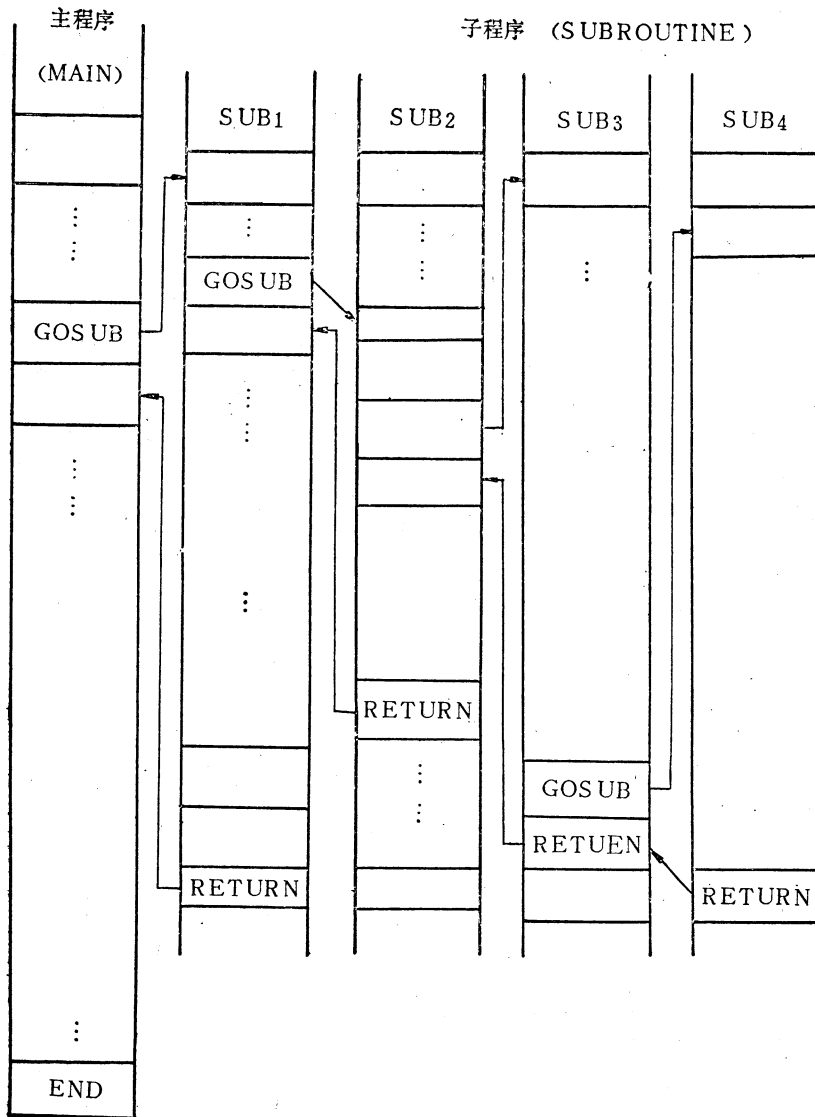


图 10.2

```

500 REM SUBROUTINE—1
510 PRINT "$—1"
520 GOSUB 600
530 PRINT "$ $"
540 RETURN
600 REM SUB—2
610 PRINT "*—2"
620 RETURN
700 REM SUB—3
710 PRINT "*—*—*"

```

```

720 GOSUB 500
730 PRINT "RETN—井—5"
740 RETURN

```

(3) 消除返回地址的命令—POP 这种命令的作用是把刚执行过（最近的一次执行）GOSUB而在堆栈中“记下”的返回地址（行号）去掉。

语句格式： 行号 POP

```

如：  30 GOSUB 300
      40      350 GOSUB 1000
      360      1050 POP
      1100 RETURN
      390 RETURN

```

1050 POP的作用是去掉360行这个返回地址，而使1100行的RETURN返回到40行。

2. 选择转子程序与返回主程序

根据给定的条件，有选择地转到某个子程序去执行，执行完后再返回主程序接下去继续运行，可用如下命令。

选择转子程序命令 ON GOSUB

返回主程序命令 RETURN

使用这些命令在程序中组成语句的规则及语句的作用，基本上与选择转移(ON GOTO)及无条件转子与返主(GOSUB……RETURN)，两种语句作用的综合相同，这里不赘述。其中，在主程序中（同样可在子程序中）设置的选择转子程序语句中的GOSUB，既有GOTO的作用，又有在堆栈中存入返回地址的作用。

选择转子语句的格式：

行号 ON 数值或表达式 GOSUB 行号1，行号2，行号3

```

如： 100 ON 3 GOSUB 1000, 500, 700, 600
      150 ON X GOSUB 700, 1000, 600, 500, 800
      180 ON 4 * S + B / 2 GOSUB 500, 600, 700, 800, 1000, 2000

```

(1) ON后面可给出具体的数值或已有赋值的变量或可算出具体数值的表达式。数值只能在0~255内。数值对选择转移的作用与前面的选择转移语句中相同。

(2) GOSUB是个命令，与ON搭配再构成语句后，仍保留其作用（与转子语句中相同）。在其后要编排出所要选择转去执行的各个子程序入口行号。

3. 出栈转移和返回重做

在运行程序中，如有错误，一般是中断运行，显示出错误信息。在系统内，还会将错误代码存入十进地址为222的存贮单元。

若要使程序运行中出现错误时不致于停顿下来，而是能够让程序自动去处理错误并继续运行下去，可运用如下语句。

(1) 出错转移语句： 行号 ONERR GOTO 行号

(2) 返回出错行重做语句：行号 RESUME

一般把出错转移语句编在主程序的开头。在GOTO后面通常是处理错误的子程序（也可以是别的子程序或语句）入口行号。

返回出错行重做语句，必须在出错转移语句之后，一般都是放在处理错误的子程序末尾。若未执行过出错转移语句而执行RESUME命令，会出现 ? SYNTAX ERROR IN 65278 错误信息或其它奇异的情况。

如果在整个程序中包含有处理错误的子程序（请注意：在子程序末尾要有RESUME语句），又在主程序中编入（最好是在开头）出错转移语句，则当运行程序中出现错误（体系所能辨认的）时，就会转到所指定的行号接着运行。执行到RESUME命令后，自动返回到出现错误的语句去重新再执行。

例： 10 ONERR GOTO 30
 20 X=SQR(-2)
 30 PRINT PEEK(222)
 RUN ↵
 53 （数值超界——数值不符合规定的错误代码）

例2： 求10以内各个数的倒数。
 10 INPUT "X=" ; X
 20 ONERR GOTO 100
 30 PRINT "1/" ; X ; "=" ; 1/X
 40 IF X<10 THEN 10
 50 END
 100 PRINT PEEK(222) ; "——" ;
 110 PRINT "DIVISION BY ZERO——"
 120 PRINT "ERROR!"
 130 X=X+1
 140 RESUME

运行后，试键入9、8、7、6、5、0、2、3、4、10。

(3) 显示正在运行的行号命令——TRACE

语句格式：行号 TRACE

执行这个命令后，会将执行着的程序行号显示出来并在行号前面加井号。

(4) 消去显示行号命令——NOTRACE

语句格式：行号 NOTRACE

这个命令的作用是将前面设置过的TRACE命令取消，恢复为正常的运行显示。

4. 程序设计举例

举一例计算职工工资总数及其中各种面额钞票张数分类统计程序的例子，作为前一段内容的小结。

(1) 基本步骤

① 出错处理

② 职工工资的输入：运行中键入；元为整数，角、分为小数；以键入0为终止输入。

③钞票面额分类：10元、5元、1元、5角、1角、5分、1分。

④10元以下各类钞票数分检：调用子程序。

⑤输出：荧屏显示（或加打印机印出）。

（2）变量的选用

①结论性变量

各个职工工资： S

工资总数： Q

10元票总数： A 1

5元票总数： B 5

1元票总数： B 1

5角票总数： C 5

1角票总数： C 1

5分票总数： D 5

1分票总数： D 1

②主要的中间变量

A：有关10元票数

X：各个职工工资额数及分检（抽出）后的余额

W：各个职工工资中各类票的张数

K：为把角、分（小数）“放大”为元（整数）的系数

（3）参考程序

```
1  ONERR GOTO 350
5  INPUT "S=" ; S
10 PRINT
15 IF S=0 THEN PR#1:GOTO 110
20 Q=Q+S
25 A=S/10
30 A0=INT(A)
35 A1=A1+A0
40 REM COUNT FIVE AND ONE YUAN
45 K=1:X=S
50 X0=10*A0
55 GOSUB 200
60 B5=B5+W5:B1=B1+W1
65 REM COUNT FIVE AND ONE JIAO
70 K=10
75 GOSUB 200
80 C5=C5+W5:C1=C1+W1
85 REM COUNT FIVE AND ONE FEN
90 K=100
```

```

95 GOSUB 200
100 D5 = D5 + W5 : D1 = D1 + W1
105 GOTO 5
110 PRINT "Q = " ; Q
115 PRINT "A1 = " ; A1
120 PRINT "B5 = " ; B5
125 PRINT "B1 = " ; B1
130 PRINT "C5 = " ; C5
135 PRINT "C1 = " ; C1
140 PRINT "D5 = " ; D5
145 PRINT "D1 = " ; D1
150 END
200 REM COUNT FIVE AND ONE SUBPROGRAM
210 W1 = 0 : W5 = 0
220 W = X - X0
230 W0 = INT (W * K)
240 IF W0 < 5 THEN 255
245 W5 = 1 : W1 = W0 - 5
250 GOTO 260
255 W1 = W0
260 X = W
270 X0 = W0 / K
280 RETURN
350 PRINT PEEK (222)
360 LIST
370 RESUME

```

运行后键入各职工工资数。

- ①试键入字符，看看运行情况。
- ②试键入“非法”值（如算式），太大的值，看看运行情况。
- ③在键入正常的工资数值后，分析运行的结果。

习 题

1. 写出下列程序运行过程中，执行语句标号的顺序

```

10 REM MAIN PROGRAM
20 PRINT "$$$$ 1"
30 GOSUB 100
40 PRINT "$$$$ 2"

```

```

50 GOSUB 200
52 PRINT "$$$$ 3"
54 GOSUB 300
56 PRINT "?????"
60 END

100 REM SUB-1
110 PRINT "**** 1"
115 GOSUB 200
120 RETURN
200 REM SUB-2
210 PRINT "**** 2"
220 RETURN
300 REM SUB-3
310 PRINT "**** 3"
315 GOSUB 100
320 RETURN

RUN

$$$$ 1
**** 1
**** 2
$$$$ 2
**** 2
$$$$ 3
**** 3
**** 1
**** 2
?????

```

2. 写出下列程序的打印结果。

```

10 FOR I= 1 TO 5
20 GOSUB 50
30 NEXT I
40 GOTO 70
50 PRINT I;
60 RETURN
70 END

```

第十一讲 程序设计 (三)

——数 组

1. 数组的意义

将一批具有相同性质的数据，按照一定的方式编排成一组，就叫做数组。用数组来表示互相关联的数据是很方便的。在使用电脑处理数据的时候，若把数据做成数组，可以大大简化程序。

(1) 数组的名称和注标 表示数组要用名称和注标。名称用来区分不同的组；注标用来区分同一个组中的不同数据，也可以用来表明该数组中有多少个数据。

在APPLE的实数BASIC中，可用来做数组的名称的词，同简单变量的用法一样，如：

A、……Z； A0、……A9；

AB、……ZA； ABC、……ZXY；

A\$、……Z\$； A0\$、……A9\$；

AB\$、……ZA\$； ABC\$、……ZXY\$等（保留字除外）。

数组的注标，用（ ）号内的十进制数字、变量、表达式或注标变量表示，正整数有效。数组是几维，就有几个注标。

例如：

一维数组：DA(19) ， NA\$(24)

二维数组：M5(24, 39) ， F8\$(26, 20)

三维数组：N(9, 14, 19)

⋮

N维数组：D(2, 4, 3, ……)

N个数值

在APPLE的实数BASIC体系中，数组的维数可多达88个。当然，实际能存贮怎样的数组，还得由内存容量决定。

(2) 数组与注标变量 数组代表整个数据组，它表明该数据组的名称，数据编排方式及数据容量（个数）。如DA(29, 34)中，如DA(29, 34)中，简单变量名DA表示名称；括号内的注标有两个，表示它是二维数组，数据是按二维方式编排的。该数组所能用到最大注标值叫维限。

注标变量代表数组内的某一个具体数据。如DA(10, 15)中，简单变量名DA表明是在某个数组DA内的数据。

括号内的注标，可用0至维限中的任一个值，由这些值表明是该数组内的某一个具体数据，如上例中的是位于X坐标（列）为10，Y坐标（行）为15的数据（见图11.1）。

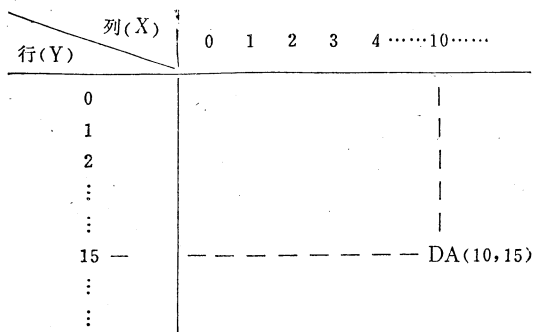


图 11.1

2. 数组的建立

(1) 设定数组命令——DIM

语句格式：行号 DIM 数组1，数组2，数组3

例：10 DIM A(14)，D(9,19)，K\$(10)

上述语句的作用是表明将要建立的数组名称、数据标志及其容量、要预留存贮空间的大小。一般可在一个语句中设定多个数组。

将某个数组括号内的各个注标加1后相乘

的积就是该数组内所含的数据个数，即注标变量个数。如A(14)，含有14+1个数据，分别是A(0)，A(1)，……，A(14)。

又如D(9, 19)，含有(9+1) × (19+1) = 200个数据，分别是：

D(0,0)，D(0,1)，D(0,2)，……，D(0,19)
 D(1,0)，D(1,1)，D(1,2)，……，D(1,19)
 ⋮
 D(9,0)，D(9,1)，D(9,2)，……，D(9,19)

要建立一个数组，首先应选定数组名。要注意数值型数据的数组，用数值型简单变量。字符型数据的数组，用字符型简单变量。

第二要确定维数和维限，定出注标个数及注标值。使用命令DIM及将要设立的数组，构成语句，再编定一个程序行。程序行可在具体置入数据前的任一位置。

(2) 置入数据 将各个具体数据分别赋予对应的注标变量，可运用置存数据与读赋数据语句及循环语句等编制出程序去实现。基本程序行及语句如下：

```
DIM.....
FOR I= 0 TO N
  ⋮
  READ.....
  ⋮
NEXT.....
DATA.....
```

具体的编程则要依数据的编排而定。如要建立二维的数组就要二重循环，三维数组要用三重循环等等。

几点说明：①对于不超过11个注标变量的一维数组（所含数据不超过11个）及不超过121个注标变量的二维的数组（所含数据不超过121个），可以不用设定数组的语句，而直接将数据置入赋予各对应的注标变量）。

②在设定数组时，系统预留一定的存贮单元；一个存贮单元可存一个字节，以便存放各有关的信息。其中：

数组名称： 2个字节

数组容量： 2 个字节

数组维数： 1 个字节

每个注标变量（实数）： 5 个字节

字符串数组的变量： 3 个字节

③在同一个程序中，可以在后面改变前面设定过的数组，如维数，维限。但新设定的数组容量不得超过原数组的容量。

例 1： 把40个数据做成一维的数组。

```

10 DIM H (39)
20 FOR I= 0 TO 39
30 READ H (I)
40 NEXT I
50 DATA 1.1, 1.3, 1.5, 1.2, 1.4, 1.9, 2.3, 2.9
60 DATA 3.5, 3.7, ..... .....4.9
70 DATA 5.3, 5.5, ..... .....7.9

```

<共四十个数据>

例 2： 将五个工厂、三种产品的产量，分别按厂及产品统计总量。

先建立数组：

工厂——FACTORY: F

产品——PRODUCTION: P

数组名: B

维数: 二维 (以列X和行Y标定数据)

数组: B (2, 4) 是一个三列五行数组

编出程序

```

10 INPUT "F=" ; F
20 INPUT "P=" ; P
30 DIM B (P, F)
40 FOR I= 0 TO F
50 FOR K= 0 TO P
60 READ B (K, I)
70 NEXT K
80 NEXT I
90 DATA 20, 30, 26, 30, 20, 25,
100 DATA 25, 50, 20, 46, 15, 10
110 DATA 35, 15, 12
120 END

```

	产品 (P)	1	2	3
厂名 (F)				
1		20	30	26
2		30	20	25
3		25	50	20
4		46	15	10
5		35	15	12

	列 X, (P)	0	1	2
行 Y, (F)				
0		B(0, 0)	B(1, 0)	B(2, 0)
1		B(0, 1)	B(1, 1)	B(2, 1)
2		B(0, 2)	B(1, 2)	B(2, 2)
3		B(0, 3)	B(1, 3)	B(2, 3)
4		B(0, 4)	B(1, 4)	B(2, 4)

运行时，键入 F = 4, P = 2。

3. 数组的运用

数组在编程处理含有大量相互关联的数据的问题时，很有用。

(1) 适用范围

- ①处理大批的数据；
- ②需要多重注明的数据的调用；
- ③需要作分类统计的大量数据；
- ④需要相互之间进行比较，选择的大量数据；
- ⑤要重新按某些要求编排的一批数据。

(2) 数组内数据的调用 数组内的数据，即是注标变量，也称为数组元素。对于已经建立起的数组，要调用其内的数据，是很方便的。只要给出数组的名称及注标——注标变量，它就代表了由变量名所指定的数组，由括号内的注标所标定的某个具体数据。其中，注标可直接给出具体的数值，也可用已有赋值的变量，或用能够算出具体数值的表达式，也还可以用注标变量。

例： 接上述的例2。

当要用到1行2列(X为2, Y为1)的数据时，由B(2, 1)就代表25。

或B(1, 3)就代表3行1列(X为1, Y为3)的数据15。

例3： 编出运用数组来做产品分类统计的整个程序。

```

10 INPUT "F=" ; F
20 INPUT "P=" ; P
30 DIM B (P, F)
40 FOR I= 0 TO F
50 FOR K= 0 TO P
60 READ B (K, I)
70 NEXT K, I
80 FOR I= 0 TO F
90 S1= 0
100 FOR K= 0 TO P
110 S1= S1+ B (K, I)
120 NEXT K
130 PRINT "FACTORY--" ; I+ 1 ; "：" ; S1
140 NEXT I
150 PRINT
160 PRINT
170 FOR K= 0 TO P
180 S2= 0
190 FOR I= 0 TO F
200 S2= S2+ B (K, I)
210 NEXT I
220 PRINT "PRODUCTION--" ; K+ 1 ; "：" ; S2
230 NEXT K
240 END
250 DATA 20, 30, 26, 30, 20, 25

```



```
260 DATA 25, 50, 20, 46, 15, 10
```

```
270 DATA 35, 15, 12
```

例4：将运动员按成绩排出名次（10名运动员——10个号码）。

```
10 INPUT "N=" ; N
```

```
20 Q = 0
```

```
30 PRINT
```

```
40 DIM M$(N), X(N)
```

```
50 FOR K = 1 TO N
```

```
60 READ M$(K), X(K)
```

```
70 IF M$(K) = "END" THEN 100
```

```
80 Q = Q + 1
```

```
90 NEXT K
```

```
100 PRINT "ORDER-I", "NUMBER-", "RESULZ-X"
```

```
110 FOR K = 1 TO Q
```

```
120 FOR J = K TO Q
```

```
130 IF X(K) < X(J) THEN 160
```

```
140 T = X(K) : X(K) = X(J) : X(J) = T
```

```
150 T$ = M$(K) : M$(K) = M$(J) : M$(J) = T$
```

```
160 NEXT J
```

```
170 PRINT K, M$(K), X(K)
```

```
180 NEXT K
```

```
190 DATA N-207, 14.5, N-156, 14.2, B-453, 15.1, C-96, 15.7
```

```
195 DATA M-339, 14.9, P-77, 15.1
```

```
200 DATA C-231, 14.7, B-176, 13.9, P-122, 13.7, M-302
```

```
205 DATA 14.5END, - 1
```

```
210 END
```

例5：某单位有4位检测员（分别姓Li, Lu, Zhou, Pan）每天工作三段时间（上午、下午、晚上）每人各段时间检测样品数如下表所示。

姓	上午	下午	晚上
Li	8	12	7
Lu	13	14	6
Zhou	17	9	4
Pan	11	15	5

每人各段时间检测样品数如下表所示。

若检测一件收费2.54元，试编出程序：每天各人检测样品的件数多少？每天各人可收费多少？每天单位共检测的件数多少？每天单位共可收费多少？按各人收费的多少排出名次。

编出程序如下（参考）：

```
10 INPUT "NANO=" ; N : INPUT "TIME=" ; T
```

```
20 INPUT "K=" ; K
```

```
30 DIM A$(N+1), B(N+1, T+1), C(N+1), D(N+1)
```

```
40 FOR I = 1 TO N
```

```
50 READ A$(I)
```

```

55 C = 0 : D = 0
70 FOR J = 1 TO T
80 READ B (I, J)
90 C = C + B (I, J)
100 D = D + K * B (I, J)
110 NEXT J
120 C (I) = C
130 D (I) = D
135 PRINT A$(I) ":", C, D,
140 E = E + C
150 F = F + D
160 NEXT I
165 PRINT
170 PRINT "CUM:", E, F
190 DATA "LI", 8, 12, 7, "LU", 13, 14, 6, "ZHOU",
        17, 9, 4, "PAN", 11, 15, 5, 0, -1, -1, -1
210 PRINT
220 PRINT
300 PRINT "ORDER:", "NAME:", "RESULT:"
301 PRINT
305 FOR I = 1 TO N
310 FOR J = 1 TO N
320 IF D (I) > D (J) THEN 390
330 M = D (I)
340 D (I) = D (J)
350 D (J) = M
360 S$ = A$ (I)
370 A$ (I) = A$ (J)
380 A$ (J) = S$
390 NEXT J
420 PRINT I, A$ (I), D (I)
430 NEXT I
440 END

```

运行后，键入 $N = 4$ ， $T = 3$ ， $K = 2.54$ 。

(3) 上机练习 将如下三题编出程序后，上机练习。

①将例2、例3改为五列三行的二维数组再作分类统计。

②将例4的数据提供改为运行后输入，并把实际参加的运动员人数改为15个，再按成绩排出名次。

③仿照例5，编出一个车间全月的产量产值统计的程序。

习 题

1. 用双注标变量和READ/DATA语句,把下列数字放入S数组中,写出一个程序,并将运行结果,按一定格式打印出来,并列出S(2, 2)、S(2, 7)、S(3, 1)、S(1, 3)各数组元素的值。

```
1  3  5  7  9 11 13 15
2  4  6  8 10 12 14 16
11 13 15 17 19 21 23  2
5  22 24 26 28 30 32 34
```

2. 写出下列程序运行的结果:

```
10 DIM A(2, 3)
20 FOR R=1 TO 2
30 FOR C=1 TO 3
40 LET A(R, C) = R + C
50 PRINT A(R, C); " ";
60 NEXT C
70 PRINT
80 NEXT R
90 END
```

3. 写个程序,将A数组改写成B数组:

A

$$\begin{pmatrix} 1 & 2 & 3 & 4 \\ 5 & 6 & 7 & 8 \end{pmatrix}$$

B

$$\begin{pmatrix} 1 & 5 \\ 2 & 6 \\ 3 & 7 \\ 4 & 8 \end{pmatrix}$$

第十二讲 程序设计 (四)

——字符处理和光标(显示)定位

1. 字符处理

作为信息资料,不但有数值性的而且有文字性的数据。现在通用的微电脑除了能作数值计算外,还要求可以处理文字。而一般的扩展BASIC都是按照这个要求来设计的。所以,处理字符也是微电脑的一种重要功能。利用这些功能可以设计出程序来处理文字资料。

(1) 字符串的意义 字母,数字,符号的组合就叫做字符串 (STRING)。字符串也是一种数据,叫字符型数据。它可以是英语词句,专用的号码,汉字等等。许多处理数值型数据的命令,语句同样适用于字符串的处理,只要把数值型数据换为字符串就行了。

另外还有一些专门用作处理字符串的命令。这些命令大都跟求数学函数值的命令一样不能单独使用,而要与其它命令组成语句联用,也称为函数。

(2) 字符串的输入——字符型数据的提供 凡是用作提供数值型数据的命令或语句都可以使用来作字符串的输入。

①直接赋值: A\$ = "NAME"

D7\$ = "BOOK—M12—5037"

字符串常量要采用“ ”内给出,在“ ”号内可允许0~255个字符(不同机型可能有所不同)。

②运行中输入: GET A\$

INPUT A4\$, BN\$

键入时不要加引号。

③置存数据与读赋数据语句:

READ F6\$, PA\$, MN\$

DATA BOOK, "N-103", "GOOD BY!"

如带有某些特别符号(如·、—、?、!、空格等)需要在“ ”内给出。

(3) 字符串的处理

①调用字符串 凡已赋值的字符串,可用字符型简单变量来代表该字符串,而对已建立了数组的字符串,则用字符型注标变量。在程序中需要用到某个字符串时,只要给出其对应的变量就行。

②合并(加法) 用加号(+)可将两个或多个字符串合并为一个新的字符串,但合并后不能超过255个字符。如“OVER”+“DUE”变为“OVERDUE”在“ ”内若有空格照样保留。

如果要将三个字符串A\$, B\$及D\$合并为一个字符串,可用下式表达:

C\$ = A\$ + B\$ + D\$

③计算长度——LEN(字符串) 命令(函数)LEN(“ ”)或LEN(A\$)要与其它

命令联用，通常与PRINT联用。

例如： PRINT LEN ("GOOD MORNING!") ↵

13

又如： 10 B\$ = "OK!"

20 PRINT LEN (B\$)

RUN ↵

0

3

④取出部分字符 下面有几个可用于从给定的字符串中取出字符的命令（函数）（不能单独使用）。

1) 取出最左边的几个字符——LEFT\$ (字符串, 数值)

如： 10 A\$ = "GOOD MORNING!"

20 PRINT LEFT\$ (A\$, 4)

RUN ↵

GOOD

2) 取出最右边的几个字符——RIGHT\$ (字符串, 数值)

如： 10 A\$ = "GOOD MORNING!"

20 PRINT RIGHT\$ (A\$, 8)

RUN ↵

MORNING!

3) 从第 n 个起取出 m 个字符——MID\$ (字符串, 数值, 数值)

命令（函数）格式：MID\$ (A\$, n , m ,)

若要从第 n 个起到最后全取出：MID\$ (A\$, n)

如： 10 A\$ = "GOOD MORNING!"

20 PRINT MID\$ (A\$, 5 3)

30 PRINT MID\$ (A\$, 5)

RUN ↵

MOR

MORNING!

⑤字符串的比较

1) 长度不同为不相等，字符个数多的为大。

2) 字符个数相同，则依次逐个比较，顺序不同为不相等。

字母大小，数字大小，符号大小按ASCII (American Standard Code for Information Interchange) 美国信息交换标准代码的缩写码与字符对照表(见表12.1)的数值而定。

如

$A < B < C \dots < Z$

$0 < 1 < 2 \dots < 9$

⑥转换 下面有几个可用来作字符与数字之间转换的命令(函数),这些命令不能单独使用。

1) 将括号内字符串的第一个字符转换成ASCII码(见表12.1), 可用ASC (字符串), 即ASC ("字符") 或ASC (A\$)。如:

④ PRINT ASC ("APPLE") ↵

65

⑤ 10 A\$ = "APPLE"

20 PRINT ASC (A\$)

RUN ↵

65

2) 将括号内的ASCII码值转换成字符, 可用CHR\$(数值)。

如 PRINT CHR\$(65) ↵

A

使用此命令, 可以等效于按键。如CHR\$(4) 等效于CTRL-D。

3) 将括号内的数值(包括表示数值的符号)转换为字符串, 根据其数值的大小转换成原样的数字串或浮点数形式的字符串, 可用SLR\$(数值)。如:

④ PRINT STR\$(999999999) ↵

999999999

⑤ PRINT STR\$(1234567891234) ↵

1.23456789E+12

4) 将括号内的字符串转换成数值可用VAL(字符串)。这里的字符串应是表示数值的数字及专用符号(10的幂次号E、小数点·、正号+、负号-等)所组成的(即表示数值的字符串), 才会被转换。如遇字符串中有上述以外的字符, 则从该字符起不再转换。这个命令与STR\$(数值)是互为反转的。如:

④ PRINT VAL("58") ↵

58

⑤ PRINT VAL("-9.6E4") ↵

-96000

⑥ 10 X\$ = "27" : Y\$ = "31"

20 PRINT VAL(X\$ + "." + Y\$)

RUN ↵

27.31

⑦ PRINT VAL("7250 BEL3") ↵

7250

各个字符的代码值参看ASCII码与字符对照表(表12.1)。

表12-1

ASCII 码与字符对照表

ASCII 码	显示字符	对应键
H	D	CTRL-
00	0	CTRL-
01	1	CTRL-
02	2	CTRL-
03	3	CTRL-
04	4	CTRL-
05	5	CTRL-

续表

ASCII 码	显示字符	对应键
06 6		CTRL-
07 7	(bell)	CTRL-
08 8	(backspace)	CTRL-
09 9		CTRL-
0A 10	(linefeed)	CTRL-
0B 11		CTRL-
0C 12		CTRL-
0D 13	(carriage return)	CTRL-
0E 14		CTRL-
0F 15		CTRL-
10 16		CTRL-
11 17		CTRL-
12 18		CTRL-
13 19		CTRL-
14 20		CTRL-
15 21	(forwardspace)	CTRL-
16 22		CTRL-
17 23		CTRL-
18 24	(cancelline)	CTRL-
19 25		CTRL-
1A 26		CTRL-
1B 27		ESC
1C 28		n.a
1D 29		CTRL-SHIFT-M
1E 30		CTRL-Λ
1F 31		n.a
20 32	Space	
21 33	!	!
22 34	"	"
23 35	#	#
24 36	\$	\$
25 37	%	%
26 38	&	&
27 39	,	,
28 40	((
29 41))
2A 42	*	*
2B 43	+	+
2C 44	,	,
2E 45	-	-
2F 46	o	o
2F 47	/	/
30 48	φ	φ
31 49	1	1

续表

ASCII 码	显示字数	对 应 键
32 50	2	2
33 51	3	3
34 52	4	4
35 53	5	5
36 54	6	6
37 55	7	7
38 56	8	8
39 57	9	9
3A 58	:	:
3B 59	;	;
3C 60	<	<
3D 61	=	=
3E 62	>	>
3F 63	?	?
40 64	@	@
41 65	A	A
42 66	B	B
43 67	C	C
44 68	D	D
45 69	E	E
46 70	F	F
47 71	G	G
48 72	H	H
4A 73	I	I
4B 74	J	J
4C 75	K	K
4E 76	L	L
4D 77	M	M
4E 78	N	N
4F 79	O	O
50 80	P	P
51 81	Q	Q
52 82	R	R
53 83	S	S
54 84	T	T
55 85	U	U
56 86	V	V
57 87	W	W
58 88	X	X
59 89	Y	Y
5A 90	Z	Z
5B 91	[n.a
5C 92	\	n.a
5D 93]	SHIFT-M

ASCII 码	显示数字	对 应 键
5E 94	^	^
5F 95	—	n.a

注：①n.a表示APPLE键盘上无对应键；

②H 为十六进制数；

③D 为十进制数。

2. 光标（显示）定位

荧屏上正常显示字符时纵向有24格（1~24），横向40格（1~40）。在荧屏上光标所在的位置，就是接着要显示字符的位置。为了设计输出的形式，可运用下述的几个命令，构成语句对光标定位。

（1）行内间接定位——SPC（数值） 括号内的数值必须在0~255内。可以直接给出具体数值，也可以给出有赋值的变量或表达式。

此命令的作用是使光标从当前的位置起，跳过括号内所给数值的位置，即空出括号内所给数值的空格。

①命令（函数）不能单独作语句使用，通常跟PRINT联用，放在PRINT后面。

②命令可以一个个接连着使用。

如：PRINT SPC (15) ; “*APPLE*” ↵

从第16格开始印出 *APPLE*

PRINT SPC (250) SPC (139) SPC (255) ; “#” 从当前光标所在处起，接连逃过荧屏上16行又4个格再印出#。

（2）行内直接定位（打印格式函数）——TAB（数值） 括号内的数值必须在0~255内。可直接给出具体数值，或以变量、表达式给定。这是一个不能单独作语句使用的命令（函数），通常放在PRINT后面使用。其作用是把光标移到括号内数值所指定的位置上。注意：若括号内的数值为0，则将光标移到第256格（在荧屏上逃过6行又15格）上。

如：PRINT TAB (10) ; “#”

在本行的第10格上印出#。

下列两个语句是等效的：

```
PRINT SPC (10) ; “*”
```

```
PRINT TAB (11) ; “*”
```

例：求学生考试六门功课的总分和平均分，并按名次排列打印出学生的学号和成绩。设各门功课以X1、X2、X3、X4、X5、X6来代表，学生学号以S代表，UM代表每个人六门科合计成绩，AV代表平均成绩，I指名次。

参考程序：

```
80 INPUT N, X
100 PRINT
110 PRINT "I S X1 X2 X3 X4";
120 PRINT "X5 X6 UM AV"
130 PRINT
140 DIM T (N, X) , S(N), A(N), V(N), Y(N)
150 FOR I= 1 TO N
```

```

160 READ Y (I)
170 FOR K= 1 TO X
180 NEXT T(I, K)
190 NEXT K
200 NEXT I
210 FOR I=1 TO N
220 LET S(I)= 0
230 FOR K=1 TO X
240 LET S(I)= S(I)+ T(I, K)
250 NEXT K
260 LET A(I)= S(I)
270 LET V(I)= INT(S(I)/X *10)/10
280 NEXT I
290 FOR I=1 TO N— 1
300 FOR K=I+1 TO N
310 IF A(I)>= A(K) THEN 350
320 LET A1= A(I)
330 LET A(I)= A(K)
340 LET A(K)= A1
350 NEXT K
360 NEXT I
370 FOR I= 1 TO N
380 FOR K=1 TO N
390 IF A(I) < S(K) THEN 440
400 PRINT I;TAB(6); Y(K); TAB(13); T(K, I);
410 PRINT TAB(19); T(K, 2); TAB(25); T(K,3);
420 PRINT TAB(31); T(K, 4); TAB(37); T(K,5);
430 PRINT TAB(43); T(K,6);TAB(49);S(I); TAB(55);V(K)
440 NEXT K
450 NEXT I
470 DATA 2301, 65, 57, 71, 75, 82, 69
480 DATA 2304, 80, 90, 91, 88, 95, 99
490 DATA 2307, 78, 82, 77, 86, 83, 73
500 DATA 2303, 45, 38, 44, 48, 61, 52
510 DATA 2305, 83, 82, 79, 85, 77, 84
520 DATA 2302, 70, 68, 83, 59, 73, 64
530 DATA 2306, 98, 92,100, 97, 90, 97
540 DATA 2308, 85, 73, 80, 77, 83, 86
9999 END

```

150~200语句将学生成绩读入T数组，把学生学号放在Y数组中。

210~280语句运算学生成绩总分并分别放入S和A数组中，学生平均成绩放入V数组。

290~360语句利用A数组将学生成绩按高低排列。

370~450语句把学生成绩按高低分顺序打印出来。

执行结果显示并印出

```
      RUN ↵
      ? 8 ? 6
      I   S  X1  X2  X3  X4  X5  X6  UM  AV
      1  2306 98  92  100 97  90  97  574 95.6
      2  2304 80  90  91  88  95  99  543 90.5
      3  2305 83  82  79  85  77  84  490 81.6
      4  2308 85  73  80  77  83  86  484 80.6
      5  2307 78  82  77  86  83  73  479 79.8
      6  2301 65  57  71  75  82  69  419 69.8
      7  2302 70  68  83  59  73  64  417 69.5
      8  2303 45  38  44  48  61  52  288 48
```

(3) 屏内横向定位——HTAB 数值

语句格式： 行号 HTAB 数值

数值在 0~255内。可直接给出具体数值或用变量或用表达式给定。语句作用是将光标从当前的位置沿水平移到数值所指定的位置上。

数值为 1：光标移到第 1 格（最左端）；

数值为 40：光标移到第 40 格（最右端）；

数值为 55：光标移到第 55 格（屏内下一行离左端 15 格上）；

⋮

数值为 0：光标移到第 256 格（在屏内移到下 7 行的第 16 格）。

(4) 屏内纵向定位——VTAB 数值

语句格式： 行号 VTAB 数值

数值在 1~24内。语句作用是使光标从当前的位置沿纵向移动到数值所指定的位置上。

数值为 1：光标移到第 1 行（屏顶）；

数值为 2：光标移到第 2 行；

⋮

数值为 24：光标移到第 24 行（屏底）。

上述的屏内横向定位和屏内纵向定位两个语句，可分别单独使用，也可前后联用，将光标调到预定位置，再用输出语句，这样可在预定位置上印出指定的内容。

```
例1： 10 HOME
      20 HTAB 10
      30 VTAB 20
      40 PRINT "*"
```

```

例2： 10 HOME
      20 FOR X=1 TO 24
      30 FOR Y=1 TO X
      40 HTAB X
      50 VTAB Y
      60 PRINT "*"
      70 NEXT Y, X
      80 GOTO 20

```

习 题

1. 写一个程序，在荧屏上显示下列图形（习12.1图）。

```

      ?
     ???
    ?????
   ??????
  ???????
 ????????
?????????
?????????
?????????

```

习12.1图

2. 写一个程序，在荧屏上显示以下图形（习12.2图）。

```

V V V V V V V V
 V V V V V V V
  V V V V V V V
   V V V V V V V
    V V V V V V V
     V V V V V V V
      V V V V V V V
       V V V V V V V
        V V V V V V V

```

习12.2图

3. 编出程序，在荧屏上任意显示一条正弦函数的曲线。

第十三讲 华字—200(LASER—200、 LASER—310)的使用与保养

1. 华字—200(LASER—200)微电脑简介

华字—200微电脑，是引进日本电路组装的练习机，是与香港LASER—200家用电脑同样功能的机型。其操作简单，易于维护，价钱低，而用途较广泛，是83年出现的新电脑系统，主机使用8-BIT微电脑中央处理器，配彩色显示器可以显示八种颜色，还配有内存扩展器、游戏操纵杆、光笔等。有较广泛的强效软件支持，包括BASIC自学程序，游戏及家庭电器控制、个人数学、科学、财经、商业等应用程序。

华字—200微电脑是目前优选的练习机，适合作初学的高、初中学、职业中学培训班、技工、中专教育的练习机。它配有最简单MIOROSOFT—BASIC来编写多种不同用途的程序，用户可自行编写高级语言的程序，制作图表，还有可发出悦耳的音乐和做有趣的游戏。

(1) 华字—200微电脑主要技术指标见表13·1。

表13·1 华字—200微电脑主要技术指标

主存储容量	ROM 16K RAM 8K (可扩展至64K)
语 言	MICROSOFT BASIC
显示输出	字符显示 32字×16行 图象字符混合64×32点 (九种彩色) 高分辨图象128×64点 (八种彩色)
键 盘	45个字符功能键；16个可定义字符/图象键，带“的”声键输入。
声音输出	单频道输出
卡式接口器	600 bps baud rate (内置) 接口器可连接录音机
打字机接口板	可接ceatronics用线传送的打印机
传输线扩展	内存记忆及外部存储扩展
显 示	PF输出至家用彩色电视机 (内置)、视频输出至监控器
电 源	DC 9V

(2) LASER—310型微电脑主要技术指标，见表13·2。

(3) 键盘 (见附录)

①华字—200 (LASER—200) 键盘图

②LASER—310键盘图

2. 微电脑系统的启动与保养

表13.2.

LASER—310型微电脑主要技术指标

项 目	内 容
记 忆	ROM 16K (内置)
主存储容量	RAM 18K (内置) 可扩展至66K
语 言	MICROSOFT BASIC
显示输出	字符显示 32×16行 图象: 低分辨图象64×32 (4色) 高分辨图象128×64 (8色) 图象字符混合 64×32 (9色)
键 盘	45键全式QWERTY型, 双SHIFT键及SPACE键, 自动重复效能, 自动输入音响提示
电视讯号	接口已置于主机内
卡式收录机	接口已置于主机内
单磁盘机	通过DI—40磁盘控制连接板, 可与DD—20单磁盘机连接使用。主机上有磁盘机的配接口。
LASIC立即反应光笔	通过LP—20光笔控制连接板, 与LASIC光笔连接使用(带光笔配件, 主机上有接口)。
PP—80 (80—0312) 图象打字机 PP—40 4色 印字描绘机 (80—0682)	通过PI—20控制连接板, 与打字机、描绘机连接使用(带打字机配件, 主机上有接口)
电 源	10V 配有电源变换器

(1) 微电脑的工作场所, 要有良好的卫生条件, 操作人员穿戴要清洁、整齐, 环境温度在0~40℃。

(2) 放置硬件, 避免靠近电动机、变电设备和磁场较强的地方。

(3) 使用前按照随机手册, 连接好各部件, 如RAM、打印机、录音机、磁盘驱动器(RAM在32K以上才用)、显示器、电源连接线。将设备的开关置于关的位置(OFF位置)。

(4) 通电前必须查对电源电压稳定值是否相符, 如LASER—200机为9V, LASER—310机为10V, LASER—2001、LASER—3000机为220V。

(5) 连接家用电视机, 要调至U频道, 电脑主机显示输出线接在电视机天线引入接线柱上。开显示器, 调整微调旋钮, 取得同步、清晰为准。

(6) 开主机电源(置主机开关在ON位置)后, 荧幕显示出READY, 表示电脑准备就绪等待接受命令。(不同机型, 显示符有差别, 请看随机手册, 华字—3000和APPLE II的提示符为]。)这时就可以将编写好的程序, 用键盘操作输入电脑。

(7) 发现故障要立即关机。

(8) 在排除故障中, 如需换集成电路块时, 烙铁不能带电直接焊接, 必须加热后断电再焊接。手必须戴好消除静电的戒指。一般不要直接触摸电路块, 拿电路块的两边, 最好用

不带电的夹具、工具。

(9) 微电脑系统工作完后, 按与开机相反顺序进行关机, 然后切断电源, 拆下连接线, 同时将各硬件, 特别是主机、磁盘驱动器、打印机必须包盖好, 达到防潮、防尘。

3. 打键操作注意事项

(1) 放置微电脑系统的桌子要适合。

(2) 操作员坐的椅子要舒适, 可调节高低。

(3) 操作员姿势要正确, 身体安稳, 两肘稍靠紧身体, 可支承在桌面上, 上体稍向前倾, 头向左斜注视原稿和顾及显示器。总之要自然、舒服。

(4) 指法要讲究, 双手十指各有分工, 熟记键盘上各键分布, 按键要平稳。联用键更要注意同时性, 打键要敏捷, 轻打快提。

习 题

1. 在教师指导下实际联接操作。
2. 熟悉开机与关机操作。

第十四讲 华字-200(LASER-200、LASER-310)

键盘操作(一)

华字-200(LASER-200)微电脑的键盘装在主壳上,有42个方长键,可分为几类(参看附录华字-200及LASER-200键盘图)。

1. 普通键

普通键,即字符键及功能键。

符号/数字 双字符键 如:

1

 !

9

)

符号/符号 双字符键 如:

:

 *

;

 +

符号/字母 两用键 如:

L

 ?

符号/数字或字母,指令语句四用键 如:

TABC

COLOR

K

 '

N

 ↑

PRINT

USING

2. 号符键的作用

+ 表示加号、正号或作字符+用。

例: PRINT 6 + 3 ↵

执行结果:

9

- 表示减法、负号或作字符-用。

例: PRINT 5 - 3 ↵

执行结果:

2

* 表示乘号或作字符*用。

例: PRINT 5 * 2 ↵

执行结果:

10

/ 表示除号或作字符/用。

例: PRINT 20/4 ↵

执行结果:

5

↑ 表示乘方号或作字符↑用。

例: PRINT 2 ↑ 3 ↵

执行结果:

8

分号(;)及逗号(,)用于分隔数据或作标点符号用。

冒号(:)用于句与句的分隔或作标点符号用。

例1: 键入

```

10 FOR I=1 TO 5
20 PRINT I; : NEXT I
RUN↵

```

荧幕显示:

1 2 3 4 5

例2: 键入: “ ” 内的数字照印, 不作运算。

```

10 PRINT "ABC15+2"
RUN

```

荧幕显示:

ABC15+2

3. 比较符(关系符)号键的作用(见表14.1)

表14.1 比较符(关系符)号键的作用

BASIC使用的比较符	含义	相当算术符号
=	等于	=
<	小于	<
>	大于	>
<=	小于或等于	≤
>=	大于或等于	≥
<>	不等于	≠

注: 除=符号外, 只有在条件语句才可以用这些比较符, 其他语句不允许键入比较符。

例:

```

10 IF B>=100 THEN 60 (变量与数值比)
10 IF X>Y THEN 70 (变量与变量比)
10 IF T<A^3 THEN 80 (变量与表达式比较)
10 IF 4*A*Y<'>6*LOG(A) THEN 50 (两个表达式比较)
10 IF X*Y<>0 THEN 50 可以简化为:
10 IF X*Y THEN 50

```

即条件中比较符为“< > 0”时, 在条件语句中的“< > 0”可省略不写。

4. 函数符键的作用

函数包括LASER-3000的非本电脑BASIC原设计所有函数可按下式计算。

1) 求基本三角函数值

SIN (X) 正弦 X应用弧度表示

COS (X) 余弦

TAN (X) 正切

ATN (X) 反正切 $(-\frac{\pi}{2} < X < \frac{\pi}{2})$

2) 求算术函数值

SQR (X) 平方根

ABS (X) 绝对值

EXP (X) “e” 幂乘

INT (X) 取整数

SGN (X) 取符号

FNY (X) 自定义函数符

SEC (X) = 1/COS (X) 正割

CSC (X) = 1/SIN (X) 余割

COT (X) = 1/TAN (X) 余切

ARCSIN(X) = ATN(X/SQR(1 - X * X)) 反正弦

ARCCOS(X) = 1.570796 - ATN(X/SQR(1 - X * X)) 反余弦

ARCSEC(X) = ATN(SQR(X * X - 1)) + (X < 0) * 3.14593 反正割

ARCCSC(X) = ATN(1/SQR(X * X - 1)) + (X < 0) * 3.141593 反余割

ARCCOT(X) = 1.57096 - ATN(X) 反余切

SINH(X) = (EXP(X) - EXP(-X))/2 双曲正弦

COSH(X) = (EXP(X) + EXP(-X))/2 双曲余弦

TANH(X) = (EXP(X) - EXP(-X))/(EXP(X) + EXP(-X)) 双曲正切

COTH(X) = (EXP(X) + EXP(-X))/(EXP(X) - EXP(-X)) 双曲余切

SECH(X) = 2/(EXP(X) + EXP(-X)) 双曲正割

CSCH(X) = 2/(EXP(X) - EXP(-X)) 双曲余割

ARCSINH(X) = LOG(X + SQR(X * X + 1)) 反双曲正弦

ARCCOSH(X) = LOG(X + SQR(X * X - 1)) 反双曲余弦 (X ≥ 1)

ARCTANH(X) = LOG((1 + X)/(1 - X))/2 反双曲正切 |X| < 1

ARCCOTH(X) = LOG((X + 1)/(X - 1))/2 反双曲余切 |X| > 1

ARCSECH(X) = LOG((1 + SQR(1 - X * X))/X) 反双曲正割

ARCCSCH(X) = LOG((1 + SGN(X) * SQR(1 + X * X))/X) 反双曲余割

习 题

键盘操作，输入下列各式并将运行结果写出：

1. $X = 6 + 3$

2. $Y = 6 + 3 - 2$

3. $I = 3 * 2 + 4 / 2$

4. $I = (3 * 2 + 4) / 2$

5. 10 READ R1, R2, R3, R4, U

20 DATA 100, 150, 240, 36, 24
30 LET I=1/(1/R1+1/R2+1/R3+1/R4)
40 PRINT "I=" ; I
50 GOTO 10
60 END
6. 10 PRINT "XM - 2 + 3"

第十五讲 华字-200(LASER-200、LASER-310)

键盘操作(二)

1. 直接按(打)键

在华字-200(LASER-200)微电脑的键盘上直接按(打)42个方、长键,发出“的”声,表示键功能已输入电脑。

例1: 按下列顺序打各键(空格为空格键)

`[1!]` `[φ@]` `[L]` `[E]` `[T]` `[X]`

荧幕显示:

10 LET X

例2: 键入

`[2"]` `[φ@]` `[SPACE]` `[L]` `[E]` `[T]` `[SPACE]` `[A]` `[- =]` `[B]`

荧幕显示:

20 LET A-B

2. 特殊键及其联用

(1) 选择各键上方字符要起作用,先按住`[SHIFT]`键,再打各功能键,则功能键上方符号起作用。

例3: 先按住`[SHIFT]`键,再按:`[+]` `[8(]` `[9)]` `[- =]` `[K']` `[L?]` `[<]`

`[.>]`

荧幕显示:

+ () = / ? < >

(2) 转换控制键`[CTRL]`的联用:

①先按住`[CTRL]`键同时再打所需要的功能键,则功能键正上方标明的字符起作用。

例4: 先按住`[CTRL]`键,同时按`[0LET[`,则LET输入电脑。

荧幕显示:

LET (不显示0或[)

②先按住`[CTRL]`键,同时分别按`[M←]` `[,<]` `[.>]` `[SPACE↓]`则按照箭指方向移动荧

幕上的光标,同时起保护已输入电脑的字符作用。如果不按`[CTRL]`键,单独按`[SPACE]`键,则使荧幕上光标位置移动,按键一次,删除右边一格的字符。如果配合标号,则可删掉整句。配合功能键也可作为修改方法之一来使用。

例5： 设荧幕显示（已输入电脑）

10 RET Y = X + 3 （光标位）

操作：1)先按住`CTRL`键，同时打`M`键，连续九次，置光标在R位上；

2)再键入`L?`（L）；

3)按住`CTRL`键的同时，打`,`键八次使光标移原位。

荧幕显示：（改写输入电脑）为：

10 LET Y = X + 3

例6： 设刚按键后（未按RETURN键），荧幕显示：

10 LET X = 3 + 1 （光标位）要改写为：

10 PRINT "X="；

操作：1)先按住`CTRL`键，同时打`M`键连续数次，置光标于L位置；

2)再按住`CTRL`键，同时打`PRINT` `P1` 键；

3)单独按`SPACE`键一次；

4)按住`SHIFT`键，同时打`2`键一次；

5)打`X`键；

6)按住`SHIFT`键，同时打`-`键；

7)按住`SHIFT`键，同时打`2`键

8)打`;` `+`键

9)10语句改写完毕，校对无错后，键入`RETURN`，则通知电脑输入完毕并存入内存。

（3）插入字符键`INSERT`，其作用是使光标所在位置的字符左移一格而空出一格，按一次移一格。

例7： 设荧幕显示（已输入电脑）

10 PRNT X （光标位）

需要在R、N之间插入I。

操作：1)先置光标在N位置上；

INSERT

2)按住`CTRL`键，同时打`L?`键一次；

- 3)再单独打 `I` 键一次;
- 4)按住 `CTRL` (保护原有字符) 打键使光标移开;
- 5)键入 `RETURE` 则荧幕显示 (已输入电脑) 出:

```
10 PRINT X
```

- (4) `RUBOUT` 键, 作整句删除使用。

例 8: 设荧幕显示 (已输入电脑)

```
50 IF Y < 0 THEN 100
```

```
60 IF Y > 0 THEN 80
```

需要删除整个50号语句。

操作: 1)置光标于 5 字位置;

2)先按住 `CTRL` 键, 同时按 `RUBOUT` 键;

3)键入 `RETURN` ;

4)则删去了50号语句。

(5) `RETURN` 键 (回键) 作命令结束并立即执行或作程序行的结束并存入内存, 及使光标换行使用。一般用 ↵ 表示此键。

(6) `NEW` 清除内存。

(7) `SPACE` 除可作空格、删掉数、字母、字符外, 还可在程序运行 (显示) 中, 第一次按 `SPACE` 作强迫中断, 显示某程序中某行。第二次按 `SPACE`, 程序继续运行。

(8) 打 `END`, 程序结束。

(9) `LIST` 键, 是将已输入的程序, 在荧幕上按标号大小顺序显示。如果要仔细查对或修改某一个或几个语句, 则在 `LIST` 键入后, 再键入所需要的标号。

例 9: 要查10~900号语句中的60~80号语句。设程序是:

```
10 COLOR 8
  ⋮
50 PRINT TAB (20) ; "□"
60 PRINT TAB (10) ; "□" TAB (20) ; "□"
70 PRINT TAB (10) ; "□" TAB (20) ; "□"
80 PRINT TAB (12) ; "□" TAB (20) ; "□"
90 PRINT TAB (10) ; "□" TAB (20) ; "□"
  ⋮
900 END
```

操作: 键入 `LIST` 60 TO 80, 则荧幕显示:

```
60 PRINT TAB (10) ; "□" TAB (20) ; "□"
```

```

70 PRINT TAB (10); "□" TAB (20); "□"
80 PRINT TAB (12); "□" TAB (20); "□"

```

查对后，有错误可以进行修改，要注意使用↵才能有效。

(10) 打印命令PRINT

例10: 输入程序并运行

```

10 PRINT 2
20 PRINT 5
30 PRINT 9
RUN↵

```

荧幕显示:

```

2
5
9

```

例11: 输入程序并运行

```

10 PRINT 4;
20 PRINT 7;
30 PRINT 9
RUN↵

```

荧幕显示:

```

479

```

例12: 输入并运行

```

10 PRINT "A = B + 4"
RUN↵

```

荧幕显示:

```

A = B + 4

```

习 题

写出下列各题的结果。

1. 先按一次 L ? 键，接着按 E 键，再按 T 键，结果是什么？

2. 按住 SHIFT 键，同时按 TAB K 键，结果是什么？

3. 按住 CTRL 键，同时 TAB K 键结果是什么？
POINT

4. 按住 CTRL 键，同时按 M 键，结果是怎样？

第十六讲 磁盘系统的操作

1. 磁盘系统的组成

现以FD—100、FDM—100系列5 1/4英寸磁盘机为例，介绍磁盘系统的组成。

(1) 磁盘驱动器（又称磁盘机）简介：

起动时间 500ms

稳定时间 15ms

转 速 300转/分

磁轨跳越速度：

使用的磁盘是单密度的为12ms

使用磁盘是双密度的为6ms

资料传送速度250位元/秒（一位元即一个信号）

工作环境温度 4~46°C

工作环境湿度 20~80%（不结霜的情况下）

电源电压 5V ± 5%、12V ± 5%

(2) 控制连接板，也称控制板，是一个印刷电路板，可插入APPLE II、LASER—3000的主机板架上。

(3) 连接电缆，是一条带状电缆，用于磁盘驱动器与控制连接板的相连接。

(4) 系统主磁盘，也称主磁盘。

(5) BASIC磁盘，有语言译解程序。

(6) 可用的磁盘有下列几种：

单面（指磁盘只有一面运行使用）单密度，磁迹密度每英寸48条，1个盘面磁轨数为40。

单面双密度，磁迹密度每英寸96条，盘面磁轨数为80。

双面（指磁盘两面都可运行使用）单密度，磁迹密度每英寸48条，一块磁盘磁轨数为80。

双面双密度，磁迹密度每英寸96条，一块磁盘磁轨数为160。

2. 磁盘机的连接方法

(1) 首先断开主机，如APPLE II，LESER—3000的电源。

(2) 把主机后端指与主机键盘距离最远的那一端稍向上提，并往后推一下，使顶盖脱离主机壳。

(3) 主机内电路板的后端，有8个长插座，又叫“沟槽”，从操作者面对键盘向前看为准，最左边的那个沟槽叫# 0沟槽（也称0号），最右边的# 7。控制连接板除绝对不允许插入0号（就是最左边的一个沟槽）外，其他沟槽都可以插（可看随机手册），一般内约定的是插在沟槽6，即是最右边向左数第二个位置。连接前，要用酒精擦干净。

(4) 调整带状电缆,使它平放,通过主机盒后开口之间平坦部分,把盖子合上,将电缆夹住。避免盒外受到拉力时,拉动插座,容易造成接触不良。

(5) 多个磁盘机的连接时,每一个控制连接板,可以连接两台磁盘机(驱动器),其中一台连接到控制连接板上方的DRIVE 1组接脚,另一台则接到控制连接板下方的DRIVE 2组接脚,连接板插入沟槽6中。第3及第4台磁盘机连到沟槽5的控制连接板上,第5及第6台磁盘机连接到沟槽4的控制连接板上。并把每个磁盘机的前端标上它的沟槽号码,以及本系统第几号磁盘机,以便编写程序时,考虑到某些磁盘机的调用。

3. 磁盘机及磁盘的保养

磁盘机是一种机械装置,里面有电动机和可移动的部分、读写头等。是一种比微机主机更容易损坏的机器,操作时必须细心,要防止掉在地上或者受到外物碰击,否则容易损坏。

不将磁盘机放在磁场较强的地方,就是电视机,也要保持2尺以上的距离才好。

软性磁盘是一片小的(直径约5英寸)塑料盘子,它的表面上有磁性材料,使得资料、档案可以存入,取出或消除。外面的这层磁性材料与录音带上的磁性材料相同,磁盘是被封在一个方的黑色塑料套子里,这套子可保护磁盘,使它干净和自由旋转,这套子是禁止打开的。因为磁盘上覆盖物含有润滑剂和清洁剂,对磁盘起保护作用,虽然软磁盘有一些弹性,也禁止过度弯曲,避免造成对磁盘的损害。

操作时,将磁盘从纸袋取出,只能合乎黑色塑料套子部分,禁止任何东西接触到磁盘表面棕色或灰色的区域。不用时,及时将磁盘放入纸袋内。这纸套子能防止由于静电场的建立而吸引灰尘,要尽可能使它立放。

磁盘能储存非常大量的资料,一个磁盘能储存1,146,000单元的资料。单一的一个单元资料在磁卡上占的位置极小,抓伤一点儿甚至一个手指印,都能使资料产生错误。所以必须把磁盘保管好,避免灰尘和外磁场干扰。

在书写磁盘的编号时,不能用太尖的笔,也不能重压。最好是先写好标签,再贴到磁盘套子上面去。

磁盘对过热或过冷都很敏感,不能放在热源旁边,磁盘(正常保管4~48°C)在50°C以上容易受损害,如果表面弯曲或黑色塑料部分隆起,则说明磁卡已损坏。

磁盘使用寿命约为40小时,一般建立登记卡,便于适时读写转录更换新磁盘,以免使已存资料损失。

将磁盘插入或移出磁盘机(驱动器)的操作方法如下:

磁盘机的开门是从下边打开,磁盘有标签的面朝上,有椭圆形口的一端先进入磁盘机。放置时要把磁盘平放滑入,慢慢地、轻轻地将磁盘推入磁盘机内,不能硬推。然后把磁盘机门拉下关上,即可使用。

在移出磁盘时,首先打开磁盘机的门,磁盘机内的读写头便举起,离开磁盘,再轻轻抽出。如果将一片新磁盘放入磁盘机内,需隔数小时才用,最好把磁盘机的门打开,避免读写头长期与磁卡接触。

磁盘机在使用过程中,当红灯亮时,绝对禁止移出磁盘。否则,磁盘将受到不能恢复的损害。

4. 资料或程序存入、调出磁盘

如果需要将贮存在APPLE II、LASER—3000的内存区(RAM)中的程序,转录存在磁盘上,以便获得长久性的记录时,可使用下列命令:

(1) 首先确定RAM的程序是否完整正确,可以先用RUN↵显示查对。

(2) 给定程序名称,例如设名称为STRIPES。

(3) 键入指令——SAVE STRIPES, 这个名称为STRIPES的程序便存入磁盘中。

(4) 如果需要列出(显示)磁盘内已存入的所有资料,可键入CATALOG↵,显示器的荧幕上,即将该磁盘内的所有档案或程序的名称一一显示出来。

(5) 如果要运行某个程序,如上述的STRIPES程序,键入 RUN STRIPES ↵,主机立即通过磁盘驱动器,从磁盘中读取出名称为STRIPES的程序,调入内存(RAM)中,并立即运行该程序。

(6) 如果只要将磁盘内的STRIPES的程序读取出,调到内存(RAM)中,而不立即运行,可键入 LOAD STRIPES↵。

凡是已调入内存(RAM)中的程序,就相当于打键输入的程序,如要查阅、修改、运行或清除,所用命令及操作完全与前面讲过的相同。

卡式录音机对程序的贮存或取出,除指令与上述相同外,其他步骤方法见第十八讲。

第十七讲 华宇-200微电脑系统使用 (一)

1. 打印机的控制

(1) 打印命令 LPRINT LPRINT命令(语句)和PRINT很相似。LPRINT是通知电脑启动打印机打印,在其后面可指定要打印的内容及格式。

(2) 复制命令COPY COPY命令(语句)是通知电脑将荧幕上显示的程序或结果的内容送往打印机(也称打字机)。如果按BREAK键,即打印机停止打印。

(3) 印程序命令 LLIST LLIST命令(语句),类似在荧幕显示的LSIT功能,将程序在打印机上印出。

例1: 设下列程序

```
10 LET A = 245
20 LET B = 43
30 LET C = 3
40 LPRINT USING "###", A
50 LPRINT USING "###", B
60 LPRINT USING "###", C
70 END
RUN
```

执行结果在打印机上打印出:

```
245
43
3
```

例2: 设下列程序

```
5 LPRINT
10 LET A = 243
20 LET B = 43
30 LET C = 3
40 PRINT A, B, C
50 END
RUN
```

执行结果:

```
243      43      3
```

上述程序和结果也全部在打印机上打出来。

例3: 设已输入电脑程序为:

```
10 LET A = 5
```

```

20 LET B = 702
30 LET C = 20
40 PRINT USING "###", A
50 PRINT USING "###", B
60 PRINT USING "###", C
70 END

```

键入LLIST命令，上述程序也可由打印机打出。如果键入LLIST 20 TO 30则打印出下列内容：

```

20 LET B = 702
30 LET C = 20

```

2. 几个语句的用法

(1) PRINT @…… 这个命令（语句）使荧幕的特定位置上显示出某些内容，荧幕按32×16陈列，划成共512个位置（格），这些位置可用数字、变数或算术表达式来指定，数值须在0—511之间。印出的内容可以是可变数、数值或字符串，在语句末尾应该用分号（；），以免把该行后面的内容删掉。

如 PRINT @ 50, 580;

(2) PRINT TAB (数值表达式) 这个命令（语句）使光标在一行上移到某个位置。位置由TAB后面的(X)来指定。数值必须在0~255之间，如果数值大过63，电脑会计算出这个数值大于64倍的最大整数，然后光标位于余数位置上。有效值为0~63。

例4: 40 PRINT @ TAB(b); 1; TAB(20); 1
RUN

1 (空14格)

(3) PRINT USING…… 这个命令（语句）能够控制各行以特定的格式显示字符串或数值，适用于报告和图表。

语句格式: PRINT USING 字符串和数值

语句中的字符串和数值可以是一个常数或变数。执行这个命令时数值会依照字符串格式印出。

用PRINT USING命令加上下列不同的符号可印出设定格式:

① “!”号，指定只印出字符串的第一个字符。

例5: 10 A\$ = ASDF
20 PRINT USING “!”; A\$
RUN
A

② “#”号，用来指定每一个数字的位置。位数位置通常是填满的，如果印出的数字比指定位置的位数少，数字会自动在栏内向右调整（即左边留出空格）。

如果是一个小数点，可以在栏内任何一个位置加入。若在格式字符串中，小数点前指定了一个位数，那个数便按指定位置印出。如果没有值便会自动加零，或数字被调整。

例6: 20 PRINT USING “##.##”; .78
RUN

0.78

③ “+”号，在格式字符串的最前或最后，会使数字的正或负号，印（显示）在数字之前或之后。

“-”号，在格式字符串的末尾出现，会使负数字的值后面印出（显示）一个负号。

例7: 20 PRINT USING "+###.##"; -68.95
RUN

-68.95

例8: 20 PRINT USING "##.##-"; -68.95
RUN

68.95 -

④ “**”号，用在格式字符串前头，会使数字栏内开头的空白部分填满*号。但双星**在引号内亦可作指定两个位数的位置。

例9: 20 PRINT USING "**###.##"; -0.9
RUN ↙

* -0.9

⑤ “\$\$”号，双线符号会使一个钱符号在格式数字左面印出。\$\$指定两个位置，其中一个便是钱符号（有时表示字符串）。但有两种情况不能使用，就是在指数形式和负值的数字均不能使用\$\$，除非负号放在右方。

例10: 30 PRINT USING "\$\$###.##"; 456.78
RUN

\$456.78

⑥ “,”号，如果出现在小数点的左方格式字符串中，它会在小数点的左方每三个数字之间印出一个逗号。如果逗号在格式中字符串最后出现，则会作为字符的一部分。一个逗号则指定另一个位的位置。

例11: 40 PRINT USING "###,##"; 1234.5
RUN

1,234.50

⑦ “%”号 如果印出的数字超过指定数字栏的位置，一个%的符号，会在数字之前标印出。如果是舍入的数字，使数字超出这个栏，一个%的符号也会在舍入数字之前标印出。

例12: 50 PRINT USING "##.##"; 111.22
RUN

% 111.22

例13: 70 PRINT USING ".##"; 999
RUN

%999.00

或 % 1.00 (1前无位时)

(4) GOTO语句。

例14: 10 INPUT A\$
20 B\$ = B\$ + A\$

```

30 PRINT B$
40 GOTO 10
RUN↵
? T↵
T
? H↵
TH
? I↵
THI
? S↵
THIS
?

```

(5) INP (I) 语句 这个语句(命令)是通知电脑返回出入口 I 读出来的值。I 必须在 0—255 之间, INP 是 OUT 语句的辅助功能。

例: 100 A = INP(255)
OUT I, J

这个语句(命令)是通知电脑往主机的输出埠(指接口)传送出一个数值, I 和 J 是整数表达式, 在 0—255 之间, I 是接口的数值(接口号数), 而 J 是转输的数据。

例: 100 OUT 32,150

其常用在联机调用, 也可用在网络上。

(6) USR(X) 语句 这个语句(命令)是利用 X 变元来呼唤用户的汇编语言的子程序, 必须是从卡式录音机输入或者是由 POKE 命令写入, 应用时要特别小心, 查对程序, 因为用错会把程序毁坏。

例: 110 A = USR(B/2)

当执行此命令时, 电脑记讲中的 16 进制 788 E、788 F (即为十进制的 30862 和 30863) 调出资料, 这些资料就成为用户的汇编语言程序的开始地址。

在执行这个 USR (X) 命令之前, 用户必须用命令 POKE, 将它的子程序放进记忆中(写入存贮器), X 的数值也会被送至子程序中。

第十八讲 华宇-200微电脑系统使用 (二)

1. 荧幕字体的两种选择

电脑一般配置绿底色显示器,若要采用白色字体,按下 **CTRL** 及启动电脑就可以实现。

如果电脑是配置黑白电视机作显示器使用,采用黑色字体(比较清晰),要键入

```
POKE 30744, 0 ↵
```

如果需要转换为白色字体,则键入

```
POKE 30744, 1 ↵
```

2. 图象

(1) 低分辨率图形和文字显示 电脑系统启动后,会自动处于正常的文字模式 MODE (0) :

设定以32(字) × (16)行的显示形式,或以64 × 32个点,9种颜色显示低分辨率图象。

(2) 高分辨图形模式 MODE (1) 这模式能够以128 × 64点和8种颜色显示图象。

若要回到正常的文字模式,只要输入 MODE (0) 即可。

(3) 图形字符 本机有16种图形字符,要这些字符显示,只要按着 **SHIFT** 键同时连用相对的图形字符键,就可以组合图象。

应用这些字符,可看下列程序的运行。

```
例 1: 10 REM COLOR  
20 FOR I = 1 TO 52  
30 FOR J = 1 TO 8  
40 COLOR J  
50 PRINT "□";  
60 NEXT J  
70 NEXT I  
80 GOTO 20 (或80)  
90 END  
RUN ↵
```

如果键入 **BREAK**,可以停止这个程序。

```
例 2: 5 COLOR 8  
10 FOR X = 0 TO 30  
20 PRINT TAB (4); "□"; TAB (10); "▨";  
30 PRINT TAB (20); "□"; TAB (30); "▨";  
40 NEXT X
```

```
50 END
```

```
RUN ↵
```

荧幕显示：红色柱（见图18·1）

对于LASER-3000荧幕显示为四条柱子；
华宇-200，LASER-200为两条柱子。

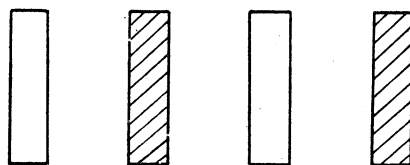


图 18·1.

（4）反转显示INVERSE 若要将字符
反转（白底黑字）显示，只要按下`INVERSE`

键，便可以实现。若再按下`INVERSE`键或`RETURN`键则回到正常（黑底白字）显示。

（5）MODE（1） 模式下的三个命令 SET、RESET、POINT可应用在MODE
（1）模式下绘画。

①SET（X，Y） 命令在X、Y值指定的位置上绘点，X的有效范围由0~127，而
Y值则由0~63。

②RESET（X，Y） 它把SET命令绘出的点清除掉，与SET命令中的X、Y一样，
X、Y值用于指定位置。（这个命令实际上是使点变为与背景同色。）

③POINT（X，Y） 这个命令，是检查荧幕上指定点有没有被SET命令绘上，若该
点已被SET绘上，此命令会给出该点的颜色代码。如果没有被SET绘上，此命令只给出颜
色代码为0。通常与条件转移语句——IF……THEN……ELSE联用。

例如：80 SET（40，20）；IF POINT（40，20） THEN POINT “YES”
ELSE POINT “NO”

```
例3： 10 MODE（1）；COLOR 2
        20 FOR I=0 TO 31
        30 SET（I，I/2）
        40 NEXT
        50 FOR I=0 TO 31
        60 RESET（I，I/2）
        70 NEXT
        80 MODE（0）
        90 END
        RUN ↵
```

在此例中电脑会绘出一对角线，然后自动抹去，当这程序完成后，电脑会进入MODE
（0）模式。若要维持图象模式MODE（1），只要将80号语句改为：

```
80 GOTO 10
```

若要停止执行，则键入BREAK；若要继续执行则键入POINT（返回命令）即可。

```
例4： 10 MODE（1）；DIM A（4）
        20 FOR I=1 TO 4
        30 COLOR I
        40 SET（I，I）
        50 A（I）=POINT（I，I）
        60 NEXT
```



```

70 MODE (0)
80 FOR I= 1 TO 4
90 POINT A (1)
100 NEXT
105 END
RUN)

```

在这个例子中，用四种颜色画点，然后用POINT得出颜色代码，以及存贮在数组A(1)，最后将一数组数据印出。

3. 颜色命令 (COLOR I, J)

可以使用此命令(语句)设定不同颜色。其中参数I是用于设定前景颜色，由1~8。而J用于设定背景颜色，由0至1。具体的颜色代码如下。

在MODE (0) :

代码	颜色
1	绿
2	黄
3	蓝
4	红
5	浅黄
6	浅蓝
7	紫
8	橙

背景颜色可以是绿色(0)或橙色(1)。要改变背景颜色，只要键入COLOR(1)，便可得橙色。要返回到原有绿色，键入COLOR(0)。

在MODE (1) :

如果背景颜色是绿色(0)，前景颜色便是

代码	颜色
1	绿
2	黄
3	蓝
4	红

如果背景颜色是浅黄(1)，则前景颜色是

代码	颜色
5	浅黄
6	浅蓝
7	紫
8	橙

例: 10 COLOR 2, 0
前景变为黄色，而背景则为绿色。

```
10 COLOR 3, 1
```

前景转为蓝色，而背景浅黄色。

```
10 COLOR 4
```

前景转为红色，而背景保持不变。

```
10 COLOR 0
```

背景转为绿色，而前景保持不变。

当用命令COLOR I 或COLOR J 设定颜色后，一直保持到用的颜色设定命令才改变。

```
例： 10 FOR I=0 TO 15
      20 FOR J=0 TO 31
      30 COLOR(1/2)+1
      40 PRINT "□"
      50 NEXT ; NEXT
      60 FOR I=1 TO 1000:NEXT
      70 END
```

把SHIFT和J连用会在字符上得到这里所说的八种颜色。

```
例5： 5 COLOR 4
      10 PRINT TAB(10); "□";
      20 FOR I=16 TO 22
      30 PRINT TAB(I); "□";
      40 NEXT I
      50 PRINT TAB(20); "□"
      60 PRINT TAB(10); "□"; TAB(20); "□"
      70 PRINT TAB(20); "□"
      80 PRINT TAB(12); "□"; TAB(20); "□"
      90 PRINT TAB(11); "□"; TAB(20); "□"
     100 PRINT TAB(10); "□";
     110 FOR I=15 TO 23
     120 PRINT TAB(I); "□";
     130 NEXT I
     140 END
```

4. 发声命令 (SOUND

微电脑的BASIC配有发声的功能，利用主机内的拍声器，可以使命令（语句）设定声音的频率、拍节等以发出各种声音或编成乐曲。

```
例： 10 FOR I=1 TO 8
      20 READ X
      30 SOUND X, 7
      40 NEXT I
      50 DATA 16, 18, 20, 21, 23, 25, 27, 28
      60 RUN
```

这个程序可产生八个音阶的音调，程序中的变数 X 是频率值，而常数 7 是音调的持续时间。运行此程序能产生 31 个不同频率和 9 种不同音调时间所组成的响声。随机手册中有附表，列出了产生不同频率和音调时间的代码。

第十九讲 程序和资料的外贮存与调用

电脑的广泛用途,是通过程序运行来实现的。为了使电脑执行多种职能,如财务管理、计划管理、人事档案、各种运算等,一个程序可多次使用,又能腾出电脑作别的工作,这就需要将各类的资料、程序编成文件(档案)存入外贮存器,使用时再由外贮存器与电脑联接调用。一般外贮存器有:磁带机(录放机)、磁盘驱动器等。

1. 将资料或程序存入外贮存器

(1) 命令格式: CSAVE “文件名”

或 SAVE “文件名”

(2) 操作步骤及方法:

①选用优质磁带放入卡式录放机(或用磁盘驱动器,另述),放在录位、音量调节放至适当位置。

②在电脑正常工作状态下键入整个程序或资料,在监视器荧屏上显示、检查修改直至无误。

③键入命令CSAVE(或SAVE) “档案名”(SAVE是对华宇—3000机用的),暂时不按RETURN键。

④按下录放机的PLAY和RECORD键,即录的位置(磁盘驱动器也是放在录位)。

⑤按下RETURN键,闪动的光标便会消失,而开始存入外贮存器。

⑥当闪动的光标重新出现的时候,贮存便完成了。

⑦按下卡式录放机STOP键。把磁带取出,编上名称和记录位置。

2. 从外贮存器中查找或调用程序

(1) 命令格式: CLOAD “文件名”

或 LOAD “文件名”

(2) 操作步骤和方法:

①将录存有程序的磁带放入卡式录放机。

②将磁带转到记录程序的开始位置。

③键入命令CLOAD(或LOAD) “文件名”,暂时不按RETURN键。

④按下卡式录放机的PLAY键。

⑤按一下RETURN键。

⑥如果运行中找不到所需要找的程序,荧屏上出现“WAITING”信息。如果不想继续找,按下BREAK键(CTRL……)即可。

⑦如果找到的程序文件名和命令中的文件名不相符,荧屏上就显示“FOUND T:FILENAME”程序便会被略过去。

⑧如果电脑找到指定的程序,荧屏上会显示:LOADING T: FILENAME(文件名)。

⑨当荧屏显示READY时，把录放机的STOP键按下。

例：若磁带上三个程序，各个文件分别为：

```
PROGRAM 1  
PROGRAM 2  
PROGRAM 3
```

如果你希望调用“PROGRAM 3”，在键入CLOAD（或LOAD）“PROGRAM 3”后（假定磁带机已在工作状态且从头开始查找），按RETURN键。荧幕上将顺序显示：

```
WAITING  
FOUND T: PROGRAM 1  
FOUND T: PROGRAM 2  
LOADING T: PROGRAM 3  
READY
```

3. 核对文件

(1) 命令格式 VERIFY “文件名”

此命令可用来将磁带上所存入的程序与电脑内存的程序核对，检查有无错漏。

(2) 操作步骤和方法

①列出在电脑内存中的程序，以证实程序的存在。

②键入命令VERIFY “文件名”，暂时不键入RETURN键。

③按下卡式录放机PLAY键。

④按下RETURN，闪动光标便消失，核对过程便开始。

如：VERIFY “PROGRAM 2”

荧屏显示：

```
WAITING  
FOUND T: PROGRAM 1  
LOADING T: PROGRAM 2  
VERIFY OK  
READY
```

⑤如果显示“VERIFY OK”，表示磁带上程序和电脑内存中程序相同。

⑥如果显示“VERIFY ERROR”，则表明磁带上与内存的程序不相同。

4. 把磁带上的程序载入内存并运行的命令

CRUN “文件名”

将录存有需要用到的程序的磁带放入磁带机，并使磁带机处于工作状态。从电脑的键盘键入上述命令，便可以将文件名所指定的程序调入内存并即运行。运行后程序仍存于内存。

5. 向磁带文件传送资料的命令

PRINT # “文件名”，……

把指定变数的数值或数据转送到卡式录放机的磁带上时，录放机应处于工作（录存）的状态。

例：10 PRINT # “NAME”，9，8，7，6

RUN↵

可将常量数据 9, 8, 7, 6 以文件名NAME存入磁带。

6. 从磁带文件读取数据并赋予指定的变量命令

INPUT # "文件名",

例: 设已用 PRINT # "NAME", 9, 8, 7, 6 语句存入资料

20 INPUT # "NAME", I, K, T, P

30 PRINT I; K; T; P;

RUN↵

FOUND D: NAME

9 8 7 6

第二十讲 华字—3000(LASER—3000)

微电脑系统操作

华字—3000 (LASER—3000) 与APPLE—II原理基本相同。BASIC的命令(语句)及编程使用也基本上一样,故上述的内容均可引用。这里只对一些不同的部分予以简述。

1. 华字—3000的功能键操作

华字—3000 (LASER—3000)的键盘上附加有8个功能键(F1~F8)。利用这8个功能键及与其它键联用,可直接由按键而执行24种命令(语句),从而省去逐个打键输入的麻烦。各命令(语句)对应的键符如下。

(1) 只按功能键F1~F8,获得对应F1~F8的作用。

(2) 按住SHIFT键再同时按F1~F8,获得对应F9~F16的作用。

例如:同时按SHIFT及F1两键,执行CALL-151,即调用机器语言程序使系统转为监控状态。

若同时按SHIFT及F4两键,执行SOUND,即调用发声功能。

(3) 按住CTRL键再同时按F1~F8,获得对应F17~F24的作用。

例如:同时按CTRL及F2两键,执行HGR,即进入高分辨图形模式。

若同时按CTRL及F6两键,执行PR#1,即启动磁盘机。

各功能键符对应的命令(语句)对照关系见表20.1。

表20-1

功能键符与命令对照表

键符	命令(语句)	F ₁₃	DRAW SCIRCLE (
F ₁	LIST	F ₁₄	DRAW HSCIRCLE (
F ₂	RUN <u>RETURN</u>	F ₁₅	DRAW SSQUARE (
F ₃	HOME	F ₁₆	DRAW HSQUARE (
F ₄	TEXT	F ₁₇	PR#1 ↙
F ₅	TEXT NORMAL BLACK	F ₁₈	HGR
F ₆	WIDTH 80	F ₁₉	HCOLOR =
F ₇	WIDTH 40	F ₂₀	HPLOT =
F ₈	PR#8 ↘	F ₂₁	PAINT (
F ₉	CALL-151 ↘	F ₂₂	PR#6 ↘
F ₁₀	SOUND DEF	F ₂₃	NOISE
F ₁₁	SOUND TEMPO	F ₂₄	PRINT USING
F ₁₂	SOUND		

上述命令（语句）的含意及作用，请查LASER系列，APPLE II实数BASIC的命令说明（见附录I）。

2. 打印机的使用

本电脑配点矩阵式打印机，既可打印字符又可印出图象。

(1) 打印字符：

①将接口连线（带状电缆）通过配接器（也称接口），将电脑与打印机连接好。应注意，连接前必须断开打印机及主机的电源。

②经检查无误后，接通电源。

③启动打印机，使打印机处于工作状态，随时可接受打印内容并打出。

在 BASIC状态，用命令 PR # 1

在 KERNE 1 状态，用命令 LCTRL—P

④使用命令（语句）设置打印内容，执行后将会在打印机上打印出来。

⑤停止打印命令。

在 BASIC状态，用 PR # 0

在 KERNE 1 状态，用 OCTRL—P

(2) 打印图象 除按上述打印字符的①、②、③条外，还要使用BASIC的 PRINT SCREEN 命令来设定。

另外，命令 HGR 1~HGR 6 可用来选择或不印出，方法是在使用 PRINT SCREEN 命令之前，把一个数存入存储器零页的某个地址。

如果 PRINT SCREEN 命令及HGR 1~HGR6 仍不够用时，可按如下顺序操作：

①启动打印机：

在BASIC 状态，用 PR # 1

在KERNE 1 状态，用 ICTRL—P

②将需要印出的图象程序（资料或外存贮的资料）准备好 把产生图象的程序输入内存（键入或由外存载入）并运行，先在荧屏上显示出所需要的图象。

③输入下列两个字符串：

\$6F9（十六进制数，对应十进制为#1785）

\$779（十六进制数，对应十进制为#1913）

④键入 CTRL Q 使之开始打印出图象。

⑤若要停止打印机，方法同前面所述相同：

在 BASIC 状态，用 PR # 0

在 KERNE 1 状态，用 OCTRL—P

(3) 绘图命令 DRAW 使用这个命令与下列的词组成语句可画出某些形状的实心或空心图形，实心的用代号S表示——中间着色，空心的用代号H表示——只有轮廓。

①圆形CIRCLE——SCIRCLE和HCIRCLE（实心与空心） SCIRCLE和 HCIRCLE的形状参数及语句如下：

DRAW SCIRCLE (X, Y) , r, C, sr, er

其中：X指荧屏的X坐标（水平方向的位置）；

Y指荧屏的Y坐标（竖直方向的位置）；
 r是要画的椭圆形的主轴线长度；
 C应在0至1范围内，决定圆度形状及其平直度，
 当C = 1时圆即画成；
 sr是起始弧度；
 er是终止弧度，并应大于sr。

若C、sr和er没有规定，则内值为C = 1，sr = 0，er = 2 π 。还要注意，若在用DRAW SCIRCLE或DRAW HCIRCLE命令之前，先用ROT（旋转命令），可以使所画的椭圆朝任何一个方向。以上的形状参数也适用于HCIRCLE（画空心圆）。

注：低分辨率图形和高分辨图形的模式下，X，Y参数如下：

X坐标的范围是0—279；
 双倍高分辨（高清晰度）图象X坐标范围是0—559；
 全图模式Y坐标范围是0—191。

②正方形 SQUARE——SSQUARE 和 HSQUARE（实心与空心） SSQUARE 和 HSQUARE的形状参数及语句如下：

DRAW HSQUARE (X1, Y1 TO X2, Y2 TO X3, Y3)

式中的每一对坐标用来设定正方形的对角。

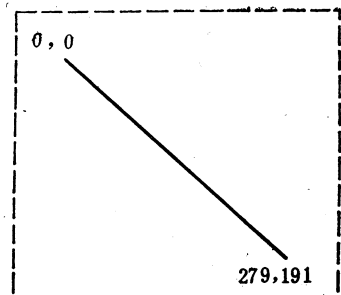
X和Y坐标的数值与DRAW SCIRCLE和DRAW HCIRCLE命令中的X和Y坐标的数值范围相同。

例1：
 10 HGR 5
 20 HCOLOR = 3
 30 DRAW SCIRCLE (140, 96) , 50
 40 END

执行结果，画出中心在（140，96）半径50的实心绿色圆形图。

若要画出实心的黄色正方形，（0，0）与点（10,10）处于斜对角状态，参考程序如下：

例2：画直线的程序
 10 HGR 1
 20 GET H
 30 HCOLOR = H
 40 HPLOT 0, 0 TO 279, 191
 50 GOTO 20
 RUN



显示见图20·1。

图 20·1

第二十一讲 LASER-3000及APPLE II

微电脑系统对图形的处理

LASER—3000、APPLE II系统可以很方便地进行图形处理工作。通过控制矩阵点(280×192)可以在荧屏上构成多种颜色的图形。构图有低分辨和高分辨两种模式。

1. 低分辨图形

以7×4的矩阵点小方块为一个单元,再通过控制各个小方块可构成各种图形。由于这种模式下的图案(小方块)较大,所构出的图形比较粗糙,分辨率较低。

(1) 混合式命令 GR

语句格式: 行号 GR

执行此语句(命令)后,系统处于低分辨图形与字符的混合模式。荧屏上除显示图形外,还可在下边显示四行字符。

可用于构图的方块有40×40个(见图21·1)。若用彩色显示器,可使用的颜色有16种。用于显示字符的位置是荧屏下部四行。

(2) 全图式命令 { GR POKE—16302,0

执行 GR后,再执行 POKE—16302,0(或POKE 49234,0),系统处于低分辨全图模式的状态。这时整个荧屏显示图形。可用来构图的方块有40×48个(见图21·2)。若使用彩色显示器,可使用16种颜色。

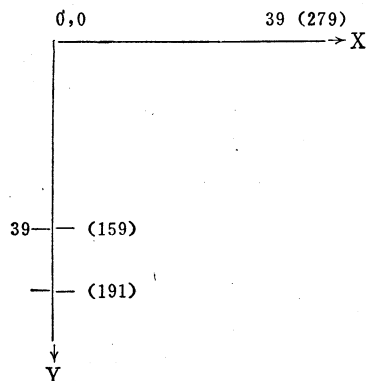


图 21·1

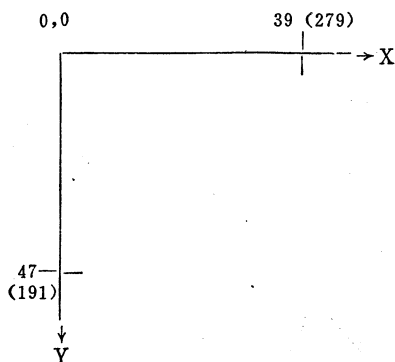


图 21·2

几点说明: ①当系统处于图形状态下,若要清屏,需要用命令 GR。

②若要转为字符状态,需要用命令 TEXT

③由低分辨全图式转为低分辨混合式,使用命令 POKE—16301, 10

(3) 设置颜色的命令 COLOR

语句格式: 行号 COLOR = 数值 (颜色码见后表)

(颜色清屏可用COLOR = 0)

例1: 将荧屏显示图形区着于不同颜色

```
10 GR
20 FOR I= 1 TO 15
30 COLOR=I
40 X= 0
50 VLINO, 39 AT X
60 X=X+ 1
70 IF X<40 THEN 50
80 FOR K= 1 TO 2000; NEXT K
90 NEXT I
```

例2: 在荧屏上显示图案

```
10 GR
20 FOR Y= 0 TO 30 STEP 10
30 FOR X= 0 TO 35 STEP 5
40 COLOR=16*RND(15)
50 FOR K= 0 TO 9
60 HLIN X, X+ 4 AT Y+ 4
70 NEXT K
80 FOR N= 1 TO 500:NEXT N
90 NEXT X, Y
100 GOTO 20
```

(4) 求颜色代码位置的命令(函数) SCRN (X,Y) 若对某位置的图(小方块)或字符的颜色代码有要求,可用此命令(函数)与PRINT联用。如:

```
PRINT SCRN (5, 10)
```

其中5为横向位置的数值,10为纵向位置的数值。

使用此命令应注意以下三点:①这命令不能单独使用,通常PRINT组成语句;

②此命令只限在低分辨率图形状态下使用;

③括号内数值要与实际位置相符。

(5) 方块的产生 在已设定颜色下,若要产生一个小方块,可使用如下语句:

```
行号 PLOT 数值1, 数值2
```

其中数值1(X)指横向(水平)上的位置,混合式为0—39,全图式为0—39;

数值2(Y)指纵向(竖直)的位置,混合式为0—39,全图式为0—47。

```
例: 10 GR
20 COLOR=13
30 PLOT 15, 20
RUN ✓
```

在第20行的第15格上出现一个黄色小方块(见图21·3)。

(6) 线段的产生

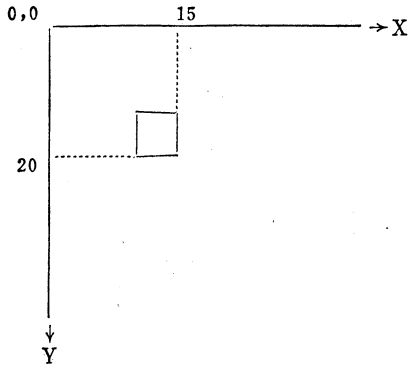


图 21.3

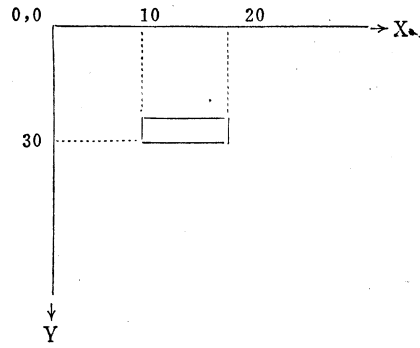


图 21.4

①在已经设置颜色情况下，若要构成一条水平线段，可使用如下语句：

行号 HLIN X1, X2 AT Y

其中Y指纵向（竖直）位置；X1是线段起点；X2是线段终点。

例： 10 GR
20 COLOR = 12
30 HLIN 10, 20 AT 30

执行结果（见图21.4）。

②在已设置颜色之后，若要构成一条竖直方向的线段，可使用如下语句：

行号 VLIN Y1, Y2 AT X

其中：Y1指线段的起点；Y2指线段的终点；X指线段的水平位置；Y1、Y2、X均可用数值或表达式表示。

例： 10 GR
20 COLOR = 3
30 VLIN 10, 20 AT 30

执行结果（见图21.5）。

2. 高分辨图形

高分辨图形是直接由显示点来构成图形。这种模式下的图案（小点子）细，构出的图比较精致，分辨率较高。

（1）混合式命令 HGR 执行这命令（语句）后，系统处于高分辨图形与字符的状态。可用于构图的单元有 280×160 个；若用采色显示器，有用的颜色有8种；荧屏下边四行是显示字符的位置，如图21.6所示。

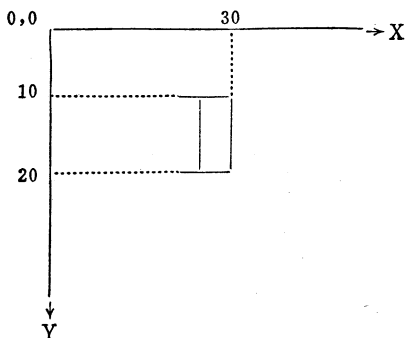


图 21.5

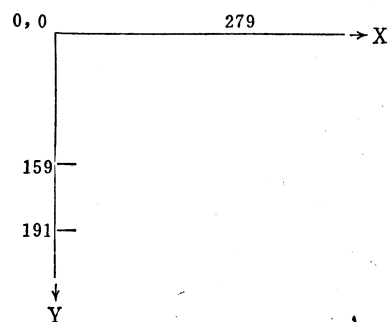


图 21.6

图形部分清屏命令 HGR

较为全字符状态命令 TEXT

(2) 全图式命令 HGR 2 若要使系统处于整个荧屏都显示图形的状态, 可使用下述两种语句之一。

① 行号 HGR 2

② 行号 HGR

行号 POKE—16302, 0

或 HGR : POKE—16302, 0

执行上述语句之后, 系统进入高分辨全图模式的状态。可用于构图的单元有 280×192 个点 (见图21.7); 可用的颜色有8种。

清屏可用命令 HGR2;

转为正常字符状态可用命令 TEXT;

转为低分辨图形状态时, 先用TEXT, 再用GR。

(3) 设置颜色的命令 HCOLOR = N 在高分辨图形状态下, 要设置颜色, 可使用如下语句:

行号 HCOLOR = N

其中数值N是颜色的代码 (见附录 I), 清屏时可用 HCOLOR = 0。

在使用颜色绘图前, 可先调用主磁盘中文件名为COLOR DEMOSOFT 程序, 先调试好色彩 (记住各对应代码)。

(4) 点的产生 若要在荧屏上某坐标 (X, Y) 处产生一个亮点, 需先用设置颜色的语句 (否则, 自动定为黑色), 再用下述语句:

H PLOT X, Y

其中: X指水平方向的位置;

Y指垂直方向的位置。

X、Y均可直接给出具体数值, 或给定变量、表达式。

例: 10 HGR
20 HCOLOR = 2
30 H PLOT 20, 20

执行结果 (见图21.8)。

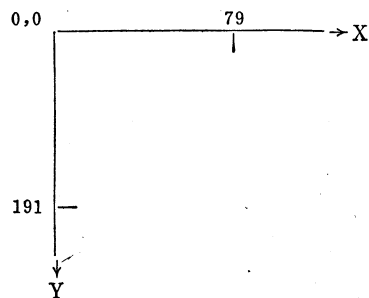


图 21.7

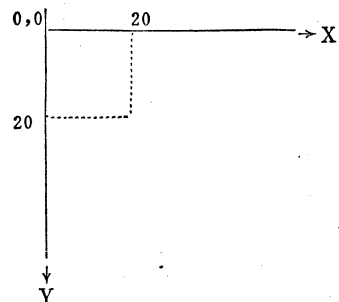


图 21.8

(5) 线段与折线的形成:

①在高分辨图形的状态下，若已经设置了颜色，要得出任意一条线段，可用语句：

```
H PLOT X 1, Y 1 TO X 2, Y 2
```

其中：X 1、Y 1 指线段的起点位置；

X 2、Y 2 指线段的终点位置。

X 1、Y 1 和 X 2、Y 2 可直接给出具体数值，或用变量、表达式给定。

```
例： 10 HGR
      20 HCOLOR = 3
      30 H PLOT 10, 20 TO 80, 100
```

执行结果（见图21·9）。

形成的线段是由小点构成的。

②折线（曲线）的形成 将形成线段的语句扩展成如下语句即可。

```
H PLOT X 1, Y 1 TO X 2, Y 2 TO X 3, Y 3 TO.....
```

执行后，可在 (X₁, Y₁)，(X₂, Y₂)，(X₃, Y₃) 各位置间以线段连成折线或曲线。

```
例： 10 HGR
      20 HCOLOR = 2
      30 H PLOT 1, 0 TO 279, 0 TO 279, 159 TO 1, 159 TO
          1, 0
```

执行结果（见图21·10）。

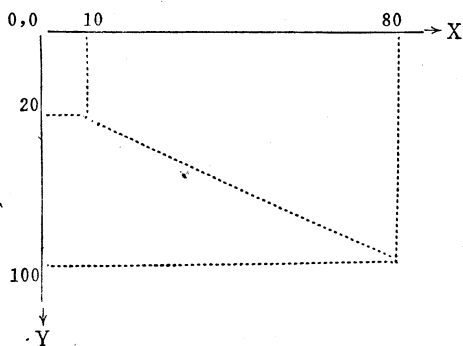


图 21·9

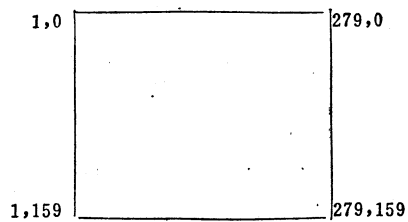


图 21·10

例3： 在荧屏上以15种颜色显示中国两字：

```
10 FOR I = 1 TO 15
20 GR
30 READ T, X, Y, Z
40 IF T = 0 THEN I 120
50 COLOR = I
60 IF T = 1 THEN 80
70 IF T = -1 THEN 100
80 V LIN X, Y AT Z
90 GOTO 30
100 H LIN X, Y AT Z
```

```

110 GOTO 30
120 RESTORE
130 FOR J= 1 TO 2000 : NEXT J
140 NEXT I
200 DATA 1, 5, 12, 5, -1, 5, 13, 5, 1, 5, 12, 13, -1,
      5, 13, 12, 1, 2, 15, 9
210 DATA 1, 2, 16, 28, -1, 28, 36, 2, 1, 2, 16, 36, -1,
      28, 36, 16, -1, 30, 34, 4, -1, 30, 34, 9
220 DATA -1, 30, 34, 14, 1, 4, 14, 32, 1, 11, 11, 34, 0,
      0, 0, 0

```

例4: 在荧屏上以不同颜色显示“电脑”两字:

```

10 HGR
20 FOR I= 1 TO 7
30 HCOLOR=I
40 HPLOT 28, 28 TO 112, 28 TO 112, 80 TO 28, 80 TO
      28, 28
50 HPLOT 28, 55 TO 112, 55
60 HPLOT 70, 16 TO 70, 96 TO 126, 96 TO 126, 82
70 HPLOT 152, 96 TO 161, 72 TO 161, 16 TO 182, 16 TO
      182, 36 TO 174, 90
80 HPLOT 161, 57 TO 182, 37
90 HPLOT 161, 63 TO 182, 63
100 HPLOT 224, 16 TO 224, 30
110 HPLOT 196, 30 TO 252, 30
120 HPLOT 196, 40 TO 196, 96 TO 252, 96 TO 252, 40
130 HPLOT 203, 50 TO 245, 84
140 HPLOT 245, 50 TO 203, 84
160 NEXT I

```

例5: 在荧屏上显示出波纹图案:

```

5 HOME
10 VTAB 24
20 HGR2
30 A = RND (1) * 279
40 B = RND (1) * 191
50 N = INT (RND (1) * 4) + 2
60 HTAB 15
70 FOR X=0 TO 278 STER N
80 FOR S=0 TO 1
90 HCOLOR = 7 * S

```

```
100 HPLOT X+S, 0 TO A,B TO 279-X-X, 191
110 NEXT S, X
120 FOR Y=0 TO 190 STEP N
130 FOR S=0 TO 1
140 HCOLOR = 7 *S
150 HPLOT 279, Y+S TO A, B TO 0, 191-Y-S
160 NEXT S,Y
170 FOR K= 1 TO 1500
180 NEXT K
200 GOTO 30
```


第二十二讲 微电脑系统对汉字处理

英文电脑系统，经过一定的扩充，也可以具有处理汉字的功能。一般是用键盘将汉字输入电脑，用英文电脑的功能把汉字作为字符型数据，对汉字进行处理，然后输出汉字资料。

对LASER-3000, APPLE II系统，只要配置一块英汉外围接口板，便可使系统保留原有的英文电脑功能，又具有汉字处理功能，实现对汉字的处理。

1. 英/汉接口板

英/汉接口板（下面简称为英/汉板，亦称汉卡），是一种包含有软件和硬件的外围接口板，其中已置入了专用程序和建立了汉字库。把它联接到英文微电脑系统后，可使系统具有处理汉字的功能。

能够与LASER-3000, APPLE II PLUS系统配接的英/汉板有：C-PLUS][和C-PLUS][A，这里只介绍C-PLUS][A。

(1) C-PLUS][A的功能 可以使APPLE II系统以英文电脑状态和汉字数据状态交替使用。

①英文电脑状态，标示符为]-。基本保留原APPLE II的实数BASIC功能，只有很小的差别，为后面的叙述简便起见，称这种状态为准APPLE SOFT状态，简称为准AP状态。


1) 在原 BASIC 的命令中，除 TEXT、VTAB、HTAB、HOME、TAB、POS、SPC、FLASH、INVERSE、NORMAL、GR、COLOR、PLOT、HLIN、VLIN、SCRN、HGR、*MOISE、SOUND、TROFF、TRON、SWAP（带有*的是LASER-3000机的命令），共计21个命令无效外（APPLE II为21个，LASER-3000为22个），其余的命令（语句）均具有原功能，可以按原格式运用。

2) 键盘作用 基本保持原来的功用，只有几个特别的作用。

命令 CTRL-L

是准AP状态与汉字数据状态之间的转换控制。单数次按键作用，转为汉字数据状态。偶数次按键转为准AP状态。

命令 CTRL-D

则是在准AP状态下未转为汉字状态前，用来选择当转为汉字状态时是否有中文标志字样。奇数次按键，使荧屏左下角有“中文”两字的标志并将组合汉字过程中的各个字元（偏、傍、部首）显示出来。偶数次按键则取消上述作用。若按 SHIFT-就出现↑，其余几个键与对应出现的符号见表22.1。

3) 荧屏显示 每幕10行，每行34个字，每个字由8×16的矩阵点构成。

4) 有一些特别的功能，需要使用专用的命令或语句调用。

在准AP状态下，除上述一些特殊之外，原BASIC系统完全可以照用。

表22·1

SHIFT—	$\boxed{\begin{matrix} (\\ 8 \end{matrix}}$	[
SHIFT—	M]
SHIFT—	K	[
SHIFT—	O	←
SHIFT—	L	\
SHIFT—	$\boxed{\begin{matrix}) \\ 9 \end{matrix}}$]

②汉字数据状态，光标为]_，还会在荧屏左下角显示出“中文”字样，组汉字时还可将字元显示出来。

这种状态下，主要是把汉字作为字符型数据由键盘输入，同时可在荧屏上显示出来。除了少数的几个键盘命令（功能）之外，不能输入APPLE II、LASER或其他体系的命令。

1) 键盘作用 主要是输入汉字及一些字符。

(a)原ASCII码键盘中A至Z键，改为汉字作用，每个键对应于一个汉字，可作单键产生汉字，也可用几个键去组成汉字。各个键对应的汉字及其作用，请看键盘图及有关的汉字组构法。

(b)其余的字符键，基本保持原字符作用。

(c)键盘命令（功能）

CTRL—RESET

复位、返回TEXT（实数BASIC）状态，光标为]□。

CTRL—L

转为准AP状态，光标为]—。

CTRL—D

在汉字状态下，用来设定在组合汉字时有无字元的显示，单次按键为有，偶次按键为无。

(d)先按ESC后再按下述各键，其作用与原BASIC时的功能相同。

- A 使光标右移一个位置；
- B 使光标左移一个位置；
- C 使光标下移一个位置；
- D 使光标上移一个位置；
- I 使光标上移一个位置（可连续）；
- J 使光标左移一个位置（可连续）；
- K 使光标右移一个位置（可连续）；
- M 使光标下移一个位置（可连续）；
- E 清去光标至本行末的字符；
- F 清去光标至本行荧屏末的字符；
- @ 清去全部荧屏的字符。

(e)按→ 使光标经过的字符重新输入（要注意正确操作，否则，不能成功）。

(f)按← 删改刚键入(输入)的一个汉字(要小心,连续使用时,往往会转到准AP状态)。

(g)按长条键——空格及构字键,除了移动光标向前,产生一个空格外,在组构汉字时,当键入字元后,要按长条键才能得到刚键入的字元构成汉字。

2) 荧屏显示 每幕10行,每行17个汉字,每个汉字由16×16的矩阵点构成,每个汉字横向显示。

3) 打印输出 在打印机上输出汉字,可在准AP状态下由命令设定以四种型式印出。

(a)横式大字

(b)横式小字

(c)竖式大字

(d)竖式小字

每行印出的字数及各行的间隔,视具体的打印机可由命令去设定(指定)。

(2)英/汉接口板的连接 有专门为APPLE II LASER系统配备的英/汉板,把英/汉板做成能够很方便地连接到APPLE II、LASER—3000系统内的形式。

① 对微电脑系统的要求:

1) 一般为APPLE—II或LASER等相应类型。

2) 系统内存48K RAM,其中\$4000—\$5FFF作汉字,中文数据显示的存贮区;\$9000—\$95FFF作组构汉字时的存贮区。

其余的内存分配见表22.2内存分配表。

3) 要有稳定的供电

+ 5 V	2.5 A
- 5 V	0.5 A
+12 V	1.5 A
-12 V	0.5 A

②将英/汉板接入系统的方法:

1) 断开主机电源。

2) 揭开主机壳上后盖。

3) 将英/汉板插入机内2~5号中的任一个插座(记住插入的座号),通常插在3号插座上。

4) 插入(或检查)打印机(通常插在1号),磁盘机(通常插在6号)等接口板。

5) 启动系统,试运行,键入

PR # 3 ↵

在荧屏上显示出有关汉卡的信息,则英/汉板接入正常。

6) 按CTRL—RESET(复位后)检查磁盘机,打印机等是否正常,若一切正常,关机。

7) 盖上后盖,放置好显示器。

③使用英/汉板的有关命令 在准AP状态下,原BASIC中有一些命令不能使用而这部分命令可用另外语句来实现。另外还有一些专用的命令,一起在下面列出。为了叙述方便,先设定一些变量:

表22·2

内存分配表

存储地址		用途
十六进制 ^H \$	十进制	
0000 } 0800 }	0 } 2048 }	系统运行管理 RAM
0801 } 3FFF }	2049 } 16383 }	可供用户使用
4000 } 5FFF }	16384 } 24575 }	(显示高分辨图形的资料存贮英/汉板及显示汉字中文资料的存贮)
6000 } 8FFF }	24576 } 36863 }	用户可用的存贮区
9000 } 95FF }	36864 } 38399 }	英/汉板及汉字组构字存贮
9600 } BFFF }	38400 } 49151 }	磁盘管理体系的存贮 (DOS)
C000 } CFFF }	49152 } 53247 }	有关输入/输出的安排、管理
D000 } F7FF }	53248 } 63487 }	(实数BASIC) 语言编译程序照写
F800 } FFFF }	63488 } 65535 }	自动启动监控程序

CN 代表英/汉板所插入的外围接口插座号, 如CN = 3。

PT 代表打印机接口板所插入的外围插座号, 如PT = 1。

BS 代表与某些子程序入口地址有关数字如BS = 49152 + 256 * CN。

1) 调用系统内已有的子程序去实现某种功能的命令(语句):

CALL BS + 21 清屏(与HOMF的作用相同)。

CALL BS + 24 清去由于前面使用过PRINT语句所留下短横线(—)光标。

CALL BS + 36 清去由光标至该行末的字符, 光标换行(类似ESC—E)。

CALL BS + 39 清去光标所在行的字符, 光标跳到下一行开头。

CALL BS + 42 清去由光标至荧屏末的字符, 光标跳到下一行开头(类似ESC—F)。

CALL 64098 从准AP状态转为实数BASIC的TEXT状态(正常显示字符)。

2) 用软件开关去调用某些功能 使用命令POKE, 可将某种数字置入某个特定的内存地址。

(a) POKE 214; 数值

作用与HTAB 数值 相同, 使光标沿水平移到数值(0—33)所指定的位置。

(b) POKE 215; 数值

等效于VTAB 数值, 这里数值可用0—9。

(c) POKE 253, 255

使在汉字数据状态下, 荧屏的左下角有“中文”字样的标志及组汉字时有字元的显示。与单次按CTRL—D键相同。

(d) POKE 253, 0

使在汉字数据状态下, 荧屏的左下角无标志。与偶次按CTRL—D同。

下面主要是关于在准AP状态下, 调用打印机及设置打印输出的格式等方面的语句(命令)。

(e) POKE 1400+CN, PT

如POKE 1400+3, 1(设已把英/汉板插在3号外围插座, 打印机接口板插在1号外围插座上)。其作用是, 在打印机已接通电源的情况下, 启用打印机。

(f) POKE 1400+CN, 0

撤消打印机作输出的作用。

(g) POKE 1656+CN, 0

设定打印机以横式大字的格式印出汉字。

(h) POKE 1656+CN, 1

设定打印机以竖式汉字的格式印出汉字。

(i) POKE 1656+CN, 2

设定打印机以横式小字的格式印出汉字。

(j) POKE 1656+CN, 3

设定打印机以竖式小字的格式印出汉字。

若不用上述的语句3) — 6) 设定印出汉字的格式, 则系统自动取横式大字的格式。

(k) POKE 1912+CN, N

设定打印机印出的英文字母符(数字)之间, 相隔N个矩阵点, 而印出的汉字之间则相隔2N个矩阵点。如不给定N值, 自动取N=0。

(l) POKE 1784+CN, M

设定打印机印出的字符行之间相隔M个矩阵点。若不给定M值, 自动取M=4。

(m) POKE 2040+CN, Q

设定打印机印出英文字符时, 每行最多为Q个。印出汉字时, 每行最多则为Q/2个。若不给定Q值, 自动取Q=50。

对不同型号的打印机, 可设定每行最多印出的字符个数, 有所不同。如EPSON MX—80打印机, 每行最多为120个英文字符, 60个汉字。

以上除(e)、(f)、(l)语句外, 其他在用C—PLUS][型的英/汉板时无效。

(3) 英/汉板的使用 英/汉板是一种外围设备接口板, 当接入系统后, 便成为一个整体系统, 增添了准AP工作状态及汉字数据的输入功能。这样, 便可以使用命令(语句)作各种状态的转换及调用各种功能。既可以立即的方式进行, 也可以延缓方式(编成程序)去进行。

一般来说, 只是在要输入汉字时或运行与汉字有关的程序段时, 才启用英/汉板, 然后又转回实数BASIC状态。其用法如下:

①开机、启动系统(包括导引磁盘, 接通打印机电源等)。

②输入命令PR # CN↵ CN为英/汉板所插入的外围插座号,如插在3号,则为PR # 3↵。

③在] _ (准AP) 状态下,输入有关处理汉字的程序行号,命令(语句)等。当要输入汉字时,要把汉字当作字符串(字符型数据)去输入和处理。

④用CTRL-D↵及CTRL-L↵或按下POWER键*(奇次)转到汉字状态。这时荧屏左下角会有“中文”字样作标志(如不用CTRL-D则无)。在该状态下,可由键盘键入汉字及一些字符。同时在荧屏上显示出来。凡要输入或处理的汉字及字符必须在“ ”内给出。

注有*号的键表示有些机种,按下电源指示键后,处于小写英文键入状态;在已插入英/汉板并已启用时,可等效于此处的CTRL-L作用,只是无中文字样的标志。

⑤用CTRL-L↵(偶次),转到准AP状态,再按RETURN键以结束并存入刚才键入的有关汉字的程序行。

⑥如果运行有关汉字的程序或要打印输出汉字,则必须在准AP状态下,使用有关的命令。如:

```
LIST
RUN
POKE 1400 + CN, 1
```

⑦若要正常地转为实数BASIC状态,使用如下命令:

```
CALL 64096
```

例如:要输入一个含有“中大——LK”的程序行,并在打印机上印出,操作方法如下:

- 1) 在] □时,键PR # 3↵转为] _;
- 2) 在] _时,键10 PRINT “_”;
- 3) 紧接上述状态,按CTRL_D,再按CTRL-L;
- 4) 在荧屏左下角显示出‘中文’后,键 $\begin{matrix} L \\ \text{中} \end{matrix}$ 后按长条键,显示:

```
10 PRINT “中
```

键 $\begin{matrix} K \\ \text{大} \end{matrix}$ 后,按长条键,显示:

```
10 PRINT “中大
```

键 $\begin{matrix} = \\ \text{—} \end{matrix}$,显示:

```
10 PRINT “中大_
```

- 5) 按CTRL-L,左下角中文两字消失,键入LK荧屏显示为:

```
10 PRINT “中大—LK”
```

- 6) 按RETURN键(↵);

- 7) 在] _时,键入LIST 10↵

```
10 PRINT “中大—LK”
```

```
键RUN 10↵
```

显示: 中大—LK

8) 接通打印机电源时, 键 POKE 1403, 1 ↵
启动打印机打字头。

9) 键入LIST ↵

在打印机纸上印出:

```
] LIST  
10 PRINT "中大—LK"
```

10) 键入RUN ↵

在打印纸上印出:

```
] RUN  
中大—LK
```

11) 键入POKE 1403, 0 ↵

在打印纸上印出:

```
] POKE 1403, 0
```

这样便撤消打印机的打印作用。可试键入LIST ↵, 再键入RUN ↵, 这时只在荧屏上有显示。

12) 键入CALL 64098 ↵ 转为] □状态。

13) 在] □时键入LIST 10 ↵

```
10PRINT "IK—LK"  
RUN 10 ↵  
IK—LK
```

2. 汉字数据处理与实数BASIC综合程序设计

可将实数BASIC程序行与准AP程序行结合, 构成一个包含有汉字数据处理及实数BASIC各种功能的综合性程序。设计这种程序, 主要是以实数BASIC程序行为主体, 在适当地方插入启用英/汉板的语句及处理汉字的程序行, 再在适当地方转回实数BASIC程序行, 也可以先编制成子程序转有关的各种处理, 再以实数BASIC程序为主体, 适当地调用子程序。

下面有一个可供参考的程序段:

```
100 HIMEM:36864  
102 POKE 37984, 0  
104 PR # 3  
106 PRINT  
108 POKE 43603, 3  
110 POKE 43604, 192 + 3  
112 POKE 43605, 48  
114 POKE 43606, 192 + 3  
116 POKE 54, 189  
118 POKE 55, 158  
120 POKE 56, 129
```

```

122 POKE 57, 158
124 FOR I=1 TO 15
126 GET A$
128 NEXT I

```

这个程序段可接入实数 BASIC 程序内，在运行程序时，会自动地启用英/汉板，转入准AP状态。

在一个含有汉字处理及输出汉字的综合性程序中，大体上应包含下述的几个程序段（行）：

- (1) 含实数 BASIC 程序段。
- (2) 含启用英/汉板的程序行。
- (3) 含准AP程序段，其中有：
 - 汉字的输入，
 - 汉字的处理，
 - 汉字的输出（显示与打印输出），
 - 其它功能的程序行，
 - 转回实数 BASIC 状态的程序行。
- (4) 含继续运行实数 BASIC 程序的命令。

设计综合性程序的关键是编排好状态转接的语句及各种功能调用，各种外围设备的启用与编撤消命令（语句）等。

此外，可在程序中适当加入注解性的词语，用来提示操作者，使操作的人明白怎样去运行程序，运行中如何配合操作。

3. 汉字的组构

汉字的数目繁多，字的构成繁杂。因此，必须将汉字分解、归纳，从中抽取出构成汉字的基本成份来，作为基本的汉字字键。然后再用基本字键，按照一定的法则去组构出各个汉字。

这里以 C-PLUS] [A 英/汉板为例，用的是繁体汉字，组构汉字的方法是仓颉汉字输出法。这种方法，定出24个汉字为基本字母，作为字键。它们除了各自本身是一个汉字外，还各自分别代表若干种组构汉字的成份。

(1) 构字成份

①基本字母24个 主要是根据汉字的形成，演变的缘由而抽取出来的。

1) 由日常生活或自然界中常见的实物形象演变而来的，称为哲理类。有七个：

日 月 金 木 水 火 土

2) 由通常构字的基本笔划为特征而抽取的，称为笔划类。有七个：

竹 戈 十 大 中 一 弓

它们是分别由丿（斜）、丶（点）、+（交）、×（叉）、丨（中）、一（横）、丿（钩）的特征而定出的。

3) 以人体的器官形象演化而来的，称为字形类。有四个：

人 心 手 口

4) 取某些有代表性的汉字字形的象征而来的，称为字形类。有六个：

尸 艹 山 女 田 卜

它们分别象征] (侧)、艹 (并)、山 (仰)、女 (纽)、口 (方)、卜 (卜)。

这24个基本字母见表22.3, 组成构汉字的字键, 每一个字键可单独构成一个汉字, 如: 日、月……。也可以其本身作为一个构字的部分去构成别的汉字, 如日与月构成明。

表22.3 仓颉法构字的字母/字元表

哲理类	笔划类	人体类	字形类
<p>日</p> <p>A</p>	<p>尸</p> <p>竹</p> <p>[斜] H</p>	<p>人</p> <p>人</p> <p>人</p> <p>O</p>	<p>コ</p> <p>尸</p> <p>匚</p> <p>[侧] S</p>
<p>月</p> <p>B</p>	<p>戈</p> <p>点</p> <p>[点] I</p>	<p>心</p> <p>心</p> <p>心</p> <p>P</p>	<p>艹</p> <p>艹</p> <p>[并] T</p>
<p>金</p> <p>C</p>	<p>十</p> <p>[交] J</p>	<p>手</p> <p>手</p> <p>手</p> <p>Q</p>	<p>山</p> <p>山</p> <p>[仰] U</p>
<p>木</p> <p>D</p>	<p>又</p> <p>大</p> <p>[又] K</p>	<p>口</p> <p>R</p>	<p>女</p> <p>女</p> <p>[纽] V</p>
<p>水</p> <p>E</p>	<p>丨</p> <p>中</p> <p>[纵] L</p>		<p>口</p> <p>田</p> <p>[方] W</p>
<p>火</p> <p>F</p>	<p>工</p> <p>一</p> <p>[横] M</p>		<p>上</p> <p>ト</p> <p>ニ</p> <p>ネ</p> <p>Y</p>
<p>土</p> <p>G</p>	<p>丨</p> <p>弓</p> <p>[钩] N</p>	<p>注: 重难字 = x</p>	

②扩展字元 仅由上述的24个基本字母是不可能组构出很多汉字的。因此必须将上述的24个基本字母键的作用加以扩展，使每个字键还能充当若干种构成汉字的成份（字元），属于这一类的构字部分（字元），称为扩展字元。

各个基本字母键，主要依据下述的二个原则，扩充为扩展字元：

1) 与字母的形状相近，有关或其变形的部分。如：

人⇒ 亻、亼、亻、丿

月⇒ 夕、冂、冂、夕

田⇒ 口、口

2) 与字母的含义相近或有关的部分。如：

戈⇒ 丿、广、厶

竹⇒ 丿、厂

大⇒ 乂、ナ

弓⇒ 丿、冂、冂

一⇒ 工、厂

扩展字元并非一个独立的汉字，它只是在构成汉字中起到一个单元（字元）作用。

若要产生一个与扩展字元同样子的独立汉字，必须再用其它扩展字元去构成。如：

一（一）、丨（中）、一（一）构成工；

丿（竹）、丿（人）构成八；

冂（山）、丿（竹）构成匕。

这样，在仓颉汉字输入法中，有二十四个基本字母，由它们又可延伸约六十多种构字的单元（字元），故可用来组构汉字的单元（字母）约有六十多种，用来组构汉字的成份总共约有九十种之多。在使用英/汉板时，由它们可能组构出约两万三千个“字”，而其中有意义的汉字约六千多个。

（2）字的分类

①基本字（单键字） 二十四个基本字母键中的每一个，本身就可成为一个汉字。因此，可以独立用来构成一个汉字，这种字称为基本字或单键字。

要想在电脑内得出某个基本字，只要在电脑处于汉字数据状态下，打入该字的对应键后，再按下长条键。

如：日 打

A
日

 键，按长条键。

心 打

P
心

 键，按长条键。

山 打

U
山

 键，按长条键。

②组码字（多键字） 要打入若干次（2至5次）基本字母键（作为扩展字元）后，再按长条键才能得出的字称为组码字或多键字。每打一次基本字母键，也叫做取一个码，所以，一个码就是构成汉字时的一个基本部分，即构成汉字的一个单元（字元）。基本字以外的汉

字，都属于多键字，其构成比较复杂。

基本字以外的汉字，可以分为分体字和连体字两种。每一个字又可分解为字首与字身两个部分。

1) 分体字 一个字可以划分为若干个单元(字元)的字称为分体字。一般可按照左右，左中右，上下，外内等方式划分，分为字首与字身，或字首、次字身。

a 对并列式的字可按左右划分(分解)。如：明(日与月)、肚(月与土)、佳(亻与圭)、特(牛与寺)、标(木与票)等。

b 对多列式的字可按左中右划分。如：例、川、微、班、缀等。

c 对上下式的字可按上下分解。如：昌、字、学、最、量等。

d 对内外式的字可按外内分解。如：困、周、回、区、凶等。

2) 连体字 构成一个字的笔划相互交连，无法分离出独立的部分(单元、字元)。如：步、桌、业、角等。

对这种难于分解出字首与字身的字作连体字处理。

(3) 构字取码规则 要将一个汉字输入到电脑，即在电脑内得出一个汉字，首先要将该字划分成若干个相对独立的部分(单元)，使每个部分对应于一个基本字母或扩展字元，然后按照一定的法则取码，即取定若干个基本字母或扩展字元，再打入对应的键，最后按下长条键。取码的基本原则如下：

①取码要从简 当一个字可以有几种取码时，应按码数最少的情况取。

如：“王”有两种方法取码，第一种是取“一土”，第二种是取“一十一”，应取第一种方法；“容”的第一种取码方法是“十金人口”，第二种取码方法是“戈月金人口”，应取第一种方法；

“非”可取“中一卜卜”，也可取“中一中一一”，应按前一种方法取码。

②取码要完整 当一个字可有几种取码而码数又相同时，应考虑按字形较完整的码来取，在顺序上先取代表形状复杂的码。

如：“丰”可取“手十”，也可取“十手”，应取前一种；

“莹”可取“廿中手一”，也可取“廿中十土”，应取前一种；

“民”可取“口山心”，也可取“尸山心”，应取前一种。

③取码要注重字形特征 一个字的取码，除首先从上述的两个原则出发外，还要考虑到所选取的码是否确切地表示出该字的特征，在取的码数和繁杂程度差不多的情况下，应按较能反映该字特征的码来取。

如：“力”可取“大尸”，也可取“大弓”，应取前一种；

“也”可取“心木”，也可取“廿弓山”，应取前一种；

“乙”可取“弓山”，也可取“一山”，应取前一种；

“之”可取“戈弓人”，也可取“卜竹人”，应取前一种。

(4) 取码构字法 对于比较复杂的汉字，要分类依照上述的取码原则，把一个汉字分解为若干个单元(部分)，必须使每个单元对应于一个基本字母或扩展字元。在输入汉字时，就取对应的字键——取码去构成该汉字。方法如下：

①连体字 连体字有下列三种情形：

1) 笔划相互交连，无法分解的。如：步、焉。

2) 卜、マ、夕等上部分与下部分笔划相交的。如：桌，甬，角、业。

3) 儿，八等下部分与上部分笔划相连的。如：免，页，兒。

对于连体字，第1码要取为首的字元；第2、3、4码，则对该字的其余部分由上而下顺序划出的字元取。若最多只有4码，则可全部取。若一个字可划分为五个字元以上的，则只能按第一部分，第二部分，第三部分及最后一个部分相应地取第1、2、3、尾四个码。

如：函（弓山水） 第1码“了”，第2码“L”，第3码“=<”。

求（戈十水） 第1码“、”，第2码“十”，第3码“=<”。

舟（竹月卜戈） 第1码“、”，第2码“刀”，第3码“一”，第4码“、”。

鸟（竹口卜火） 第1码“ノ”，第2码“口”，第3码“卜”，尾码“灬”。

乘（竹木中心） 第1码“ノ”，第2码“木”，第3码“丨”，尾码“匕”。

商（卜金月口） 第1码“一”，第2码“ノ”，第3码“冂”，尾码“口”。

②分体字 对分体字的取码比较复杂。首先要将一个汉字分解为字首与字身两个部分，然后再分别取码。

1) 字首与字身的分解 主要是划出字首，其余部分为字身。

(a) 凡可以作上与下或左与右独立分解的字，上部分或左边部分，划为字首。

(b) 一个字的上、外、左侧部分凡有口、匚、厂、广、疒、尸、户、夂、彳、走、風、毛、戈、气、瓜、支等独立部分的，划为字首。

(c) 一个汉字，若作上下分解后，上部可分离成八、人、父、夹、冂、𠂇、高、冂、𠂇等独立部分的，划为字首。

(d) 一个汉字若可分离成戈、气、弋、夂、戊、产等独立部分的，划为字首。

2) 取码法则 对一个分体字，可取2至5个码，最多5个码，要分别对字首和字身取码。

(a) 字首部分可取1至2个码，最多2个。如果字首部分可以再分解为几个字元的话，也只能对应于第一个字元和最末的字元取首码与末码。其余的中间部分略去。

如：枝 字首为“水”，取1码“木”。

休 字首为“亻”，取1码“人”。

量 字首为“日”，取1码“日”。

疫 字首为“疒”，取首码“戈”，末码“卜”。

照 字首为“昭”，取首码“日”，末码“口”。

(b) 字身部分可取1至3个码，依次为次1码，次2码至尾码。

若整个字身部分只划分为三个字元以下，则对应各字元依次全都取码。如：

科 字身为“斗”，取次1码“卜”，尾码“十”。

破 字身为“皮”，取次1码“木”，次2码“竹”，尾码“水”。

若整个字身部分属于连体字，则只能按次1码，次2码，尾码依次取码，不得超过3个。如：

睡 字身为“垂”，取次1码“竹”，次2码“木”，尾码“一”。

若整个字身部分属于分体字，又要再分解为次字首与次字身，分别两种情形去取码。

第一种：若次字首只是一个字元，则次字首取此字元相应的码为次1码。次字身可取次2码及尾码。如：

係 字身为“系”，次字首为“丿”，取次1码“竹”，次字身为“系”，取次2码“女”，尾码“火”。

第二种：若次字首还可分解为二个字元以上，则只能对应于次字首的第一个字元取次1码，对应于次字首的最后字元取次2码。而次字身部分不管还可如何分解，也只能对应其最末尾的字元取为尾码。如：

略 字首：田；字身：各；次字首：夕，取次1码“竹”，次2码“水”；次字身：“口”，取尾码“口”。

缀 字首：系；字身：段，次字首：扌，取次1码（竹），次2码（十）；次字身：爻，取尾码“水”。

3) 取码构字省略某些部分的原则 在取码构字的过程中，如遇到可分解的部分繁多（含有五个字元以上）的字，由于最多只能取4码（连体字）或5码（分体字），因此必须省略某些部分，省略的原则为：

(a) 省略字的中间部分 如：

页 作字身取码为“一竹金”，省去“目”。

(b) 省略内含部分 当遇到口、刀、匚、凵、儿、工、王、夕、土等内部还含有其他笔划，而又要作省略时，就省去其内含的部分。如：

商 作字首取码为“卜月”，省去刀内的“台”部分。

齿 作字首取码为“卜凵”。

面 作字身取码为“一竹田”。

(5) 例外情形

由于汉字的构成非常复杂，除了上述的法则之外，还有一些无法归纳进去的情形。在取码构字时，要作特别的处理。

① 复合字元 有一些可用来构字的独立部分，它本身又包含着复杂的结构，难于再分解取码。则将这些独立部分划为复合字元。它们不论是作字首还是字身（一般不能独自作一个汉字），都规定只取其第一与最后的字元所对应的两个码。规定如下：

門	“日弓”	們	“人日弓”
目	“月山”	眼	“月山日女”
鬼	“竹戈”	魁	“竹戈卜十”
九	“竹山”	肌	“月竹山”
門	“中弓”	鬲	“中弓卜中月”
卩	“弓中”	部	“卜口弓中”
佳	“人土”	淮	“水人土”
气	“人山”	氛	“人山金尸竹”
幾	“女戈”	幾	“女戈人”
声	“卜心”	虐	“卜心尸一”
盲	“卜口”	羸	“卜口月女山”

② 难字 有些汉字（或相对独立的一部分）很难明确划分为对应基本字母或扩展字元的部分，因而很难明确取出若干个码来构字。这一类的字不多（约12个），可借助于“难字键”，以“难码”代替无法取出的码。

1) 若字的第一码与尾码容易取出, 则取第一码、难码、尾码。

如: 身 “竹难竹”
兼 “甘难金”
鹿 “戈难心”
疒 “中难竹”
册 “中难中”

2) 若其尾码也难取, 则取第 1 码及难码二个码。

如: 白 “竹难”
肅 “中难”
齋 “卜难”

③特殊字形 在一个汉字中, 凡含有大、水、火三种字形的话, 不论有无其它笔划(字元)横贯于其中, 首先按该字形取码, 然后再取其馀部分。

如: 東 “木田”
束 “木中”
束 “木月”
東 “木田火”
末 “木十”
秉 “竹木中” (丿不算横贯于木)
本 “水一”
夷 “大弓”
爽 “大大大大”
券 “火手尸竹”

④同码异字 有些汉字, 虽字形不同, 但所分解出的字元却相同, 因此取码也就相同了。这就会造成同码异字的情况。为了不致混淆, 把日常中比较起来多用, 常见的字, 定为“本字”, 而另者为“重字”。在取码构“重字”时, 要先按“重”字键(X键, 难键)。若该字要取的码已有 5 个, 则要删去尾码。如:

代码	“本字”	“重字”
(日)	日	日
(日月十十)	晖	晕
(日卜口火)	晾	景
(月一)	且	肛
(木竹水)	皮	板
(日弓日山)	晚	冕

有时, 也可以在按一般取码而构出“本字”时, 再补打“重”字键, 便即构出“重字”来。

如键入“日月十十”, 按长条键, 构出“晖”, 再打“重字”键则构出“晕”。

汉字的组构是一项比较繁琐而困难的工作, 尤其是对一些笔划结构繁杂的字。但只要遵循上述的法则和方法(参看图 22.1 汉字分划取码流程图)去做, 一般是可以构出所要求的汉字来的。当然, 也会有不成功的情况, 这时可再改试另一种取码或查阅随机所附的“字典”。

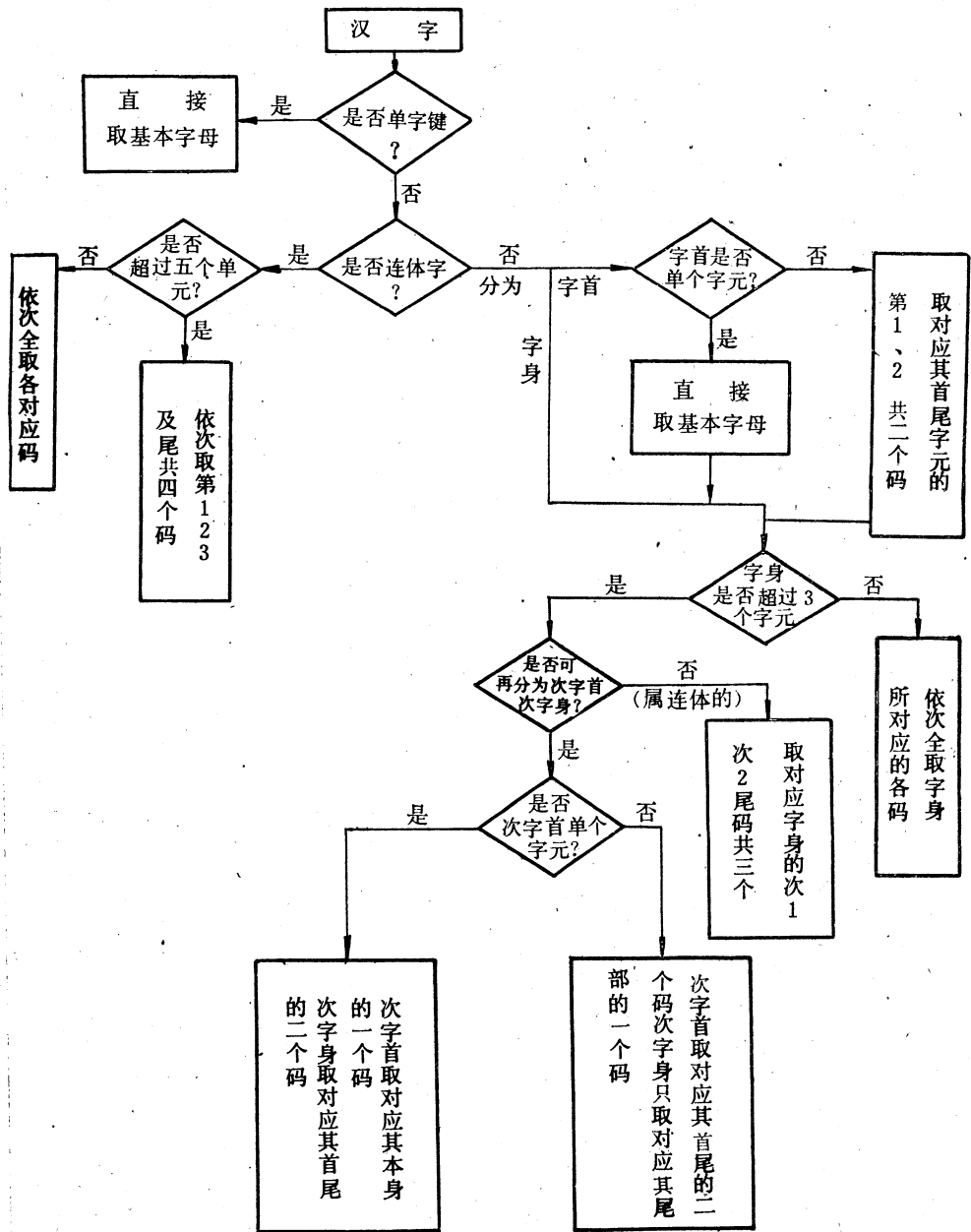


图22.1 汉字分划取码流程图

4. 微电脑汉字功能的运用

APPLE II PLUS (或同类性能的) 英文微电脑系统, 加入C-PLUS Ⅱ A (或同类) 英/汉板后, 具有汉字终端的功能。可通过键盘把汉字输入系统或把已存入磁盘的汉字载入系统内存, 再运用程序对汉字进行处理及输出 (荧屏上显示及打印机上印出)。运用汉字终端的要点如下:

(1) 把汉字作为字符型数据,就可按实数 BASIC 体系去输入,处理及输出。
(2) 在实数 BASIC 状态下,转为准AP状态,再转为汉字状态,才能通过键盘组构汉字。

- (3) 要在准AP状态下,才能把组构出的汉字输入内存。
- (4) 有关汉字的程序段(行),必须转到准AP的状态运行。
- (5) 运用语句(命令)在程序的运行中作状态的转换,连接及调用各有关功能。
- (6) 可把有关汉字处理,输出的程序段(行)编排入实数 BASIC 程序中。

下面举一个应用的例子。

例:把三个汉字名称(中山大学物理系、技工学校、化工仪表厂)合并为一个名称,并在打印机上以四种格式(横式大字、小字,竖式大字、小字)印出。

设英/汉板插入3号插座,打印机接口板插入1号插座。程序如下:

```
10 HOME
20 INPUT "CHINESE CARD SLOT CN=? "; CN
30 HM=49152+256*CN+21
40 PR# CN
50 CALL HM
60 A$="中山大学物理系"
70 B$="技工学校"
80 C$="化工仪表厂"
90 PRINT A$;PRINT B$;PRINT C$
100 E$=A$+"-" +B$+"-" +C$
110 PRINT E$
120 CALL HM
130 PRINT "请开打印机"
140 FOR I=1 TO 1000:NEXT I
150 PRINT "开了打印机?(Y/N)"; P$
160 GET P$: PRINT P$
170 IF P$ <> "Y" THEN 130
180 POKE 1400+CN, 1
190 FOR K=3 TO 0 STEP -1
195 POKE 1656+CN,-K
200 PRINT " ";
210 PRINT E$;
220 PRINT " * *"
230 PRINT:PRINT
240 NEXT K
250 POKE 1400+CN, 0
260 CALL HM:CALL 64098
```

在LASER-3000机上练习操作以下程序


```

10 POKE 1405, 1
20 POKE 1661, 18
30 PRINT "      欢      迎      "
40 POKE 1661, 2
45 PRINT "同志们参加微机师资班学习"
50 PRINT "      江门市技工学校 1984.3"
60 GOTO 10
70 END
RUN

```

操作方法：微机开机后在 BASIC 状态下，由于LASER—3000汉卡是插在第5号插座内，所以键入PR# 5使进入AP状态，即调入中文（汉卡）发生器在AP状态下键入10至30 PRINT “ 时，键入CTRL—L，这时荧幕左下方显示中文两字，这时用仓颉法键入中文，即NKNO（欢）IOHUL（迎）”，接着键入 CTRL—L 退出中文状态/换行，键入40至45 PRINT “CTRL—L，键入BMR（同）GP（志）XOIN（们）IKHH（参）KSR（加）HOUUK（微）LWU（电）BYKU（脑）LLMLB（师）YOBO（资）MGILG（班）EBND（学）SY（习）” CTRL—L/

50 PRINT “CTRL—L，按6—8次空格后，继续键入EM（江）IN（门）JB（市）QJE（技）MLM（工）EBND（学）DYCK（校）” CTRL—L/换行，60……。

注：

POKE 1661, 2	横向小字
POKE 1661, 3	直向小字
POKE 1661, 16	横向大字
POKE 1661, 17	直向大字
POKE 1661, 18	横向中字
POKE 1661, 19	直向中字

习题参考答案

第三讲 微电脑的内部工作过程

1. 分三步。

2. 10 READ A, B, C
 20 DATA 2, 1, X
 30 LET X = 4
 40 LET Y = A * C ↑ 2 + B
 50 PRINT "Y =", Y
 60 END

第四讲 电脑中数的表示方法

1. 微电脑机器的机理对导电 1 和断电 0 很敏感,它能够接收 1 和 0,而 1 和 0 与二进制相适应。

2. 01 数为高位, 10 为低位。

3. A D F
 ↓ ↓ ↓
 1010 1101 1111
4. 1100 1110 1011
 ↓ ↓ ↓
 C E B

第六讲 BASIC 及其编程知识

3.

- (1) 10 LET X = 4
 20 LET Y = 5 * X ↑ 2 + 2 * X + 2
 30 PRINT Y
 40 END
- (2) 10 LET X = 6 * 5 + 2 * 2 + 4 / 2
 20 PRINT X
 30 END
- (3) 10 LET X = (6 * 5 + 2 * 2 + 4) / 2
 20 PRINT X
 30 END
- (4) 10 LET X = 6 + 4 / 2
 20 PRINT X

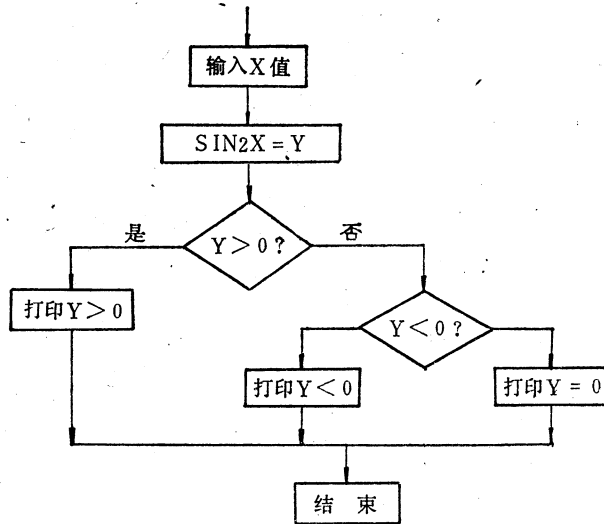
```

30 END
(5) 10 LET X = (6 + 4) / 2
20 PRINT X
30 END

```

第九讲 程序设计（一）——转移及循环

1. 框图是：



程序是：

```

10 PRINT "X=";
20 INPUT X
30 LET Y = SIN(2 * X)
40 IF Y > 0 THEN 80
50 IF Y < 0 THEN 100
60 PRINT "Y = 0"
80 PRINT "Y > 0"
100 PRINT "Y < 0"
110 END

```

第十讲 程序设计（二）——主程序与子程序

1. 顺序：

10	20	30	100	110	115	200
210	220	120	40	50	210	220
52	54	300	310	315	100	110
115	200	210	220	120	320	56

60

2. 打印结果

1 2 3 4 5

第十一讲 程序设计（三）——数组

```

1. 10 DIM S(4, 8)
    20 FOR Q=1 TO 4
    30 FOR R=1 TO 8
    40 READ S(Q, R)
    50 PRINT " ", S(Q,R);
    60 NEXT R
    70 PRINT
    80 PRINT
    90 NEXT Q
100 DATA 1,3,5,7,9,11,13,15,2,4,6,8,10,12,14,16
110 DATA 11,13,15,17,19,21,23,2,5,22,24,26,28,30,32,34
120 END

```

RUN

```

 1  3  5  7  9 11 13 15
 2  4  6  8 10 12 14 16
11 13 15 17 19 21 23  2
 5 22 24 26 28 40 42 34
S(2,2) = 4
S(2,7) = 14
S(3,1) = 11
S(1,3) = 5

```

2. 运行结果

```

 2  3  4
 3  4  5
3. 10 DIM A(2,4),B(4,2)
    20 PRINT "MATRIX A"
    30 FOR I=1 TO 2
    40 FOR J=1 TO 4
    50 READ A(I,J)
    60 PRINT A(I,J);
    70 NEXT J
    80 PRINT
    90 NEXT I
100 PRINT "MATRIX B"
110 FOR I=1 TO 4
120 FOR J=1 TO 2
130 LET B(I,J) = A(J,I)
140 PRINT B(I,J);
150 NEXT J

```

```
160 PRINT
170 NEXT I
180 DATA 1,2,3,4,5,6,7,8
200 END
RUN
MATRIX A
  1  2  3  4
  5  6  7  8
MATRIX B
  1  5
  2  6
  3  7
  4  8
```

第十四讲 华宇—200(LASER—200、LASER—310)键盘操作(一)

1. 结果: 9
2. 结果: 7
3. 结果: 7
4. 结果: 5
5. 结果: $I = .566667$
ERR 15 Q 10
6. 结果: $XM - 2 + 3$

第十五讲 华宇—200(LASER—200、LASER—310)键盘操作(二)

1. 结果: LET
2. 结果: /
3. 结果: TAB
4. 结果: 光标向左移一格, 并保护该右一格内的字符不被删掉。

附 录

附录 I 华字—3000 (兼容LASER—3000、APPLE II)

实数 BASIC 命令汇集



以下是华字—3000 (兼容LASER—3000,APPLE II) 实数 BASIC 的命令汇集,共105个命令(语句)格式。凡能举例的,一般都有范例。凡命令后有*号的,不能单独使用,要与其他命令联用。


1. ABS (数值) *


对括号 () 内的数值取绝对值。

[例] ? ABS (-2.73)

执行结果为2.73。

2.  及 

按一下  键,光标右移一格,并将经过的字符输入。荧幕上不立即消除。

按一下  键,光标左移一格,并将经过的字符删去。

3. ABC (字符串) *

得出括号 () 内的字符串第一个字符的ASCII码。

[例] ? ABC ("QUT")

执行结果为81。J

4. AND

逻辑“与”。

[例] 1 AND 1

执行结果为1;

0 AND 1

执行结果为0;

1 AND 0

执行结果为0;

0 AND 0

执行结果为0。

5. ATN (数值)

对括号 () 内的数取反正切函数值,单位为弧度,在 $-\frac{\pi}{2}$ 到 $\frac{\pi}{2}$ 之间。

[例] ? ATN (2)

执行结果为1.10714872弧度。

6. CALL

转去执行其后面所给存贮地址的机器语言（汇编语言）子程序。

〔例〕 CALL—151

执行结果是转到监控状态，荧幕上显示*。

7. CATALOG

列出磁盘内所贮存的各文件名，属DOS命令。

8. CHR\$ (数值)

得出括号 () 内ASCII码所对应的字符，括号内的数值必须是0~255。

〔例〕 CHR\$ (65)

执行结果为A。

9. CLEAR

置所有的变数为0；置所有的字符串为空白；置数组读数指针于始点；内存的程序不变。

10. CLR

清除全部变数。

11. COLOR =

在低分辨的图形显示时，设定显示的颜色。

〔例〕 COLOR = 13

设定为黄色。

12. CONT

使中断执行后的程序继续执行。

13. COS (弧度值)

取括号 () 内数值为余弦函数值，括号内的数以弧度为单位。

〔例〕 ? COS (2)

执行结果为1.415146836。

14. [CTRL]

控制键，属于键盘命令，要与其它键连用。

〔例〕 [CTRL] [C]

中断正在执行的程序；中断列表。

[CTRL] [X]

将刚打入的一行作废。

15. DATA 数据，数据，数据

把数据依次存入内存的数据区。

〔例〕 DATA JOHN, "CODE 32", 23.45, -6

将文字JOHN和字符串 "CODE 32" 数23.45及数-6依次存入内存。

16. DEF FN 变量名 (变量) = 表达式

使用者在程序中自己定义一个函数。

〔例〕 DEF FN C(R) = 2 * 3.14 * R

用户自定义圆周长的函数C(R)，当R的值为3时，由? FN C(3)可得出函数值为

18.84。

17. DEL 行号 1, 行号 2

删除行号 1 到行号 2 的程序。

〔例〕 DEL 5,20

执行结果将第 5 行到第 20 行的程序删去。

18. DIM 变量名 (数值)

设定数组 (包括名称, 维数, 容量)。

〔例〕 DIM A(80)

设定名称为 A 的一维数组, 有 81 个数据。

DIM N\$(50)

设定名称为 N\$ 的字符串数组, 字符串的个数为 51 个。

19. DRAW N AT X, Y

在高分辨绘图状态下从 (X, Y) 位置开始绘出 N 所对应的图案。内存已有图案表。

〔例〕 DRAW 4 AT 50, 100

执行结果从 X = 50, Y = 100 处开始绘出图表中 4 所对应的图形。

20. END

结束程序的执行。

21. ESC

脱开键, 属键盘命令, 要与其他键联用。

〔例 1〕 ESC A

使光标向右移一格。

〔例 2〕 ESC B

使光标向左移一格。

〔例 3〕 ESC C

使光标向下移一格。

〔例 4〕 ESC D

使光标向上移一格。上述四例, 不影响内存及荧幕上的字符。

22. EXP (数值)

取 e 为底, 括号内的数值为指数值。

〔例〕 ? EXP(2)

执行结果为 7.3890561

23. PLASH

置荧幕显示为黑白交替 (闪烁) 方式。

24. FOR X=X1 TO X2 STEP X3

⋮

⋮

NEXT X

不断地依次给变量赋值, 可用来组成循环语句。

〔例〕 FOR I = 0 TO 30 STEP 3

⋮
⋮

NEXT I

第一次取 I = 0，执行到NEXT后，第二次取 I = 3，又执行到NEXT。第三次取 I = 6，又执行到NEXT……。直至 I > 30；执行NEXT I 的下一条语句。

25. FRE (0)

示知使用者，存储器中还有多少字节的存储容量。

〔例〕 ? FRE (0)

会显示出当前还剩下多少字节的存储量。

26. GET

设置在程序运行中，由键盘打入一个字符。键入后，字符不在荧幕上显示。

〔例〕 GET A\$

程序执行到此句时，自动暂停。等操作者打入一个键盘字符后，会将其存入字符串变量 A\$ 内，并接着运行程序。

27. GOSUB 行号

使顺序执行着的程序跳到指定行号去继续执行。用来调用子程序，要与RETURN 联用。

〔例〕 GOSUB 120

转到入口为120行号的子程序去继续执行。

28. GOTO 行号

使顺序执行着的程序跳到指定行号去继续执行。

〔例〕 GOTO 215

无条件地转到215行继续执行。

29. GR

将显示荧幕设置为低分辨图字混合型式。图有40×40个格，并在下面留4行作程序语句的显示用。低分辨图形显示时各种颜色及代表数见附表1。

附表1 低分辨图形显示时各种颜色及其代表数

代表数	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
颜色	黑	靛	深蓝	紫	深绿	灰	中蓝	浅蓝	棕	橘	灰	粉红	绿	黄	水	白

30. HCOLOR =

设定在高分辨图形时绘图的颜色。八种颜色的代码见附表2。

附表2 高分辨图形显示时各种颜色及其代表数

代表数	0	1	2	3	4	5	6	7
颜色	黑1	绿*	蓝*	白	黑2	橘*	蓝*	白

* 表示实际颜色会因不同的显示器（电视机）有差异。

〔例〕 HCOLOR = 2

设定绘图的颜色为蓝色。

31. HGR

将显示荧幕设置为高分辨图字混合型式，有 280×160 个小点，显示存贮器第一页。并在下面留4行作显示程序语句用。有下列范围：

HGR

HGR 1

HGR 2

HGR 3

HGR 4

HGR 5

HGR 6

〔例〕 HGR 2

设置为全荧幕显示高分辨图型，有 280×192 个小格，显示存贮器第二页。

32. HIMEM

设定 BASIC 程序在存贮器中可用的最高地址。

33. HLIN X1, X2 ATY

已经设定颜色，图形为低分辨图形时，在指定的位置上画一条水平线。

〔例〕 HLIN 10, 20 AT 30

假定已设定黄色。执行结果由(10, 30)为起点到(20, 30)为终点，画出一条黄色水平线。

34. HOME

清除荧幕上全部字符，但内存不变。

35. HPLOT X, Y

HPLOT X1, Y1 TO X2, Y2 TO.....

已经设定了颜色，在高分辨图形时画出点或线。

〔例1〕 HPLOT 12, 24

假定已设定了黑色在高分辨图形时，在(12, 24)处画一个黑点。

〔例2〕 HPLOT 32, 41 TO 56, 68

假定已设定为蓝色，在高分辨图形时，由(32, 41)到(56, 68)画出一条蓝色的线条。

36. HTAB X

把光标沿水平方向移到X所指定的位置。

〔例〕 HTAB 23

把光标从所在行移到第23格。

37. IF 条件 THEN 执行内容

如果条件满足，则按执行内容执行，否则，接下一语句执行。

〔例〕 IF X > 8 THEN 35

如果变数X大于8，则跳到第35行继续执行。否则继续执行下一行。

38. IN #

选定一个外围输入插座（或信号），插座代表1—7，实际是0—7号。

〔例〕 IN # 4

选编号为4的外围插座作输入。

39. INPUT

设置在程序运行中由键盘输入资料。

〔例1〕 INPUT "X=" ; X

程序执行到此句时，荧幕上出现 X = 的信号，等待操作者输入一个数据赋予X。

〔例2〕 INPUT X

只显示出?号，以提示操作者输入数值。

40. INT (数值) *

求括号内的数为不大于它的最大整数。

〔例1〕 ? INT (2.999)

执行结果取2。

〔例2〕 ? INT -45.123345

执行结果取为-46。

41. INVERSE

荧幕的显示设定为白底黑字（反常显示方式）。

42. LEFT \$ (字符串, 数值) *

取所给定字符串中最左边的由数值指定个数的字符。

〔例〕 ? LEFT\$ ("APPLE SOFT",5)

执行结果显示APPLE（5个字符）。

43. LEN (字符串) *

计算括号内的字符串的长度，空格也算一个字符。

〔例〕 ? LEN ("AN A")

执行结果为4。

44. LET X =

对变量赋值（有的电脑LET可以省去）。

〔例〕 LET A = 28.75

让A的值为28.75。

45. LIST 行号1, 行号2

〔例〕 LIST 25, 365

将当时内存的程序中25行至365行列示出。

或 LIST

将内存中的程序列示出来，列出当时内存所有的程序。

46. LOAD

将存于磁带或磁盘中的程序读取到主机的内存RAM中。读取磁盘时应给定文件名。

47. LOG (数值)

取括号内的数的自然对数。

0 OR 1

执行结果为1。

1 OR 0

执行结果为1。

0 OR 0

执行结果为0。

58. NOT

逻辑“非”。

〔例〕 NOT 1

执行结果为0。

NOT 0

执行结果为1。

59. PDL (号数)

求出括号内所指定的控制摇杆当前数值。

〔例〕 ? PDL (3)

显示出第3号摇杆的当前数值。摇杆号码0至3有效，当前值0至255。

60. PEEK ()

求出括号内所指地址的存储器内容。PRINT用选定的颜色使荧幕的某一范围着色。括号内的数用十进制。

〔例〕 ? PEEK (37)

执行结果显示地址为37的存储器的内容。

61. PLOT X, Y

已设定颜色，在低分辨图形时，在指定的位置(X, Y)画出一方块。

〔例〕 PLOT 15, 28

设定为黄色，图形为低分辨图形时，在(15, 28)处画出一个黄色方块。

62. POKE 地址, 数值

把数值存入指定地址的存储器内，数值只能0~255。

〔例〕 POKE 16321, 255

将数值255存入地址为16321的存储器内。

63. POP

在放RETURN返回地址的堆栈顶去掉一个地址，使子程序中RETURN返回地址被改变。

64. POS (参数)

*

给出当前光标在水平方向的位置(0至39)，括号内的参数，可任意给定一个数或赋值变量。

65. PR # 数值

设定某个外围插座为输出，插座号数由0至7。

〔例1〕 PR # 0

把输出的资料送到荧幕显示。

[例2] PR # 3

把输出的资料送到第3号插座。

66. PRINT 或 ?

把字符在荧幕上或其他输出设备上显示出来。是设定输出的语句。其后面可跟几种型式的内容。

[例1] PRINT

执行结果是显示中空一行。

[例2] PRINT 10 - 4

执行结果显示6。

[例3] ? "10-4=" , 10 - 4

执行结果, 显示10 - 4 = 6

[例4] PRINT "ONE THIRD IS EQUAL TO" ; 1/3

执行结果, 显示出ONE THIRD IS EQUAL TO .333333333。

67. READ A, B, X *

从程序中的DATA语句中读取数据, 逐一依次赋给READ后面的变量A、B、X。要与DATA语句联用。

68. RECALL MX

从磁带中读取存在其中的整数或实数数组, 事先需使磁带机正常运转。

[例] RECALL MX

从磁带中取出数组MX的数元MX(0), MX(1), MX(2)……依次存入内存中。

69. REM……

注释语句。用在程序中插入一些注释, 以便阅读程序的人看懂。这个程序行不占用执行时间。

[例] REM THIS IS SUB-1

注明这是第1个子程序。

70. REPT

重复键, 属键盘命令。当按住任何一个字符键时, 再按此键, 则重复出现该字符。

71. RESTORE

将读取数据的起点, 重新设定在第一个DATA后的第一个数据, 以便重新使用。

72. RESUME

将此命令放在错误处理子程序最后面, 当该子程序执行完后, 即转回到刚才发生错误的该语句再继续执行。(主要用于调试程序。)

73. RETURN

返回最近执行过GOSUB的下一个语句去继续执行, 即在执行过子程序后返回主程序。要逐字键入的命令, 一般以✓代表。

RETURN

回复键, 属键盘命令。在RUN命令之后或键完一行程序语句后, 要按此键。

74. RIGHT\$ (字符串, 数值) *

将括号内的字符串中最右边由数值指定的几个字符取出来。

〔例〕 ? RIGHT\$ ("COM APPLE" , 5)

执行结果显示APPLE。

75. RND (数值) *

得出一个随机函数的随机数, 该随机数在 0 与 1 之间 (不包含 1), 依据号内的数不同而不同。

76. ROT = 数值

对DRAW或XDRAW所得出的图案再设定其转动的角度。

〔例 1〕 ROT = 0

由DRAW或XDRAW所得出的图案不再转动。

〔例 2〕 ROT = 16

由DRAW或XDRAW所得出的图案沿顺时针转90°。

〔例 3〕 ROT = 32

将图案沿顺时针转180°。

77. RUN

运行已在内存中的程序, 从行号最小的语句开始。

〔例 1〕 RUN 155

从155行开始执行程序。

〔例 2〕 RUN ANNUITY

执行存于磁盘中文件名为ANNUITY的程序, 属于磁盘操作命令。

78. SAVE

将存于内存RAM中的程序, 通过磁带机录存于磁带, 即向磁数据带写入一个程序。

〔例〕 SAVE ADDRESS

将存于RAM中的程序, 以文件名ADDRESS录存于磁盘, 属于磁盘操作命令。

79. SCALE = 数值

对DRAW或XDRAW绘图案时, 设定比例。数值为 0 ~ 255 (扩大或缩小)。

〔例 1〕 SCALE = 1

按图案表定义一点一点画。

〔例 2〕 SCALE = 0

取最大的尺寸。

80. SCRN (X, Y) *

在低分辨图形时, 求出括号内X、Y所指定的位置上的颜色代码。

〔例〕 ? SCRN (15, 22)

得出在X = 15, Y = 22处的色点的颜色代码。

81. SGN (数值)

根据括号内的数而取符号。

〔例 1〕 SGN (正数)

取 1 (取正号)。

〔例 2〕 SGN (负数)

取 - 1 (取负号)。

[例3] SGN(0)

取0。

82. SHLOAD

从磁带中把图案定义表载入内存RAM中(放在HIMEM所定的地址下面)。

83. SIN(弧度值) *

得出括号内(单位为弧度)的数的正弦值。

[例] ? SIN(2)

执行结果为.909297427。

84. SPC(数值) *

用在PRINT语句中,使空出括号内所定的数目的空格。

[例] ? SPC(8) "***"

空8格后再显示***。

85. SPEED=数值

设定字符传送到荧幕显示或其他输入/输出装置的速率。数值0至255之间。

[例] SPEED=255

设置最大的传送速率。

86. STOP

暂停程序的运行,并回到键盘命令状态,可由操作者控制机器下一步做什么。

87. STORE

把内存RAM中的某个指定的数组录存于磁带上。磁带机的运转要操作者控制。

88. STR\$(数值) *

把括号内的数值变换为数字式字符串。

[例] ? STR\$(12,45)

执行结果为字符串"12.45"。

89. SQR(数值) *

取括号内的数的算术平方根。

[例] ? SQR(2)

执行结果为1.41421356。

90. TAB(数值)

通常用在PRINT语句中,使光标沿水平方向向前移到数值所规定的位置后,再开始显示或打印。数值可由0至255。

[例] ? TAB(10); "#"

在第10格上显示"#".

91. TAN(弧度值) *

取括号内弧度的正切值。

[例] ? TAN(2)

执行结果为-2.18503987。

92. TEXT

把荧幕显示设定为字符方式。有24行,每行40个字符。

93. TRACE

把执行着的程序的行号显示出来，以便跟踪查错。

94. TROFF

阻止程序语句，执行程序到这语句时，停止运行并且将该行号显示出来。

95. USR (数值) *

把括号内的数值送至机器语言子程序中，给予程序一个参数。

96. VAL (字符串) *

将括号内的数字式字符串转换成数值。当遇到非数字字符时，便不再转换。

[例] ? VAL (" -3.7E 4A5PLE")

执行结果为 -37000。

97. VLIN Y1, Y2 AT X

在低分辨图形显示时，从指定的起点到终点画出一条垂直线。

[例] ? VLIN 10, 20 AT 30

执行结果是由 (30, 10) 到 (30, 20) 间画出一条直线。

98. VTAB (数值)

把光标沿竖直方向移到括号内的数值所指定的行上。行数由 1 到 24。

[例] VTAB(15)

将光标移到15行。

99. WAIT

使程序的执行暂停下来，一直到某个特定的地址的存储器内的数值达到预先设定的值时，再继续执行程序，即等待检测。

100. XDRAW M AT X, Y

在高分辨图形显示时，在 X, Y 所指定的位置为起点，把 M 所代表的图案画出来，所用的颜色为原来各点的颜色之互补色。

101. NOISE

从指定程序内产生声音。

102. SOUND

从各程序内产生声音。

103. SWAP

交换两个变数的数值。

104. TRON

使一个程序的行号在被执行时显示出来。

105. XDRAW

除去一个已画出的图形。

附录Ⅱ 微电脑LASER—310BASIC摘要

1. 数学符号

+ 加号

- 减号

* 乘号

/ 除号

↑ 乘幂号

AND 逻辑“与”运算符

OR 逻辑“或”运算符

NOT 逻辑“非”运算符

> 大于号

< 小于号

= 等于号

>= 大于等于号

<> 不等于号

2. 求数学函数值命令 (又称为数学函数)

SQR () 求括号内数值 (应大于零) 的平方根。

INT () 取括号内数值的整数。

RND () 取随机值。

ABS () 取括号内数值的绝对值。

SGN () 依括号内的数而取符号, 当括号内的数大于零时取+; 当括号内的数小于零时取-; 当括号内的数等于零时取0。

SIN () 求括号内数值的正弦值。

CON () 求括号内数值的余弦值。

TAN () 求括号内数值的正切值。

ATN () 求括号内数值的反正切值。

EXP () 求e为底括号内数值为指数的幂。

LOG () 求括号内数值的自然对数值。

3. 字符转换与处理命令 (又称字符函数)

LEN () 求括号内字符串的长度 (字符个数)。

STR\$ () 将括号内的数值转换为字符串形式。

VAL () 将括号内的数字性字符串转换为数值形式。

ASC () 将括号内字符串的首字转换为ASCII码。

CHR\$ () 将括号内的ASCII码转换为字符。

LEFT\$ () 取出括号内所指定的字符串的左段。

RIGHT\$ () 取出括号内所指定的字符串的右段。

MID\$ () 取出括号内所指定的字符串的中间段。

4. 操作及功能命令

(1) 与运行程序有关的命令

LIST 列示内存中的程序。

RUN 运行内存中的程序。

NEW 清除内存中的程序。

CONT 继续运行中断(暂停)执行的程序。

CLS 清去屏幕上的显示。

(2) 与外存贮(磁带机)有关的命令

VERIFY 核对外存(磁带)与内存中的程序。

CSAVE 将内存中的程序录存于磁带。

CLOAD 将录存于磁带的程序载入内存。

CRUN 将录存于磁带的程序载入内存并即运行。

(3) 与打印机有关的命令

LLIST 在打印机上印出程序。

LPRINT- 在打印机上印出资料(数据)。

COPY 将屏幕上显示出的内容在打印机上印出。

(4) 有关输入/输出的命令

POKE 数值, 数值 将给定数据存入指定的内存地址。

POKE () 从指定的内存地址中取出其内容

INKEY\$ 设定从键盘输入一个字符, 而赋予指定的字符型变量。

INP () 设定资料的输入口。

OUT (..., ...) 设定资料的输出口并将给定的资料输出。

USR () 与机器语言子程序联接。

5. 编程语句

(1) 有关数据传送的命令

LET = 运算并赋值于变量。

INPUT 设定在程序运行中才通过键盘输入数据。

INPUT # 设定从录存于磁带上指定文件名中读取数据。

DATA 将一批数据依次存入内存的数据区。

READ 从内存数据区中依次读取数据, 并赋予变量。

RESTORE 恢复已被读取过的数据的作用, 使下一次读取数据时又从头开始。

PRINT # 设定将数据送存于磁带上指定的文件名中。

(2) 有关屏幕显示的命令

PRINT 设定显示的内容及格式。

PRINT TAB (.....) 由括号内数值(0~255)指定行内的显示位置并显示给定的内容。

PRINT USING "....." 以“ ”内的符号(! # + - * * " \$ \$ * * \$ / %)设置特定格式显示。

PRINT@, 以@后面的数值(0~511)指定屏内显示的位置, 并显示,

号后面的内容。

(3) 有关图形的命令

COLOR , 设定显示图形的颜色。

SET () 在屏幕上指定的位置显示一个点。

POINT () 将屏幕上指定位置的显示点的颜色转换为代码。

RESET () 将屏幕上指定位置的显示点清除掉 (变为背景色)。

MODE () 设定屏幕显示方式。当括号内的数值为 1 时, 是高分图形方式, 可有 8 种颜色, 128 × 64 个显示点。

(4) 有关发声的命令

SOUND , 设定发声的音调和音长。

(5) 有关转移的命令

GOTO 行号 无条件转到指定的程序行去执行。

IF THEN ELSE 当 IF 后面的条件符合 (满足) 时, 执行 THEN 后面的内容, 然后跳过 ELSE 而按下一个程序行运行。否则, 跳过 THEN 而执行 ELSE 后的内容。

(6) 循环命令

FOR TO STEP...

⋮

NEXT

(7) 转子程序与返回主程序命令

GOSUB

⋮

RETURN

(8) 设定数组的命令

DIM

(9) 其它

REM 设定在列示程序时, 显示出编程输入时, 对程序的解释。

STOP 设定运行程序中间, 暂停运行。

END 设置程序结尾。

附录Ⅲ LASER—3000、APPLE SOFT 实数BASIC 的错误信息 (ERROR MESSAGE) 表

当操作者打入的命令或程序有错误 (不符合电脑的要求) 时, 在荧幕上会显示出一定的信息, 见附表 3。其格式如下:

? ×× ERROR IN X1

其中: ××代表错误的类型 (或名称);

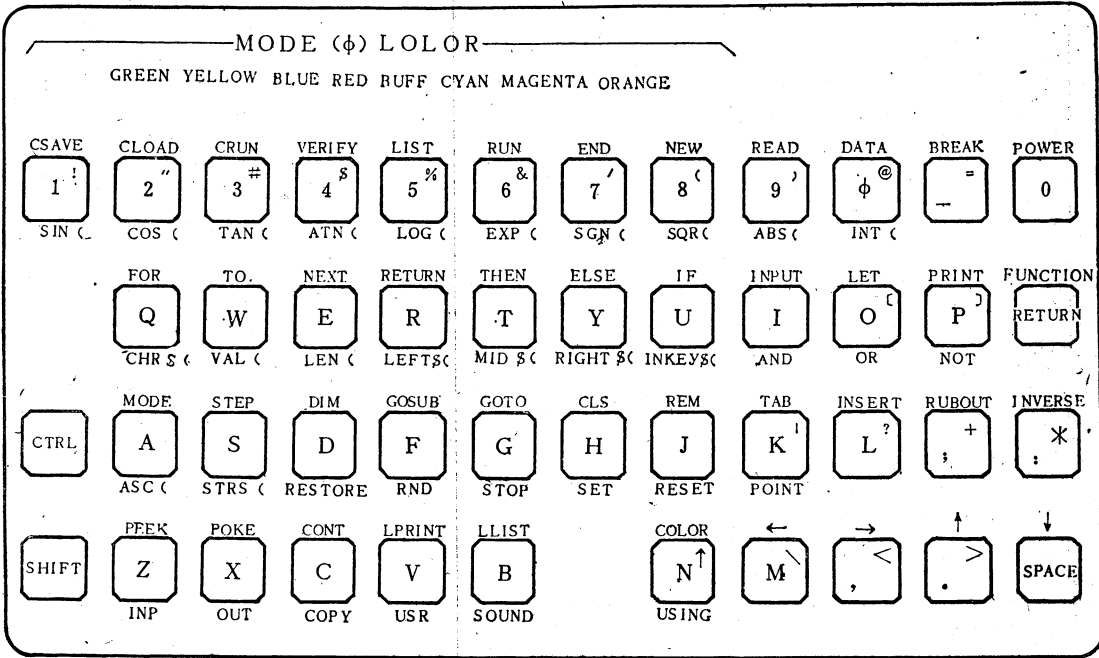
X1代表有错误的语句所在的行号。若是直接执行的命令, 则无此项。

附表 3 LASER-3000、APPLE SOFT 实数BASIC错误信息表

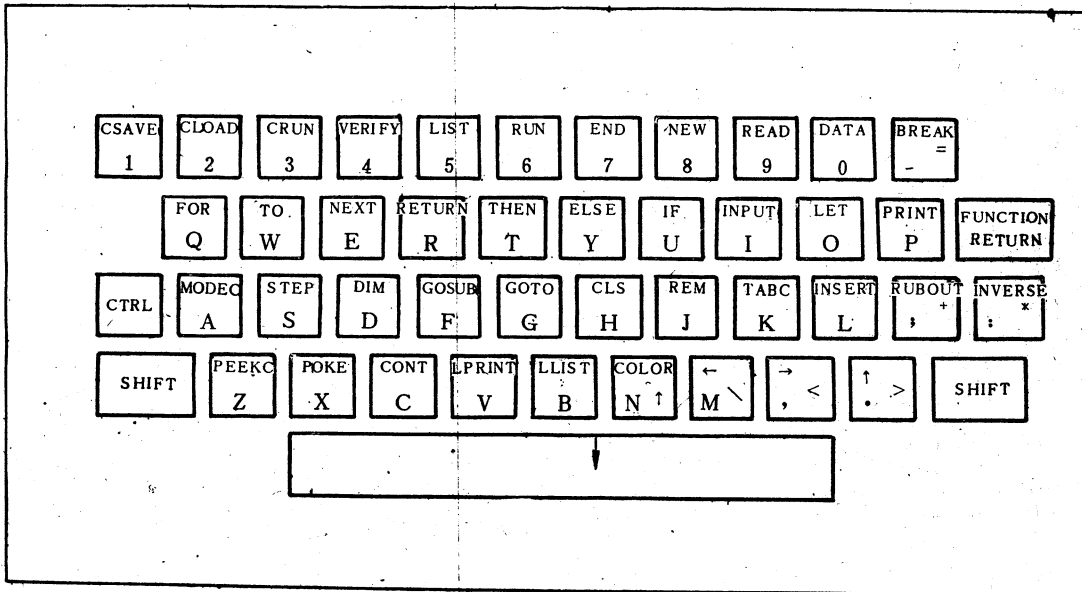
错误类型 (或名称)	错误代码	说 明
BAD SUBSCRIPT	107	注标用错。所调用的数组元素的注标不对或超过容量。
CAN T CONTINUE		不能继续。执行到END后仍用CONT或运行中出现了错误。执行中遇到中断命令, 停下后修改程序仍用CONT。
DIVISION BY ZERO	133	程序内的算式中有除数 (分母) 为零的情形。
FORMULA TOO COMPLEX	191	方案 (式子) 太复杂。遇有语句格式特别, 而超出了语言定义范围。
ILLEGAL DIRECT		非规定的直接命令, 用INPUT, DEF FN或GET, DATA等作立即执行的命令。
ILLEGAL QUANTITY	53	不合规定的数值。使用的数值超出范围。如: LET A = (- 1) = 0 以负数作数组注标。 SQR (- 2) 括号内数值不能是负数。
NEXT WITHOUT FOR	0	执行到一个没有FOR对应的NEXT语句。
OUT OF DATA	42	DATA语句中的数据用完了, 还要再输入READ。
OUT OF MEMORY	77	存贮空间不够。如程序太大; 变数太多; FOR 套迭循环超过 10 层; 子程序嵌套的层次超过24; 括号的层次超过36等等。
OVERFLOW	69	“溢出”。输入的数或程序计算出的数太大。
RETURN WITHOUT GOSUB	22	执行到一个没有GOSUB与之对应的RETURN语句。
STRING TOO LONG	176	在合并字符串后, 所得出的长度超过255。
SYNTAX ERROR	16	句法错误。命令拼错或语句格式不对。
TYPE MISMATCH	163	类型不匹配。如等号左边是数值型变量, 而右边却是字符串。
UNDEF D FUNCTION	224	使用到一个并未定义的函数。
UNDEF D STATEMENT	90	在程序中转到一个不存在的行号去执行。

附录IV 微电脑键盘图

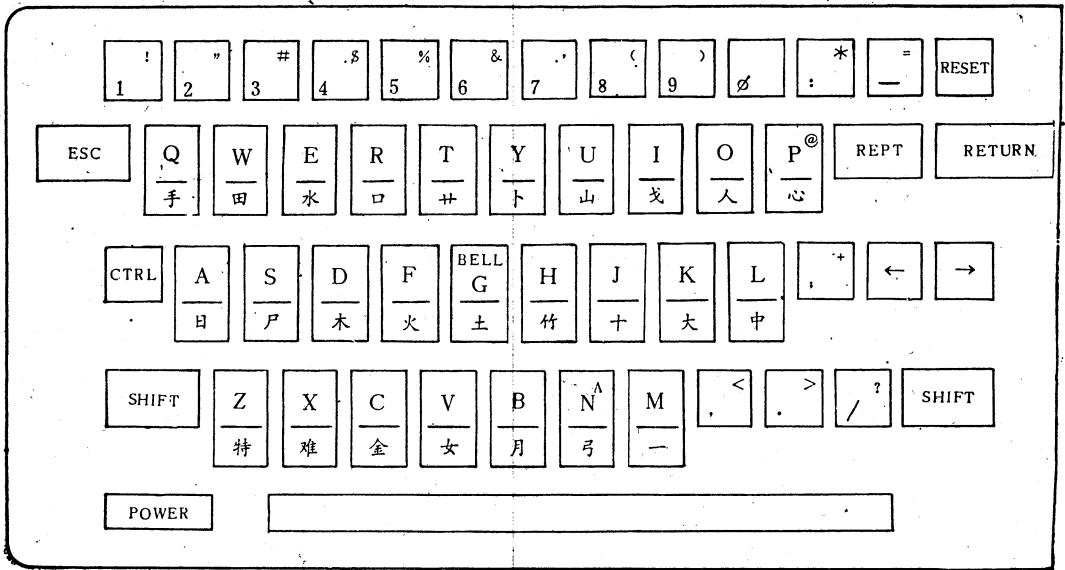
华字—200 (LASER-200) 键盘图



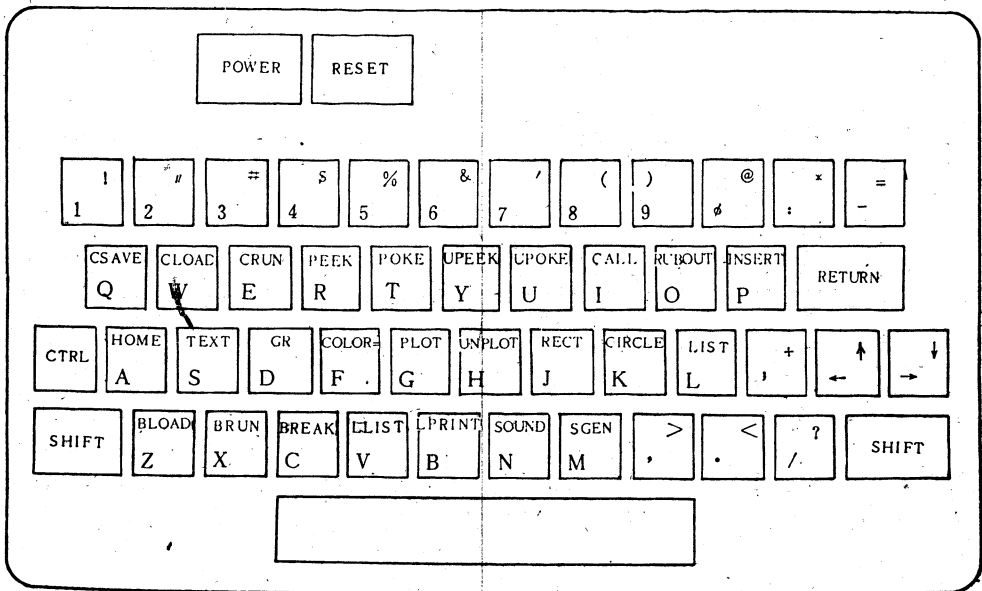
LASER-310 键盘图



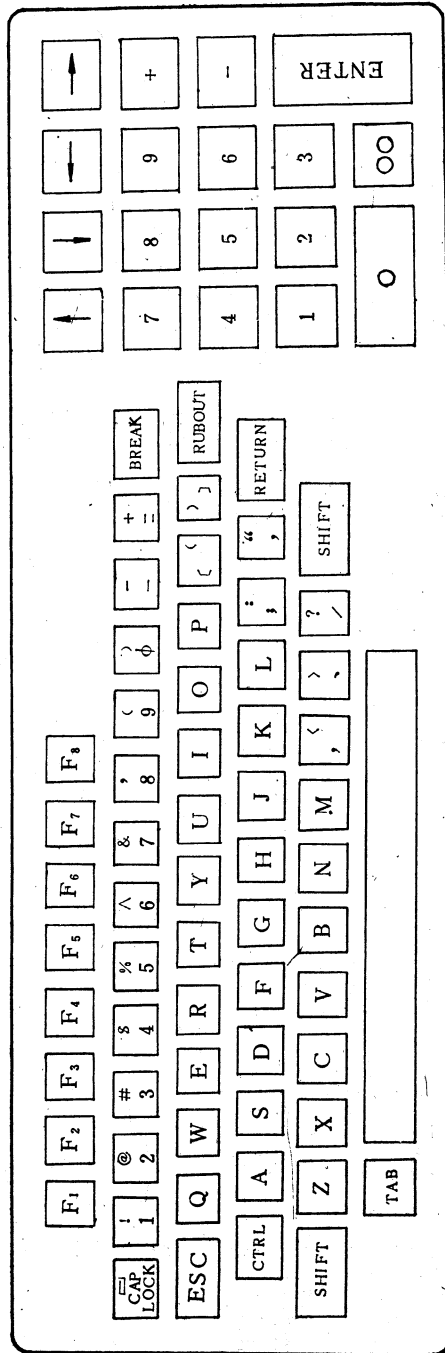
APPLE 键盘图



LASER 2001 键盘图



华字-3000 (LASER-3000) 键盘图



ZD-065 键盘图

