少年微型计算机入门

庄钢铭　黄立宏　韩莫奇　编著

福建科学技术出版社

封面设计：陈崇涛

书号：15211·62

定价： 0.96 元

少年微型计算机入门

庄钢铭
黄立宏　编著
韩莫奇

福建科学技术出版社
一九八五年·福州

责任编辑：王水俤

少年微型计算机入门

庄钢铭　黄立宏　韩莫奇　编著

*

福建科学技术出版社出版

（福州得贵巷27号）

福建省新华书店发行

福建新华印刷厂印刷

开本787×1092毫米　1/32　6.5印张　142千字

1985年10月第1版

1985年10月第1次印刷

印数：1—21,080

书号：15211·62　　定价：0.96元

前　言

《少年微型计算机入门》是一本为少年朋友编写的微型计算机普及读物，选材新颖、内容丰富。它根据孩子们初学计算机的心里状态，在内容安排上注重趣味性和实用性。本书共为六章。第一章，以生动的事例说明少年是计算机时代的主人，完全可以学懂计算机，学会使用计算机；第二章，以国内最流行的微型机为例，教会孩子们如何使用微型计算机编乐曲，进行＋、－、×、÷等的简单运算，使孩子们对微型计算机有一个感性认识，以引起他们对微型计算机的浓厚兴趣；第三章，以生动的比喻，讲解微型计算机的基本原理，为学习计算机高级语言打下基础；第四章，以浅显文字讲解BASIC语言的基本语句和基本规则；第五章，讲解如何使用BASIC语言编简单的程序，并列举了实用和趣味程序十五例，其中包括中学数理化中各种计算程序，日常实用程序以及若干自然现象的趣味程序等。附录中汇编了计算机必备的知识。

本书可作为少年微型计算机入门读物，也可以作为未学习过微型计算机的成年人自学和中、小学教师教学参考用书。

该书在编写过程中，得到了林曰钿工程师的指导和陈布峰、宋秀珍等同志的帮助。在此一并表示衷心的感谢！

由于我们水平有限，书中难免存在不妥之处，敬请广大读者和小朋友批评指正。

编著者

1984年11月

目　录

第一章　少年——计算机时代的主人

少年朋友们，微型电子计算机是当今世界上最令人神往的机器，你们一定非常向往能亲自操纵它，让微型计算机帮助自己做练习，算难题，学外语，做游戏吧！我们相信，只要你们一接触上这种令人神往的机器，你们就会被它巨大的魔力所吸引。小小的微型计算机键盘，将在你们的指尖下，使你们获得无穷的乐趣和广博的知识。

但是你们当中的一些人，可能觉得微型计算机是很神秘的，不曾想过自己能去操纵它，使用它。即使在你们的父母当中，也一定有不少人认为计算机太复杂，怀疑连大人都不曾触摸过的东西，孩子们能够学会吗？本书将列举大量生动的事例，说明少年必将成为计算机时代的主人。

那么，学习微型计算机应该从那里开始呢？本书将首先教会你使用一种目前在国内中学里，少年宫中最流行的一种微型计算机。大家只要按照书上的做法去操作，用不了几天的功夫，你们就能学会使用微型计算机的键盘，并在微型计算机上编一首你自己最喜欢的乐曲，和进行加减乘除的运算。

少年朋友都有一种强烈的好奇心和求知欲望，大家一定想知道计算机为什么会有那么多神奇的功能？书中将用一些生动的比喻，简介计算机的发展历史、计算机的基本结构、计算机中数的表示方法、计算机运算过程以及计算机程序的基本知识。并为学习计算机高级语言打下良好的基础。

微型计算机终究是一种机器，人们怎样才能让计算机听人的指挥，让它演奏乐曲就能演奏乐曲，让它算题就能算题？这就要有一种人、机通用的语言。有了这种语言，人们就可以要计算机干什么，不要干什么，怎么干，计算机在人们的指挥下就会准确地为人们服务。大家一定非常想学会这种计算机和人都能懂的语言。本书将以浅显易懂文字向大家介绍一种高级的计算机语言——BASIC语言。这种语言本身具有简单易学的特点，加上大量的实例，使大家在边学边练中学会最常用的BASIC语句和BASIC语言的各种规则，并初步掌握如何使用BASIC语言编程序。

本着实用性与趣味性相结合的原则，本书还列举了实用和趣味程序十五例。应用这些程序，少年朋友就可以帮助老师统计分数，协助父母算帐统计和再现课本中曾经学习过的图表、曲线。本书还试编了一些天文、地理和一些自然现象方面的程序实例。附录中还列举了一些主要的计算机方面的名词解释以及有关名词的中英对照。

本书虽然是为少年朋友编写的，但对未学习过计算机的成年人，也能从书中得到教益。

下面我们将给大家介绍中国和外国的少年朋友学习计算机，使用计算机的故事。

一个小时就可以学会编乐曲

小华和小明是姐弟俩，小华15岁（上初中），小明12岁（上小学），从来没有接触过计算机。一天他俩到父亲的办公室去玩，正碰见父亲在微型计算机上编程序，他们看到屏幕上一行行跳动的数字和图形都非常好奇，非要自己试一试不可。父亲只好告诉他们编乐曲的“MUSIC”语句的作用

和用计算机编曲的方法。接着姐弟俩，找来了“我的中国心”的歌片，嘀嘀咕咕地讨论开了，父亲根本不指望他们能学会什么，还是埋头干自己的事。可是没过多久，小华说，她已经编好了一首曲子，要在计算机上试试。父亲只好让她试一试。结果，小华正确地应用了“MUSIC”语句和编程序最简单的规则，一首“我的中国心”的乐曲，立即在室内回荡。姐弟俩高兴地跳了起来。

问路电脑——轰动的新闻

在上海市少年科技站电脑培训中心有一位小学生叫谭引。别看她只有12岁，一脸稚气，而她编出的电脑程序“上海北站问路”，获得了上海市青少年电脑程序竞赛小学组第一名，还成了颇为轰动的社会新闻呢！小谭家住上海闸北区，常和父母到北站去买东西，几乎每次都遇到外地来上海的旅客问路。1983年12月，谭引到区少年科技站学习计算机知识后，下决心解决“问路难”的问题，不到半年，就设计出“上海北站问路程序”，只要旅客说出到什么地方去，计算机就能准确地告诉旅客要去目的地的方向，乘什么车，并能为旅客打印出一条汉字小纸条。

孩子成了父母学习电脑的老师

小刚是某大学附中的初一学生。最近迷上了微型计算机，他决心在全国少年电子计算机编程竞赛中取得好成绩。一天，父亲正在编写一段复杂的计算程序，但两段程序总是联不起来。小刚看见父亲着急的样子，就说：“爸爸，让我来给您看看吧！”小刚来到计算机旁，认真地检查起来，不一会就高兴地说：“爸爸，您取的循环变量前后矛盾，所以

两段程序不能联接。”父亲恍然大悟，没有料到，没有高深数学知识的儿子，却如此迅速地查出了计算机程序中的问题。

农民孩子成了电脑迷

不少人认为那些少年电子计算机迷都是大城市的孩子，而他们的父母肯定都是知识分子。然而在上海市川沙县高桥中学的电子计算机房里，一批家在农村的学生，端坐在电子计算机前，各自按动电子计算机的按键，荧光屏上迅速跳出各种数字和中文、英文字样，并不时闪出美丽而变化万千的图案，传出悦耳的音乐。

下面再讲讲外国小朋友学习计算机的故事：

模拟心脏跳动，模拟核反应堆工作情况

在美国新泽西州的一所初级中学里，一天的课程已经全部结束了。可是在教室里还有十几个学生在全神贯注地操纵着微型计算机。计算机键盘在嘀嘀哒哒地发出悦耳的声音，荧光屏上闪烁着各种数字和图象、图表，打印机上传出表示结果的纸带。14岁的马米尼斯刚刚打完模拟心脏工作的图象程序，程序运行时，屏幕上清楚地显示出心脏的房室、活动瓣膜和心脏周围的主动脉与静脉。另一位14岁的小朋友米勒，正在编制一个模拟核反应堆工作状况的程序。一个叫洪美龄的15岁的女孩子，正在精心检查她的商业数据管理程序中所出现的故障。她仅仅是在上一学期才开始学习排除程序故障的技术，如今她已经是一位排除故障的老手了。

七、八岁的娃娃玩电脑

一些先进的国家里，孩子们进入幼儿园时就开始接触计

算机，这是毫不奇怪的。在美国有一个叫肖勃尔的计算机专家，曾目睹一个8岁的小孩在创造性地玩一种简单的计算机游戏。在这个游戏里，只要按照规定的程序按下一系列的按钮，在屏幕上就会出现一个青蛙突然跃起并吞吃一只飞翔的蝴蝶。过了一会，计算机专家发现，青蛙的跳跃动作明显地放慢了，于是专家问这个孩子是怎么回事？孩子回答说："我想让青蛙抓到更多的蝴蝶，我改变了按钮的次序，终于使青蛙慢慢地跳起来"。一个8岁的孩子打破了原来的游戏程序，而创造了一套新的程序。

编制实用程序，帮助老师统计全班成绩

初中二年级学生小高是班里的学习班长，以前每次考试完，都替老师统计全班成绩，要算出每人单科成绩，各科总成绩，全班的总成绩，平均分数，最后还要排列名次，别看全班才三十多人，但统计一次也得一、二天功夫呢！小高把刚刚学来一个实用的统计程序，使用最流行TP—81微型计算机，只用了一个小时，就把期中考试的全班总分，平均分，每个同学的总分，每个分数段的人数等各项统计数字一一打印出来了。小高从中尝到了学习计算机的无穷乐趣，又得到了老师的赞赏和表扬。

成立软件公司，出售各种游戏程序

在美国芝加哥，14岁的杜伯曼和13岁的波兹索尼成立了一个"阿里斯多软件公司"，专门出售他们编制的计算机游戏和计算机绘图的程序。在洛杉矶，14岁的伏肯罗，每星期为一个电影公司工作24小时，编制各种有关的计算机程序，为此，他每星期可得到480美元的报酬。加里福尼亚18岁的克利

斯坦森，由于他发明了一种游戏用的计算机程序，一年就获得10万美元的专利税。

少年朋友们，祖国的灿烂前程在向我们招手，世界的新技术革命在激励我们向前，投身于祖国的计算机事业吧！你们是祖国的花朵，未来计算机时代的主人，多么幸运啊！

第二章　学会操作微型计算机

少年朋友们，你们知道吗？当今世界上各种各样的电子计算机中，有价钱昂贵，体积庞大，运算速度每秒钟达一亿次的巨型计算机，象我国研制成功的“银河”计算机就是这种类型的。另外，还有大、中、小型计算机。随着科学技术的迅速发展，最近十多年来，又出现了一种微型计算机，也有人把它叫做微电脑。这种微型计算机只有一台电视机那么大，便宜的几百元一台，好一点的也只有几千元一台。现在，我们向大家介绍一种国内比较流行的普及型微型计算机。它的型号国外叫R1型，北京产的叫PC—81，另外，上海、广州等地都有生产。

第一节　如何联结你的微型计算机

微型计算机是由四部分组成的，如图2—1—1。

1.一台普通的黑白或彩色电视机。

2.一台微型计算机的主机。它包含着微机各种最基本的功能。

3.电源。电源小方盒上，联了两根导线。一根导线上，带普通的插头，这是接220伏交流电的；另一根带一个圆插头，这是插在主机上的电源插座上的。

4.联接导线。

除上面介绍的电源联接线之外，还有一根联接主机和电

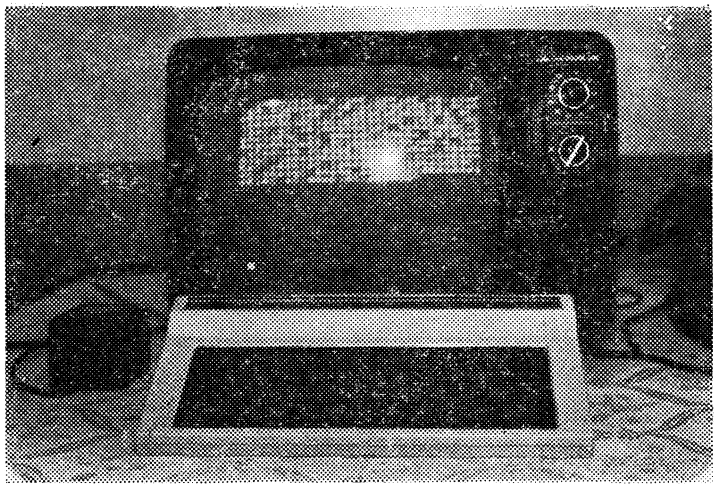


图2—1—1　PC—81（R_1）微型计算机

视的导线，二根联接主机和录音机的导线。

一般说，一台微型计算机有如下三个部分就可使用了：

* 主机（带键盘）。
* 显示器（就是用普通的电视机）。
* 稳压电源（输入：220伏交流；输出：12伏直流）。

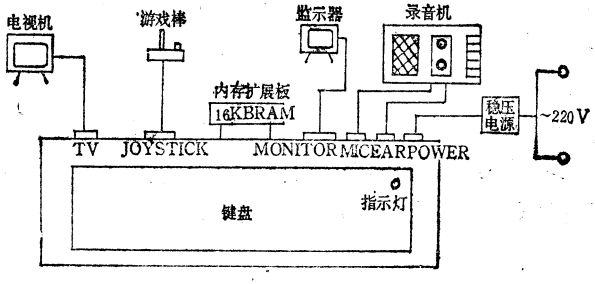


图2—1—2　PC—81（R_1）安装示意图

在主机侧面共有六个不同的插孔和一个长方形插槽，每个插孔都与那个部件联接，如图2—1—2所示。其中游戏棒，内存扩展板、监示器。录音机如果暂时还没有，可以先不用。只要把电视机、稳压电源和主机联结起来就可以使用了。

使用时要把电视机旋置在二频道上，并由75Ω外接天线输入端与微型计算机主机相联。当部件联好以后，就可以接通220伏交流电源了。在主机键盘上的红色指示灯亮了，即表示电源已经接通。当打开电视机时，电视机屏幕上显示出：如图2—1—3的字样和一闪一闪的光标。

READY

图2—1—3　（英文的意思是已经准备好）

■闪动的光标，指示你可以输入数据或信息了。

第二节　计算机键盘

几乎所有微型计算机都有一个象打字机一样的键盘（图2—2—1为普及微型机PC—81的键盘，图2—2—2为高级微型机IBM—PC/XT的键盘）。

各种微型计算机都有自己的键盘，但都是大同小异的。只要学会了一种键盘的使用方法，再去学习其他计算机的键盘就容易得多了。

下面，我们就PC—81普及型微型机键盘的用法及其英文含义作一介绍：

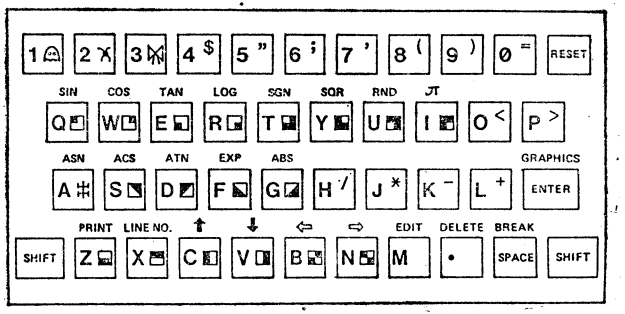


图2—2—1　普及微型机PC—81的键盘

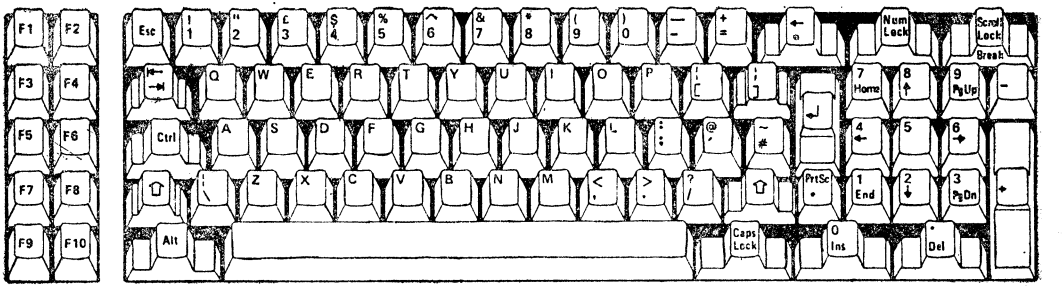


图2—2—2　高级微型机IBM—PC/XT的键盘

由于微型计算机大都是从国外引进的，所以大多数键盘上只有英文。现将键盘中的英文翻译成中文，并作简单说明。

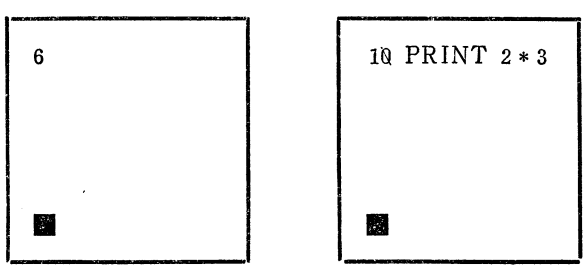
1.RESET

中文：复位。

作用：清除所有内存变量和字符串。

说明：一个程序运行结束后，在屏幕上显示结果。当按下“RESET”后，可把运算结果从屏幕上清除，并恢复原来程序清单。

举例：10 PRINT 2*3 ENTER RUN 当按键入“RUN”以后，屏幕上显示6（见图2—2—3—a）。当再按下“RESET”后，屏幕上恢复了原来的语句。（见图2—2—3—b）。



(a)　　(b)

图2—2—3，　2×3举例图示

2. SIN、COS、ASN、ACS、TAN、ATN

中文：三角函数。

作用：计算三角函数值。

说明：计算机要求用弧度表示角的大小，角和弧度的换算：2π弧度$=360°$。即：1弧度$=\frac{180°}{\pi}=57°17'44.8''$

$1° = \frac{\pi}{180}$弧度 = 0.01745弧度

3. LOG

中文：对数。

作用：$Log_e \times$（求以e为底数的对数值，即求自然对数Lnx）。

4. SQR

中文：平方根。

作用：用这键可求任何数的平方根（输入数≥0）。

5. RND

中文：随机数。

作用：用这键可得到一个个没有规律的数字。

6. π

中文：圆周率≒3.14

作用：使用时，可用π这个按键，此时屏幕上出现“PI”；也可以按键入3.14159……，其作用完全一样。

7. EXP

中文：指数函数

作用：e^x（e = 2.71828）

8. ABS

中文：绝对值。

作用：求x的绝对值为|x|。

9. GRAPHICS

中文：图形。

作用：用这个键可显示键盘上各种图形。

10. ENTER

中文：送入

作用：这是最常用的键。在计算机中，“ENTER”一般又称“回车”键。其作用是把键盘输入的各种数据送入内存，以便进行运算处理。所谓“回车”是针对打印机来说的，当用键盘输入数据并打印时，打印机头由左⟶右移动。按下“ENTER”后，机头回到起始位置，等待下一行打印。即“ENTER”能使机头回来，所以叫回车。

11. SHIFT

中文：分离、选择

说明：SHIFT是一个很重要的键。在键盘上所有按钮上都有两种以上的符号，如：

①可向计算机输入数字“8”

②可向计算机输入一个左括号“（”

①可用此键求任何数的平方根

②可向计算机输入英文字母“Y”

③可向计算机输入一个“■”图形

可见，键盘上的一个按钮可有两种或三种功能。要选择某些功能，要SHIFT配合使用才能做到。

SHIFT 使用方法：

①在键盘按钮上字型大，白色字体的字母和数字可用不用SHIFT。

② ENTER（回车），

. （小数点）

SPACE（空格）

RESET（复位）

} 可直接使用。

③除上述10个数字，26个大写英文字母和图中四种功能以外的各种符号和功能都应先按下 SHIFT 键。

12. SHIFT + Z■ ⟶ PRINT

说明：PRINT 是最常用的功能键。使用 PRINT 时，可一个个字母输入。为了提高输入速度，按 SHIFT + PRINT Z 即可显示PRINT。

13. SHIFT + X■ ⟶ LINENO 自动行号

说明：用BASIC 语言编程序都要用行号，行号由小到大依次排列。按 SHIFT + X 能自动显示由小到大的行号，而不需要单独输入一个行号。

14. SHIFT + ↑ C ⟶ 编辑符号上移一行。

15. SHIFT + ↓ V ⟶ 使编辑符号下移一行。

16. SHIFT + ← B ⟶ 使光标左移一行。

17. SHIFT + → N ⟶ 使光标右移一行。

18. SHIFT + EDIT M ⟶ 可对已编好的程序进行修改。

说明：如右图。

10 A = 2，

现要改成A = 3，

方法如下：

①SHIFT + ↑

把编辑符号■移到第10句。

②按 SHIFT + EDIT M

第10句即出现在屏幕下部。

```
10■A = 2
20  B = 3
30  P = 2 * 3
40  PRINTP

10  A = 3
```

③ SHIFT + → N 把光标移2后 如A = 2■

DELETE

④ SHIFT + . 即可把2删除。

⑤按 3 即可得A = 3

⑥按 ENTER 原来程序中第10行：

10 A = 2，即可改成：

10 A = 3。

DELETE

19. SHIFT + . ——→可删除光标前的一个字符。

BREAK

20. SHIFT + SPACE ——→中断正在运行的程序。如要程序继续运行可打入CONT。

说明：BREAK 是“中断”“屏蔽”。

CONT。是“继续”由键盘打入

SHIFT + SPACE 'BREAK就是命令计算机停止工作。

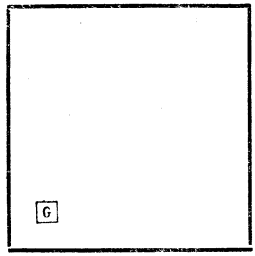
打入CONT。即命令计算机继续工作。

21. SHIFT + GRAPIHICS ENTER ——→使计算机处于显示图形的准备状态。

说明：在计算机键盘上有19个方格图形，如▭……■等。为了使这些图形显示出来。首先通过按 SHIFT + GRAPHICS，使计算机处于显示图形准备状态。GRAP- ENTER

HICS 是“图形”的意思。

显示图形步骤：



①当按下 SHIFT + GRAPHICS后，屏幕显示提符 ENTER 号[G]

②当屏幕上显示提示符 G 时，按 SHIFT + 所需图形按钮，即可显示各种图形。

22. SPACE　空格　按下这个按键，可使光标移动一格。

在一些计算机中，程序中的两个字符之间一定要留空格。如：

10␣FOR␣A = 1O␣STEP␣1␣表示空格。

但在我们的R1（PC—81）微型计算机中，可以不留空格。所以使用起来是非常方便的。

23. +，−，×，÷　在键盘上+，−，和平常数学书上的符号是一样的。但×和÷在键盘上表示方法如下：

×……用 * 表示

÷……用/表示

第三节　计算机编乐曲

让计算机编乐曲是很有趣的事。你可以编你平时最喜欢

的乐曲，也可以让计算机教你唱一首新歌。

少年朋友，下面就请你试一试。

```
10  MUSIC "C5D5E5F5G5A5B5C>5"
20  RUN
RUN
```

计算机立即一遍又一遍地奏出1 2 3 4 5 6 7 i 八个音符。若要让演奏停止下来，只有按住 SHIFT 和 BREAK 两个按键就行了。

下面是一首大家非常熟悉的乐曲，到底是什么歌？就让计算机演奏给你们听听吧！

请你用键盘打入（见图2—3—1）

```
10  MUSIC "E<16E<16A<4E<4G<4C4F<4A<4C4G4A8A<4B<4C
       8D8E16E16.8C4D4E8A8B8.16B>16.4E2E2E2E2
       E2E2.8C>16E8G8D32G12D4E8G8A32D>16E>8D>
       8C>24D>4C>4B16A8G8A32C>12A4G8E8C>12A4G
       8E8C>4C>8A4G8C>8A4A4.8E12C4D8E8G8A4G4E
       8.8D12G4E8C8A<4A<4.32A<4C4G<4A<4D8G8E3
       2A4G4E4G4C>8B4A4A32C>12A4G8E8D8G8E8C8A
       <4C4D4E4G4E4C>4B4A32C>12A4G8E8C>12A4G8
       E8C>4C>8A4G8C>8A4A4.24E12C4D8E8G8A4G4E
       16C>4C>8A4D>8C>4B4A32"
RUN
```

图2—3—1　乐曲打入键盘

这是一首电视连续剧“霍元甲”的主题歌。如果你还不会唱，就请计算机教你唱吧！但怎样才能让计算机反复演奏，你还得动动脑筋。

下面简单介绍计算机编乐曲的基本知识（以PC—81微型机为例）。

1. MUSIC 英语是“音乐”的意思。在计算机里就是

命令计算机演奏乐曲的语句。所以，MUSIC 是必不可少的。

2．请大家记住：

C相当于1（音符）　　G相当于5（音符）

D相当于2（音符）　　A相当于6（音符）

E相当于3（音符）　　B相当于7（音符）

F相当于4（音符）　　＞是高八度，＜是低八度

·代表停顿

3．CDEFGAB 后面的数字，代表乐曲中的节拍的长短。数字越大，发音越长。

如：6　35　|2——|

可以编成：A6　E3G3　D12

大家会发现：6+3+3=12

6　35　|2——|

这说明，在编程序时应该注意，每一小节的数字应该相等，每一拍的数字也应该相等。至于每节拍数字的大小，可以根据每个音符节拍的长短和整个曲子快慢来适当考虑。

根据上面的简单介绍，大家不妨自己亲自动手编几首自己喜爱的乐曲。有关编乐曲更详细的介绍，请看本书的第四章。

第四节　计算机算题

让计算机做数学题是少年朋友所关心的。本节将采用“跟我学”的办法，让大家有一个直观的印象。

请大家用键盘打入：

(1) 10 PRINT　2+3　ENTER屏幕上立即显示出5

（见图2—7）。

原来，计算机做了一道2＋3＝5的加法。

（2）10 PRINT　1234567＋7654321

ENTER

屏幕上立即显示出8888888（见图2—8）。

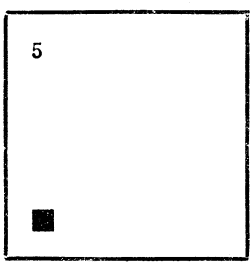


图2—7

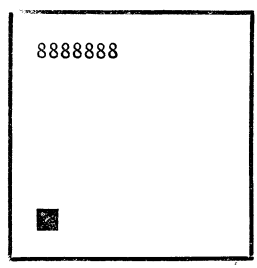


图2—8

原来，计算机又做了一道多位加法。

大家如果有兴趣的话，可让计算机做一道多位数的加法、减法、乘法或除法。即键盘打入的格式完全一样，只是在“＋”号的地方换成－、×、÷就可以了。如

n　PRINT　×××××[＋，－，×，÷]×××××

现在，请大家用计算机计算下列各题：

a、PRINT　25＋25　ENTER（得数：　　）

b、PRINT　225／25　ENTER（得数：　　）

c、PRINT　1984—1921　ENTER（得数：　　）

少年朋友们，你们可能已经发现了，计算机算题的方法和用一般计算器算题的方法是一样的。若微型计算机当作计算器使用时，打入 [PRINT]（PRINT在英语中是“打印”的意思，在我们的计算机中，PRINT 是一条命令），即是命令计算机进行表达式运算后，把结果显示在屏幕上。

或许少年朋友会问，微型计算机和计算器有什么不同？计算器能代替计算机吗？

我们说，微型计算机比普通的计算器的功能强多了。计算器能干的事，微型计算机都能干，并且微型计算机还具有能按预先编好的程序进行运算，一般普通的计算器是没有的。

现在，请少年朋友们，用键盘打入下面的一段程序：

```
10  FOR  A=1  TO  9
20  FOR  B=1  TO  9
30  P=A*B
35  SCROLL
40  PRINT  A;  "*";  B;  "=";  P
50  NEXT  B
60  NEXT  A
RUN
```

该程序运行的结果，是一个大家熟悉的“九九表”（见图2—9）。而普通的计算器只能作一道道的乘法。例如，只能做9×8=72，但不能从1×1=1直至做9×9=81连续的计算结果打印出来。所以说，能不能编程序是计算器与计算机最主要的区别之一。

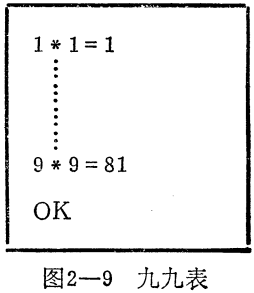


图2—9　九九表

由此可见，要学会使用微型计算机，首先就得学会计算机的语言，但仅仅学会计算机语言还不够，还得学会编制解决实际问题的程序。如上面我们练习过编一段“九九表”的程序，计算机按照这段程序运行，一份完整的“九九表”就会打印出来了。这说明学会计算机语言和学会编程序都是非常重要的。

第三章　微型计算机基本知识

第一节　从电子管计算机到微型计算机

微型计算机所以有如此神奇的功能，完全归功于一块只有指甲大小的硅片。这块硅片叫做大规模集成电路。

别看它只有指甲那么大，然而在这一块小小的硅片上却容纳了几千个甚至几万个、几十万个电子元件呢！这些电子元件大部分是晶体管，其次还有电阻、电容、二极管和成千上万条连线。如果这块硅片放大几百倍，它真象一座星罗密布四通八达的大城市，有密如蛛网的街道、一座挨一座的楼房店宇。那些象楼房店宇就是各种各样的晶体管、二极管、电阻电容等元件，而那些象马路一样交通线就是这些元件之间的联线。这种神奇的硅片是科学家和工人们用非常精密的仪器设备，在非常干净的超净车间里制造出来的。

我们可以用这些神奇的硅片，来组装成收音机，组装成电子手表，组装成计算器，组装成电视机，而用这些硅片还能组装成微型计算机。你们所看到PC—81型微型计算机主机差不多只有一本厚书那样。

然而在1946年发明的第一台计算机，却是一个庞然大物。第一台计算机名叫“ENIAC”，因为那时，晶体管还没有问世，这台计算机是用当时认为很先进的真空管制造的。这一台计算机一共用了18000个真空管（现在，许多少年朋友可能还没有见过真空管了）。原来老式收音机和电视机

都是用真空管装配成的，一个真空管和一个5w的电灯泡差不多。另外第一台计算机还用了七千个电阻，一万个电容和六千个继电器，再加上五万条以上导线。大家可以想象，这么多的东西组装在一起，肯定是一个庞然大物。据说，这台计算机共占地170多平方米，有6个普通房间那么大，总耗电量达150KW。但这计算机的运算速度只有5000次/秒。而如今我们看到微型计算机只有书本那么大，稍微高级的微型计算机也只不过和普通电视机差不多。从发明第一台计算机算起，至今还不到40年，但这40年中计算机的发展却是令人惊叹不已。

有人作了分析比较，现在的微型计算机和第一台计算机相比：

体积　　缩小了四万倍
速度　　提高了几十倍
用电量　节约了几千倍
可靠性　提高了几千倍
价格　　降低了一万倍

在若干年前，个人或家庭想买计算机简直是不可想象的，因为一台计算机少则几万元，多则几十万元，几百万元。现在一台PC—81型计算机主机只有几百元，如果需要的话，一个中等收入的家庭就能买得起。

随着集成电路技术的迅速发展，在一块小小的硅片上将能制造出几十万个甚至几百万个元件。到时候，一块硅片，就是一台完整的计算机，这种计算机或许和手表一样带在手上呢。电子手表，手表式计算器，手表式彩色电视机，手表式电子计算机……这些现代科学技术的结晶，必将使你们大开眼界，并极大地促使你们更加勤奋上进，做一个名符其实

的计算机时代的主人。

第二节　计算机的工作过程及计算机的基本结构

计算机不仅能算题，编乐曲，而且还能指挥火箭的发射，控制人造卫星的飞行。现在还有一种电脑医生，能为病人看病开方……所有这一切，都给电子计算机蒙上了神秘的色彩。

然而，当我们掌握了计算机工作的基本原理时，我们就不会觉得计算机是神秘的东西了。

计算机和算盘

算盘是我们祖先给我们创造的一种简单有效的计算工具，至今还在我国，甚至在世界上许多国家被广泛地使用。

现在让我们回想一下用算盘来演算一道最简单的计算题：

$5\times6+8=?$

我们来分析一下这道题，如用算盘运算的过程：

第一步：把题写在纸上。$5\times6+8=?$

第二步：在算盘上拨上5和6。

第三步：用乘法口诀，5和6相乘得30。

第四步：在算盘上把（$5\times6=$）30再加上8得38。

第五步：把得数38写在纸上。

这样，$5\times6+8=?$ 这道计算题的演算才算结束了。

由上面我们可以看出，要用算盘计算一道题，必需具备下面几个条件（见图3—2—1）；

1.要有进行计算的装置——算盘；

2.要有能存放数据和计算步骤，计算方法的装置——即用笔在纸上做记录；

3.要有进行控制的装置——即要知道取那个数，进行什么计算，那几项先计算，那几项后计算——这种控制装置就是人的脑子。一切计算都是在人的控制下进行的。

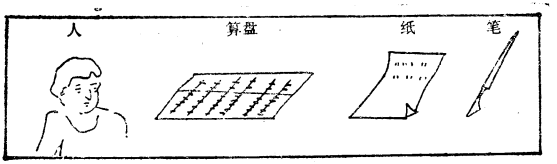


图3—2—1　用算盘计算应具备的条件

在使用计算机算题时，也必需具备几种设备：

1.运算器——相当于算盘；

2.存贮器——相当于纸和笔。在计算机中存贮器用于保存和记录各种原始数据，运算步骤，以及运算中间结果；

3.控制器——相当于人脑和手的作用。控制器是计算机的指挥中心，它指挥计算机先算什么，后算什么，算好的结果往那里送等。

用计算机算题时，是通过控制器从存贮器中取一个数并送到运算器中去。

4.输入、输出设备。

人们要把自己的意图告诉计算机，就得用各种手段和设备，如用键盘把各种命令和数据送入计算机，键盘就是一种输入设备。

计算机的计算结果，只有显示在电视屏幕上或用打

印机打印在纸上，才能使人们看得见。显示器和打印机就是输出设备。

所以一台计算机，有四个必不可少的部分，那就是：运算器，控制器，存贮器和输入输出设备（见图3—2—2）。

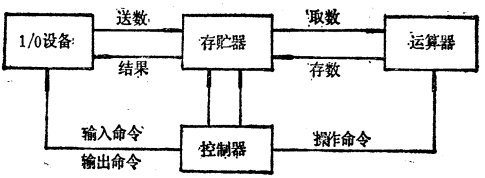


图3—2—2　计算机工作原理图

根据计算机原理图，我们再把计算机工作过程再总结一下：

由控制器发出接收数据命令，输入设备把数据送入存贮器，在控制器控制下，运算器从存贮器中取数并进行运算，运算结果再送回存贮器。如果是最终结果，控制器就把结果送至输出设备，输出设备把结果显示出来或打印出来。至此，计算过程结束。

前面已经提到，由于科学技术的进步，现在计算机的各个部件都可高度集成到一小块硅片上，所以微型计算机的结构与前面介绍的传统的计算机结构图，已经有了很大的变化。当今微型计算机的结构图如下（见图3—2—3）。

从微型计算机结构图中我们看到微型机的主机由三部分组成：

第一部分叫微处理器。实际的微处理器只是一块指甲大的小硅片。但别看它小，它却包含了控制器、运算器和存放中间结果的寄存器等传统计算机的主要部件。所以又把微处

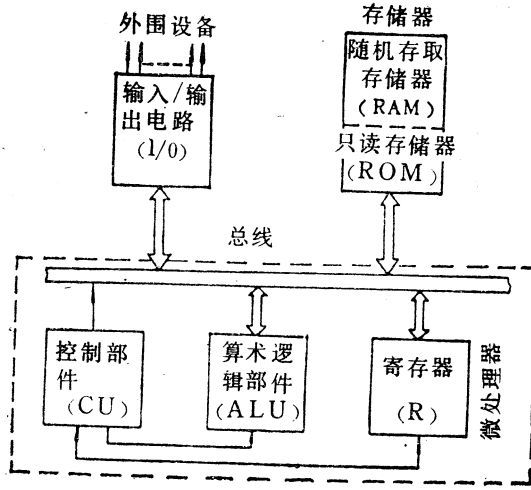


图3-2-3微型计算机基本结构框图

理器称为微型计算机的中央处理单元，英文的名称叫CPU；

第二部分叫存贮器。这一部分与传统的计算机差不多。但做存贮器的材料不同了。传统的计算机的存贮器一般都是用很多个小型的圆环状的磁芯来制造，而微处理机的存贮器却用半导体材料来制造的。半导体存贮器比磁芯存贮器体积小得多，但它的速度却比磁芯存贮器快得多。

存贮器有两种类型：一种叫随机存贮器（英文缩写为RAM），一种叫只读存贮器（英文缩写叫ROM）。随机存贮器——就是随时可存随时可取的存贮器。就象我们在银行用活期存款，什么时候存钱，什么时候取钱都可以。所谓只读存贮器，顾名思义，只读存贮器中存放的内容，只能取出来，而不能把新的内容存进去。可能少年朋友都知道，当你要长久保存一盘已录好音的磁带而不至于被抹掉时，可以把

磁带盒上二个小凸起部分去掉，这样，这盘磁带就不会被抹掉，而只能放音了。只读存贮器和这种能永远保存的录音带很相似。

第三部分是输入/输出电路，英文的缩写能I/O。

前面已经说过，为了把人们的意图告诉计算机；就得有输入设备，如键盘。为了把计算机的计算结果让人们看得见，就得有输出设备，如显示器或打印机。

从下一节我们就会学到，计算机只认得“0”和“1”两个数，也就是说，在计算机内部不管进行什么运算都用“0”和“1”两个数来表示。但人们都习惯于十进制和汉语或英语这些日常习惯使用的文字。为此，就需要一种电路，把人们用十进制或字母，文字等信息变成计算机能懂得的0和1组成的数码；而计算机用0和1所得出的运算结果，也要换成人们懂得的十进制数，字母和文字等。能起这种变换作用的电路我们叫做输入/输出电路，也叫接口电路。

第三节　计算机中的数

电子计算机是一种具有复杂功能的机器。就是这种神奇的机器，就其智能水平来说还不如一个学龄前的小孩呢！因为不论多复杂的计算机也只认得0和1。

日常生活中的数制

每个人都有两只手，每只手有五个手指，两只手一共有十个手指，两、三岁的小朋友刚学数数时，就会按着手指数数，从1一直数到10。所以人们最习惯于逢十进一的十进制。

但在日常生活中不仅有逢十进一的十进制，而且还有各种各样的进位制。例如，六十秒是一分钟，六十分钟是一小时，这就是六十进制。24小时是一天，就是24进制。七天是一星期，就是七进制。12个月是一年，12件是一打，就是十二进制等。

在生活中也常用二进制。如两只鞋是一双，两只手套算一双，两支筷子算一对。又如人的两只脚算一双，两只手也算一双等都是逢二进一的二进制。

二进制是所有数制中最简单的，因为在二进制中只有两个数“0”和“1”。

计算机中的数制

我们知道电子计算机是用电子元件制造的，它在电的作用下才能工作。当初，科学家们曾设想制造一种使用十进制的计算机，但十进制计算机中，用电气来实现0，1，2，3，4，5，6，7，8，9十个稳定状态是很困难的。

人们发现在电气设备中，两个状态的现象是很多的。如电灯开关，开时灯亮，关时灯灭，这就是两种状态。电压高和低，晶体管的导通和截止，电容的充电和放电，都是两种状态。所以，在电气上实现二进制是很容易做到的，科学家们决定在电子计算机中采用二进制。

在计算机中，我们设定高电压为1，低电压为0，在计算机中的术语是，高电平为1，低电平为0。我们说，计算机认得的0和1，其实不是我们写在纸上的0和1，而是电压的高或低。只不过我们假定高电平就是1，低电平就是0罢了。

要让计算机工作。例如：做复杂的计算，还是做有趣的

游戏，都得把方法、步骤编成由0和1组成的各种编码，计算机才能认识，才能处理。难怪人们说计算机其实只认得0和1呢！

在通常的使用中，象0 1 2 3 4 5 6 7 8 9和A B C D E……X Y Z 等一些数字、字母和符号是用得最多的，但在计算机中都把这些常用数字、字母和符号进行二进制编码。现世界各国一般都采用美国信息交换标准代码，即所谓 ASC Ⅱ（阿西码）作为编码标准。

大家一定会问，这些编码太复杂，很难记住。怎么办？在计算机发明初期，由于还没有创造出编程序的高级语言，计算机专家叫计算机算题时，就得用成千上万条用0和1编成的指令。现在大家可不用着急了，我们甚至可以不懂得二进制，也能很好地使用计算机，因为计算能能自动地把人们习惯使用的十进制数字、字母、符号，甚至汉字翻译成计算机能识别的二进制码。

二进制基本运算规则

首先让我们一起做几道二进制的加法练习：

```
  0
 +1
---
  1   ……（相当于十进制的1）
+、1
---
 10   ……（相当于十进制的2）
 +1
---
 11   ……（相当于十进制的3）
+、1
---
100   ……（相当于十进制的4）
+ 1
---
101   ……（相当于十进制的5）
```

大家如果有兴趣，还可以一直往下加去。下面的一个表是十

ASCⅡ编码表（美国信息交换代码）表1

输出、入文字	对应的编码		输出、入文字	对应的编码	
	2 进 制	16进制		2 进 制	16进制
LF	00001010	OA	>	00111110	3E
CR	00001101	OD	?	00111111	3F
SP	00100000	20	@	01000000	40
!	00100001	21	A	01000001	41
▼▼	00100010	22	B	01000010	42
#	00100011	23	C	01000011	43
$	00100100	24	D	01000100	44
%	00100101	25	E	01000101	45
&	00100110	26	F	01000110	46
▼	00100111	27	G	01000111	47
(	00101000	28	H	01001000	48
)	00101001	29	I	01001001	49
*	00101010	2A	J	01001010	4A
+	00101011	2B	K	01001011	4B
,	00101100	2C	L	01001100	4C
−	00101101	2D	M	01001101	4D
•	00101110	2E	N	01001110	4E
/	00101111	2F	O	01001111	4F
0	00110000	30	P	01010000	50
1	00110001	31	Q	01010001	51
2	00110010	32	R	01010010	52
3	00110011	33	S	01010011	53
4	00110100	34	T	01010100	54
5	00110101	35	U	01010101	55
6	00110110	36	V	01010110	56
7	00110111	37	W	01010111	57
8	00111000	38	X	01011000	58
9	00111001	39	Y	01011001	59
:	00111010	3A	Z	01011010	5A
;	00111011	3B	[	01011011	5B
<	00111100	3C	]	01011101	5D
=	00111101	3D	←	01011111	5F

注：LF（Line Feed）：改行；CR（Carriage Return）：复归；SP（Space）：空白。

进制从1到15和相应二进制数的对照表。

十 进 制	二 进 制	十 进 制	二 进 制
0	0000	8	1000
1	0001	9	1001
2	0010	10	1010
3	0011	11	1011
4	0100	12	1100
5	0101	13	1101
6	0110	14	1110
7	0111	15	1111

通过上面的练习，我们已可以看到，二进制的运算规则比十进制的运算规则简单多了。下面是二进制的加、减、乘、除的运算规则：

1．二进制加法规则：

$$\begin{array}{cccc} 0 & 0 & 1 & 1 \\ +0 & +1 & +0 & +1 \\ \hline 0 & 1 & 1 & 10 \end{array}$$

即 $0+0=0$

$0+1=1$

$1+0=1$

$1+1=10$

例如：$4+5=?$

二进制	十进制
100	4
+101	+5
1001	9

2．二进制减法规则：

$$\begin{array}{cccc} 1 & 0 & 0 & 1 \\ -0 & -0 & -1 & -1 \\ \hline 1 & 0 & 1 & 0 \end{array}$$

↑——（向高位借1）

即：1－0＝1

0－0＝0

0－1＝1（向高位借1）

1－1＝0

例如：13－11＝？

二进制	十进制
1101	13
－1011	－11
0010	2

3．二进制乘法规则：

$$\begin{array}{cccc} 1 & 0 & 0 & 1 \\ \times 0 & \times 0 & \times 1 & \times 1 \\ \hline 0 & 0 & 0 & 1 \end{array}$$

即 1×0＝0

0×0＝0

0×1＝0

1×1＝1

例如：7×8＝？

二进制	十进制
0111	7
×1000	×8
0000	56
0000	
0000	
111	
111000	

二进制的除法运算规则：

除法和乘法是逆运算。

例如：56÷8=？

二进制

```
        111
1000)111000
     1000
     1100
     1000
     1000
     1000
        0
```

十进制

```
   7
8) 56
   56
    0
```

从二进制除法中，我们可以看出，二进制除法实际上就是采用移位的方法由被除数一次一次减去除数，直至除尽为止。在计算机中，除法运算实质上就是减法运算。

二进制和十进制的转换

前面已经讲到，在日常生活中数的进制是多种多样的，人们根据不同的需要可以采用不同的进制。也可以把一种进位制转换成另一种进位制。

在计算机中，最常用的是二进制，但有时也要用八进制，十六进制。同样根据不同的目的和要求，也要进行各种进位制的转换。前面我们已经说过，在计算机中，各种进位制的转换都是自动进行的。下面只是简单地向大家介绍一下二进制和十进制的转换方法。

从列表中，少年朋友可能已经发现，十进制中的1 2 4 8分别为二进制的001，0010，0100，1000，而且1，2，4，8是等倍增加的。从中我们可以推想出表

十进制	二进制
1	00000000001
2	00000000010
4	00000000100
8	00000001000
16	00000010000
32	00000100000
64	00001000000
128	00010000000
256	00100000000
512	01000000000
1024	10000000000

表中1 2 4 8 16 32……称为“权”。任何一个数都可以由“权”相加而得。

例 1 二进制1111换算成十进制是多少？

$1111 = 1000 + 100 + 10 + 1$

因为：

1000	是十进制的	8
100	是十进制的	4
10	是十进制的	2
+ 1	是十进制的 +	1
1111		15

可知二进制数1111是十进制的15。

例 2 十进制531换算成二进制是多少？

$531 = 512 + 16 + 2 + 1$

查表可知：

512……	1000000000
16……	0000010000
2……	0000000010
+ 1 …… +	0000000001
531	1000010011

可知十进制531转换成二进制的数为：1000010011

第四节　计算机语言

计算机是一种机器，要使它懂得人们要它干什么，要它怎么干，也得创造出一种人和计算机都能理解的语言。这就是我们常说的计算机语言。

计算机并不懂得人类的语言，不管是汉语还是英语，计算机只能识别0和1两种状态。在计算机发展的初期，向计算机发出的各种命令都是用0和1组成的一条条指令。做一道最简单的数学题，例如2＋3×5＝？就得编上十多条由0和1组成的指令。因为这些指令都是用计算机能直接识别的0和1组成的，所以这种指令叫做机器指令。能表达一个完整意思的若干条指令的集合就叫机器语言。

要解决一个复杂问题，就要编上成千上万条由0和1组成的指令，这些指令要是写在纸上，尽是0和1，密密麻麻一大片，令人眼花缭乱。所以，机器语言，不仅难编，难记，而且出了错也难检查出来。在计算机发展的初期，只有计算机专家才会编程序，才能使用计算机，这样就大大地阻碍了计算机的发展和计算机的推广应用。

科学家们为了使计算机更便于使用，希望能使用人们的自然语言编写指令，这样普通的人，即使不懂得0和1组成的机器指令，也能使用计算机了。

通过多年的努力和实践，科学家们终于研究出各种各样的计算机用的高级语言。这种计算机高级语言主要特点是，它很接近人们的自然语言和数学语言，用这种语言编写出来一条条命令，计算机就能够接收，执行。

例如计算5×3+8=？

用BASIC高级语言可以写出如下一条命令：

```
10 PRINT 5*3+8
```

当按下 ENTER 键以后，屏幕上立即显示：23。

从上面的例子中我们可以看到，只要写上已经规定好的一条命令，把通常的数学算式进行小小的变化（例如把“×”号改成“*”），计算机就能了解人们的意图，并得出正确的结果。这样，一个人只要通过短时间的学习，掌握一些计算机高级语言的最基本语句，它就能很快地学会使用计算机了。

前面不是说过，计算机只能识别0和1吗？为什么这里又说，计算机可以识别一些人们通常语言呢？如PRINT在英语中是打印，印刷的意思，计算机怎么会认得这个字呢？如果计算机配上汉字，我们还可以用汉语命令计算机为我们干事呢？

这里的奥妙，一说大家就懂。大家知道，两个语言不通的人进行交谈时，就得请来一个翻译。现在我们分析一下这两个人在翻译帮助下进行交谈时的过程：

1.甲用自己的方言先说第一句话；

2.翻译通晓甲的方言，所以听懂了甲说的第一句话是什么意思；

3.翻译用乙的方言把自己听懂了的甲的第一句话告诉乙；

4.虽然乙不能直接听懂甲说的是什么，但通过翻译，乙已经完全理解了甲说的全部意思。就这样，由于翻译的作用，两个语言不通的人顺利地交流了思想，互通了感情。从这里我们也可以看到，当好一名翻译，其最基本的条件就是

通晓两种以上的语言。

同样，我们也可以在计算机里预先存贮一个象“翻译”那样的编译程序。在设计编译程序时，使它既能识别人们常用的语言和数学公式，同时它又能把这些自然语言和数学公式等自动地翻译成计算机能识别的由0和1组成的编码。

由于解决了程序的编译问题，为不同目的的编译程序都一个一个的研制出来了。如：1. BASIC语言：

*BASIC是Beginner's All-purpose Symbolic Instruction Code（初学者通用符号指令代码）一词的缩写。

2. FORTRAN语言：

*FORTRAN是FORmula TRANslation（公式翻译）的缩写。

3. COBOL语言：

*COBOL是COmmon Business Orienteb Language（通用商业语言）的缩写。

另外还有：PL/1（大型通用语言）

PASCAL（结构程序设计语言）

C语言 ……等等

我们应该着重指出，要使用那种高级语言，预先就应该有这种编译程序。例如，我们向大家介绍的PC-81型微型计算机，在其主机里就预先存放有BASIC解析程序，所以我们才能用BASIC高级语言和计算机通话。

大家会问，前面提编译程序，为什么又出来一个BASIC解析程序呢？

原来，在计算机中设“翻译程序”时，通常有两种做法：

(1) 编译方式：在计算机中先存放一个编译程序，当把

用高级语言编写的程序（我们称为源程序）送入计算机时，编译程序便把源程序整个地翻译成机器指令（我们称为目的程序），然后计算机才开始执行目的程序，最后得出结果。

在编译方式中，最主要特点是把源程序整个地翻译以后才去执行。这就好象一个搞笔译的翻译，领导给这个搞笔译的人一篇文章，这个人把整篇文章都翻译好以后，再交给领导，而不是翻一段交一段。

(2) 解析方式：和编译方式最主要差别是，在解析方式中，不是整体翻译，而是逐句翻译，即译出一句立即执行一句。这就好象一个口头翻译一样，人家说一句，就得翻译一句，而不是等人家都说完了再去翻译。BASIC语言就是这种解析程序。

计算高级语言的出现，对计算机的发展起了具大的推动作用。甚至可以这样说，没有计算机高级语言，计算机的推广应用是不可能的。自从有了高级语言以后，一般科技人员，甚至中、小学生，可以不去弄懂计算机内部构造和原理，只要了解高级语言的使用方法，就能得心应手地操纵计算机，进行设计，进行科学计算，进行各种事务管理。

第四章　BASIC语言初步

第一节　什么是BASIC语言

“语言”对我们来讲并不陌生，日常生活中无处不接触语言，若是没有语言简直是不可想象的事。因而，要使计算机能够按我们的要求做事，那我们就必须通过一种特殊的“语言”，把要做的事情告诉了它，而计算机也是用这种“语言”把它工作的结果告诉我们。这样一种特殊的“语言”叫做“计算机语言”。现在让我们简单地说明一下计算机语言——BASIC语言。

BASIC语言是一种比较简单易学的语言。BASIC是英语Beginer's All-purpose Sgmbolic Instruction Code的缩写，它的中文意思是“初学者通用符号指令代码”，是由两个美国人在1964年为一些初学计算机的人编制的一种计算机语言。所以具有易学习、好掌握和使用方便的特点。但对我们使用汉语国家的人来讲有一个语种的问题，遗憾的是现在还没有一个类似于BASIC语言，面向广大初学者的较为实用的汉字语言供大家学习。这一部分工作虽有进展，但还不够完善，故还希望你们当中某些愿献身于计算机科学事业的少年朋友来完成。

第二节　BASIC语言基础

前面我们已经或多或少地使用了一些BASIC语句了，

现在不妨再看一段用BASIC语句写的程序：

```
10  LET  A=9
20  LET  B=25
30  LET  C=5
40  LET  D=1
50  LET  Y=(A*5+B*4+C*3+D*2)/40
60  PRINT  Y
70  STOP
RUN↙ (↙是按下ENTER回车键的意思)
```

幕上显示

4.05

这段程序是计算 $Y=(A\times5+B\times4+C\times3+D\times2)/40$ 表达式的值。

在这段程序里，可以看到BASIC语言编写的程序一般由这几个部分组成。首先，是写在每一行左边的“数字”，我们叫它为行号或序号。行号是由小到大递增的，例如，10、20、30、40……，它只是表明作一件事情的顺序。比如，当我们要解一道数学题或且要做别的什么事时，总要有一个先做什么，后做什么的顺序。虽然并不要求给这个顺序编个号码，但我们要按照这个顺序来解决问题的。同样，当我们在同计算机“交谈”时，不但要告诉它你想让它做什么事，还要把这个顺序告诉它。这样计算机才可能懂得你的想法而正确地为您工作。这和日常生活中的谈话所不同的是：你要把这个顺序用号码表示出来。那么，就要求你在每个语句的最左边写上个标号，用以说明这语句能被计算机接受并能执行的顺序。我们常把写在每一行左边的符号叫做语句标号或行号。这个行号在BASIC语言中的要求是：没有符号(+、

－）的整数，即不能是小数、分数或零。

再有，我们看到这段程序一共从上到下有7行（RUN不算）。我们把每一行叫做一个BASIC语句行，或称为一个语句行。每一行都是一个完整的句子，这类似于我们谈话中的一句话。而每一个语句行都有它的确切含意，它至少说明计算机一个的“动作”。但是，正象我们不能用一句话就能把事情说清楚一样，也不能用一个语句就能使计算机把我们的思想表达清楚。所以，实际上在编写一个程序时，要求由若干个语句组成，而它们之间的顺序关系用语句标号来加以说明。计算机执行程序的顺序是：按行号从小到大的顺序执行。

另外，我们还看到在每一行里都是由一个或几个英文单词和表达式（或没有表达式）组成。其中英文单词的意思是指让计算机做什么样的工作，而表达式的含义就是指计算机工作的内容。我们就把英语单词叫做“BASIC”语言中的关键字。有些类似于我们在数学中遇到的算术式、函数式的表达式，我们称它为BASIC语言中的表达式，简称表达式。

我们学习BASIC语言，主要就是学习BASIC语言中的关键字的使用方法，及一些有关的规则。

第三节　BASIC 语句及使用

一、BASIC 语句（一）

现在我们向大家介绍BASIC语句，首先介绍三个BASIC语句和一个BASIC命令。

1. LET 赋值语句：

LET这个英语单词是“让、假设、使”的意思。

我们在数学里常碰到这样的问题“设A等于100，B等2（A=100，B=2）。如果用英语表示则是：Let A be equal to 100,let B be equal to 2 (LET A=100，LET B=2)。

计算机中的BASIC语言，也正是引用了英语中的表示习惯。即BASIC语言中写成：LET A=100,LET B=2。为称呼上的方便，我们把LET A=100叫做把100赋给A，所以把LET语句叫做赋值语句。

赋值语句的格式：

〔行号〕LET变量=表达式

其中表达式可以是复杂的表达式或者是常数。

赋值语句（LET语句）的用途：

可以用它来假设一个数：如 设 A=100

可写成： LET A=100

还可以用它来进行一个算式的运算：如

X=100+2+5-3-9

可写成： LET X=100+2+5-3-9

2. PRINT 打印语句：

PRINT的英文意思是“打印”。BASIC语言就是引用了这个意思。

打印语句的格式：

〔行号〕PRINT 打印项1$\left(\begin{matrix};\\,\end{matrix}\right)$打印项2$\left(\begin{matrix};\\,\end{matrix}\right)$…

其中括号里的“；”和“，”可以根据不同的需要而任选其一。打印项可以是一个算术式或一个变量，还可以是常数。当要打印一个或几个字符时，可以用引号（“ ”）把

要打印的字符内容放在引号当中。

打印语句（PRINT语句）的用途：

（1）可以在显示器（或家用电视机）上显示出要求显示的数、字符。

应当向大家介绍一下字符的概念，BASIC 中字符是 指英文字母和数字或是符号（象+，－，*，/，＝等）。

如：

```
5  PRINT "100" ↙ (按下ENTER 键)
RUN↙
```

显示出：

```
100
```

又如：

```
5  PRINT "HELLO" ↙
RUN↙
```

显示出：

```
HELLO
```

由上面例子，可以看出，把在两个引号之间的任何字符都将原样被显示出来，不做任何变更。

（2）可显示一个变量的值或一个计算结果。

如：

```
5    LET    A=100↙
10   PRINT  A↙
RUN↙
```

显示出：

```
100
```

```
10   LET    X=100+2+5-3-9↙
20   PRINT  X↙
RUN↙
```

显示出：

```
95
```

（3）可以计算一个算术式并同时显示出这个算术式的计

算结果。

如：　5 PRINT　100+2+5-3-9↙

RUN↙

显示出　95

打印语句（PRINT语句）的这种功能与赋值语句（LET语句）有所不同，它是把一个算术式写成一个打印项的形式，而不是写成变量=表达式的形式。这也是与赋值语句有差别的地方。

另外关于打印语句（PRINT语句）的其它用途作为深一步的内容，将在后面介绍。

3. STOP停止语句：

STOP的英文意思是“停止”。BASIC语言中的STOP的意思也是停止。

停止语句的格式：

〔行号〕STOP

停止语句的用途：当我们需要程序在那个地方停止下来时，都可以使用STOP语句

如：　10 LET　A=5 ↙

20 PRINT A ↙

30 STOP ↙

RUN

显示出：5

程序在30号语句处停止下来。

又如：　10 LET　A=5 ↙

20 PRINT A ↙

30 STOP ↙

40 PRINT “A” ↙

显示出：5

并不显示A这个字符；说明程序在30号语句处停止下来并没有执行40号语句，所以没有显示字符A。

4．RUN运行命令：

也许我们熟悉英文“RUN”是“跑”的意思，这是对的，但这里我们选取了“RUN”的另一个意思是“运行”。不过都没有什么关系，只不过是一个更易为大家所能接受的习惯问题。

当我们用BASIC语言编好了一个程序，那么怎样才能开始工作？我们打入“RUN”这个指令，命令计算机工作。所以，我们习惯地把“RUN”命令译成③“运行命令”。

运行命令的格式：

RUN

不要在RUN前面加行号，这一点也是语句和命令的不同之处。

5．BASIC语言中的一些符号的表示习惯和一些基本术语：

字符：在BASIC语言中我们把英文字母“A、B、C…X、Y、Z”和数字“0，1，…，9”，及一些常用的符号“=，+，·，…”等，称为字符。

变量：BASIC语言中使用的变量类似我们数学中的变量，可以用字母和数字命名，但打头的第一个字符一般要求是字母。

表达式：BASIC语言中的表达式类似代数式或函数式，一个BASIC表达式可以由变量和算符组成。

BASIC算符：在BASIC语言中使用的算术符号基本上与数学中相同，但有些符号为了与一些英文字母产生混淆而有一些修更，比如：

	BASIC 中使用的算符	常用算符
加法	+	+
减法	-	-
乘法	*	×
除法	/	÷/
指数	* *或↑	指数形式
如：3的5次幂	3 * *5或3↑5	3^5

以上我们可以看出BASIC 语言中的乘法和指数运 算 的符号与我们常用算符有所不同。这是由于乘法算符“×”易与英文中的小写字母“x”混淆，而选用了一个星号“*”来表示乘法运算。其次，指数运算的表示方法也不同，这是由于在计算机的显示器上一般不容易在同一行上显现出不同字体，同时英文打字键盘上也没有那种字体。所以，就选用了二个星号“* *”表示指数运算。

另外，数字零（0）的表示法，在BASIC语言中，或一些别的计算机语言中，数字零（0）一般都写成“Ø”，这也是为了区别于英文字母“O”。不过，在键盘上数字零和英文O是能辨认出来的。

键盘上的圆点“.”充当小数点。

在数字中不能使用逗号“,”，比如：二百万应记作2000000，而不应记成2,000,000。

BASIC语言进行运算的顺序如下：

(1) 做括号内的运算。

(2) 做函数运算。

(3) 做幂运算（* *或↑）。

(4) 做求负运算。

如：-5 * *3 意思是-(5 * *3)，先作5^3运算之

后作求负（-5^3）运算。

$-5*3$ 意思是 （-5）$*3$，先作求负（-5）的运算，之后做乘法运算（-5）$*3$。

（5）做“*”或“/”，假如这二者同时出现，那么先做最左边的。

如： $8*3/2$ 意思是 $(8*3)/2$

先做乘法（$8*3$），而后做除法（$8*3$）/2运算。

$8/3*2$ 意思是 $(8/3)*2$，

先做最左边的除法（8/3）运算，而后再做乘法（8/3）*2。

（6）“+”或“-”运算最后做，假如它们出现在同一个表达式内，则先做最左边的。

如： $8+4-2$ 意思是 $(8+4)-2$

$8-4+2$ 意思是 $(8-4)+2$

以上我们谈了有关BASIC运算顺序的问题，希望大家能自己动手试一试，以便加深记忆。当大家真正了解了这个问题之后，在进行设计程序过程中就不会出现计算顺序上的错误。大家可以试做这样一个题，即4/3*6，4/(3*6)，(4/3)*6看看有什么不同结果，为什么？

6.BASIC语言中我们使用的数：

“数”在我们日常计算中并不陌生，所以也许有人要问“BASIC中使用的数与我们现在使用的数难道不一样吗？”其实，BASIC语言中的数与我们常用的数区别是不大的。不过在有些数当中，它更规格和严紧一些，而不象我们使用的那样随便就是了。但还有一些数，是由于计算机计算有误差，而引起了这些数也有误差。所以我们要向大家介绍BASIC中的数有些什么特点，以便大家更好的使用它。

在BASIC语言中，你能从显示器上看到的数字通常是有

限的几位。比如，12位、8位或6位等。这要根据你使用的计算机而定。但不管你使用那种计算机，它显示给你的数字的位数也是有限的，而这些数字又都是以整数或小数形式给你的。比如：

输入	输出	输入	输出
3	3	4＊100000	400000
3/4	0.75	4＊10000000000000	4E＋13
2/3	0.66666667	9000＊＊2	810000000000
14/3	4.66666667	30000＊＊3	2.7E＋13

注意2/3是由舍入的8位数字来表达的。数10000000000或更大的数，输出用指数（科学记数）形式。如上：4E＋13，即 $4\times10^{13}=4\times10000000000000$ 2.7E＋13，即 $2.7\times10^{13}=2.7\times10000000000000$

在应用计算机计算过程中可能会出现由指数（科学记数）形式输出的数。大家应当记住它。

另外，计算机在任何时候都不会给我们显示一个分数，即使输入的是一个“类似”分数的数。我们说“类似”分数，是由于它形状有些象分数。如：$\frac{3}{4}$，我们若把它看成是“4分之3”显然没有错，不过$\frac{3}{4}$在BASIC语言中的意思是“3被4除”的意思。所以，当你把分数输入计算机时，最好进行一些变更。如$2\frac{3}{4}$可写成2.75或$2+\frac{3}{4}$。虽然，有一些数并不需要变更，如$\frac{3}{4}\rightarrow\frac{3}{4}$，你最好也把它理解成是“3被4除”。这样做且可以防止错误。

我们举两个例子，介绍一下BASIC语言中的算术表达式的正确表示方法。

例 1 $\dfrac{3+8\times 9}{6^3-5}$

BASIC语言表达式：

(3+8*9)/(6**3-5)

或 (3+8*9)/(6*6*6-5)

但不允许写成：$\dfrac{(3+8*9)}{6**3-5}$

或 (3+8*9)/6**3-5

例 2 $\{[(2+3)\times 4]^2-5\}^3$

BASIC语言表达式：(((2+3)*4)**2-5)**3

注意：BASIC也允许按照一组带有括号的算符来书写一个BASIC表达式；但应注意你所使用的括号的层次和左右括号的对称。计算机是连同括号及算符、数字一起翻译的。

下面是一些例子：

传统的记法。	BASIC表示法
(1) $3x^2+x-6$	3*X**2+X-6
(2) $\dfrac{5}{2x}$	5/(2*X)
(3) $[(A+B)^2-C^2]^3$	((A+B)**2-C**2)**3
(4) $1+\dfrac{1}{1+\dfrac{2}{1+\dfrac{3}{1+4}}}$	1/(1+(2/(1+(3/(1+4))))) 或：1/(1+2/(1+3/(1+4)))

至此，我们已知学习了三个BASIC语句和一个命令了。它们是：

(1) LET 赋值语句；

(2) PRINT 打印语句；

(3) STOP 停止语句；

(4) RUN 运行命令。

还了解了一些BASIC语言中的基本概念。如：字符、函数、表达式，算符的使用及“数”和书写表达式。

下面我们就应用已学习到的这些概念，试用BASIC语言做几道练习题，来进一步掌握它们是如何被使用的。当然，现在还不要求大家能够自己独立去做，而是希望大家能够看懂例题，理解它的意思，如果你能轻松或不怎么费力就完全理解了例题的每一步，那么说明你是个相当了不起的学生，在以后的更进一步学习中一定会学得很不错。如果，你是经过一段思考、复习而能理解了例题的，那么经过了你的努力你就会更好地掌握住前面学习的东西，而为以后的学习打下基础。如果你对例题的某些地方倘不能独立地通过思考以达到理解的目的，希望你能重新再回过头来重新阅读一下前面有关的章节。

例 1 计算 $100-10+9=?$

解

```
10  PRINT 100-10+9
```

或

```
10  LET  X=100-10+9
20  PRINT  X
```

运行后得到结果为：99

从这个例子里我们可以看到LET赋值语句和PRINT打印语句虽然都可以用来进行计算，但它们在使用上是不同的，我们把100－10＋9可以看成是一个打印项，所以，可使用PRINT打印语句进行运算。但是如果写成以下形式：

```
10  PRINT  x=100-10+9
```

或

```
10  LET 100-10+9
```

都是不对的，这是由于从格式上就不符合PRINT打印语句和LET赋值语句的要求，从英文的意思也不合乎语法的要求。比如：

LET 100－10＋9，LET是赋给某个变量值的意思，而LET 100－10＋9中只有一个式子而没有变量。

例 2 一球从h高处以初速度$V_0=100$cm/s向下扔，经10秒到地．求h＝？

解 $h=V_0t+1/2gt^2$

其中 $V_0=100$；$g=980$；$t=10$

BASIC表示：H＝V0＊T＋1/2＊G＊T＊＊2

程序，
```
10  LET  V0=100
15  LET  G=980
25  LET  T=10
35  LET  H=V0*T+1/2*G*T**2
45  PRINT  "H=";H
55  STOP
```
运行后得到结果为：

50000 （即50000cm或500m）

由例2我们可以看到，语句标号或行号是由小到大递增的。计算机进行计算或对你的程序进行处理的步骤也是由行号小的开始到行号大的顺序处理的。即10→15→25→35→45→55完毕．而对于行号是有要求的，这一点在前面已讲过，所以，在解题过程中，行号的使用应当按照讲过的规则使用。这道例题计算机具体工作的步骤是（按行号顺序）：

(1) 10语句　把100这个数赋给变量V0；

(2) 15号语句　把980这个数赋给变量G；

(3) 25号语句　把10这个数赋给变量T；

(4) 35号语句　计算V0＊T＋1/2＊G＊T＊＊2
并把计算的结果赋给变量H；

(5) 45号语句　显示变量H的内容，其中“H=”是把引号内的字符原封不动的显示出来；

(6) 55号语句　停止程序的执行。如果没有55号语句55STOP　程序也会自动停下来，因为后面没有任何BASIC语句了。

例 3　已知　一直角三角形的a、b二边，求c边？(a=3；b=4)

解　a=3，b=4，

由直角三角形定理（勾股定理）

知：$c=\sqrt{a^2+b^2}$

BASIC表示　C=(A**2+B**2)**(1/2)

或C=(A*A+B*B)**(1/2)

程序：10　LET　A=3

15　LET　B=4

25　LET　C=(A**2+B**2)**(1/2)

35　PRINT “C=”，C

运行后结果是：C=5

从例3中可以看出，计算机解一道数学题的顺序实际上就是我们习惯上所使用的解题步骤。但是它的顺序是明显表示出来的，在BASIC语言中是使用行号来表示的。所以，我们利用BASIC语言来解习题，也就是利用BASIC语言把我们惯用的解题方法描述出来的过程。

例 4　计算半径 $2\frac{3}{4}$ 的圆周长和面积

解　已知　$R=2\frac{3}{4}$　求H=？S=？($\pi=3.1416$)

BASIC表示　R=2.75　H=2*3.1416*R

S=3.1416*R**2

程序：

```
10  LET  R=2.75
20  LET  H=2*3.1416*R
30  LET  S=3.1416*R**2
40  PRINT  "H=";H,"S=";S
```

或

```
10  LET  R=2.75
20  PRINT "H=";2*3.1416*R,"S=";
    3.1416*R**2
```

运行结果是：

```
H=17.2788   S=23.7584
```

由于计算机只能采用整数或小数，我们不能用$2\frac{3}{4}$直接运算，就要转换成2.75或2+3/4来代替$2\frac{3}{4}$。表达式2*3.1416*R是表示圆周长的积，注意关于乘法的符号是*．而表达式3.1416*R**2是表示面积的积，其中的指数运算是使用了**符号。对于表达式3.1416*R**2的计算顺序是（1）R**2，（2）3.1416*(R**2)，而不是3.1416*R 之后在做指数运算，即 (3.1416*R)**2。

另外，对于例二中的表达式：

H=V0*T+1/2*G*T**2

及例三中：

C=(A**2+B**2)**(1/2)

也有一个运算顺序的问题。对于以上的几个表达式的写法，你是否已经了解它们的运算顺序？如果有什么问题可以再看一看有关BASIC运算顺序那一段的叙述。

最后，建议大家用计算机验算一下以上的几道例题，以便加深理解。也可以按你自己的计算方法和想法更改一下程序中的参数、变量和符号，看一看结果如何。当然，也可以用你认为可靠的方法验证一下你的改动是不是对的，如果不对，找一找问题出在哪里，并试着修改它。总之，你要大胆地去试，不要有任何顾虑。

练习题一

1.写出等价于BASIC表达式的传统表达式：

① $-5*3**2$　　② $-3**2$

③ $2+3/4$　　④ $2*3/4$

⑤ $2*3/4/4$

⑥ $(((1+2)**3+4)**5+6)**7$

⑦ $A+B/C/D+A/B**3$　　⑧ $N**(1/N)$

⑨ $H=V0**T+1/2*G*T**2$

⑩ $C=(A**2+B**2)**1/2$

⑪ $3.1416*R**2$

2.写出等价于传统的表达式的BASIC表达式：

① $3.14\cdot R^2$　　② $\dfrac{1-X^2}{1+X^2}$

③ $\dfrac{1-X}{1-\dfrac{Y}{2}}$　　④ $\dfrac{4\pi R^3}{3}$

⑤ A^2+B^2-2AB　　⑥ $\dfrac{N(N-1)(N-2)}{1\times2\times3}$

⑦ $\{[(1-a)^2-b]-c\}^3$

3.修正下述各BASIC表达式：

① 2,000　　② A÷L

③ 2(L+W)　　④ $\frac{1}{2}$BH

⑤ $2\frac{1}{2}*4\frac{1}{3}$　　⑥ 2*3

⑦ (7+8)/2

4.写出下述各式的BASIC 程序并且写出执行的结果：

① (92+93+94+95)/4　　② 986÷3

③ 3.1416×5.5^2　　④ $\frac{4}{3}(3.1416)\cdot(5.5)^3$

⑤ $\frac{1}{2}+\frac{1}{3}+\frac{1}{4}+\frac{1}{5}+\frac{1}{6}$

⑥ $\frac{1}{2}+\frac{1}{2\times3}+\frac{1}{2\times3\times4}+\frac{1}{2\times3\times4\times5}$

5.下述程序错误在哪里？并写出正确的程序：

① 10　PRINT　$3.1416*2.71828^2$

② 10　LET　A+B=10.01*5

③ 5　PRINT　6(5280+333)

4　PRINT　"A="

6.指出下列表达式中的运算次序：

① 5*6.2**3/2-6*(-3)

② 64.5+6.3*(SIN(3+6))**2/2+6

注：②中SIN等于sin正弦函数

二、BASIC语句（二）

1. GOTO　转移语句：

GOTO 在 BASIC 语言中表示"转到某个地方"，即"转移"的意思。

GOTO 转移语句的格式：

〔行号〕　GOTO〔行号〕

GOTO 转移语句的用途：

前面已经说明，在一般情况下，程序是按行号由小到大的顺序执行的。但是，如果需要的话，也可改变这种顺序。

这时，只要使用“GOTO”语句就可以达到这一目的。当程序执行到“GOTO”语句时，就不再按行号了，而是按你的要求，即：“GOTO”语句中右边你指明的那个行号开始并由那里再顺序往下执行，直到再一次遇到一个“GOTO”转移语句而改变执行顺序，否则，将执行下去。所以，我们说当程序转到“GOTO”语句中指明的（右边的）那个行号时，它又开始了按行号由小到大的顺序执行了。例如，钟表总是按顺时针方向运行的，当把钟表的表针向反方向（逆时针方向）转动时，表针就会由于你的转动而被转到你要求的某一位置（这显然不是表针自动运行的顺序）。但只要到达这一位置后，表针又会按顺时针方向自动运行。也就是说你可以任意转动，改变表针的位置，表针永远按顺时针方向运动的这一性质是不能改变的。

例如：
```
10  LET  N=0
20  LET  N=N+1
30  PRINT  N
40  GOTO  20
```

运行后结果是：

```
1
2
3
⋮
```

（若想停止该程序的运行，请按下 SHIFT 键并同时按下 BREAK 键）

这些程序在BASIC语言中是这样执行的呢？

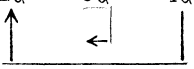
（1）计算机首先把零赋给变量N，N=0。为了理解上的方便，我们可以想象“有一个‘小黑匣子’N，你可以随便往里面放入数，如：一个表达式或一些字符”。而语句10LET

N=0 就是把0 放到N的那个“小黑匣子”里去。

(2) 20 LET N=N+1 计算机把你在10号语句中放入的零，或者说变量N中放入的数据加上1，然后再放回到N中去，这时N中的数就会比以前的数据大1。

(3) PRINT语句 打印出N当前的数。

(4) GOTO转移语句，程序运行到这里就转移到GOTO右边的那个行号（如：40 GOTO 20 中的20号）去执行程序。例中，就是转移到20号语句，把N里面的内容“再加1”之后，“重新放回”到N中去之后执行30号语句等，就这样：10⟶20⟶30⟶40循环不停地执行下去。



对于GOTO转移语句，我们一般又称它“无条件转移语句”。这是由于程序执行到GOTO语句时计算机不做任何别的选择，而立即转移到GOTO语句指明的那个行号继续执行。

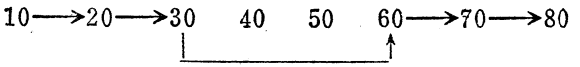
例如：

```
10  LET  A=22
20  PRINT  A
30  GOTO  60
40  LET  B=33
50  PRINT  B
60  LET  C=44
70  PRINT  C
80  STOP
```

运行结果：22

44

这段程序运行的顺序是：



所以我们只得到了A和C的结果，而没B的结果。这是由于“30 GOTO 60”使程序执行的顺序变动了，由30号语句跳过了40、50号语句而执行下面的60、70、80号语句了。

2. INPUT 输入语句：

INPUT这个英文的原意是“输入”。BASIC语言中借用了这个意思。当我们需要把一些计算用的数据输入到你的BASIC程序中去的时候，就可以使用INPUT输入语句。

INPUT输入语句的格式：

〔行号〕 INPUT 变量

其中，行号就是INPUT输入语句在你程序中的位置或顺序。变量是指那个准备放数据的“小黑匣子”。当计算机执行到INPUT输入语句时就会停下来等待你的数据打入。这时，你若打入一个数据并按下回车键（ENTERH），计算机就会把这个数据自动地放入那个“小黑匣子”中去，以供你的程序在运行中使用。

例如：

```
10 INPUT A
20 PRINT A
30 STOP
RUN↙
```

运行结果：首先在显示器上方出现一个光标。

这时打入500↙

显示器上将显示出：500

程序开始运行后，首先遇到10号语句，即（INPUT语

句），这时，计算机等待你打入一个数（本例中500）。否则，计算机就一直处于等待状态。当打入500并按下回车键（ENTER）之后，计算机才认为你的数据已经输入完毕。再执行20号语句（20 PRINT A）时，你所通过“INPUT”语句输入的数据就被20号语句（20 PRINT A）显示在显示器上了。

例：设需要计算如图4—3—1所示的面积S（阴影部分），试编写BASIC程序？

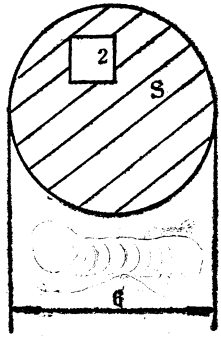


图4-3-1

解：面积S等于一个半径是$\frac{6}{2}$的圆面积减去一个边长为2的正方形面积。

$$S = E_{圆} - F_{方}$$

$E_{圆} = \pi \cdot R^2 = 1/4\pi D^2$；$F_{方} = A \times A$

其中 $\pi = 3.1416$；$A = 2$；$D = 6$

程序：

```
10 PRINT "D=";
20 INPUT D
30 PRINT
40 PRINT "A=";
50 INPUT A
60 PRINT
70 LET E=3.1416*D**2/4
80 LET F=A*A
90 LET S=E-F
100 PRINT "S=";S
110 STOP
```

RUN↙

6↙（键盘输入并按下回车键）

2↙（键盘输入并按下回车键）

运行结果 D=6

A=2

S=24.2744

在这个程序中，10～60号语句是用来输入圆的直径D的数值和正方边长A的数值。而当和10、30、40、60号语句是为了输入方便设置的，若把这四条语句去掉也不会影响程序的正常执行。但20、50号语句是不能去掉的，这两条语句是用来输入D和A的数值的。70号语句计算圆面积E，80号语句计算正方形面积F，90号语句计算面积S。100号语句是为把计算的结果显示出来，110号语句表示程序执行到这儿将停止。在这个程序中我们还可以看到PRINT打印语句的应用情况，当我们需要显示计算结果及显出一些对我们有所提示的信息时，一般也都使用PRINT打印语句。

另外，还有一点请大家注意，INPUT语句的格式是：

〔行号〕 INPUT 变量

在INPUT的右边要求是一个变量，而不能是一个表达式或者常数，建议在INPUT输入语句的前面或后面使用一个PRINT语句，用来显示一个提示信息，以防止输入数据时出错。

3. IF…THEN（如果…则…）条件转移语句：

IF … THEN … 的英文意思是“如果…则…”，在BASIC语言中称它为条件判别转移语句或条件转移语句。

IF …THEN 条件转移语句的格式：

〔行号〕 IF 〔条件〕 THEN 〔语句〕

其中，“条件”是指由大于“>”号，小于“<”号，等号“=”，…，等这一类关系符号构成的关系式。“语句”就是我们学习的BASIC语句，可以是任何BASIC语句，比如PRINT打印语句，LET赋值语句，GOTO转移语句等。

IF…THEN 条件转移语句的用途：

至今我们所看到的BASIC语句能使我们输入信息，计算并打印显示出结果。但这并不够用，很多情况下，我们希望计算机能面对不同的情况作出种种判断。IF…THEN 条件转移语句就允许我们在程序中去做判断。

为了告诉大家怎样使用条件判别语句，首先介绍一下构成条件转移语言中使用的关系符号和关系式。

在BASIC语言中，有以下几种关系比较符可以使用：

BASIC关系比较符	含义	相当的算术符号
=	等于	=
<	小于	<
>	大于	>
<=	小于或等于	≤（≤）
>=	大于或等于	≥（≥）
<>	不等于	≠

大家使用时应注意：≤与<=，≥与>=，≠与<>的不同和BASIC语言中的用法。用一个关系符把两个变量或两个算术表达式连系起来，就构成了一个BASIC关系式或者说是构成一个“条件”。我们在介绍条件判别语句的格式时所说的“条件”，就有很大一部分是由这些关系符及变量、表达式构成的。

例如：　10　IF　X>Y　TIEN　PRINT X

其中　X>Y　就是一个“条件”，PRINT X 就是个

“语句”。

又如：

```
10  INPUT  A
20  IF  A>100  THEN  GOTO 50
⋮
50  STOP
```

其中：A>100是“条件”，GOTO 50是“语句”。它的意思是“如果A大于100则转到50号语句”。

当然，“条件”还可以复杂点，看下面的例子：

```
⋮
50  IF  X+Y>A/B-C  THEN  STOP
⋮
```

这里的“条件”就比较复杂一些了。也就是说条件判别语句中的“条件”部分是由一个关系比较符和左、右两个表达式构成的。那么，这里的表达式是否具有象PRINT语句一样的计算功能？回答是表达式是可以完成计算功能的。也就是说，关系比较符的两边的表达式都是首先计算表达式的值，然后用这个表达式的值去进行比较的。

现在为了帮助大家理解IF…THEN…条件判别语句的执行情况，用图4—3—2来说明。

IF…THEN…（条件判别语句）

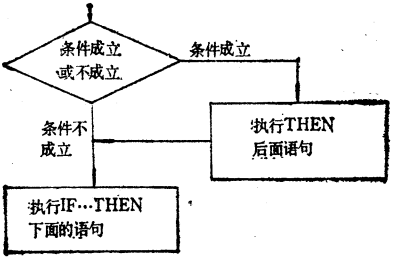


图4—3—2

如：
```
5   Y=500
10  INPUT X
20  IF X>Y THEN PRINT X
30  GOTO 10
```

左边这张（见图4—3—3）仅表示了条件判别语句（20号语句）的执行情况，有关更加详细的整个程序的执行图（流程图）在第五章中讲。

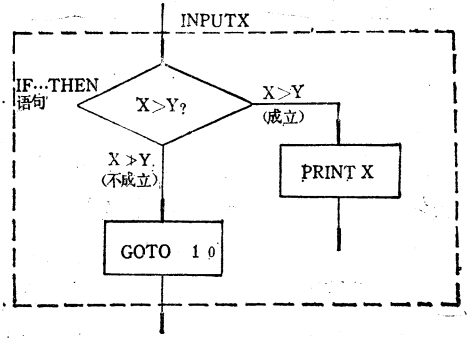


图4—3—3

例 求$1+\frac{1}{2}+\frac{1}{3}+\frac{1}{4}+\cdots+\frac{1}{1000}$的和？

解 求这个和的时候，我们由和S=0（S是个变量名）开始，然后计算S=0+1=1，计算$S=1+\frac{1}{2}=1\frac{1}{2}$，再计算$S=1\frac{1}{2}+\frac{1}{3}=1\frac{5}{6}$，如此等等类推，直到我们得到和$S=1+\frac{1}{2}+\frac{1}{3}+\frac{1}{4}+\cdots+\frac{1}{1000}$。当这最后一项$\frac{1}{1000}$已经加上时，这个求和的过程就结束了。这个重复加

法的过程按照下述的框图进行，并注意（图4—3—4）中的回路（重复）部分。

```
10  LET  S=0
20  LET  N=1
30  LET  S=S+1/N
40  IF  N=1000  THEN
        GOTO  70
50  LET  N=N+1
60  GOTO  30
70  PRINT  S
80  STOP
```

运行结果：7.4854709

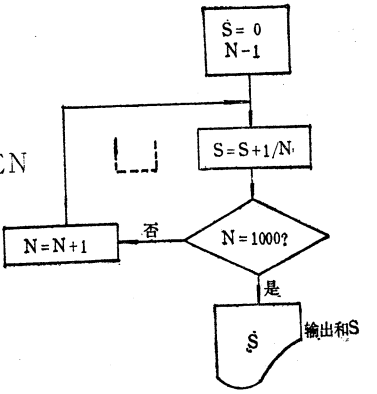


图4—3—4

这个框图由S=0和N=1开始，然后我们置 $S=S+\frac{1}{N}$，它的意义是S以前的值（“老”值）加上$\frac{1}{N}$就得到S的新值。菱形部分是判断条件“N=1000”，若N≠1000，N的值再加1，然后使用N的新值再产生S的新值。若N=1000，那么，就显示（打印）出S的值，即求得的全部和，此程序就告结束。

对于这个问题的BASIC程序能够直接从框图写出。在更为复杂的问题中，画出该问题的框图能帮助我们去设计该问题的BASIC程序。有关如何根据需要解决的问题，绘制出一张有逻辑性的框图并据框图设计出BASIC程序，在第五章里会详细介绍。现在仅要求大家能够理解。

下面介绍求解一元二次方程的例子：

例 编写求解一元二次方程

$$ax^2+bx+c=0$$

的BASIC程序，其中系数a，b，c的具体数值由键盘输入（INPUT语句）。

解 从代数中知道，一元二次方程的根的可能情况如下：

(1) 如果$a\neq 0$，$D=b^2-4ac>0$，则有二个相异实根

$$x_{1,2}\frac{-b\pm\sqrt{b^2-4ac}}{2a}$$

如果$a\neq 0$，$D<0$，则有一对共轭复根

$$x_{1,2}=-\frac{b}{2a}\pm\frac{\sqrt{-(b^2-4ac)}}{2a}i$$

如果 $a\neq 0$ $D=0$，则有重根

$$x_1=x_2=-\frac{b}{2a}$$

(2) 如果$a=0$，$b\neq 0$，则一元二次方程退化成一元一次方程，所以只有一个根

$$x=-\frac{c}{b}$$

(3) 如果$a=0$，$b=0$，则方程无意义。

BASIC程序：

```
10  PRIN   "A*X**2+B*X+C=0"
20  PRINT  "A=";
30  INPUT  A
40  PRINT  A
50  PRINT  "B=";
60  INPUT  B
70  PRINT  B
80  PRINT "C=";
90  INPUT  C
```

```
100  IF  A=0  THEN  GOTO  260
110  LET  D=B*B-4*A*C
120  IF  D>0  THEN  GOTO  160
130  IF  D<0  THEN  GOTO  200
140  PRINT "X1=X2=" ; -B/(2*A)
150  GOTO  300
160  LET  X1=(-B+D**(1/2))/(2*A)
170  LET  X2=(-B-D**(1/2))/(2*A)
180  PRINT "X1=" ; X1, "X2=" ; X2
190  GOTO  300
200  LET  R=-B/(2*A)
210  LET  M=(-D)**(1/2)/(2*A)
220  IF  M>0THEN  GOTO  240
230  LET  M=-M
240  PRINT "X1=" ;R; "+" ; M; "I" , "X2
       =" ; R; "-" ; M; "I"
250  GOTO  300
260  IF  B=0  THEN  GOTO  290
270  PRINT "X=" ; -C/B
280  GOTO  300
290  PRINT "A=0" , "B=0" , "ERROR"
300  STOP
```

可以看出，如果把最后的“300 STOP”语句换成“300 GOTO 10”。则计算机又询问有没有第二个方程求解。这样一来，无论多少个方程，都可以用一个程序，一个接一个地求解出来。

为了帮助大家能更好地理解这个程序，解释一下：

10号语句至90号语句是用于输入A、B、C这三个变量的数值和显示出一些提示信息。

100号语句是判别A是不是等于零（A=0？），如果A=0，则转移到260号语句，260号语句是判别B是否也等于零（B=0？）。如果B=0，则转移到290号语，显示出“A=0 B=0 ERROR”的信息，告诉你因为你输入的变量A=0，B=0所以此方程无意义。英文写ERROR是“错误”的意思。如果A不等于零(A≠0)则执行110号语句。

110号语句是用来计算一元二次方程根的判别D的数值(D=B*B-4*A*C)。

120和130是判别D是否大于或小于零。(D>0？ D<0?)。如果，D>0则转移到160号语句

如果 D<0，则转移到200号语句。

140号语句是用于计算 D=0时，一元二次方程的重根 $x_1=x_2=-B/2A$，由于120，130号语句已经把D>0和D<0的两种情况考虑到了，所以只剩下 D=0的情况了，则140号语句用于求D=0情况的方程的根。

150号语句是转移语句，由于已经计算完了一组变量(A，B，C）使方程产生的根，所以将转移到程序的最后，停止程序执行。

160号语句至190号语句用于计算当D>0时，二个不等实根的根。

200号语句至250号语句用于计算当 D<0时，一对共轭复数的根。

260号语句（见100号语句解释）

270号语句用于计算当变量A，B，C，中A=0，B≠0时，仅存在的一个实根 X=-C/B

300号语句：程序到此结束。

上面例子是我们见到的语句最多的例题，也是比较完整的例题。如果你有条件，可以拿到计算机上试一试，也可以把你认为很好的语句，而程序中并没用到的语句，大胆换上去试试看。也许，你可以把这么长的程序读懂的。

4. FOR…TO…STEP…循环语句：

英文中“FOR…TO…STEP”的意思是：从哪里开始到哪里终止，步长是多少。也许大家对“STEP”步长这个词不太了解，步长的意思我们可以理解为“我们走路时，每走一步的长度。”

那么，对循环语句，我们就可以理解成：从A地点出发，到B地点终止（即B地点是目的地），每走一步的距离不大不小应当是C。若用循环语句来描述这个问题就应写成：“FOR A TO B STEP C”。所以我们可以把A叫做起始值或初值，B叫做目的值或终值，C叫做每步的长度或步长值。

循环语句的格式：

〔行号〕FOR〔循环变量＝初值〕TO〔终值〕STEP〔步长〕

⋮

〔行号〕NEXT〔循环变量〕

说明：NEXT的英文意思是“下一个”初值必须是个表达式，如：I＝0。表示由零开始，其中I叫做“循环变量”。终值、步长可以是常数，如：3、2＊A＋B，A＊B，100等。

循环语句的用途：

正向前面举的例子一样，如做一个游戏，让你闭着眼向

前走100步去拿事先放在那儿的一件东西，那你必须记住从开始的地方到放东西的地方是100步远，也就是初值和终值。那么，假定你每一步迈出的距离都一样远，即步长为1。现在你就可以闭眼向前走了，但每走一步都要记数字。否则，你就可能走不到或走过了头。所以，你若是采用累加的办法或渐减的办法，即就应每走一步在你原来的那个数上加1或减1，直到累加到100或减到零。否则，你就要错过拾东西的机会。我们也正是在这种情况下，要使用循环语句。比如这个游戏就可以用循环语句表示如下：

```
10    FOR X=1  TO  100  STEP  1
20    PRINT"A";  (表示你向前走了一步)
30    NEXT  X
```

如果这段程序执行完毕，你将在显示器上看到100个A。我们把FOR…TO…STEP…与NEXT之间的一个或几个语句，叫做循环体。

构成一个循环，必须具备下面三个条件：

(1) FOR语句。如上面例子的10号语句：

```
10    FOR  X=1  TO  100  STEP  1
```

它是循环说明语句。它的一般形式可以写为：

〔行号〕FOR X=A TO B STEP C

其中，X叫做循环控制变量或叫做循环变量。也可以是你自己任意指定的名字。如I，J，K等。由它的值来决定循环是否继续进行下去。

A叫做循环变量的初值。如X=1中的"1"。

B叫做循环变量的终值。如100

C叫做循环变量的步长。

FOR…TO…STEP语句的作用是：说明循环变量从初

值（A)开始变化，到终值(B) 结束。循环变量每次变化的增量即步长为C。因此，初值、终值、步长都是控制循环次数的必要数据。没有这几个数据，循环就不能够进行。

FOR…TO…STEP语句中的初值A，终值B，步长C可以是常数、变量或表达式。A、B的值可以为正、负和零。C不能为零，否则循环将无止境。当步长C等于1时，可以省略。

(2) NEXT 语句。

如上例中30号语句：

```
30    NEXT X
```

它表示这一语句为终点，NEXT语句以下的语句就不属于循环之内了。所以看一个循环的范围只要找到FOR…TO…STEP…和相当的NEXT语句即可。

NEXT X的意思是：到执行此语句时，循环变量的值就要变化，取下一个X′值，即在原来的X值的基础上增加一个步长值C。

(3) 循环体

循环体是指在FOR…TO…STEP语句与NEXT语句之间的全部语句，可以是一个或者一组语句。

如：

```
10  FOR  X=1  TO  100  STEP 1
20  PRINT"A";
30  NEXT  X
```

或 10 FOR X=1 TO 100（由于步长为1，可以省略）

其中20号语句PRINT"A"；就是循环体（注意FOR语句和NEXT语句不包括在内）循环体就是你所要求多次循

环执行的那部分语句。

具体的循环语句功能，可用框图表示见图4—3—5。

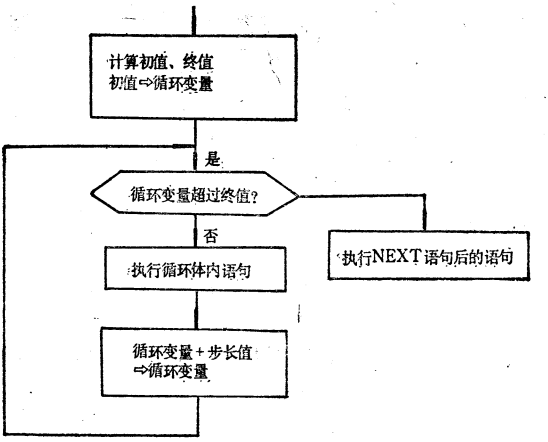


图4—3—5 循环语句功能框图

对照框图说明循环语句是怎样执行的。

(1) 在FOR〔循环变量=初值〕TO〔终值〕STEP〔步长值〕语句中，计算初值、终值和步长值，并记下其各自的值，然后把初值（计算出的）赋给循环变量。其中初值、终值和步长值都可以写成表达式的形式，初值赋给循环变量：就象我们已经学习过的LET赋值语句的功能一样。

(2) 进行循环变量是否超过终值的判断。若循环变量的值“超过”（步长值为正时，是指“大于”；步长值为负时是指“小于”）终值时，则转到循环终端语句 NEXT 的下一句去执行；如果循环变量的值不超过终值，则执行循环体内的全部语句。

(3) 当执行完循环体内语句后，循环变量的值加上一个步长值，并赋给循环变量此新值。再转去执行(2)。

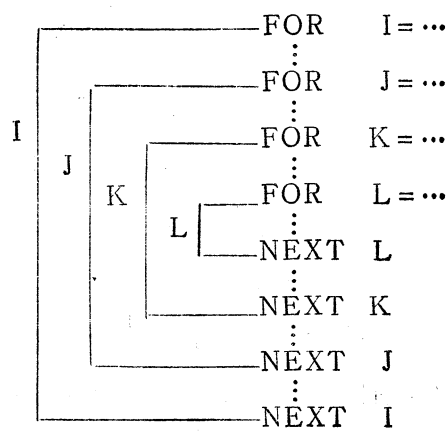
循环语句的使用规则：

(1) FOR语句和NEXT语句必须成对出现，缺一不可。两者使用的循环变量的变量名必须一致。

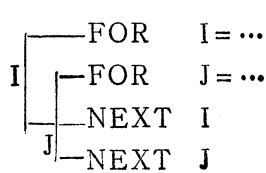
(2) 循环可以套循环。即循环语句的循环体内可以使用循环语句，但内外循环层次要清楚，循环变量的名字不能错乱，不能使循环语句之间交叉，循环嵌套的层数可以是多层的。

(3) 不得从循环体外转到循环体内。也就是说不允许在未先执行FOR语句时，使用GOTO或 GOSUB 转移语句直接转进循环体内。反之，允许从循环体内转到循环体外。

现在，让我们用图表示一下：

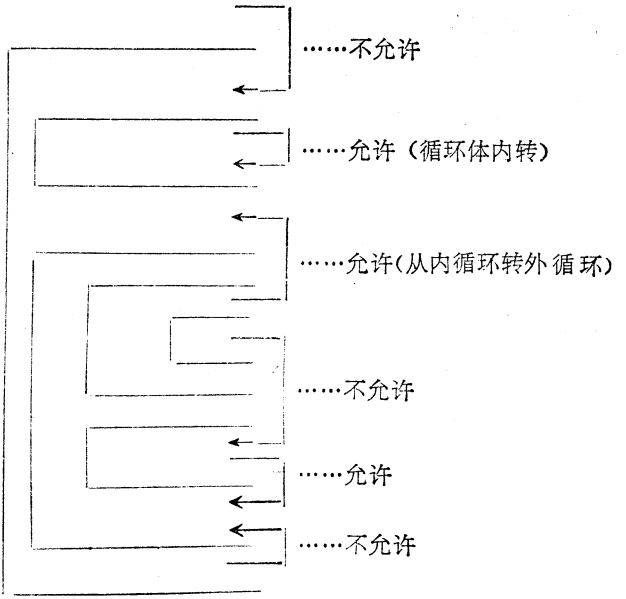


这表示的是正确嵌套循环语句。



这表示的是错误的嵌套循环语句。

下图每个框子表示一个循环语句。



至此我们已了解了FOR循环语句的功能及使用规则了，下面就说明一下它的具体应用。

例1 计算 $1+\frac{1}{2}+\frac{1}{3}+\frac{1}{4}+\cdots+\frac{1}{1000}$ 的和。

这道题我们在讲条件语句（IF语句）时说过。现在，我们试用循环语句运算一次，比较一下两者的区别。

解 按循环语句作如下说明：

程序：
```
10  LET  S=0
20  FOR  N=1   TO  1000  STEP  1
30  LET  S=S+1/N
40  NEXT  N
```

```
50  PRINT  S
60  STOP
RUN  ↙
```

如果：7.4854709

说明：

语句　　功能

10　置变量S的值为零。

20　循环变量N将取从1到1000的正整数值（包括1及1000在内）。现在N 是1，即N＝1

30　由N＝1开始，S现在的值是零（S＝0），由S 的旧值加上1/N 得到S 的新值，于是S 的第一个新值是$0+\frac{1}{1}$即1。

40　NEXT N是对于N 的下一个值，它是2（这里我们使用的计算变量是借助了循环变量，利用循环变量的特点来完成计算的），由于2＜1000，所以计算机返回到20号语句处，而现在新S是旧S 加1/2。因此，S现在是$0+\frac{1}{1}+\frac{1}{2}$，再用N 的下一个值来计算S 的下一个值，即 $0+\frac{1}{1}+\frac{1}{2}+\frac{1}{3}$，如此继续循环，直到N＝1000。然后执行NEXT N下面的语句（50号）

对于50号语句，它相当于 把N＝1000的S的最后值打印显示出来。

在计算

$$1+\frac{1}{2}+\frac{1}{3}+\frac{1}{4}+\cdots+\frac{1}{1000}$$

总和的过程中，我们对 N 每次增加的步长值 是1，用 FOR－NEXT语句步长值不是1时，也可以进行计算。让我们计算：

$$\frac{1}{2}+\frac{1}{4}+\frac{1}{6}+\frac{1}{8}+\cdots+\frac{1}{2000}$$

```
程序：10  LET  S=0
      20  FOR  N=2  TO  2000  STEP  2
      30  LET  S=S+1/N
      40  NEXT  N
      50  PRINT  S
      60  STOP
      RUN↙
```

显示结果：3.74273545

这个结果只是例 1 运算的一半，你看为什么？

提示：$\frac{1}{2}\times\left(1+\frac{1}{2}+\frac{1}{3}+\frac{1}{4}+\cdots+\frac{1}{1000}\right)$

$$=\frac{1}{2}+\frac{1}{4}+\frac{1}{6}+\cdots+\frac{1}{2000}$$

例2. 求在x轴上且在x＝0和x＝1间，$f(x)=\frac{1}{1+x^2}$下的近似面积（见图4—6）。这个面积能由10个底长为 0.1 的矩形面积的和近似表示：

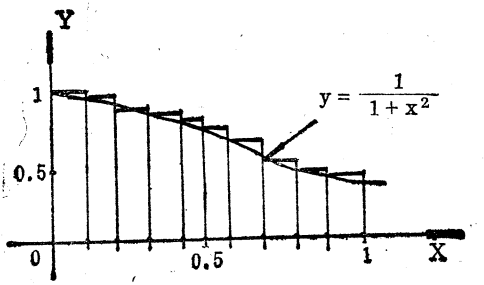


图4—3—6

近似面积 $S=0.1\times f(0)+0.1\times f(0.1)+0.1\times f(0.2)+\cdots+0.1\times f(0.9)$

程序：
```
10  LET  S=0
20  FOR  X=0  TO  0.9  STEP  0.1
30  LET  F=1/(1+X*X)
40  LET  S=S+0.1*F
50  NEXT  X
60  PRINT  S
70  STOP
RUN↙
```

显示结果：0.75473288

计算时，我们把面积分为10个底长为0.1的矩形面积的近似值。

现在，我们又把图形分成100个底边长0.01的矩形来计算它的近似值

只要把上面程序中FOR语句改为

```
⋮
20  FOR  X=0  T  0.99  STEP  0.01
⋮
40  LET  S=S+0.01×F
⋮
```

就可以了。这时得出S的值为0.787894。

请考虑一下这是为什么？哪一种方法更为准确。

5. REM 注解语句：

REM是英文REMARK（注解、说明）的缩写。

格式：〔行号〕 REM〔注解内容〕

REM注解语句的用途：

我们可以用REM注解语句来注明关于程序目的或在程序中语句的说明。在一个程序中，特别是在长而复杂的程序中，我们可以插入“REM”语句去适当解释这个程序要做什么，以及解释要去做这个程序的每段的目的是什么。对于我们使用汉字的人来讲，这个REM的内容当然无法写入汉字，但我们可以写入汉语拼音来注明程序的功能及内容。REM语句是注解语句。仅当你需要在程序中那个地方加上一些注释时，才加上这个语句。所以，它是可有可无的。但，使用REM语句时，一定要按照REM语句的格式，不要忘记REM语句同别的BASIC语句一样，也必须要有一个行号。

6. CLS 清理屏幕语句：

CLS是英文CLEAR SCREEN的缩写。clear screen的意思是“清理屏幕”。BASIC语句CLS的意思也即：清理屏幕。

CLS清理屏幕语句的格式：

〔行号〕CLS

注意：CLS清理屏幕的语句格式与前面讲过的一些语句的格式有些不同。由于它是以命令的形式出现在语句中，所以CLS语句中没有选择项或其它项。如

```
100  CLS
105  PRINT "ABC"
110  GOTO  100
RUN  ↙
```

当你在计算机上执行这段程序时，会看到CLS的用途和作用。

CLS（清理屏幕）语句的用途：

使用CLS语句，清理掉屏幕上我们所不期望看到的那一

部分。有些计算机，当所显示的“行”的数目超过了屏目的“行”数时，就会自动地移出屏幕顶部一行，而显示新进入屏幕的一行，从而使整个屏幕自动上移一行。而有些计算机则不提供自动上移屏幕的功能（这并不说明该计算机的性能的好坏，只是考虑不同的需要）。R1型计算机就是属于后一种。这时，就需要我们处理掉屏幕上已显示的内容。处理的方法之一就是使用CLS清理屏幕语句。

如：打印60至100的平方和立方值。

程序：

```
10  LET  N=0
20  FOR  X=60  TO  100
30  LET  N=N+1
40  IF  N=22  THEN  CLS
50  PRINT  X**2,  X**3
60  NEXT  X
70  STOP
```

这里，我们设屏幕的行数是22行。由于我们使用了40 IF N=22 THEN CLS语句，就可清除屏幕一次，从而使60至100之间40个数据的计算结果全能显示一遍。

7. BASIC命令介绍（NEW，LIST，ENTER，BREAK，CONT）：

(1) NEW命令。NEW英文意思是“新”。我们使用计算机编制或键入一个新程序时，就要使用到它。前面介绍RUN运行命令时，说过了命令和语句有什么不同之外，但那只是直观上的，对于计算机来讲，如果接收到了你的一个命令，就会立刻作出反应。而语句则不然，你若仅写了一个程序段中的一条语句，计算机并不马上去执行它所指明的动

作，因为它尚不明白这一句话的前言和后语，只有待你写入了全部完整的程序之后，并告诉它，你的程序已经全部表达完毕。这时，计算机才按你的程序中指明的前后顺序去执行程序中的语句所指明的动作。所以，一般说来，语句与命令是不完全相同的。

NEW的格式：

NEW ↙

我们只要打入 NEW 这三个英文字母并按下回车 ENTER 键，即可。

NEW的用途：

当你开始向计算机打入程序时，或你需要处理第二个、三个…程序时，而前面的程序不希望再留在计算机里，就可以使用NEW命令。NEW命令一旦被计算机接收，它将把内存贮器全部清新，用于运行一个新程序。

(2) LIST命令。LIST是“列表”的意思，BASIC中LIST命令是列出或重列你已打入计算机内的程序的意思。

LIST命令格式：

LIST ↙

或：LIST〔行号〕 ↙

LIST命令的用途：当我们使用LIST时，就会把已写入计算机的程序列表一次。有时，由于已经运行了一次你的程序，则会发现屏幕上只留下了程序的运行结果，而没有了你的程序，这可能你会着急，以为辛苦半天的成果被破坏掉。其实不然，你只要使用 LIST 命令，就可以恢复你的程序列表，显示在屏幕上。你若使用了，LIST〔n〕则可以在屏幕上显示〔n〕指定的语句和这条语句下面的全部语句(n是程序中的行号)

如：循环语句中的例2。

a. 你应把程序打入计算机。

b. 利用RUN运行命令运行它。这时，你就会发现程序在屏幕上消失了。

c. 现在，你打入LIST并按下回车（ENTER）键，程序就会重新显示在屏幕上。

然后，请你再做一遍a，b，但在c中你打入LIST40并按下回车键，看一看结果是什么？

(3) ENTER命令（键）。ENTER是“进入”的意思。但它和别的命令是不一样的，ENTER是一个命令键，它象我们所使用的键盘上任何一个键（如[A]，[9]等）一样，只要你一按下，立刻就会产生反应。

所以，我们又称它为命令键。有些计算机则标明RETURN键，或↙键，但它们的功能是一样的。当写完一个BASIC语句，或者一个BASIC命令，那么，就按下ENTER键，告诉计算机你已经写完了这条语句或这个命令。所以，也可以说ENTER命令键是一条结束语句或一个命令的标志。

除了上述的功能外，ENTER还有着与LIST一样的功能，即列表程序。但这只是在一些计算机上是这样的。以后，我们为了书写的方便用符号“↙”来表示按下ENTER命令键。

(4) BREAK命令（键）。BREAK的意思是“损坏”或“打坏”的意思。BASIC语言中BREAK命令键的意思是损坏一个程序的正常运行。

如：
```
10 PRINT "ABC"
20 CLS
30 GOTO 10
RUN ↙
```

这时，你会看到屏幕上不停地闪着ABC这三个字母，

即使过了好长时间，它也不会停下来。这时你只要按下SHIFT键和按下BREAK键，程序就会停住了。

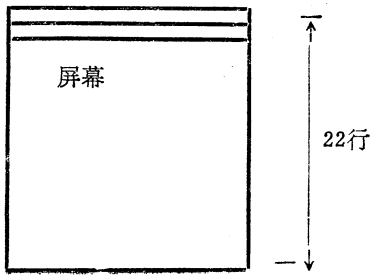
(5) CONT 命令。CONT是英文CONTINUE的缩写。BASIC语言中，使用CONT命令可以“继续”一个程序的执行。这里所说的“继续”是恢复一个由BREAK或不是因出错而停止运行的程序，从而使程序继续运行。CONT是有别于RUN命令的，CONT是由某一地方继续执行程序，而RUN是由程序的起始处开始执行。

下面举例说明一下这几个命令的具体使用方法。

例 计算并打印乘法九九表。

```
      NEW↙
程序: 10 FOR A=1 TO 9 ↙
      20 FOR B=1 TO 9 ↙
      30 LET P=A*B ↙
      40 PRINT A; "*"; B; "="; P↙
      50 NEXT B ↙
      60 NEXT A ↙
      70 STOP ↙
      RUN ↙
```

注：屏幕显示行数是22行，如下图。



当按下↙后，显示结果如下：

```
1*1=1
1*2=2
1*3=3
 ⋮        } 22行
3*4=12
SF IN 40
```

这时，计算机便停下来，并在屏幕底部显示SF IN 40这是提示报告码或称为提示信息(有关提示信息的内容在后面介绍)，它的意思是屏幕已放满数据。计算机显示出SF IN 40便是等待你处理这一问题。这时，你只要从键盘上依次打入CONT并按下ENTER键，则程序又可继续执行。屏幕显示如下：

```
3*5=15
3*6=18
3*7=21   } 22行
 ⋮
SF IN 40
```

此时，又一次出现了屏幕已放满数据的提示信息。现在，你就知道如何使用CONT命令来保证你的程序运行完毕。

现在，你的程序如果还正在运行中，请你按下[SHIFT]键并按下[BREAK]键（若程序已经停止，你应重新打入RUN↙运行程序），这时你会看到程序会停止下来，你可能会发现屏幕上并没有放满数据，也没出现“SF IN 40”这样的语句提示信息。而是有另外一个提示信息出现“BK

IN nn ”。这是表示你已经按下了BREAK键，使正在运行的程序，停止了。此时，你可以打入CONT↙，你又会看到程序会继续执行下去了。当程序运行完毕后，屏幕上仅留有：

```
⋮
⋮
9 * 8 = 72
9 * 9 = 81
ST  IN  70
```

提示信息：ST IN 70说明程序在70号语句遇到了一个“STOP”停止语句。这时，你还可以打入“CONT”命令，并按下↙键。那么，这次你又看到了什么？

好了，现在屏幕上并没有保留你的程序，仅是一些处理结果或提示信息。你若要看到程序，可打入“LIST↙”，立即你就会看到程序了。若仅使用“ENTER”键，其结果也是一样的。

另外，你想写个程序让计算机工作，那就打入“NE-W”，并按下了“ENTER”键。那么，计算机内部也不会保留刚才使用过的那个程序了。也就是说，计算机又开始为下一个程序做好了准备。

三、BASIC 语句使用（一）

在上面一节里，我们又学习了六个BASIC语句和五个BASIC命令。它们是：

语句：(1) GOTO 转移语句

(2) INPUT 输入语句

(3) IF…THEN…条件语句

(4) FOR—NEXT…循环语句

(5) REM　注解语句

(6) CLS　清理屏幕语句

命令：(1) NEW　清内存贮器命令

(2) LIST　列程序清单命令

(3) ENTER　“回车”命令

(4) BREAK　停止程序执行命令

(5) CONT　继续程序执行命令

有关BASIC的语句和BASIC命令，除了已经向大家介绍的之外，还要做一点说明，希望大家在具体的使用中注意。

BASIC的CLS清理屏幕语句，它可以是命令，也可以是语句。

BASIC的RUN命令同样也可以当做语句使用。但是，它们各自的功能不因在语句中或在命令中出现而改变，无论是在语句中还是在命令中其功能都是一样的。注意：它们各自在语句和命令中使用的格式。

语句格式：

〔行号〕RUN

〔行号〕CLS

命令格式：

RUN

CLS

现在，我们利用已经学习过的BASIC语句和命令做几道练习，便于大家掌握它们的使用规则和使用方法。

例1　计算　$1+2+2^2+2^3+\cdots+2^{63}$之和。

解　计算这道题，有两种方法。

$$T=1+2+2^2+2^3+\cdots+2^{63}$$

由于：$2^0=1$，$2^1=2$

所以 $T=2^0+2^1+2^2+2^3+\cdots+2^{63}$

我们设 $t_0=2^0$；$t_1=t_0+2^1$；$t_2=t_1+2^2$；$t_3=t_2+2^3$

$\cdots$ $t_{63}=t_{62}+2^{63}$

$T=t_{63}=t_{62}+2^{63}$

可以看出，只要我们分别由t_0开始求出t_1，t_2，$t_3\cdots t_{63}$就可以得到T，那么问题的关键在于怎么按顺序得到2^0，2^1，2^2，2^3，$\cdots$，2^{63}这些数，如果已经得到这些数，只要按$T=t_{63}=t_{62}+2^{63}=\cdots\cdots$的顺序累加起来便可以得到T=？。

〔方法1〕使用框图（见图4-3-7）说明：

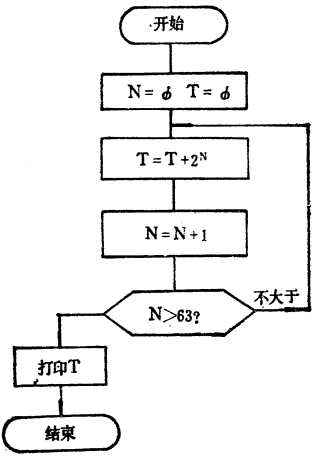


图4—3—7

程序：

```
10  N=0
20  T=0
```

```
30  T=T+2**N
40  N=N+1
50  IF N<=63  THEN GOTO 30
60  PRINT"T=";T
70  STOP
```

说明：对于赋值语句，“LET”可以省略不写。

如 10 LET N=0

可简写成：10 N=0

N>63与N<=63 是判别一个条件的两种关系表达式，为了程序上的方便，我们使用了条件N<=63。如果使用（条件）N>63那么，请你动手改一改这个程序。

第10、20号语句是把两个要在计算中使用的变量清零。这是由于我们需要把每次求得的2^n累加在T当中，那么T这个变量，在我们第一次放入数据之前，即放入2^0之前，应当是零(T=0)。然后，由30号语句计算出第一T的值$T=T+2^0$。这样，原来等号(确切地说是赋值号)右边的T等于0，现在得到$T=2^0+0=1$。40号语句N=N+1，使原来的N值(N=0)加上一个“1”成为N=1。50号语句利用条件“N<=63”，对N进行判别，如果N<=63，则转移到30号语句。否则，N>63就打印出T的值。当转移到30号后，T由旧值(T=1)加上一个2^1，得到一个新值$T=T+2^1=2^0+2^1=1+2=3$。然后，又进行40号语句，N=N+1直到条件“N<=63”第一次被超过，即“N>63”时，便由60号语句打印出T的结果，运行结果T=1.84467E+19

写成习惯表示为：T=18446700000000000000

（对于数的科学表示法，在BASIC语言里我们将使用的数一段中已经做了说明）

〔方法2〕　与〔方法1〕的方法差不多，但是，利用FOR-NEXT循环语句来控制实现条件“N＜＝63”。

程序：

```
10  T=0
20  FOR N=0 TO 63 STEP 1
30  T=T+2**N
40  NEXT  N
50  PRINT"T=";T
60  STOP
RUN↙
```

运行结果　T=1.84467E+19

相传古代印度国王舍罕要褒赏他的聪明能干的宰相达依尔（国际象棋的发明者），问他需要什么，达依尔回答说：“国王只要在国际象棋的棋盘上第一个格子上放一粒麦子，在第二个格子上放二粒，第三个格子上放四粒，以后按此比例在下一格子中加一倍的麦子，一直放到第64格（国际象棋是8×8=64格），我就感恩不尽了，其它什么我都不要了”。国王想：“这有多少，还不容易”，让人扛来一袋小麦，但很快全用没了，再来一袋，又很快用没了，结果当时全印度的小麦全部用完还不够。国王纳闷，怎样也算不清这笔账。我们把它用一个式子表达出来即：

$$T=1+2+2^2+3^3+\cdots+2^{63}$$

我们已经计算出来的T是小麦的颗数。若1立方米小麦约有1.42×10^8颗。设S为体积。

$S=\dfrac{T}{1.42\times10^8}$。我们只要在程序中加一条语句就可以求出T和S。利用〔方法2〕如下：

程序：

```
10  T=0
20  FOR N=0 TO 63 STEP 1
30  T=T+2**N
40  NEXT  N
50  S=T/1.42*10**8
60  PRINT "T=";  T,  "S=";S
70  STOP
RUN↵
```

运行结果：T=1.84467E+19　S=1.29906E+11

即$S\fallingdotseq1.3\times10^{11}m^3$约为一千三百亿立方米小麦。可以在中国的全部土地上铺上1.3cm厚的小麦。相当于中国几百年的产量。

例2　用一元人民币兑换一分、二分、五分的硬币，共有几种不同的兑换法。

解　按兑换方法中五分硬币有0，1，2，…，20种，设每种为A，而每种A还要与每种二分硬币的分法，以及每种一分硬币的分法一一组合。也就是说，首先把一元钱分为每种有0个，1个，2个，…20个五分硬币的21种情况。而这21种分法当中每一种又都要与二分的硬币分法，以及一分的硬币方法相组合，这样就可以计算出共有多少种换法了。现在，也许你觉得虽然能理解如何进行计算的方法，但计算起来并不那么容易。

我们可以利用计算机解决这个问题，而且十分简便。使用FOR-NEXT语句就可以得到每一种组合。让我们象解一道数学题一样，做一些说明。令五分钱硬币的个数为I。I的取值范围为0～20。令二分钱硬币的个数为J。J的取值范围为

0～(100-I×5)/2。一分钱硬币的个数为100－5×I－2×J。如果以M为组合的计数，那么只要将M放在循环体内，每次加1，当循环结束后M的值即为所求的共有多少种换法。

框图（见图4-3-8）如下：

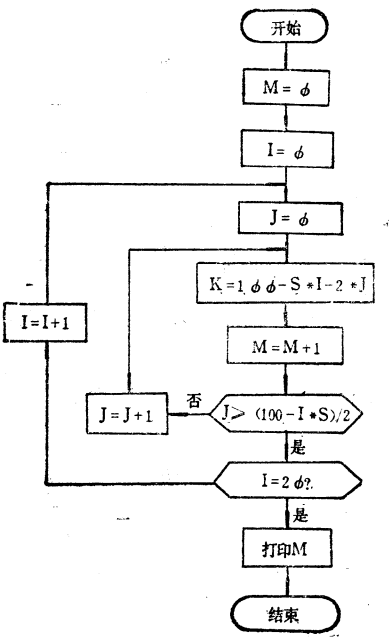


图4—3—8

程序：
```
10  LET  M=0
20  FOR  I=0 TO 20
30  FOR  J=0  TO  (100-5*I)/2
40  LET  k=100-5*I-2*J
50  LET  M=M+1
60  NEXT J
```

```
70  NEXT I
80  PRINT "M=";M
90  STOP
PUN↵
```

结果：M=541

说明：这段程序中共有两个循环。内层J的循环和外层I的循环。I代表五分硬币的个数。J代表二分硬币的个数。我们把一块钱看成是数字100，一、二、五分硬币看成是数字1，2，5。当外层循环变量取零时（第一次I=0），进入内层循环。此时内层循环变量取零（第一次J=0）。于是由I=0，J=0，即五分硬币和二分硬币都取零个及K=100-5*I-2*J，得出K=100，即一分硬币取一百个，第一种换法便产生出来五分为零个，二分为零个，一分为一百个。于是程序执行NEXT J之后J由零变成1(J=1)，此时I=0，于是由I=0，J=1及K=100-5*I-2*J，得出K=90，第二种换法又产生出来五分为零个，二分为一个，一分为九十八个。当又一次执行到NEXT J语句之后，J=2，此时I=0，J=2，由K=100-5*I-2*J 得出K=96。第三种换法又产生出来。类推，可求出I=0的全部五十种换法，此时内层循环变量J已经达到终值(100-5*I)/2。程序自动退出内层循环，执行NEXT I语句之后I=1，于是程序又自动进入内层循环。去求I=1，J=0至(100-5*I)/2的全部四十七种换法。类推，就可以求出全部换法，现在由于题目要求计算出全部换法有多少种，所以只要在内层循环内加一个计数的变量M就可以了。当每求得一种换法，M就加上1即 M=M+1 当程序结束后，打印出M的值，就是有多少种换法的值。现在，若要求你打印出每一种换法的具体组合，那么只要在内

层循环体内再加上一个打印语句就可以了。

如：加上

```
55 PRINT M;"*";I;"--5--";J"--2--";K;"--1--";"□";
```

就可以打印出全部541种换法的组合。我们仅使用了二层循环就把一个较复杂的组合问题解决了。当然，希望你能把这个程序再进一步扩大，一块钱，若兑换成一分、二分、五分及一角、二角、五角的换法有多少种？

例3 输入十个数，打印出其中最大的一个数。

这道例题要求大家先试着读一下程序，看看已经能懂得多少，那些地方还不能理解，然后，看后面的说明部分。

程序：

```
10 INPUT M
20 FOR I=1 TO 9
30 INPUT X
40 IF M>=X THEN GOTO 60
50 LET  M=X
60 NEXT I
70 PRINT  M
80 STOP
```

说明：10号语句INPUT M 是让你由键盘打入第一个数，由于是第一个数，所以，我们暂时认为M是最大的数。当程序进入循环体后，你由键盘输入第二个数X。40号语句利用条件"M>=X"来比较M和X二个数那个大，如果M大于X，或等于X，那么转移到60号语句，使循环变量I加1，返回到20号语句，比较终值9，看I是否超过终值，若没超过终值则执行循环体内语句。否则，执行循环语句后面的语句。30

号语句让你再输入一个数X，若 X大于 M，即“M>=X”条件不成立，执行50号语句把那比较出的那个较大的数放入到M当中，以便输入下一个数时进行比较。直至循环体被执行九次，循环变量超过终值为止，打印出M的内容，便是十个数当中最大的一个数。

例4 使用BASIC程序求方程 $f(x)=x^3+4x^2+x-6$，$x=-10, -9, \cdots, 9, 10$时，$f(x)$的值，并分析 $f(x)$的根的情况。

程序：

```
10  FOR X=-10 TO 10 STEP 1
20  LET F=X**3+4*X**2+X-6
30  PRINT X, F
40  NEXT  X
50  STOP
PUN↵
```

结果显示：

-10	-616
-9	-420
-8	-270
-7	-160
-6	-84
-5	-36
-4	-10
-3	0
-2	0
-1	-4
0	-6
1	0

2	20
3	60
4	126
5	224
6	360
7	540
8	770
9	1056
10	1404

说明：上面在没加分析的情况下，给出了程序和程序运行的结果。希望大家能在仔细思考之后再看说明。也许，你自己的分析结果要比说明更加精确。

这个程序本身并不复杂，但由于它是帮助我们解一元高次方程的方法之一，所以认真地分析一下它，对我们应用计算机解一些数学问题是会有很大启发的。

在数学中，对于给定的自变量，如果函数f(x)值的变化由正变化到负或由负变化正，那么在自变量的取值范围之内，就一定有方程f(x)=0的根存在。即对于x_1有$f(x_1)>0$（或$f(x_1)<0$）；x_2有$f(x_2)<0$（或$f(x_2)>0$），那么在x_1与x_2之间，方程f(x)一定存在根。我们有了这个数学上的概念，再看看程序的输入结果。由结果可以看出，当x=-3，到x=2之间，f(x)三次等于零。即：f(-3)=0；f(-2)=0，f(1)=0。所以，可以分析出方程f(x)=0，在(-10，10)之间至少有三个根。如果我们还要进一步得到根的准确值。更简单有效的方法是把程序中的10号语句改为：

```
10  FOR X=-3  TO 2 STEP 0.5
```

于是，通过运行程序，就又可以得出一些对于f(x)=0

方程根的更精确一些的结果。

结果为：

-3	0
-2.5	0.875
-2	0
-1.5	-1.875
-1	-4
-0.5	-5.625
0	-6
0.5	-4.375
1	0
1.5	7.875
2	20

如果，你仍然对其结果有怀疑，那么还可以改变一下10号语句，就会得到更精确的结果，对于这道题 f(x)=0 这个方程的根是$x_1=-3$，$x_2=-2$，$x_3=1$。

当用计算机去求解一个高次方程，方法之一，就是给定一个自变量的取值范围，然后，选择一个合适的步长值。观察分析一下其结果，再次修改取值范围和步长值，直到求出方程的根。

练 习 题 二

1. 对于下述每一个程序输出的是什么？

(1)
```
10 FOR X=1 TO 5
20 LET Y=X-5
30 NEXT X
40 PRINT Y
50 STOP
```

(2)
```
10 FOR X=1 TO 5
```

```
20  LET  Y=X-5
30  PRINT  Y
40  NEXT  X
50  STOP
```

2. 描述这两个程序的输出

(1)
```
10  FOR  I=1  TO  100  STEP 2
20  PRINT  I
30  NEXT  I
```

(2)
```
10  FOR  I=1  TO  100  STEP 2
20  PRINT  I-1
30  NEXT  I
```

3. 写出并且执行一个能打印（PRINT）出对于N=1,2,3…,20的平方表的程序，输出应包括N和N^2。

4. 设计和执行一个BASIC程序去计算并打印出下述的和。

(1) $\frac{1}{1^2}+\frac{1}{2^2}+\frac{1}{3^2}+\cdots+\frac{1}{n^2}+\cdots+\frac{1}{100^2}$

(2) $\frac{1}{1}-\frac{1}{2}+\frac{1}{3}-\frac{1}{4}+\cdots+\frac{(-1)^{n+1}}{n}+\cdots-\frac{1}{1000}$

5. 令 $f(x)=5x^2-6x-1$ 对于$x=-5,-4,\cdots 5$，计算$f(x)$的值。

(1) 在关于$f(x)$的这些值中，你注意到什么规律？

(2) $f(x)=5x^2-6x-1=0$ 方程的根近似值是多少？

(3) 计算$f(x)$关于$x=-1$到$x=0$每隔0.1的值。

(4) 用相同的办法，计算更小的区间，估计$f(x)=0$的近似根。

6. 使用IF…THEN语句去终止求和的过程，写出并且执行一个BASIC程序去求和：

(1) $T=1+2+3+\cdots+100$

(2) $P=1+2+4+8+\cdots+2^N+\cdots+2^{10}$

7. 仔细思考一下本节例2的解题方法。试说出有什么特点？

8. 在你的计算机上作下列练习，并观察其结果，总结规律及使

用方法。

(1) 10 LET M=5
20 PRINT "M"
30 PRINT M
40 PRINT "M=" ; M

(2) 10 PRINT "A" , "B"
20 PRINT "A" ; "B"
30 PRINT 3+5, "A" ; "B" ; "3+5"

(3) 10 PRINT 1, 2, , 3

(4) 10 PRINT 1; 2; ; 3

提示：逗号与分号的区别？引号的使用方法。

9. 指出下列两个程序的同异。

(1) 10 LET K=1
20 IF K>100 THEN GOTO 60
30 PRINT K
40 LET K=K+1
50 GOTO 20
60 STOP

(2) 10 FOR K=1 TO 100
20 PRINT K
30 NEXT K
40 STOP

四、BASIC 语句（三）

1. READ 读数据语句 及
DATA 提供数据语句：

READ和DATA的英文意思分别为"读"和"数据"。在 BASIC 语言中使用 READ 语句可以把 DATA语句中的数据读到变量中去。

READ 读数据语句格式

〔行号〕READ〔变量1〕，〔变量2〕，…

DATA 提供数据语句格式

〔行号〕DATA〔数据1〕，〔数据2〕，…

READ 读数据语句和 DATA提供数据语句的用途：

用赋值语句或输入语句（LET或INPLT）给变量赋一个值或改变一个值，固然是最直观的方式。但是当初始数据较多时，用它来给所有的变量赋值，将使程序显得冗长，且不灵活。比较紧凑和灵活的形式是使用READ 读数据语句和DATA提供数据语句。它们两者配合使用，便可以把一组数赋给一组变量。例如：

```
10  READ  X, Y, Z
20  PRINT "X=" ; X,  "Y=" ; Y,  "Z=" ; Z
30  DATA  72.3, 74.11, 3
```

计算机执行10号语句时，便把30号语句中的72.3赋给变量X；把74.11赋给变量Y；把3赋给变量Z。执行20号语句时，便把变量X、Y、Z的值打印出来。

READ读语句和DATA提供数据语句不一定要紧接着写。提供数据语句在计算机执行程序时，不进行任何操作，它可以放在程序的任意位置。但是，READ读数据语句的位置不能那么任意，一定要放在将要使用那些变量的语句之前。

例

```
10   READ  X
 ⋮     ⋮
100  READ  Y, Z, W
 ⋮     ⋮
200  READ  I, J, K
 ⋮     ⋮
```

```
400  DATA  5, 4, 3.8, 9
 ⋮      ⋮
500  DATA  5, 4, 3.8
```

从这个例子可以看出，读数据语句READ和提供数据语句之间并不需要有一一对应的关系。程序的执行是这样的：当执行到第一个读数据语句时，首先从程序中找到第一个提供数据语句开始处，按读数据语句中的变量个数，依次往下从提供数据语句中读数据。读完一个数据语句中的数，又顺序找下一个提供数据语句的数据继续读给变量，直到把这个读语句中所列的全部变量赋值完毕为止，并记住从数据语句中读数的当前位置，以便再出现读数据语句时，继续读数据。

例子中读数据语句中的变量与提供数据语句中数据的相互关系见下表所列：

执行语句10之前	X，Y，Z，W，I，J，K的值为零
执行语句10之后	X = 5 Y，Z，W，I，J，K的值与执行语句10之前一样
执行语句100之后	Y = 4，Z = 3.8，W = 9 其余变量与执行语句100之前一样
执行语句200之后	I = 5，J = 4，K = 3.8 其余变量与执行语句200之前一样

在程序中使用READ读数据语句和DATA提供数据语句时，应注意：二者缺一不可。但READ语句与DATA语

句的个数可以不等。

例　计算97、88和95的平均数。

```
10  READ  A, B, C
20  LET  V= (A+B+C) /3
30  PRINT  V
40  DATA  97, 88, 95
50  STOP
RUN↵
```

运行结果：

```
93.33333
```

在执行这个程序的过程中，计算机分配给READ语句中变量的值，是按照DATA语句中数据的次序分配给A=97，B=88，C=95。计算机再执行20号语句计算，V=（A+B+C)/13，然后打印输出V的值。注意在READ和DATA语句中，每个变量或数据之间必须用逗号分开。

2. GOSUB转子程序语句及RETURN返回语句：

GOSUB是英文GO SUB-program的缩写。意思是："到子程序。"RETURN是"返回"的意思

GOSUB转子程序语句和RETURN返回语句的格式：

〔行号〕GOSUB〔子程序开始语句的语句行号〕

⋮

〔行号〕〔子程序开始的第一条语句〕

⋮

〔行号〕RETURN

GOSUB转子程序语句和RETURN返回语句的用途：

在一个程序中，往往会遇到要重复使用某一段程序的情况。BASIC语言中，把这样一段程序编写成相对独立的程序段并称之为子程序，然后加以重复使用，从而使程序简短。包含调用子程序的那个程序，相对于子程序来说，它称为主程序。主程序执行中，需使用子程序时就用转子程序语句（GOSUB 语句）调用子程序；子程序执行完毕，便由返回语句（RETURN 语句）返回到主程序。由返回语句（RETURN）控制的返回，是返回到转子语句(GOSUB)的下一个语句，继续执行主程。由于 RETURN 返回语句中，并没有指定任何语句行号，因此，在程序的任何不同地方执行 GOSUB 转子程序语句后所遇到的 RETURN 返回语句，但程序都能自动返回到相应转出的那个位置上。

例　计算当X＝0.5，1，3.1415926，111 时，

$$Y=9x^2+2x^3-1$$

的值。

程序：

```
10  LET  X=0.5
20  GOSUB  100
30  LET  X=1
40  GOSUB  100
50  LET  X=3.1415926
60  GOSUB  100
70  LET  X=111
80  GOSUB  100
90  STOP
100  LET  Y=9*X**2+2*X**3-1
110  PRINT  "X=";X,"Y=";Y
```

120 RETURN

关于子程序，需注意如下几点：

(1) 主程序只能通过 GOSUB 语句调用子程序，子程序必须使用 RETURN 语句返回主程序。这就是说，不允许直接进入子程序或通过GOTO，IF 等语句进入子程序，也不允通过 GOTO 语句，IF语句直接转出子程序。

(2) 子程序可以有多个入口和出口，但每一个出口都必须使用 RETURN 语句实现返回，这就是说，用几个 RETURN 语句在一个给它的子程序里从不同的地方退出子程序，这是可以的。

(3) 主程序与子程序的关系可用图4—3—9表示。图中表示的是前面例题的转子程序和返回关系。子程序从100号语句到120语句，转子程序语句有四个，分别为第20、40、60、80号语句。

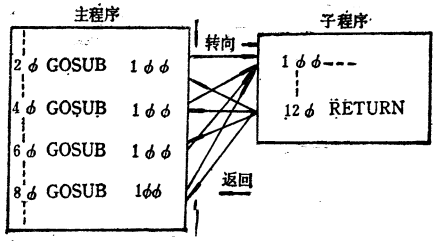


图4—3—9 主程序与子程序的关系图

3. END 终止语句：

英语中 END 的意思为“终止、结束”。BASIC 语言中使用 END 表示一个程序执行的结束。

END 终止语句格式：

〔行号〕 END

END 终止语句的用途：

我们已经学习了 STOP 停止语句，并且在前面一直使它代表一个程序的结束。但 END 终止语句和 STOP 停止语句是不一样的两个语句。END 终止语句一般用于表示程序结束、终止的概念。STOP 停止语句一般用于表示程序被指定停止或停在某个地方的概念。虽然，END 语句和 STOP 语句是不同的两个语句，但在有些地方使用时能达到同样的效果。一般说来，所有使用 END 的地方都可以使用 STOP 语句。但，反之并不是全部都可以的。如果当我们仅是给程序一个结束标志时，使用二者，都无关紧要。请大家注意，我们可以利用 CONT 继续命令恢复由 STOP 语句引起的停止。但 END 标志的结束和终止，则是无法被恢复的。在用于中、小学教学使用的计算机仅提供了 STOP 语句，而没有提供 END 语句。所以，我们把一些例题及习题中标志程序结束的地方使用了 STOP 语句，这是为了方便读者，有关一些类似的使用注意事项在后面专有说明。

例

```
10  PRINT  3.1416*2.71828**2
20  END
```

4. 字符串概念：

在前面已经向大家介绍了有关 BASIC 语言的基本概念。如，“数”、“字符”。现在向大家介绍另一个概念——字符串。

我们把 BASIC 语言中使用的英文字母 A到Z、数字0到9和全部运算符号及标号（逗号，句号，引号等）统称为字符。由这些字符构成的一串符号及字母、数字称为字符串。我们可以利用字符串赋值给一个“变量”，这个变量是

与我们常使用的数字变量不同，它的表示用一个变量名后跟一个美元符$表示。如：A$，HELLO$，CHINA$，A0$54等。

例如 A$ = “ABCDEFG”

的意思就是把 ABCDEFG 赋值给字符串变量A$。

如：

```
10 LET A$ = “ABCDEFG”
20 PRINT A$
30 STOP
RUN↙
```

结果显示： ABCDEFG

我们还可以把多个字符串“相加”这一点类似于数字的相加。但字符串的“相加”实际上是把多个字符串的内相互并接起来，一个排在另一个字符串的后面，所以，确切讲是二个或多个字符串相并。

如：

```
10 PRINT “ABC” + “DEF”
RUN↙
```

结果显示： ABCDEF

又如：

```
10 LET A$ = “ABCDEFG”
20 LET B$ = “HIJKLMN”
30 LET C$ = “BEIJING␣” *
40 LET D$ = “CHINA”
50 PRINT A$, A$ +B$
60 PRINT C$ +D$
```

*C$= “BEIJING␣”中␣代表一个空格。空格可以象别的字母一样使用，空格也是 BASIC 字符中的一个（见 BASIC 字符集）。

RUN↙

结果显示：　ABCDEFG　ABCDEFGHIJKLMN

BEIJING␣CHINA

但是字符串不能相减，乘、除。也不能对字符串乘方。

我们可以测试字符串长度。

如

```
10  A$ = "CHEESE"
20  B=LEN  A$
30  PRINT  B
RUN↙
```

结果显示：6

另外，还可以使用 CHR$这个字符串函数得到全部字符集中的任意一个字符。

如：

```
10  FOR  I=0  TO  255
20  PRINT  CHR$ (I) ; "␣" ;
30  NEXT  I
RUN↙
```

结果显示：〔全部 BASIC 语言字符集内容〕

对于字符串还有一些其它的函数运算，但那些对于我们初学者来说并不是很重要的。有兴趣的读者请查看一下后面的附录。

5. 数组的概念和 DIM 数组说明语句：

数组的概念是比较重要的。在实际生活中，“数组”的使用是很多的。一组有序（有规律）的数被称为数组，其中“序”的表示在数组中称为下标。

排队，这件生活中常见的事情也可以用数组这个概念表示。如：有10个人按先来后到的次序排队，可表示为：

A（1），A（2），…，A（10）。

那么，有些例子就无法用上面的方法表示。如：电影院中座位。当你进入电影院时，先按你票上标明的排号来找你的座号。如：第十排第3号（座位），可表示为：A(10,3)；第七排第7号（座位），可表示为：A（7，7）。

前者，如：A（1），A（2），…，A（10）称为单下标数组或一维数组。后者，如：A（10，3），A（7，7）称为双下标数组或二维数组。其中A称为数组变量名，或简称数组变量。对于数组变量名可以是前面使用的变量名，但由于有些计算机不能使用复杂的变量名，仅能使用简单变量名。所以，大家在使用时应查阅一下该机的手册。

当我们准备使用数组之前，首先要说明所使用的这个数组有多大。计算机好预先把可能要用的地方为你准备好。如：前面排队这个例子，若共有10个人排队，则可说明指定一个数组的大小如下

DIM A(10)

DIM 是英文 DIMENSION 的缩写，意为“尺寸”、“范围”、“大小”。而 BASIC语言中 DIM 语句的意思是指一个数组的大小。

DIM 数组说明语句的格式：

〔行号〕 DIM 〔数组变量名〕(n1，n2)

其中 n1，n2 表示数组的大小，称为下标变量。

对于一维数组可以写成：

〔行号〕DIM 〔数组变量名〕(n)

例 10 DIM A(10)

表示：A数组是一维的，共有10个元素。

n称为下标变量。

对于二维数组可以写成：

〔行号〕 DIM 〔数组变量名〕(n1, n2)

例　10 DIM A (10, 10)

表示：A数组是二维的，共有10×10＝100个元素。类似于一个电影院共有10排，每一排有10个座位。这个电影院总计可容纳100个人。

下面举例说明数组的使用：

例

```
10 DIM B(3, 2)
20 FOR I=1 TO 3
30 FOR J=1 TO 2
40 LET B(I, J)=I*J
50 PRINT "B(";I;",";J;")=";B(I, J),
60 NEXT J
70 NEXT I
80 END
   RUN↙
```

结果显示：

```
B(1, 1)=1        B(1, 2)=2
B(2, 1)=2        B(2, 2)=4
B(3, 1)=3        B(3, 2)=6
```

注意　下标变量I,J都是由1开始的。而不能从零或别的开始。

例　不仅在自然界，而且在数学中经常出现的有名序列是斐波那奇序列：

1,1,2,3,5,8,13,21,……

在这个序列中，头两项之后的每个项是前面两项的和。于

是：

$$1+1=2$$
$$1+2=3$$
$$2+3=5$$
$$3+5=8$$
$$\cdots\cdots$$

要求：计算并打印出斐波那奇序列的前20项。

程序：
```
10  DIM  F(20)
20  LET  F(1)=1
30  LET  F(2)=1
40  FOR  N=3  TO  20
50  LET  F(N)=F(N-1)+F(N-2)
55  NEXT  N
60  FOR  N=1  TO  20
70  PRINT  N, F(N)
80  NEXT  N
90  END
RUN↙
```

结果显示：

1	1	11	89
2	1	12	144
3	2	13	233
4	3	14	377
5	5	15	610
6	8	16	987
7	13	17	1597
8	21	18	2584
9	34	19	4181
10	55	20	6765

说明：

10号语句为20个变量：F(1)，F(2)，F(3)，…，F(20)留出空位。20号、30号语句置F(1)和F(2)的值为1。40号、50号和55号语句用于计算F(3)，F(4)，F(5)，…，F(20)。60号、70号和80号语句要求N从1到20（包括1和20）的序列项的值，打印出N和F(N)。这程序的执行，首先把数放入数组F(N)中，然后在由F(N)依次取出来。

6．BASIC语言中使用的函数介绍：

除BASIC语句外，BASIC语言还包括一组我们能在BASIC程序中使用的函数。这类函数称为特殊函数。另外，BASIC语言还允许使用一些我们熟悉的标准函数。如：SIN，COS，LOG，EXP等。

BASIC语言提供的特殊函数主要有下面四个。

(1) INT(X)取整函数。它用于对变量X的值取不大于X的最大整数。

(2) ABS(X) 绝对值函数。它用于对变量X的值，取其绝对值。即X→|x|。

(3) SQR(X) 平方根函数。它用于对变量X的值，取其平方根。即X→$\sqrt{x}$。此时，要求x≥0 这个函数才有意义。

(4) TAB(N)；“打印内容”，打印表格函数

它用于对屏幕或打字机上的输出格式进行编排。可利用TAB(N)函数组织你自己所要求的输出格式和制作一个函数曲线或其它的图形。

N—表示括号内可以是一个变量、常数或一个表达式。

TAB(N)函数是必须与PRINT打印语句一起使用的。

下面举例说明这些函数的使用。

例1 10 LET T=9

```
20 LET S=9.5
30 LET R=-9.5
40 PRINT ABS(T), INT(T)
45 PRINT SQR(T)
50 PRINT ABS(S), INT(S), SQR(S)
60 PRINT ABS(R), INT(R)
70 PRINT ABS(R)+T
80 PRINT S-INT(S)
90 END
RUN↙
```

结果显示：

```
9              9
3
9.5            9
3.082207
9.5            -10
18.5
0.5
```

例2

```
10 FOR I=1 TO 19
20 PRINT TAB(I);"*";
30 NEXT I
RUN↙
```

结果显示：

```
                    19
                ┌──────┐
* * * * * * * * * * * * * * * * * * *
```

例3

```
10 FOR X=0 TO 42 STEP 2
20 PRINT TAB(X);"A";
```

```
30  NEXT  X
RUN↙
```

结果显示：AAAAAAAAAAAAAAAAAAAAA

共显示22个“A”，每两个A间有一个空格。

BASIC语言中还可以使用一些标准函数，即数学函数。主要有以下几种：

SIN(X)	对应于数学中的	sin(x)正弦
COS(X)		cos(x)余弦
TAN(X)		tg(x)正切
LOG(X)		ln(x)以e为底的自然对数
ASN(X)		arcsin(x)反正弦
ACS(X)		arccos(x)反余弦
ATN(X)		arctg(x)反正切
EXP(X)		e^x e指数函数
RND		随机数函数

以上这些函数的有关规则，我们在数学中已经学习过了，而在BASIC程序中使用时又完全遵照数学中的定义。

例
```
10  PRINT“SIN(30*PI/180)=”;SIN
    (30*PI/180)
20  PRINT“TAN(45*PI/180)=”;TAN
    (45*PI/180)
30  PRINT“ASN(0.5)=”;ASN(0.5)
40  PRINT“EXP(2.5)=”;EXP(2.5)
50  PRINT“LOG(2.7182818)=”;LOG
    (2.7182818)
60  PRINT “RND=”,
```

```
70 FOR I=1 TO 5
80 PRINT RND,
90 NEXT I
100 STOP
RUN↙
```

结果显示：
```
SIN(30*PI/180)=0.5
TAN(45*PI/180)=1
ASN(0.5)=0.52359876
EXP(2.5)=12.182494
LOG(2.7182818)=0.99999999
RND=                    0.51483154
0.61291504              0.96907044
0.68031311              0.023834229
```

我们在使用这些标准函数时，应注意：对于正弦、余弦等三角函数，变量是以弧度表示的而不是角度，若你在计算时要使用角度，需要进行适当转换。如：SIN(30)，其中30表示的是弧度，而使用角度计算时可以换成SIN(30*PI/180)即π/6。另外LOG(x)是指以e为底的自然对数。

在有些计算机的键盘上，有π这个键其标记是PI。有些计算机没有这个键时，如果你要用π进行计算，就应当自己给出π的近似值。如SIN(30*3.14159/180)。

计算机虽然给出了许多标准函数，但仍不是全部的函数。比如：arcctg(x)，lg(x)等。当必须使用这些函数时，可以通过数学转换。比如：$\text{arctg}(x)=\text{arcctg}\left(\frac{1}{x}\right)$。

7. BASIC语言中文件的概念（保存和取出文件的方法）：

我们编制的程序，怎样才能在第一次使用之后把它保存

起来，以备再用。而不是每次使用时都要把程序重新把入。为解决这个问题，计算机都配有外存贮器。如磁带、磁盘和光存贮器等。我们比较常用的是磁带、磁盘。

在BASIC语言中，使用文件时也要给文件起一个名字，称为该文件的文件名。文件名一般由1到8个字母和数字组成，而第一个字符位置上要求是字母，以后的几个可以是字母或数字。比如："ABC"，"K12"，"SOPHIA"，都是正确的文件名。而"9AC"，"PROGARMING"是错误的文件名。

当你一旦为程序起了文件名之后，就应记住它。否则，你乃无法知道那个是你需要使用的文件。文件名是唯一的文件标识符，就象一把钥匙开一把锁一样。也不可以一个以上的文件共用一个文件名。

文件的存和取，一般是通过计算机为你提供的一对保存和取出命令来实行的，下面就介绍这两个命令。

(1) SAVE　保存命令。SAVE的英文意思是"保存"、"保护"和"救"。BASIC语言中使用SAVE是"保存文件"的意思。

SAVE　保存命令格式：

SAVE "文件名"

SAVE保存文件命令一般用于磁带和磁盘的BASIC语言中。但有些计算机有不同的规定，希望在使用之前对照一下该计算机的BASIC语言使用手册。

(2) LOAD　取出命令。LOAD的英文意思是"取"、"拿"。BASIC语言中使用LOAD是"取出文件"的意思。

LOAD取出命令格式：

LOAD "文件名"

LOAD取出文件命令，一般用于磁带和磁盘BASIC语言中。

当使用SAVE和LOAD对文件进行存取时，若你把文件存放在磁带上，还应查一下录音机与计算机的连接是否正确(具体的检查方法及连接见计算机BASIC语言使用手册)。然后，再进行存取。若你是把文件存放在磁盘上，只要按上述方法并打入回车键就可以了。

8．特殊功能语句：

近几年来，生产的微型计算机都添加了一些特殊的功能。而这些添加的（语句）并不是用于计算方面。所以，称之为特殊功能语句。如：音乐语句MUSIC，声响语句SOUND屏幕上移语句SCROLL及画图语句PLOT等。下面就对其中的几个作简单介绍一下。

(1) MUSIC　音乐语句

MUSIC是“音乐”的意思。使用MUSIC音乐语句可以让计算机为你演唱一首歌，当然，歌曲一定得由你用程序的方式书写的。

MUSIC音乐语句的格式：

〔行号〕 MUSIC “BASIC音符”

“BASIC音符”有别于我们平常使用的简谱或五线谱。但是，BASIC音符同样也是一套完整的音符。它有不同节拍、不同八度及半音等规定。不过，尽管BASIC音符的规定比较全面，但计算机本身终究不是一架音质上等的钢琴。所以请你不要过多的指责它的效果。它仅做为一种娱乐，雅俗共享。

BASIC音符中，对不同音符有不同代号。如：

简谱音符	BASIC音符
1	C

2	D
3	E
4	F
5	G
6	A
7	B

这些代号在附录中有更详细说明，可以查阅，这些代表符号是按下列安排的。

X（X可以是C、D、E…、B）代表中音

X<　　代表低八音（度）

X>　　代表高八音（度）

|X|　　代表半音（度）

其它符号只要看附表就会很清楚。

为了引入“拍”的表示法，我们应用了TEMPO（英文拍的意思）语句。格式为：

〔行号〕 TEMPO n

n：为0到255的整数，（0相当256），而1单位约为3.9ms

取n=25　则T=25×3.9ms=0.1秒。接近于一般乐曲的$\frac{1}{4}$音符的时间。经常把n=25取为一般乐曲的初值。n取越大，则代表持续时间越长。有了时间单位，就可以指定一个音符的持续时间了。

d在BASIC音符中，表示一个音的持续时间是几个时间单位那么长。d为二位数字。d=1代表$\frac{1}{16}$全音符。d=4相当常用的四分音符代表的一拍。

MUSIC音乐语句格式中的“BASIC音符”是由“xd”组成。x表示C，D，E，…，B中的任意一个音符。d表示该音乐的持续时间。

例1
```
10 MUSIC "C4C4C>4C<4"
20 MUSIC "C4C>4G4G<4A4A>4G4"
30 MUSIC "C1C1G1G1A1A1G1"
RUN↙
```

对不同的频率（高，中，低），不同的音符（C,G,A），不同的持续时间（4，1）进行比较。

例2
```
10 TEMPO 100
20 MUSIC "C4C4G4G4A4A4G4"
RUN↙
```

另外，可以把MUSIC语句中的“BASIC音符”看成字符串，进行合并。

例3
```
10 MUSIC "C4B2" + "C1D1E1F1G1"
   RUN↙
```

例4
```
10 MUSIC "E2G2C>4C>6B2A2B2C>4G6"
20 MUSIC "E2G2C>2B4F6D2F2A2G4E6 "
30 MUSIC " E2G2E>4E>6D>2C>2B2D>
   4A6"
40 GOSUB 100
50 MUSIC "G3F1E2E4E1E1E2E2|D|2E2G4
   F8"
60 MUSIC "A3G1F2F4F1F1F2 F2 E2 F2 A4
   G8"
70 MUSIC " G3 E1 E2 E4C>1C>1C> 2C>
   2B2C>2D74A6"
```

```
80  GOSUB  100
90  STOP
100  MUSIC"C>2B2A2G3A1G2F2E1D4C10.2"
110  RETURN
RUN↙
```

(2) PLOT画图语句。PLOT 的英语意思是“图”。BASIC 语言中代表进入画图方式（或许你使用的计算机是“SET”）。

PLOT 画图语句格式：

〔行号〕 PLOT X, Y

X代表X轴上的坐标，Y代表Y轴上的坐标。

例1
```
10  PLOT  INT(RND*64), INT(RND*44)
20  GOTO  10
    RUN↙
```

结果显示：〔屏幕上出现许多没规律点〕

按下BREAK 键使程序中止执行。

例2
```
10  FOR N=0 TO 64
20  PLOT N, 22+20*SIN (N/32*PI)
30  NEXT N
RUN↙
```

结果显示（见图4—3—10）：

图4—3—10

(3) SCROLL 屏幕上移语句。SCROLL是“动”、“移动”的意思。BASIC语言中使用SCROLL语句可以把屏幕内容向上移动一行。

SCROLL屏幕上移语句格式：

〔行号〕 SCROLL

SCROLL语句，一般是与PRINT语句联用的。如果你自己设计一个游戏，你会感到SCROLL语句是很重要的。它还可以帮助你解决屏幕放不下显示内容的问题，省去了不断打入CONT命令来保证程序运行的必要。

例
```
10 FOR A=1 TO 9
20 FOR B=1 TO 9
30 LET P=A*B
40 SCROLL
45 PRINT A; "*"; B; "="; P
50 NEXT B
60 NEXT A
70 STOP
RUN↙
```

结果显示：〔可以看到九九表中的每一行都是由底部向顶部移动〕

这时，不出现一个“SF IN 45”的提示信息。也不需要你打入CONT命令，可保证程序运行完毕。

对于特殊功能语句，只介绍上面三个。其它的语句，可以参阅附录。

五、BASIC 语言使用（二）

上面，我们学习了READ读数据语句，DATA提供数

据语句，GOSUB转子程序语句和RETURN 返回语句。以及END 终止语句、MUSIC 音乐语句、PLOT 画图语句及SCROLL屏幕上移语句。还介绍了字符串、数组（DIM 数组说明语句）、函数及文件的概念。这主要希望大家通过学习掌握：READ语句，DATA 语句，GOSUB 语句，RETURN语句和END语句及DIM 语句的使用。字符串的概念和数组的概念及数组、函数的使用也希望大家能掌握。对其它的内容有个了解就行了。在今后进一步的学习中，大家会逐渐掌握。现举例说明：

例1 输入10个数，按大小排序

程序：

```
10 DIM A (10)
20 FOR I=1 TO 10
25 PRINT "A(" ; I; ")=" ,
30 INPUT A (I)
40 PRINT
50 NEXT I
55 PRINT
60 FOR I=1 TO 9
70 FOR J=I+1 TO 10
80 IF A (I) >A (J) THEN GOTO 120
90 T=A (I)
100 A (I) =A (J)
110 A (J) =T
120 NEXT J
140 PRINT A (I) ,
145 NEXT I
```

```
150 PRINT A (10)
160 STOP
RUN↙
```

结果显示：A (1) =　　A (2) =

345↙

453↙

A (3) =　　A (4) =

444↙

9999↙

A (5) =　　A (6) =

1000↙

-99↙

A (7) =　　A (8) =

0.9123↙

9876↙

A (9) =　　A (10) =

555↙

6↙

9999	9876
1000	555
453	444
345	6
0.9123	-99

当你按下RUN及回车（↙）键后，屏幕上出现“A(1)=”，此时，你可打入数字。每打入一个数字应按一次回车键（↙）。

程序说明：

10号语句（DIM）说明了一个名字是A的一维数组的大小是10。20号到50号语句用于输入这个数组 A（I）的每一个元素。即：要进行排序的数据。60号语句到程序结束是对A（I）数组中的每个元素分别取出来比较大小。然后打印出比较结果。

例2 学生成绩统计

以下是一个统计学生成绩的程序。统计：得100分的学生有几个，得90分到99分之间有几个，得80分到89分之间的有几个……。

程序：

```
10  DIM S (10)
20  PRINT "C="
30  INPUT C
40  PRINT
45  CLS
50  PRINT "Q="
60  FOR J=1 TO C
70  INPUT Q
80  PRINT
90  LET R=INT (Q/10)
100 LET S(R)=S(R)+1
110 NEXT J
115 CLS
120 FOR I=0 TO 10
130 IF I=10 THEN GOTO 170
140 PRINT 10*I; "--"; 10*I+9, S(I)
150 NEXT I
```

```
160 STOP
170 PRINT "␣␣␣"; I*10, S(I)
180 GOTO 160
RUN↙
```

结果显示：

```
C=
10↙
Q=
100↙          77↙
88↙           99↙
69↙           83↙
100↙          73↙
35↙           40↙
 0——9         0
10——19        0
20——29        0
30——39        1
40——49        1
50——59        0
60——69        1
70——79        2
80——89        2
90——99        1
10  100       2
```

说明：利用数组S(I)记录各类得分的人数。S(0)记为0到9分之间的学生数，S(1)记10到19之间的学生数，S(2)记20到29之间的学生数，……，S(10)记得100分的学生数。

程序中，10号语句说明S(10)数组S的大小。20号到40号语句用于输入全部学生的总数。50号到80号语句用于输入每个学生的成绩分数。变量Q存放输入的学生成绩。90号

语句是把学生的分数分成0到9，10到19，…，90到99，100的不同级别。如：学生成绩为83，即Q=83。则Q/10=8.3取整后为8。即R=INT（Q/10）=INT（83/10）=8。于是通过下面的100号语句，则可在S(8)这个单元里计上一个数，表示有个成绩在80分到89分之间的学生。如果，成绩Q=78则R=INT(Q/10)=INT(78/10)=7于是便在S(7)单元内计上一个数，表示在70分到79分之间也有学生。当统计完毕全部学生成绩之后，S(0)，S(1)，…，S(10)里分别存在的数就是得分在0到9分，10到19分，…，100分的不同级别学生得分人数。120号到180号语句是把全部S(0)，S(1)，…，S(10)的数打印出来。

例3 打印$-\pi$到π之间的正弦曲线。要求每隔20度打一个点。（$\pi=3.14159$，使用TAB函数）

程序：

```
10  FOR X=-3.14159 TO 3.14159 STEP 3.14159/9
20  PRINT TAB (18+12*SIN(X)); "*"
30  NEXT X
40  STOP
RUN↙
```

结果：见图4—3—11显示

说明：10号语句指定了打印SIN曲线的起始和终止位置（$-\pi$，π）及每隔20度$\left(\frac{2\pi}{18}=\frac{\pi}{9}\right)$打印一个点“*”。所有曲线共由18×20度，即18加1个零点共19点组成。20号语句中“18”和“12”的选取是考虑曲线的最大值（横向）、最小值（横向）和屏幕的行、列数而设定的，与曲线本身无联系。但若选则不合适，也会影响程序的执行。

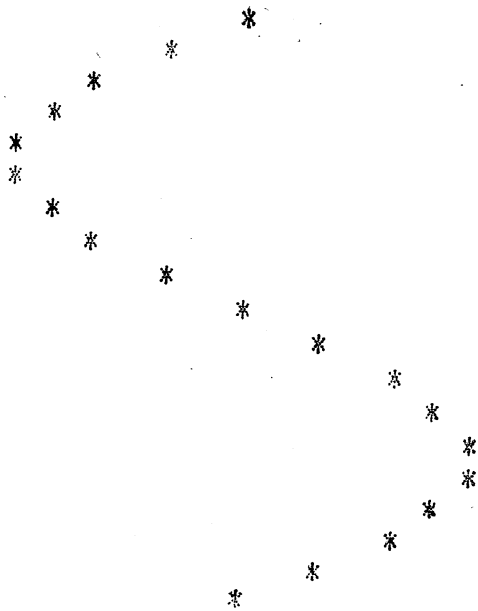


图4—3—11

练 习 题 三

1. 求下述程序的输出：

```
10  FOR X=-4 TO 4
20  IF X<0 THEN GOTO 80
40  PRINT SQR(X)
60  GOTO 100
80  PRINT ABS(X)+INT(X)
100 NEXT X
120 STOP
```

2. 已知一个三角形的顶点坐标为A（2，3）、B（4，7）、

C（-9，5），写出并且运行一个BASIC程序去计算这个三角形的周长。

3．令一个序列的头三项的每一项是1，令所有序列的项均是它前面三项的和，故这个序列的头几项是：

1，1，1，3，5，9，17，…

设计和运行一个BASIC程序打印出这个序列的前30项。

4．飞机在水平匀速飞行时投弹，不计空气对弹体的阻力，打印其弹道曲线（参考例3）。

弹道曲线方程：

$$\begin{cases} x = vt \\ y = \frac{1}{2}gt^2 \end{cases}$$

故：$$x = v\sqrt{\frac{2y}{g}}$$

其中：x——水平距离（米）

y——垂直高度（米）

g——重力加速度，等于9.8米/秒^2

v——投弹时飞机速度（米/秒）

t——时间（秒）

5．判断下列程序执行后将会出现什么结果。

```
    5  DIM A(2)
(1) 10  FOR I=1 TO 2
    20  LET A(I)=I
    40  PRINT A(I)
    50  NEXT I
    60  END
(2) 5   DIM C(20)
    10  FOR I=1 TO 20
    20  LET C(I)=2*I
    30  LET Q=C(I)
```

```
40  PRINT Q, C(I)
50  NEXT I
60  FOR I=1 TO 20
70  PRINT Q, C(I)
80  NEXT I
90  END
```

6. 利用GOSUB转子程序和RETURN返回语句设计一个求：$S=2!+3!+4!$ 的BASIC程序。

（提示：
```
100  LET P=1
110  FOR J=1 TO 4
120  LET P=P*J
130  NEXT J
140  PRINT P
```
结果显示：24

设计子程序时阶乘方法可参考此提示）

第五章　实用程序设计初步

第一节　如何用BASIC语言编程序

利用BASIC语言编制程序的方法有多种多样。现在较为成熟的方法是结构程序法。但对初学计算机的人来讲，无论是对计算机知识的了解，还是所要解决的问题，都是简单的。所以，结构程序法及一些其它的编程方法都不太适合。我们应当选择一种适合于自己特点的方法去编制程序。比如：数学基础较好的人或分析能力较强的人，就不一定要按照同一种方法来设计和编制程序。我们在上一章讲过的例题，“用一块钱人民币兑换成5分、2分、1分硬币，共有多少种换法”。在解法中，我们用分析的方法来分析每一种换法的组合情况，并让计算机为我们组合出每一种换法。最后累计得出一共有多少种换法。而不是把上述问题写成一个数学表达式（按组合的方法写出表达式），再通过计算这个表达式得到全部有多少种换法。数学基础较好的人可选用后一种方法，比较简单。而分析能力较强的人可选用前一种方法，比较直观。

现在，向大家介绍在编制BASIC程序中经常碰到的一些问题及解决的方法，帮助大家得出解决问题的步骤和方法。

例1　已知半径R，高度H，求圆的周长、圆面积，圆柱面积，圆锥体积。

公式：圆周长　　$C=\pi \cdot D=2\pi R$

圆面积　　$S_1=\pi R^2$

圆柱全积面　　$S^2=2\pi R(H+R)$

圆锥体积　　$V=\frac{1}{3}\pi R^2 H$

分析：计算机算题和人进行算题大体是相同的。但计算机的数据的输入、输出和变量的说明是人进行算题时所不要求的。所以，在编制一个程序之前，要同运算的步骤一起考虑。

我们可以按习惯上算题的步骤，画一张框图（见图5—1—1）：

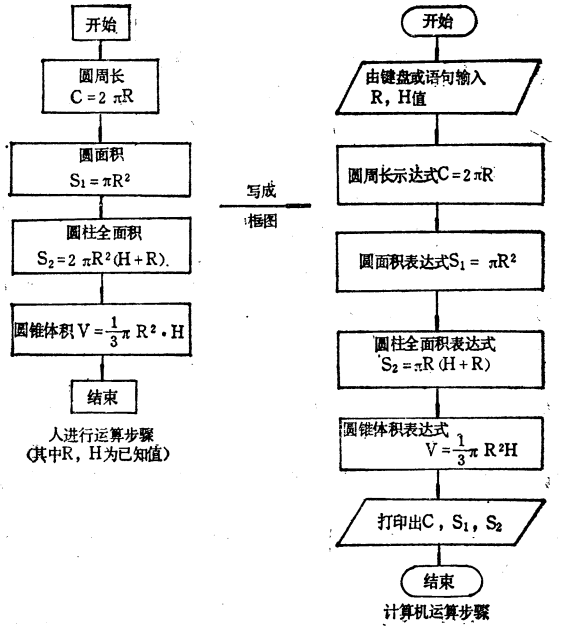


图5—1—1

用BASIC语言描述出框图的内容。

```
1  INPUT R
2  INPUT H
3  LET C=2*3.14159*R
4  LET S1=3.14159*R**2
5  LET S2=2*3.14159*R*(H+R)
6  LET V=1/3*3.14159*R**2*H
7  PRINT C, S1, S2, V
8  END
```

写在左边的行号近似代表着框图中的层次，语句内容就是框图中的内容。于是，就可以得到一个没作任何加工的BASIC程序。考虑到计算机输出格式值观的方便，也可以把程序加工成下面的形式：

```
10  PRINT "R=";
20  INPUT R
30  PRINT
40  PRINT "H=";
50  INPUT H
60  PRINT
70  LET C=2*3.14159*R
80  LET S1=3.14159*R**2
90  LET S2=2*3.14159*R*(H+R)
100 LET V=1/3*3.14159*R**2*H
110 PRINT "C=";C, "S1=";S1, "S2=";
    S2, "V=";V
120 END
```

于是，我们就得到了一个较完整的BASIC程序。

这道题并不复杂，但我们可以看到用计算机解题的特点。一旦编制了这个求解圆周长、圆面积、圆柱全面积及圆锥体积的程序，你就可以千百次地使用它，求出任意给定R、H这些参数而求V、S的值来，而不用再去重新计算及设计程序了。每次使用时仅需输入半径R及高H，即使你不了解程序的编制过程也无关紧要，计算机会把你所要求解的圆周长、面积等数据告诉你。

例2　求一元二次方程式

$$AX^2+BX+C=0$$

的根，其中A，B，C为常数。

分析：求解一元二次方程$AX^2+BX+C=0$的根。如何编制一个BASIC程序，使应能求出对任意给定的A、B、C方程$AX^2+BX+C=0$的根的全部解。从数学上可分析出：

(1) 如$A\neq 0$，$D=B^2-4AC>0$ 则有两个相异实根

$$X_{1,2}=\frac{-B\pm\sqrt{B^2-4AC}}{2A}$$

如果$A\neq 0$，$D<0$ 则有一对共轭复根

$$X_{1,2}=-\frac{B}{2A}\pm\frac{\sqrt{-(B^2-4AC)}}{2A}i$$

如果$A\neq 0$，$D=0$ 则有重根

$$X_1=X_2=-\frac{B}{2A}$$

(2) 如果$A=0$，$B\neq 0$，则只有一个根$x=-\frac{C}{B}$

即：一元一次方程

(3) 如果$A=0$，$B=0$，则方程无意义

根据以上的分析画出框图（见图5—1—2）：

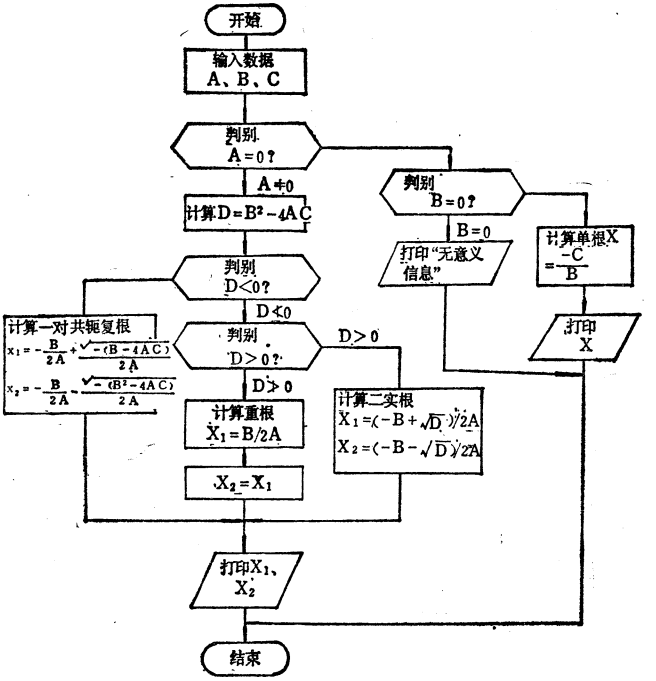


图5—1—2

由框图上可以看出对于A是否为零（$A \overset{?}{=} 0$），和根的判别式（$D = B^2 - 4AC$）是否大于、小于、等于零（$D \overset{?}{>} 0$，$D \overset{?}{<} 0$，$D \overset{?}{=} 0$）是通过计算机判别而完成的。我们对于这种判别一般在框图中表示为◇或〈□〉形。它的实现是通过BASIC语句IF…THEN…。这也是计算机解题的特点，能够完成人交给它的逻辑判断的任务。

在框图中，只给出了$D \overset{?}{>} 0$和$D \overset{?}{<} 0$的判别，而没给出$D \overset{?}{=} 0$的判别。这是由于当D不小于0（D≮0）及D不大

于0（D≯0）时，D就不大、不小而一定等于0（D=0）。所以，只要计算重根$X_1=X_2$就可以了。如果出现A=0而B=0的情况，则认为方程无意义，否则，可以求出方程的全部根。

解一个方程时，首先给出方程的系数A，B，C。然后，进行判别分析及计算根。如给出了一组数，A=1，B=2，C=3。第一求判别A是否等于0。当A=1时，计算出$D=B^2-4AC=4-12=-8$。由于D=-8<0，所以是共轭复根；

$$X_1=\frac{-B}{2A}+\frac{\sqrt{-(B^2-4AC)}}{2A}i,\qquad X_2=\frac{-B}{2A}-\frac{\sqrt{-(B^2-4AC)}}{2A}i$$

，然而打印出X_1及X_2的根。

例3 输入计算机十个数，要求打印出值最大的一个数。

分析：对于输入的数，可以采用逐个比较的方法，第一个与第二个数比较，挑出一个较大的再与第三个数比较……（见图5—1—3）

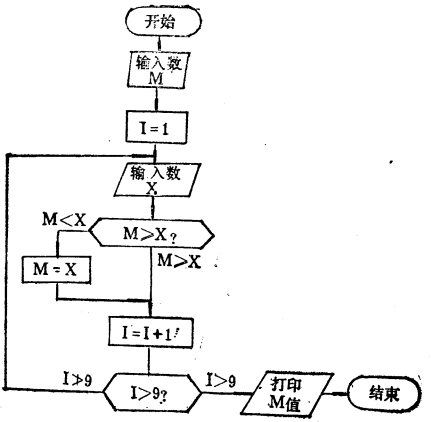


图5—1—3

设：M变量中放最大的数。

X变量中放输入待比较的数。

I变量为计数变量（器）I=I+1　意为I的旧值加1等于I的新值。

先将第一个输入的数送到M中，再输入第二个数，存放在X中。将X与M相比较，如M小，则将X送到M中。接着输入第三个数，它再与M相比较，大者送M。一直到输入完10个数为止。比较完九个数后（由于第一次输了一个M，它与后面输入的九个数相比），打印出M值。则可得到最大的数。I为计算器，程序每次通过I=I+1语句，I变量都将自动加1，计数一次，直到满足条件为止。

程序：〔见第四章第二节BASIC语句使用（二）例3.〕

通过这三道例题的分析及程序编制过程，可以看出在用BASIC语言设计程序时，一般有以下几个步骤：

(1) 分析待解决问题的数学式及其求解的方法。确定一个较满意的方法，做为BASIC程序的实现方法。

(2) 根据解题方法，确定解题步骤。

(3) 画出一个能表达这个方法的实现及实现步骤的框图。

(4) 检查框图所表达的意思是否与你采用的解决问题的方法一致，若有问题，修正框图直到能把方法及实现步骤表达完整。

(5) 用BASIC语句把框图的内容及关系表达出来。即：根据框图写出BASIC程序。

(6) 检查BASIC程序是否能完全表达框图的意思。否则，应修改BASIC程序。

(7)加工BASIC程序，使它更完美些。特别是有关输

入、输出的语句。

(8)运行该BASIC程序。

实际上，BASIC 程序设计的方法就是把一种习惯上的解题方法用 BASIC 语言描述出来。只要我们选择的解题方法是可行的，正确的。那么，用 BASIC 语言编制的程序也一定是可行的（原则上讲，不包括语法错误）。

下面介绍一下有关画框图（见图5—4）时，所采用的符号：

(1) “起止”框：两端为圆弧的框，用它表示过程的“开始”或“结束”（见图5—1—4）。

(2) “处理”框：矩形框，用它表示某一种处理或运算。它有一个入口和一个出口（见图5—1—5）。

(3) “判断”框：菱形框，用来检查和判断某些待定判断的问题，一般它有一个入口和两个出口。表示在两种选择

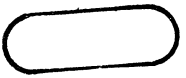


图5—1—4

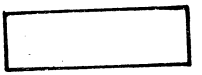
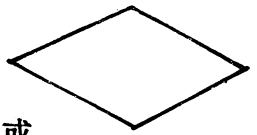


图5—1—5



或

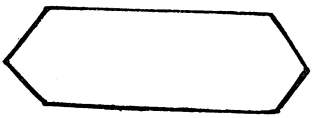


图5—1—6

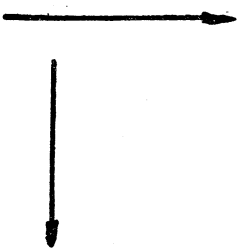
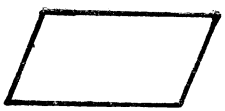


图5—1—7



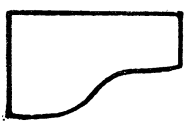


图5—1—8　　图5—1—9

（“是”与“否”）中可选择的一种的出口。根据判断结果选择其中某一个出口路径（见图5—1—6）。

（4）“流程”线：带箭头的方向线，表示框图的路径和方向，是各种框之间的连线（见图5—1—7）。

（5）“输入/输出”框：表示从计算机输出信息（如打印结果）或将数据输入给计算机（见图5—1—8）。

（6）“连接”点：圆圈作为“连接”点。它表示到本框图外的某个地方，或表示由框图外的某个地方进入。一般用于本页框图画不下时，可通过“连接”点将二个框图连接起来。（见图5—1—9）。

以上给出了绘制一张框图所常用的几种符号，希望大家习惯使用它们画框图。

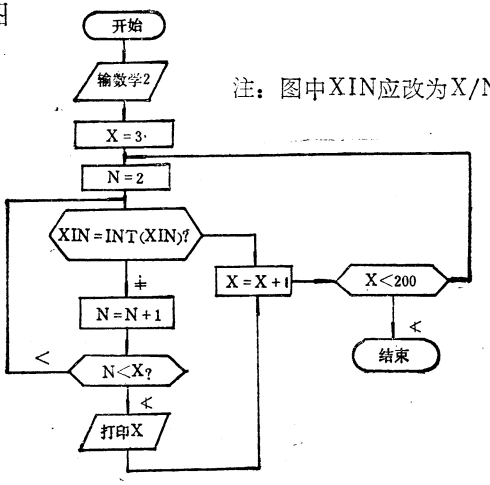
例4　求所有小于200的质数。

质数，只有它自己和1这两个因数的正整数。

分析：假如任何正整数X能被N整除($N\neq1$，$N<X$)，那么X就不是质数。这个思想指引我们造一个框图，它能概括一个过程，这个过程检验每一个整数X是不是有任何

异于1和它自己的因数，这里2＜X＜200，对于N＝2，3，4和5你自己可以检验这个框图(见图5—1—10)和程序。

框图



注：图中XIN应改为X/N

图5—1—10

程序：

```
10  PRINT 2; "␣";
20  LET X=3
30  LET N=2
40  IF X/N=INT (X/N) THEN GOTO 100
50  LET N=N+1
60  IF N<X THEN GOTO 40
70  PRINT X; "␣";
100 LET X=X+1
110 IF X<200 THEN GOTO 30
120 STOP
RUN↵
```

结果显示：

2 3 5 7 11 13 19 23 29 31 37 41 43 47 53 59 61
67 71 73 79 83 89 97 101 103 107 109 113 127 131
137 139 149 151 157 163 167 173 179 181 191 193
197 199

首先，打印出最小的质数2，对于每个大于2的整数X，用全部整数N（2≤N＜X）去除来验证。假如整数X能被一个较小的整数N整除（见40号语句），那么，X就能用一个较大的整数来代替（见100号语句）；并且假如这个X的新值是小于200的，就用以2开始的比它小的整数去除来检验。假如一个整数X不能被较小的整数N整除（见40号语句），那么N就用它下一个较大的整数来代替（见50号语句）；并且假如N的这个新值是小于X的（见60号语句），再作检验，看看X能否被它整除，假如对于N＝2，3，4，…，X－1，其结果是X都不能被N整除，那么X就是一个质数，并且打印出X（见70号语句）。这个被打印出的X的值又被下一个较大的整数来代替（见100号语句）。并且，如果这个X的新值是小于200（见110号语句），我们就开始检验它是不是一个质数。当X的值达到了200时（见110号和120号语句），那么这个程序就结束。

小结：在编制BASIC程序中应当注意的问题。

(1) 对于待解决的问题，要进行全面的考虑，从最坏、最原始的情况入手提出解决每一种可能发生的情况，应当采用的解决办法。也许，有些情况在我们习惯的解法中不予考虑，有些情况是可以避免的，因为我们多多少少已经习惯了默认一些东西的存在前提。比如在没有什么说明的情况

下，对于 $X=\frac{-B}{2A}+\frac{\sqrt{B^2-4AC}}{2A}$ 中的A，我们不会把A＝0代入去计算，而一般默认A≠0。但计算机则不然，你必需对类似的情况加以说明和考虑，要不就会出现错误。

（2）在保证正确设计程序的前提下，要尽量方便程序的使用者。

（3）尽量使程序简单、易读。

对初学者来说，过高的要求是没有必要的。诸如，程序的静态、动态所占据的内存空间、速度等，可暂时不加考虑。否则，你会感到编制程序是很复杂的事情，从而对计算机望而生畏。希望大家能多在机上练习，这是最好的学习方法。你不用担心计算机会疲劳或由于使用而被损坏。

第二节　实用和趣味程序选编

在这节里，我们选编了一些实用和趣味程序，目的是为了让大家能够多掌握一些实验程序和开阔大家的眼界和思路，从而使自己尽快地学会并掌握BASIC语言。如果大家在学习中能够多做例题及独立完成部分习题，再上机做几道程序实例。那么，你就可以独立的解决一些问题了，并且可以自学一些计算机方面的专业书籍了。

1．统计全班成绩总分和平均分数程序。

程序：

```
10  LET  N=0
20  LET  T=0
30  PRINT  "X="
```

```
40  INPUT  X
50  PRINT
60  IF  X=-1  THEN  GOTO  100
70  LET  T=T+X
80  LET  N=N+1
90  GOTO  30
100  PRINT
110  LET  M=T/N
120  PRINT  "N=";  N
130  PRINT  "TOTAL=";  T
140  PRINT  "MEAN=";  M
150  STOP
RUN↙
```

结果显示：

```
X=
99↙
100↙
88↙
67↙
87↙
-1↙
N=5
TOTAL=441
MEAN=88.2
```

程序使用方法：

当显示“X=”时，是询问X的值，即是每个学生的分数。使用者可依次打入每个人的分数，如上“99↙”并按下

回车键↙。当你把全部分数输入完毕后，打入－1（在X＝－1状况下）并按下回车↙键作为输入结束的标志。然后，计算机便把全部分数（TOTAL＝）打印出来，平均(MEAN＝)分数打印出来。N 是代表你一共输入了多少个学习的成绩。

2．求3到100之间的所有素数

程序：

```
10  FOR  M=3  TO  100
20  FOR  I=2  TO  M-1
30  IF  INT(M/I)=M/I  THEN  GOTO  60
40  NEXT  I
50  LPRINT  M,
60  NEXT  M
70  STOP
RUN↙
```

结果显示：

```
3                    5
7                    11
13                   17
19                   23
29                   31
37                   41
43                   47
53                   59
61                   67
71                   73
79                   83
89                   97
```

程序说明：以上程序使用了多种求素数的方法之一。可以修改程序中的语句，求到更多的素数。对于，更多、更大的素数，这种方法不是有效的。

3．求200以内，满足$a^2+b^2=c^2$的正整数，即勾股数。

程序：

```
10  FOR  A=3  TO  199
20  FOR  B=A+1  TO  200  STEP  2
30  LET  X=A*A+B*B
40  IF  (SQR(X)=INT(SQR(X))
    THEN  GOTO  70
50  NEXT  B
60  GOTO  100
70  LET  C=INT(SQR(X)+5E-10)
80  IF  C>200 THEN GONO 100
90  PRINT ")" ;A; "," ; B; "," ;
    C; ")",
100  NEXT  A
RUN↙
```

结果显示：

```
(3, 4, 5, )          (5, 12, 13)
(7, 24, 25)          (8, 15, 17)
(9, 12, 15)          (11, 60, 61)
(12, 35, 37)         (13, 84, 85)
(15, 20, 25)         (16, 63, 65)
(17, 144, 145)       (19, 180, 181)
(20, 21, 29)         (21, 28, 35)
(24, 45, 51)         (25, 60, 65)
```

(27, 36, 45)	(28, 45, 53)
(33, 44, 55)	(35, 84, 91)
(36, 77, 85)	(39, 52, 65)
(40, 75, 85)	(44, 117, 125)
(45, 60, 75)	(48, 55, 73)
(49, 168, 175)	(51, 68, 85)
(52, 165, 173)	(55, 132, 143)
(56, 105, 119)	(57, 76, 95)
(60, 63, 87)	(63, 84, 105)
(65, 72, 97)	(69, 92, 115)
(72, 135, 153)	(75, 100, 125)
(81, 108, 135)	(84, 135, 159)
(85, 132, 157)	(87, 116, 145)
(88, 105, 137)	(93, 124, 155)
(95, 168, 193)	(99, 132, 165)
(100, 105, 145)	(104, 153, 185)
(105, 140, 175)	(111, 148, 185)
(117, 156, 195)	(119, 120, 169)

4. 世界部分城市时间转换程序。

程序：
```
5    DIM  X$(6)
10   PRINT "BEIJING="
20   INPUT  A
110  BEIJING=A
115  PRINT  "BEIJING"; BEIJING
120  LONDON=BEIJING-8
121  IF  LONDON>0  THEN  GOTO  125
122  LONDON=LONDON+24
```

```
123  X$ = "AFTER"
124  GOTO 128
125  X$ = "BEFOR"
128  PRINT "LONDON=" ; LONDON, X$
130  PARIS=BEIJING-7
131  IF PARIS>0 THEN GOTO 136
132  PARIS=PARIS+24
133  X$ = "AFTER"
134  GOTO 138
136  X$ = "BEFOR"
138  PRINT "PARIS=" ; PARIS, X$
150  BERLIN=BEIJING-7
151  IF BERLIN>0 THEN GOTO 156
152  BERLIN=BERLIN+24
153  X$ = "AFTER"
154  GOTO 158
156  X$ = "BEFOR"
158  PRINT "BERLIN=" ; BERLIN, X$
160  TK=BEIJING+1
161  IF TK<=24 THEN GOTO 166
162  TK=TK-24
166  X$ = "AFTER"
168  PRINT "TOKYO=" ; TK, X$
170  MOSCOW=BEIJING-5
171  IF MOSCOW>0 THEN GOTO 176
172  MOSCOW=MOSCOW+24
173  X$ = "AFTER"
```

```
174  GOTO  178
176  X$ = "BEFOR"
178  PRINT "MOSCOW=" ; MOSCOW, X$
180  NY = BEIJING - 13
181  IF  NY>0  THEN  GOTO  186
182  NY = NY + 24
183  X$ = "AFTER"
184  GOTO  188
186  X$ = "BEFOR"
188  PRINT "NEW  YORK=" ; NY, X$
RUN↙
BEIJING=
23↙
BEIJING = 23
LONDON = 15          BEFOR
PARIS = 16           BEFOR
BERLIN = 16          BEFOR
TOKYO = 24           AFTER
MOSCOW = 18          BEFOR
NEW YORK = 10        BEFOR
```

程序使用说明：按下RUN及回车键↙之后，显示“BEIJING=”你可输入北京时间（以小时为单位），然后按下回车键↙，程序自动显示六个国家首都的时间。

其中：BEIJING——北京（中国）

LONDON——伦敦（英国）

PARIS——巴黎（法国）

BERLIN——柏林（德国）

TOKYO——东京（日本）

MOSCOW——莫斯科（苏联）

NEW YORK——纽约（美国）

AFTER——晚于北京时间

BEFORE——早于北京时间

5. “家庭作业”游戏程序：

这是两个由计算机为你出（作业）题，并为你判成绩的程序。第一个程序是较简单的两位数加法，第二个程序是简单的四则运算程序。每次由计算机给你出10道题，每道题10分。

你还可以根据你的计算能力，对程序进行修改，使它出题难易程度适合你的需要。

程序(1)（加法程序）

```
1    LET  E=0
5    LET  R=0
10   LET  K=100
15   LET  N=0
30   IF  N=10  THEN  GOTO  500
40   LET  A=INT(100*RND)
50  LET  B=INT(100*RND)
60   PRINT  A; “+”; B; “=”;
70   INPUT  C
80   IF  C=A+B  THEN  GOTO  400
85   LET  E=E+1
90   PRINT  C
100  K=K-10
105  PRINT “……␣E␣…”; E
110  N=N+1
```

```
120  GOTO  30
400  LET  R=R+1
405  PRINT  C
410  PRINT "……␣R␣…" ; R
420  LET  N=N+1
430  GOTO  30
500  SCROLL
505  PRINT "……〈K〉……" ; K
510  STOP
```

程序(2)（四则运算程序）

```
10   LET  K=100
20   FOR  I=1  TO  10
30   LET  A=INT(10*RND)
40   LET  B=INT(100*RND)
50   LET  C=INT(10*RND)
60   LET  D=INT(10*RND)
70   LET  E=INT(100*RND)
80   PRINT  A; "*" ; B; "+" ; C; "*" ;
     D; "-" ; E; "=" ;
90   INPUT  F
100  IF  F=A*B+C*D-E  THEN  GOTO
     200
110  PRINT  F
120  PRINT "IT IS WRONG"
130  LET  K=K-10
140  GOTO  250
200  PRINT  F
```

```
250  NEXT  I
260  PRINT "……〈K〉……" ; K
270  STOP
```

程序说明：

当你按下RUN及回车键↙后，显示出一道算术题，你再用键盘把答案打入，并按下回车键，当你做完10道题后，程序自动把成绩告诉你。"……〈K〉…… "100，这个100就是成绩。

其中，"IT IS WRONG" 表示"不正确"

"……R……" 10　数10表示正确的题数。

"……E……" 10　数10表示不正确的题数。

6。马克思做过的数学题：

伟大的导师马克思曾经解答了下面的数学题（马克思数学手稿）：

有30个人，其中有男人，女人和小孩，在一家小饭馆里共花了50先令，每个男人花3个先令，每个女人花2个先令，每个小孩花1个先令，问男人，女人，小孩各是多少？

分析：可得出　令X、Y、Z，分别表示：男人、女人、小孩，则有：

$$\begin{cases} x+y+z=30 & \cdots\cdots① \\ 3x+2y+z=50 & \cdots\cdots② \end{cases}$$

由②－①　得：$2x+y=20$……③

只要满足　X＞0，Y＞0的解，都可以满足方程。这是个不定解的方程。利用上面的分析可写出如下程序。

程序：

```
10  FOR  X=0  TO  10
20  FOR  Y=0  TO  20
```

```
30 LET Z=30-X-Y
40 IF 50=3*X+2*Y+Z THEN GOTO 60
50 GOTO 70
60 PRINT "X=";X;"Y=";Y;
   "Z=";Z
70 NEXT Y
80 NEXT X
90 STOP
   RUN↙
```

结果显示：

X=0	Y=20	Z=10
X=1	Y=18	Z=11
X=2	Y=16	Z=12
X=3	Y=14	Z=13
X=4	Y=12	Z=14
X=5	Y=10	Z=15
X=6	Y=8	Z=16
X=7	Y=6	Z=17
X=8	Y=4	Z=18
X=9	Y=2	Z=19
X=10	Y=0	Z=20

程序说明：如前，X、Y、Z，分别表示男人、女人和小孩。那么可以有11种不同的人数的组合。

7．付钱游戏：

取一分，二分，五分的硬币共十枚，要付出一角八分钱的款，问有几种不同的付法？

分析：设一分硬币为X枚，二分硬币为Y 枚，五分硬币

为Z枚。则有：

$$\begin{cases} x+y+z=10 \\ x+2y+5z=18 \end{cases}$$

同上题类似，可以写出程序。

程序：

```
10  FOR  X=0  TO  10
20  FOR  Y=0  TO   8
30  LET  Z=10-X-Y
40  IF  X+2*Y+5*Z=18  THEN  GOTO  60
50  GOTO  80
60  LET  I=I+1
70  PRINT  "*␣";I;"␣*1$␣";X;"␣
      Z$␣";Y;"␣5$␣"Z
80  NEXT  Y
90  NEXT  X
100  STOP
RUN↙
```

结果显示：

```
*  1  *1$  2  2$  8  5$  0
*  2  *1$  5  2$  4  5$  1
*  3  *1$  8  2$  0  5$  2
```

8．打印出从公元1000年到2000年中的全部闰年：

判闰年的规律是：(1) 能被4整除的年为闰年；(2) 但能被100整除的又不是闰年；(3)但能被400整除的又是闰年。只有同时满足这三个条件的一定是闰年。

程序：

```
10  FOR  Y=1000  TO  2000
```

```
20 IF Y< >INT(Y/4)*4 THEN GOTO 70
30 IF Y< >INT(Y/100)*100 THEN GOTO
   80
40 IF Y=INT(Y/400)*400 THEN GOTO 80
70 NEXT Y
75 STOP
80 PRINT Y;"␣";
90 GOTO 70
RUN ↙
```

结果显示：

1004 1008 1012 1016 1020 1024 1028 1032 1036 1040 1044 1048 1052
1056 1060 1064 1068 1072 1076 1080 1084 1088 1092 1096 1104 1108
1112 1116 1120 1124 1128 1132 1136 1140 1144 1148 1152 1156 1160
1164 1168 1172 1176 1180 1184 1188 1192 1196 1200 1204 1208 1212
1216 1220 1224 1228 1232 1236 1240 1244 1248 1252 1256 1260 1264
1268 1272 1276 1280 1284 1288 1292 1296 1304 1308 1312 1316 1320
1324 1328 1332 1336 1340 1344 1348 1352 1356 1360 1364 1368 1372
1376 1380 1384 1388 1392 1396 1404 1408 1412 1416 1420 1424 1428
1432 1436 1440 1444 1448 1452 1456 1460 1464 1468 1472 1476 1480
1484 1488 1492 1496 1504 1508 1512 1516 1520 1524 1528 1532 1536
1540 1544 1548 1552 1556 1560 1564 1568 1572 1576 1580 1584 1588
1592 1596 1600 1604 1608 1612 1616 1620 1624 1628 1632 1636 1640
1644 1648 1652 1656 1660 1664 1668 1672 1676 1680 1684 1688 1692
1696 1704 1708 1712 1716 1720 1724 1728 1732 1736 1740 1744 1748
1752 1756 1760 1764 1768 1772 1776 1780 1784 1788 1792 1796 1804
1808 1812 1816 1820 1824 1828 1832 1836 1840 1844 1848 1852 1856
1860 1864 1868 1872 1876 1880 1884 1888 1892 1896 1904 1908 1912
1916 1920 1924 1928 1932 1936 1940 1944 1948 1952 1956 1960 1964
1968 1972 1976 1980 1984 1988 1992 1996 2000

9．汉字“你好”程序：

这是利用PLOT（或TRS-80 SET）语句及二次曲线制作的汉字“你好”。

```
10  FOR  x=0  TO  7
20  PLOT  x, 8/5*(x-1)+32
30  NEXT  X
70  FOR  x=13  TO  17
80  PLOT  x, 8/5*(x-7)+26
90  NEXT  x
40  FOR  x=3  TO  36
50  PLOT  4, X
60  NEXT  X
100  FOR  x=14  TO  26
110  PLOT  x, 38
120  NEXT  x
130  FOR  x=23  TO  26
140  PLOT  x, 8/5*(x-14)+19
150  NEXT  x
160  FOR  x=3  TO  38
170  PLOT  19, x
180  NEXT  x
190  FOR  x=14  TO  19
200  PLOT  x, 8/5*(X-14)+22
210  NEXT  x
220
310  FOR  x=19  TO  25
320  PLOT  x, -5/8*(x-14)+31
330  NEXT  x
```

```
340 FOR x=31 TO 40
350 PLOT x, 8/5*(x-15)
360 NEXT x
370 FOR x=32 TO 40
380 PLOT x, -13.7/8*(x-30)+26
390 NEXT x
400 FOR x=30 TO 42
410 PLOT x, 35
420 NEXT x
430 FOR x=27 TO 42
440 PLOT x, 8/5*(x-22)
450 NEXT x
460 FOR x=45 TO 57
470 PLOT x, 33
480 NEXT x
490 FOR x=52 TO 58
500 PLOT x, 8/5*(x-38)
510 NEXT x
520 FOR x=45 TO 57
530 PLOT x, 22
540 NEXT x
550 FOR x=3 TO 22
560 PLOT 50, x
570 NEXT x
RUN↙
```

结果显示

你 好

说明：你也许看出“你好”的出现，每笔每划并不是按习惯笔顺的要求。因为这是留给你的一道有趣的习题，请你修改该程序使“你好”两字的写法按习惯笔顺。提示：你只要交换一下一些语句序号（行号）就可以了，而不需修改其内容。

现在，为大家选择了几个智力游戏的例子。这部分程序刊登在“R1微机使用及编程50例”（吉林高教局出版，1984.7月）

1．万花筒图案

```
10  LET  A=PI/12
20  FOR  E=10 TO  1  STEP  -1
30  FOR  B=0  TO  23  STEP  1
40  LET  X=11+E*COS(B*A)
50  LET  Y=15+E*SIN(B*A)
60  PRINT  AT  X, Y; "⋈"
70  NEXT  B
75  SCROLL
80  NEXT  E
90  GOTO  20
```

注：若将其中语句75的行号改成65，则变成“彩蝶纷飞”。

2．跟踪

下面是一个游戏，计算机显示一个数目，你将按下这个数，开始你需要一秒钟完成，但如果打错，则会要更长的时间才能显示第二个数目，如果按对则显示第二个数目的时间就短些，意思是使它进行的尽可能快，然后按下Q看你的得分——越高越好。

```
10  LET  T=50
```

```
 15 REM T=NUMBER OF FRAMES DER
    GO—INITIALLY 50 FOR 1 SECOND
 20 LET A$=CHR $ INT (RND*10+CODE
    "∅")
 35 REM A$ IS A RANDOM DIGIT
 36 SCROLL
 40 PRINT A$
 45 PAUSE T
 60 LET B$=INKEY$
 70 IF B$="Q" THEN GOTO 200
 80 IF A$=B$ THEN GOTO 140
 85 SCROLL
 90 PRINT "NO GOOD"
100 LET T=T*1.1
110 GOTO 20
140 SCROLL
150 PRINT "OK"
160 LET T=T*0.9
170 GOTO 20
200 SCROLL
210 PRINT "YOUR SCORE IS"; INT (500/T)
```

3. 打台球游戏：

```
  5 LET T=0
 25 FOR B=1 TO 13
 30 PRINT AT B, 1; "■"
 35 NEXT B
 40 FOR C=1 TO 16
```

```
45  PRINT  AT  13, C; "■"
50  NEXT  C
55  FOR  D=1  TO  13
60  PRINT  AT  D, 17; "■"
65  NEXT  D
70  PRINT  AT  ∅, ∅; "0□2□4□6□8□
    0□2□4□6"
75  PRINT  AT  1, 1; "1□3□5□7□9□
    1□3□5" (□翻白键)
80  FOR  E=1  TO  10
85  LET  X=INT (12*RND+0.5)
90  LET  Y=INT (12*RND+0.5)
95  IF  X<2  THEN  GOTO  85
100 IF  Y<4  THEN  GOTO  85
105 PRINT  AT  Y, X; "*"
110 PRINT  AT  15, 8; E
115 INPUT  M
120 IF  M=INT((16*Y+21*X)/29+0.5)
    THEN  GOTO  140
125 PRINT  AT  Y, X; "□"
130 NEXT  E
135 GOTO  170
140 PRINT  AT  2, 16 "□" (翻白键)
145 LET  T=T+1
150 PRINT  AT  15, 16; T
155 PRINT  AT  2, 16; "□"
160 PRINT  AT  Y, X; "□"
```

```
165 GOTO 130
170 IF T>=8 THEN PRINT "VERY
    GOOD"
180 IF T<=2 THEN PRINT "BAD"
190 IF T>2 AND T<8 THEN PRINT
    "BETTER"
```

这是一个打台球的游戏，把球向右上方打，经过两次反射，如果都进右上方的穴中，你就得分。每一次击球方向，由键盘输入进去。共有十次击球机会，得分情况计算机会给出评价。

4。踢足球游戏：

```
3  FOR S=1 TO 18
4  PLOT 20+S, 36
5  NEXT S
10 LET T=0
13 LET F=0
15 PRINT AT 13, 15; "⌓"
20 PRINT AT 6, 8; "⋈"
25 PRINT AT 6, 22; "⋈"
30 PRINT AT 20, 8; "⋈"
35 PRINT AT 20, 22; "⋈"
40 FOR K=1 TO 10
45 LET X=8
50 LET Y=19
55 PRINT AT Y, X; "○"
```

```
60  PRINT AT Y, X; "□"
65  LET X=X+1
70  LET Y=Y-1
75  IF X=16 THEN GOSUB 185
80  IF X=22 THEN GOTO 90
85  GOTO 55
90  LET X=X-1
95  PRINT AT Y, X; "○"
100 PRINT AT Y, X; "□"
105 IF X=8 THEN GOTO 115
110 GOTO 90
115 LET X=X+1
120 LET Y=Y+1
125 PRINT AT Y, X; "○"
130 PRINT AT Y, X; "□"
135 IF X=15 THEN GOSUB 185
140 IF X=22 THEN GOTO 150
145 GOTO 115
150 LET X=X-1
155 PRINT AT Y, X; "○"
160 PRINT AT Y, X; "□"
165 IF X=8 THEN GOTO 175
170 GOTO 150
175 NEXT K
180 STOP
185 LET A$=INKEY$
190 IF A$="L" THEN GOTO 200
```

```
195 RETURN
200 LET T=T+1
205 PRINT AT 10, 5; T
210 LET F=F+1
215 PRINT AT 10-F, 15; "○"
220 PRINT AT 10-F, 15; "□"
225 IF F=7 THEN GOTO 235
230 GOTO 210
235 LET F=0
240 GOTO 175
```

简要说明：

这是一个模仿踢足球的游戏。正上方横线表示球门，“○”代表球，假定你是中央的球员，看一看球到脚下时你能否果断起脚。射门按“L。

5. 游戏程序：

```
10 INPUT A$
20 LET X=11
30 LET Y=16
40 PRINT AT X, Y; "□"
50 LET K$=INKEY$
60 IF K$="1" THEN GOTO 100
70 IF K$="0" THEN GOTO 170
80 GOTO 40
100 PRINT AT X, Y; "□"
110 LET K$=INKEY$
120 IF K$="0" THEN GOTO 170
130 PRINT AT X, Y; A$
```

```
140 GOSUB 400
150 PRINT AT X, Y; A$
160 GOTO 100
170 LET K$=INKEY$
180 IF K$="1" THEN GOTO 100
190 PRINT AT X, Y; "□"
200 GOSUB 400
210 PRINT AT X, Y; "○"
220 GOTO 170
400 LET B$="B>5"
410 IF K$="C" THEN GOTO 460
420 IF K$="V" THEN GOTO 490
430 IF K$="B" THEN GOTO 520
440 IF K$="N" THEN GOTO 550
450 RETURN
460 IF X<=1 THEN GOTO 580
470 LET X=X-1
480 RETURN
490 IF X>=20 THEN GOTO 580
500 LET X=X+1
510 RETURN
520 IF Y<=1 THEN GOTO 580
530 LET Y=Y-1
540 RETURN
550 IF Y>=30 THEN GOTO 580
560 LET Y=Y+1
570 RETURN
```

```
580 MUSIC B$
590 RETURN
```

游戏程序操作说明：

该程序可用任何字符（由10号语句输入）在屏幕上画出任何图形。程序运行时首先要输入一个字符A$，然后显示提示符“—”。此时可输入两个命令：“1”表示在屏幕上画图形，“∅”表示清屏幕上的某一个字符，在0,1命令下光点可向上、下、左、右四个方向任意移动。C表示光标向上（↑），V表示光标向下（↓），B表示向左（←），N表示向右（→）。在每个方向上，一但打印位置越过屏幕打印范围，则发出警告声。

6.音乐：中华人民共和国国歌：

```
NEW
10 LET A$ = "C>6E>2G>4G>4A>4.4G
          >4.4"
20 LET B$ = "E>6C>2G>3G>2G>3E>8C
          >8"
30 LET C$ = "G3G2G3G3G2G3C>4.8G 4C>
          12C>4C> 4C>4G4A2B2C>4.4
          C> 4.8E> 4C> 4D> 2E> 2G>
          6G>2G>4.4E>4F>4C>4E>4G
          > 6E> 2D> 8D> 16A> 4.4G >
          4.4D>4.4E>4.4G>2.2E>4.4G>
          4E>4D>2E>2C>8E>16G4G4C
          >4C>4E>4E> 4G> 4G> 4D>
          4D>4A8D>12G4C> 12C> 4E>
          12E>4G>16"
```

```
40  LET  D$ = "G2G2G2G2C>4.4"
50  LET  E$ = A$ + B$ + D$
60  LET  F$ = B$ + D$ + D$ + D$ + "C>16"
70  TEMPO  15
80  MUSIC  A$ + B$
90  MUSIC  C$
100 MUSIC  E$ + F$
110 STOP
RUN
```

注：本书全部程序均在IBM-PC微型机和R1(PC-81)上运行通过，考虑到初学者能接触到的机型，采用了R1机的BASIC语言格式和符号。对于使用其它机型的人来讲，仅需要把程序中你使用的机器上没有的符号换一下，就可以使用了。

附　录

1　PC—81（R1）微型计算机 BASIC 语言简介

PC—81（R1）机 BASIC 语言中，有两种类型语句。一、只能做为程序语句使用的语句；二、既能做为命令使用又能做为程序语句使用的语句。在下面介绍中没有特别注明的语句是第二种类型的语句。

1．运行语句、结束语句、暂停语句：

RUN n n 为行号。启动程序开始运行，

当n＝0（等价于 RUN），程序将从最小的行号开始运行。

GOTO n n 为行号（n≠0），启动程序开始运行。这条命令与 RUN 不同之处在于它不清变量的值，当它做为程序语句使用时就是转语句。

STOP 结束程序运行。注意：本机器不能用“END”做为程序结束标志。

PAUSE n 暂停程序运行，暂停时间由n确定。其中：0≤n≤32767

2．清除语句、解释语句、列表语句：

NEW 清除全部内存。

CLEAR 把内存变量的值清零，字符串的内容清空。

CLS 清除屏幕信息。即执行此命令时，屏幕变为空白。

REM 解释内容 这条语句在程序运行时不被解释执行，它只是说明程序的功能及操作。

LIST n n为行号。将程序显示在屏幕上，当n=0（等价于 LIST）从最小行号开始显示。

3．赋值语句、输出语句、输入语句：

赋值语句格式：变量=表达式

把表达式的值赋给变量。

输出语句格式：

(1) PRINT 表达式

将表达式的值送到显示器上显示。表达式可以是数字、字符串或以上项目的组合，表达式之间可以用“，”或“；”隔开。前者，表达式的值按固定格式显示，后者，则按紧凑格式显示。

(2) PRINT AT m，n；

将光标移到显示器的指定位置（m，n）处显示。其中，0≤m≤21（最大行），0≤n≤31（最大列）。例如，PRINT AT 3，5；“A”即在屏幕的（3，5）处显示字符A。

输入语句格式：

(1) INPUT 变量或字符串变量（只能做为程序语句）

机器等待键盘输入。

例如，10 INPUT A

在执行这条语句时，机器等待键盘输入。

(2) INKEY $；（只能做为程序语句）

选通键盘，得到一个在选通脉冲期间由按键输入的字符（如果不按键，则为空串）。INKEY $ 与 INPUT 是有区别的，它不会等待你输入字符串。

注意：本机器不能使用 READ 和 DATA 语句。

4. 转语句、条件语句、循环语句：

转语句格式

GOTO n　其中，n为要执行语句的行号，当执行此语句时程序转到所指定的行，并继续执行。

条件语句格式

IF 关系式 THEN 语句（只能做为程序语句）

当关系式成立，执行 THEN 后的语句，否则执行 IF 语句的下一条语句。

关系式有以下形式：

表达式<比较算符>表达式

或　关系式<逻辑算符>关系式

其中，比较算符包括：

=（等于）、<>（不等于）、<=（小于等于）、>=（大于等于）、<（小于）、>（大于）逻辑算符包括：

AND（与）、OR（或）、NOT（非）

举例：(1) 10 IF I<=4 AND J≥3 THEN I=I+1

(2) 10 IF I<4 THEN GOTO 30

循环语句格式

FOR 控制变量=初值 TO 终值 STEP 步长值 循环体{要执行的语句

NEXT 控制变量（只能做为程序语句）。把初值赋给控制变量，并判断其是否到达终值：如果到达终值则跳出循环体；否则执行循环中的语句，执行后控制变量增加步长值，再进行判断、执行，直到到达终值为止。

举例：10 FOR I=1 TO 255

```
20  PRINT CHR$(1)
30  NEXT I
```

5. 转子语句、子程序返回语句、数组说明语句：

转子语句格式

GOSUB n（只能做为程序语句）

其中，n为子程序开始的行号，执行此语句，把主程序地址压入栈，并转到子程序。

子程序返回语句格式

RETURN（只能做为程序语句）

子程序结束，返回到主程序。

注意：以上两条语句是成对出现的，不能单独使用。

举例：

```
10  I=1
20  PRINT I          } 主程序
30  GOSUB 100
40  GOTO 20
 ⋮
100 I=I+2            } 子程序
110 RETURN
```

数组说明语句格式

DIM＜数组＞（只能做为程序语句）

当系统解释这条语句时，给所定义的数组分配足够大的内存单元。

＜数组＞的一般形式是：

(1) 数组名（表达式1，表达式2，…）

例：

```
10 DIM A (12)
```

(2) 字符串数组

数组名（表达式1，表达式2）

其中，表达式1是字符串的个数，表达式2是每个字符串的长度。

例：

```
10  DIM A$ (2, 5)
20  A$(1) = "ABCDE"
30  PRINT "A$(1) =" ; A$(1)
40  PRINT "A$(1,5) =" ; A$(1.5)
50  PRINT "A$(1,2TO4) =" ;A$(1,2)
```

6. 其它语句和函数：

SCROLL 将屏幕显示的文件上移一行，最顶行消失。

SOUND m，n 根据m，n所确定的值发出声响。其中，m为频率（0≤m≤256）n为延持时间。

PLOT m，n 输入象元坐标（m，n），则在屏幕的对应位置上显示一光标，其中m为横坐标，0≤m≤63，n为纵坐标。0≤n≤43

UNPLOT m，n 取出象元。即从屏幕的对应位置上消除光标。其中m为横坐标0≤m≤63，n为纵坐标0≤n≤43。

MUSIC f 根据字符串f的记号和TEMPO的选择，可奏出各种乐曲，其中，C、D、E、F、G、A、B 分别代表音符 1、2、3、4、5、6、7。并且，

X（音符之一）——代表中音

X<（音符之一）——代表低音

X>（音符之一）——代表高音

☒（音符之一）——代表半音

• 表示停顿

TEMPO n 决定乐曲的速度。其中，0≤n≤256

例：

```
10  MUSIC "G<2G<2A<4G<4C4B<8"
20  MUSIC "G<2G<2A<4G<4D4C8"
30  MUSIC "G<2G<2G4E4C4B<4A<8"
40  MUSIC "F2F2E4C4D4C8"
```

上面程序中的数字表示节拍

POKE m，n 把机器码写入内存。n代表机器码（十进制），m代表内存地址（十进制）

PEEK m 读出m地址中的内容。其中，m为内存地址（十进制）

LPRINT 表达式用打印机打印表达式的值

LLIST 打印程序清单

PC—81 函数

USR m 其中：m为程序入口地址（十进制），利用此函数可以调用由机器码编写的子程序。

注意：计算结果存于BC寄存器中，在返回时IY寄存器一定有值为4000H。

LEN 求字符串的长度。例如：LEN (ABCD) =4

VAL 求字符串的值。例如：VAL "8-4" =4

CODE 求字符代码。例如：COND "F" =43

INT 取整。例如 INT (56.8) =56

SGN 符号函数。其结果为-1，0，+1，按照自变量是负，零，或正而定。

STR $ 把数字变为字符串。例如：STR $6= "6"

ABS 求绝对值。例如：ABS-3.2=3.2

SIN
COS
TAN
ASN
ACS
ATN
} 三角函数。用弧度进行计算。

LOG 自然对数

EXP 指数函数

SQR 平方根

PI $\pi = 3.14159265\cdots\cdots$。

RND 随机数

2 PC—81（R1）微型计算机报告码

OK 完全成功。 NF 有NEXT语句，没有FOR语句或一般变量与控制变量同名。

UV 未定义变量。 BS 不良下标。

OM 内存溢出。 SF 屏幕已满。

OV 溢出。 RG 只有RETURN语句，没有GOSUB语句。

II 无效INPUT语句。 ST STOP语句被执行。

AG 某些函数自变量无效。

IR 实数在可容范围之外。

IE 无效表达式。

BK 程序由“BREAK”键中断。

NA 没有程序名，SAVE语句不允许用空字串。

MF 音乐字串格式不对。

3 PC—81（R1）微型计算机字符集

码	符 号	十六进制	码	字 符	十六进制
0	□	00	27	•	1B
1	▗	01	28	0	1C
2	▖	02	29	1	1D
3	▄	03	30	2	1E
4	▗	04	31	3	1F
5	▐	05	32	4	20
6	▚	06	33	5	21
7	▟	07	34	6	22
8	▣	08	35	7	23
9	◪	09	36	8	24
10	◪	0A	37	9	25
11	〃	0C	38	A	26
12	✕	0E	39	B	27
13	$	0F	40	C	28
14	⋈	0B	41	D	29
15	⌓	0D	42	E	2A
16	(	10	43	F	2B
17	)	11	44	G	2C
18	>	12	45	H	2D
19	<	13	46	I	2E
20	=	14	47	J	2F
21	+	15	48	K	30
22	−	16	49	L	31
23	*	17	50	M	32
24	/	18	51	N	33
25	;	19	52	O	34
26	,	1A	53	P	35

（续表）

码	字 符	十六进制	码	字 符	十六进制
54	Q	36	82	未用	52
55	R	37	83	未用	53
56	S	38	84	未用	54
57	T	39	85	未用	55
58	U	3A	86	未用	56
59	V	3B	87	未用	57
60	W	3C	88	未用	58
61	X	3D	89	未用	59
62	Y	3E	90	未用	5A
63	Z	3F	91	未用	5B
64	THEN	40	92	未用	5C
65	TO	41	93	未用	5D
66	STEP	42	94	未用	5E
67	RND	43	95	未用	5F
68	INKEY$	44	96	未用	60
69	PI	45	97	未用	61
70	未用	46	98	未用	62
71	未用	47	99	未用	63
72	未用	48	100	未用	64
73	未用	49	101	未用	65
74	未用	4A	102	未用	66
75	未用	4B	103	未用	67
76	未用	4C	104	未用	68
77	未用	4D	105	未用	69
78	未用	4E	106	未用	6A
79	未用	4F	107	未用	6B
80	未用	50	108	未用	6C
81	未用	51	109	未用	6D

（续表）

码	字 符	十六进制	码	字 符	十六进制
110	未用	6E	138		8A
111	未用	6F	139	反〃	8B
112	光标上↑	70	140	反 	8C
113	光标下↓	71	141	反$	8D
114	光标左←	72	142	反 	8E
115	光标右→	73	143	反 	8F
116	GRAPHICS	74	144	反（	90
117	EDIT	75	145	反）	91
118	ENTER	76	146	反＞	92
119	DELETE	77	147	反＜	93
120	LMODE	78	148	反=	94
121	BREAK	79	149	反+	95
122	LINE NO	7A	150	反-	96
123	未用	7B	151	反*	97
124	未用	7C	152	反／	98
125	未用	7D	153	反；	99
126	数 number	7E	154	反，	9A
127	光标 cuvso	7F	155	反．	9B
128		80	156	反0	9C
129		81	157	反1	9D
130		82	158	反2	9E
131		83	159	反3	9F
132		84	160	反4	A0
133		85	161	反5	A1
134		86	162	反6	A2
135		87	163	反7	A3
136	反 	88	164	反8	A4
137		89	165	反9	A5

（续表）

码	字 符	十六进制	码	字 符	十六进制
166	反A	A6	194	LEN	C2
167	反B	A7	195	SIN	C3
168	反C	A8	196	COS	C4
169	反D	A9	197	TAN	C5
170	反E	AA	198	ASN	C6
171	反F	AB	199	ACS	C7
172	反G	AC	200	ATN	C8
173	反H	AD	201	LOG	C9
174	反I	AE	202	EXP	CA
175	反J	AF	203	INT	CB
176	反K	B0	204	SQR	CC
177	反L	B1	205	SGN	CD
178	反M	B2	206	ABS	CE
179	反N	B3	207	PEEK	CF
180	反O	B4	208	USR	D0
181	反P	B5	209	STR$	D1
182	反Q	B6	210	CHR$	D2
183	反R	B7	211	NOT	D3
184	反S	B8	212	AT	D4
185	反T	B9	213	TAB	D5
186	反U	BA	214	**	D6
187	反V	BB	215	OR	D7
188	反W	BC	216	AND	D8
189	反X	BD	217	<=	D9
190	反Y	BE	218	>=	DA
191	反Z	BF	219	<>	DB
192	CODE	C0	220	TEMPO	DC
193	VAL	C1	221	MUSIC	DD

（续表）

码	字 符	十六进制	码	字 符	十六进制
222	SOUND	DE	239	LOAD	EF
223	BEEP	DF	240	LIST	F0
224	NOBEEP	E0	241	LET	F1
225	LPRINT	E1	242	PAUSE	F2
226	LLIST	E2	243	NEXT	F3
227	STOP	E3	244	POKE	F4
228	SLOW	E4	245	PRINT	F5
229	FAST	E5	246	PLOT	F6
230	NEW	E6	247	RUN	F7
231	SCROLL	E7	248	SAVE	F8
232	CONT	E8	249	RAND	F9
233	DIW	E9	250	IF	FA
234	REM	EA	251	CLS	FB
235	FOR	EB	252	UNPLOT	FC
236	GOTO	EC	253	CLEAR	FD
237	GOSUB	ED	254	RETOPN	FE
238	INPUT	EE	255	COPY	FF

4 微型计算机常用名词简介

二　划

二进制　binary number system

这是一种数制，只有两种数字即“0”和“1”，因此，在加法中，满2即向左进1，故称二进制。

十六进制　hexadecimal number system

这是一种特殊的数制，基数为16，逢十六进一，故称十六进制。在表示时，除使用从0到9这十个数字外，还要加上六个字母，其顺序为A、B、C、D、E、F。

三　划

个人计算机　personal computer

一般是指附有键盘和显示器，而能象打字机那样以对话方式使用的计算机，是微型计算机中的一大门类。主要用于学习、娱乐、计算、控制、事务处理、充当终端或数据库等。

小型软盘　mini-floppy disk

小型软盘比标准软盘小，采用$5\frac{1}{4}$英寸的外壳。通常，也称作“5英寸软盘”。

子程序　subroutine

通常将实现某种功能的一些常用的并带有共同性的一段程序独立出来，每当用到它时就转去执行一次。这种能实现特定功能的一段程序称为子程序。

四　划

中央处理器　central processing unit

目前通称为 CPU，由运算器和控制器两部分组成。用于数据的运算和比较、指令的解释和执行以及各个部分的控制等，是一切计算机的心脏部分。

中断　interruption

当某种事件（硬件故障、程序出错、外围设备操作结束等）发生时，为了对此事件进行处理，中止现行程序的运行而引出处理事件的程序(管理程序或操作系统的有关部分)，这个过程称为中断。

中断处理程序　interrupt handling routine

中断处理程序是管理程序（或操作系统）的一个重要组成部分。计算机响应中断后转入管态。首先保护被中断的程序的现场，然后分析中断寄存器的内容，即查明中断原因，决定转向哪一段程序来处理。各种中断都有一段预先编好的程序来处理它，这种程序即称中断处理程序。

中断屏蔽状态　interruption masked status

一个计算机的中断系统，当同时出现许多中断请求时，不可能同时进行处理，就需要将其中某些中断请求屏蔽起来，一般采用在程序状态字里设置屏蔽位的办法来实现。被屏蔽的那些中断，即处于中断屏蔽状态。

比例因子　scale factor

一般因受计算机结构和字长的限制，数值的表示只能在一定的范围内，否则会发生“溢出”。为了保证计算时不致溢出或损失过多的有效位，常把某数据缩小或扩大一定的倍数，即乘以或除以某个数，这个数就叫做比例因子。

引导程序　bootstrap

是控制计算机输入的程序。在引导程序控制下，程序和数据输入到计算机内存储器中。

文件　file

文件由一个或多个逻辑记录组成，它具有一定的逻辑上的独立含义。在计算机中往往把各种程序数据（如编辑程序、编译程序、标准程序……等）以文件的形式来保存。每个文件有自己的名字，叫文件名。每个文件又可以包括若干份文件。

元语言　meta-language

用以描述语言的语言称为元语言，被描述的语言就是对象语言。

专用语言　special—Purpose language

随着计算机的应用领域日益扩大，每个应用领域的问题也不同，反映特定应用领域要求的程序设计语言称为专用程序设计语言。

专用程序　specific routine

用以解决某个特殊的数学、逻辑或数据处理问题的程序。

反码　one's complement

反码也是一种数的表示方法，这种表示法对于正数，仍和原码一样。但对于负数，则把该数中的“0”变为“1”，“1”变为“0”，然后在其符号位放上“1”即得。

分支　branch

分支或称分支结构程序，这种程序至少有一个判定条件及若干个可能的后继程序段。程序执行时，根据判定结果，选择某个相应的后继程序段。

分时系统　time-sharing system

分时系统是操作系统的一种类型，它使一台计算机可以几乎是同时地为许多终端用户服务。实现分时系统的主要手段是把机器时间分为许多小段（时间片），轮流为与机器对话的各用户服务，保证每隔一定时间（称为响应时间，一般为数秒）每一用户都可以分到一段时间片。

分辨率　resolution

又叫分解度。用数表示某一个量时，其精确度决定于数字的位数，位数越多，最低位的数值越小，分辨率就越高。

分配内存　Storage allocation

为具体的程序、子程序和初始数据等安排存储单元或存储区，称为分配内存。

计时器　timer

在计算机系统中，为了测量时间间隔而设置的装置。

计数器　counter

可以存放数据，并可根据操作的需要对它进行定量的增加或减少的装置。它包括正向计数器和反向计数器。它类似寄存器那样的装置。

计算机网络　computer network

计算机网络是用通信线路把多个分布在不同地点的计算机联结起来的一种网络。把计算机联成网络的目的是使用户能共享网络中的所有硬件、软件和数据等资源。

计算机主频　computer main frequency

在中央处理器的控制器中，时钟脉冲发生器产生的脉冲频率称为计算机的主频。它在一定程度上决定了计算机的运算速度。

手编程序　machine code programming

直接用机器指令编写的程序称为手编程序。手编程序有很大缺点，如编写麻烦、易出错、对机器的依赖性大等。

五　划

主　机　main frame

在计算机中，通常把包括运算器、控制器和内存储器三部分的设备合称为主机。

主程序　main program; master program

主程序与子程序的概念是相对的。一般说来，称调用子程序的程序为主程序，主程序不被它的子程序调用。

示踪程序　tracer

示踪程序是检查源程序是否合乎计算意图的程序。

用户程序　user program

由用户自己编写的解决实际问题的程序。

只读存储器　ROM

仅供读出而不能写入的存储器，所以通常用来存储已经最终完成而无需再作修改的程序和数据。

代　码　code

在电子计算机中，送入的信息，如符号、字母数字等都要化成电子计算机能够识别的二进制数码，这种代表信息的二进制数码称为代码。其中表示数目的代码有原码、补码、反码三种。

可扩充语言　extensible language

它是一种功能性强、多用途的通用语言。它的特点是建立一核心语言，它本身具有一种扩充手段。用户可以利用该语言的扩充手段，按照自己的需要，将语言在数据类型、运算及控制等方面进行扩充。

汇编语言　assembly language

汇编语言是符号语言的发展，它是针对一类（甚至几类）计算机抽象出来的一种符号语言，是一种面向机器的语言。

汇集型语言　collective language

汇集型语言是一种功能强、多用途的通用语言。它的特点是把多种语言的功能汇集在同一个语言中。

汇编程序　assembler

汇编程序是把汇编语言写的程序翻译成机器指令序列的一种程序。翻译时，它把宏指令相应的一组机器指令复抄到目标程序中（若有宏指令参数，还要进行适当的替换），并且进行转换符号码、分配内存、代真等工作。

目　录　directory; catalog

文件系统中用来说明文件的名称、属性、物理位置等的那一部分叫做目录。目录本身也往往算成一个文件。如果一个目录本身所指明的那一文件又是一个目录，则，这时对应的目录就是多级目录。

目标程序　object program

目标程序是源程序经过编译程序加工最后得到的程序。目标程序一般可由计算机直接执行。目标程序也有人称为目的程序或结果程序。

目标状态　Problem Status

简称目态，也叫课题状态。机器处于这种状态时，只能执行某一部分指令，而不允许使用管态指令（或称为特权指令）。当机器在执行程序时遇到管态指令，就产生程序性错误中断。

外部中断　external interrupt

对某台中央处理机来说，它的外部非通道式装置所引起的中断称为外部中断。

外寄存器　external register

在多处理机系统中，各中央处理机之间或一台中央处理机和某台非通道式外部设备之间相互交换信息的寄存器称为外寄存器。

外部设备　external equipment

计算机除主机以外的其它设备。通常指外存储器（磁带、磁盘、磁鼓等）、输入输出设备（如打印机、穿孔机、打字机、卡片读出机等）。

打印机　Printer

把字符打印在纸上的计算机输出装置，应用最为普遍。打印机按打印字型可分为活字式或点阵式，按打印方式则可分为行式（一次打印一行）和串行式（一次打印一字）两种。

记忆码　mnemonic code

在程序设计中用以代替操作码的一种记号，目的是为了帮助人们记忆。记忆码可以根据习惯约定一些符号或者采用拼音文字的缩写。用记忆码书写程序、便于阅读、理解和交流。

六　划

字　word

在计算机中，一组数字作为一个整体来处理或运算的，称为一个计算机字，或简称字。字通常分为若干个字节（每个字节一般是8位）。

字长　word length

计算机的每个字所包含的位数称为字长。根据计算机的不同，字长有固定的和可变的两种。固定字长，即其长度不论什么情况都是固定的；可变字长，则在一定范围内，其长度是可变的。

字节 byte

字节是指一小节相邻的二进制数码，通常是8位（也有4位或6位的）作为一个字节。它是构成信息的一个小单位并作为一个整体来参加操作，比字小，是构成字的单位。

字 符 character

字符是指用来组织、控制或表示数据的字母、数字以及计算机能识别的其它符号，例如、纸带或穿孔卡上的输入符号、各种输出设备的控制和输出符号、终端上的键盘字母和符号以及存储在存储区中供处理机存取和识别的符号。通常用一个字节表示一个字符。

字符显示器 character display

它是显示文字、数字和符号的显示器。它常用阴极射线管显示；利用显示设备上附设的键盘，向计算机直接输入信息，并很快将字符显示在荧光屏上。

字符和图形CRT character and graphics

其显示按功能分有文字、数值显示和图形显示。文字、数值（即字符）显示可在屏幕上显示程序表和计算结果等，作用相当于打印机。显示字数少者为32列×16行，多者为80列×25行。图形显示可在屏幕上显示地图、物体和图表。作用相当于X-Y绘图仪。显示象点少者为96×128，多者为200×640。

寻址操作 addressing operation

按照指令给出的地址形式及特征码，找出存放在主存储

器中需要进行处理的信息的真实地址，这一过程称为寻址操作。

共　享　share

指多个计算任务对某种资源的共同享用。由于计算机系统所能提供的资源数量是有限的，为了完成大量的计算任务，通常希望对一些公共使用的资源实行共享，以提高资源的利用率，从而增大系统的吞吐能力。

交叉存取　interleaving access

解决中央处理机与主存储器速度匹配的方法之一。它把主存储器分成若干个（4个、8个或者更多个）独立的存储体，且在这些存储体之间进行交叉的编址，中央处理机对这些存储器先后交叉地进行访问。这种技术叫做交叉存取。

阶　码　exponent

在浮点制中，数的表示分为阶码和尾码两个部分，阶码即其阶数的二进制数码。见“浮点制数的表示法”。

会话语言　conversational language

会话语言即指人—机联系的语言，它是一种交互式的程序设计语言。

全加器　full adder

执行两个二进制数相加，并且加上低位进位的逻辑电路称为全加器。

任　务　task

在操作系统中往往有很多基本上独立的活动在同时进行着，一个独立的活动称之为任务。

死　锁　deadlock

在两个或多个共行过程中，如果每个进程持有某种资源而又都等待着别的进程释放它们现在保持着的资源，否则就

不能向前推进。这种相持的情形即每个进程都占有一定资源而又都不能前进时，就是“死锁”。

有效数字　significant figure

某数所含准确值的位数（从左起第一个非零数字算起），称为有效数字。不能任意将有效数字的末一位或几位舍去，舍去则影响准确性。

伪指令　Pseudo-op；pseudo instruction

是汇编语言中的术语，或称汇编指令，这些指令被包含在汇编语言写成的源程序中，但不转换为目标程序指令。

执行周期　execute cycle

送到CPU的指令，存放在CPU内的指令寄存器中，在弄清其性质后，就要把这一指令表示的动作变为电信号，送到必要的地方。这一过程即为指令执行周期。

光　笔　light pen

是一种把电视屏幕上的亮点变换为电信号输入到计算机内的光电检测器。它可以选择电视屏幕上的文字和符号，确定和修改图形位置的坐标，并把它们变换为输入信号。在使用时，用光笔轻轻一触指定的位置，即可完成输入操作。

自愿中断　voluntary interrupt

这是一种由程序员在编制程序的时候因某种需要而主动安排的（对机器硬件的）中断要求。

自编译语言　self-compiling language

这种语言的编译程序可直接用这种语言本身来写。这种语言的功能及其编译程序是可以象滚雪球一样，一级一级地扩充。它可以对编译程序作修改并且描述其它语言的编译程序。

机器语言　machine language

机器语言即计算机的指令系统。用机器语言编写的程序不需经过翻译就可被计算机直接执行。

机器代码程序　program in machine code

用机器代码（指令）表示的程序，称为机器代码程序，简称代码程序。这种程序可被机器直接执行。

机器校验中断　machine check interrupt

在计算机中设置一些校验线路，当这些校验线路发现机器故障时，便产生中断信号以引起中断，称这些中断为机器校验中断。

地址码　address code

地址码是指令的组成部分，它指明参加操作的数在存储器中的地址或指明操作结果应送到哪个存储单元。指令中包含的地址数目随机器而异，有零地址、一地址、二地址、三地址、四地址之分。目前常用的是一地址及三地址这两种形式。

七　划

补码　two's complement

补码也是一种数的表示法，这种方法对负数的加减运算相当方便。它的定义是：对于正数，其表示法与原码一样；对于负数，其补码可以表示为：

$$〔X〕_{补} = 10 + X = 10 - |X|$$

注意上式中的10是二进制，即十进制的2，所以〔X〕补也称“对2取补”。

系统程序　system program

一般地说，系统程序包括程序设计语言、编译程序、操作系统以及与计算机密切相关的程序，统称为系统程序。

系统程序设计语言　system programming language

这种语言是自编译语言的发展，即不仅能写编译程序，而且还能用来写一般的系统程序（如编译程序、操作系统）。

形式地址　formal address

形式地址只具有地址码的形式，而不直接对应一个实在的存储单元。它主要用来同变址量进行运算而得到操作数地址。对于某些操作，形式地址的含义也可和地址完全无关，例如在直接操作型指令中，形式地址本身就是操作数；而在移位指令中，形式地址则表示移位次数。形式地址在有些场合下也被称为移位量。

状态器　stater

计算机控制器中，记录数据通路执行操作时的特殊情况所使用的特征触发器称为状态器。根据它在机器中的使用，可分通用状态器和专用状态器。

应用程序　utility program

它是专门为解决某个应用领域里的一类或几类实际问题的程序。它对某一类问题有较强的通用性。这是一些采用先进算法、经过考验的比较成熟的程序，供有关用户选用，减少重复性劳动。

条件码　condition code

每条指令执行结束后，根据指令执行的结果，给出一定的标志信息，存放在特定的地方，用以判定一条指令执行后的情况。这个信息代码叫做条件代码，又称结果特征。

位　bit

二进制数据或代码的每一数位称为“位”，是计算机信息的最小单位。bit是英文binarydigit之略语，故位的音译

名为“比特”。

间接寻址 indirect addressing

由指令的地址码对应的存储单元给定的是一个地址A、而地址A指示的主存单元内存放的是参加本次操作的数据，称为间接寻址。若地址A指示的主存储单元内存放的是另一个地址B，而B地址指示的主存单元内存放的是参加本次操作的数据，则称为多重间接寻址。

库 library

库是汇集在一起的一些资料或程序。库主要包括资料库和程序库。资料库中存放一些常用数据、表格、文件等等。程序库中存放一些常用的标准程序或子程序。

诊断程序 diagnostic Program

用来帮助维护人员发现和定位计算机故障的程序。它是由管理程序调用或由计算机操作员使用。

八 划

舍入 rounding；round-off

按照某种规则（例如四舍五入）舍去数的最低一位或几位，称为舍入。这将使量的精确度变差。

定点制数的表示法 fixed-point representation of a number

数的表示法采用固定小数点位置的方法，称为定点制。通常定点机都把数限制在－1和＋1之间，即小数点规定在第一位之前，而把整数位作为符号位，即把数的正负符号分别用“0”、“1”表示。

软 件 software

也称为软设备。它是为了用户方便和充分发挥计算机效

能的各种程序的总称。包括操作系统、各种程序设计语言、编译程序以及检查和诊断程序等。也有人认为，广义地应把用户为利用计算机解决某类问题而编制的程序（称为应用程序）也包括在内。

取数时间　access time

又叫读出时间。它是存储器从接到读出命令到数码寄存器接收到来自存储单元的读出数为止的时间间隔。

录存和装入　Save and load

无论数字磁带，还是录音磁带，在用于微型机时，都是作为程序或数据的外存。把程序记录于磁带上称为录存，把记录于磁带上的程序移到计算机的存储器内称为装入。数据通常不再装入。

服务程序　service routine

是属于为其它程序服务的一类程序，如示踪程序、输入输出程序等。

周期窃用　cycle stealing

周期窃用又称中止。它是停止并延迟执行现行程序，插入传输周期进行数据交换。

码　点　code-point

微指令字段中，每一个二进制编码都表示一定的控制信息，这样的二进制编码称为微指令字的码点。

直接寻址　direct addressing

由指令地址码直接指出操作数存放的单元。

软盘驱动器　floppy disk drive

软盘驱动器简称为“FDD”，是对软盘读写信息的装置。也分为“标准FDD”和“小型FDD”。它由驱动软盘旋转的马达、读写头、控制读写头移动的步进马达以及控制

上述机构的控制装置构成。

话音输出　voice output

利用扬声器发出近似人的声音的计算机输出装置。过去话音信息是记录在磁带和磁鼓等外存上的。目前，一块可以放在手掌上的LSI基板，就能说几十种话。

实时系统　real-time system

“实时”一词是指对于外来信息要以足够快的速度进行处理并在一定时间之内作出反映。往往上一次输入的信息将立即影响到对下次输入信息的处理。具有这种功能的计算机系统称为实时系统。

物理记录与逻辑记录　physical record and logical record

把计算机中的信息按一定的大小或信息存放设备的物理性质（如行式打印机中的一行）来进行划分所得到的信息单位叫做物理记录。物理记录便于在硬件中存储与管理，它本身不具有完整的逻辑上的含义。

如果按信息逻辑上的含义来划分，所得到的单位叫做逻辑记录。

转移操作　branching operation

计算机一般按顺序逐条执行指令。只有在遇到转移指令时，则改变正常顺序，转移到指令指出的地方。转移分无条件转移和条件转移两种。无条件转移由指令指出处所；条件转移则按照运算结果或某种条件，从几个程序方向中选择一个。

规格化　normalization

规格化是在数据处理时，建立一个数的标准规格。在电子计算机中，数的规格化是使尾数的最高位等于1(通过移位

和变阶得到）。

波特 baud

表示信息传输速度的单位。在很多情况下相当于计算机的最小信息单位—bit（位）。但是，在计算机中，一个字是用一个字节（8位）表示，在进行传输时，它前面还要附加开始标志，因此波特数要略为增加。300波特相当于30个字节，即每秒传输30个字。

变址 indexing

变址就是改变地址的意思。机器在执行需要变址的指令时，必须把指令的形式地址与变址寄存器内容相加（有的机器也可以相减），才能获得操作数地址（也称操作数的有效地址）。有的计算机不设专门的变址寄存器，借用通用寄存器来完成变址功能。

变形代码 modified code

微型计算机（主要是家用计算机）所采用的代码虽以ASCII代码为基础，但大都加以若干补充和修正。例如，日本夏浦公司的MZ-80就不使用英文小写字母，而增了若干汉字（日、月、时、分等）和划图用的图示符号等，共达256种，称为JIS变形码。

固 件 firmware

固件也称稳固件，是一种具有软件功能的硬件。随着计算机的发展，出现了把大量软件功能并到硬件中去的问题，即所谓“软件硬化”方法。固件就是这种方法之一。

固定程序 fixed routine

存放在只读存储器中的程序称为固定程序。

固定性故障 constant fault

由于电路元件的变质，电路内部的短路或断线、软件设

计不周到等原因所造成的必然性故障，叫做固定性故障。

九　划

指令　instruction

规定计算机的操作种类、操作数的值或其地址的“命令”称为指令。为解决某一问题的一系列命令就构成程序，所以指令是程序中的一步，可告诉计算机在程序中某一步应如何去做。

指令寄存器　instruction register

指令寄存器是控制器中的一个基本部件，其作用是暂时存放现行指令。

指令系统　instruction repertory

对一台计算机或一个计算机系列，它所能够执行的各种不同类型指令的总合称为该机或该系列机的指令系统。

指令复执　instruction retry

为了克服随机性故障引起的计算机运算错误，将出现故障时的现行指令重复执行，其复执行的次数由硬件控制或由软件安排。这种将指令重复执行的过程称为指令复执。

逆汇编程序　inverse assembler

它是相对于汇编程序而言的，汇编程序是把符号语言写成的程序翻译成机器代码的程序。逆汇编程序则正好相反，是把机器代码指令转换为符号形式输出，即把操作码转换为记忆码、把地址码转换为符号形式地址。以符号形式输出的程序便于程序员阅读。

总线　bus line

地址信号和数据的通道称为总线。典型的微型机除地址总线和数据总线外，还有传送同步信号使各部分工作同步的

控制总线。各种总线都分别由若干条传输线构成，把各块芯片间连接起来。数据总线的条数最为重要，它可以决定CPU的类型。

相对寻址 relative addressing

指令地址码指出操作数据的相对地址，要找出该操作数的真实地址，必须加上地址码指出的相对量。这种寻址法叫相对寻址。

选择通道 selecter channel

它可连接若干台外部设备，当其中一台设备被选中工作时，其它设备则不能使用该通道，直到该选中设备工作结束后，选择通道再行开放，以便程序选择使用任意一台设备。选择通道一般用来连接速度较高的外部设备，如磁带、磁盘等。

挂起 suspend

在多道程序系统中，可同时存放若干道用户程序，由管理程序（或操作系统）根据一定的原则，控制、调度它们交替地运行。失去运行条件的程序暂时被搁置在一旁，称之为“挂起”。暂时不能运行的程序均处于挂起状态。

标准子程序 standard subroutine

通常把应用面广、使用频繁的子程序经过仔细推敲编制成比较精炼的所谓标准子程序。它比一般子程序的要求高，通常满足下列要求：1.精度比较高；2.计算尽可能快；3.程序尽可能短，以减少所占的存储空间。

标准程序 standard program

把经常遇到的计算过程，常用的计算方法，经过反复推敲，编成精度比较高，运算速度比较快而又尽可能短的典型程序，称这些程序为标准程序。

标准软盘　standard floppy disk

标准软盘是指外壳尺寸为8英寸的软盘。通常简称为软盘，也称为“8英寸”软盘。

面向机器语言　machine—oriented language

这种语言是为特定的计算机或一类计算机而设计的程序设计语言。这种语言保留了机器语言的外形，即由操作码和地址码组成指令这个外形，但面向机器的语言是用符号形式而不用机器代码形式。这种语言能让使用者摆脱计算机的一些纯事务性的细节问题（如无需硬记机器指令代码、摆脱了二～十进制的转换问题和分配内存问题等），而专心考虑程序间的内在联系。这类语言的代表是汇编语言。

面向过程语言　procedure—oriented language

这种语言是独立于计算机的。利用这种程序设计语言在计算机上解题，人们不必去了解计算机的内部逻辑，而主要集中精力考虑解题、算法的逻辑和过程的描述，由于这种语言对解题过程的描述采用了比较接近人们习惯的方式，因而易学、易懂。这种语言的代表是BCY、ALGOL、FORTRAN和COBOL。

面向问题语言　problem—oriented language

这种语言是独立于计算机的。利用这种语言，在计算机上解题，人们不仅摆脱了计算机的内部逻辑，而且不必关心问题的解法和计算过程的描述。只要指出问题、输入数据和输出形式，就能得到所需要的结果。这类语言的代表有报表语言和判定语言等。

十　划

递归子程序　recursive subroutine

递归子程序指的是具有递归性质的子程序。所谓递归性，是指在没有转出该子程序以前可以直接或间接再次调用这个子程序本身。

浮点制数的表示法　floating—point representation of a number

在二进制计算机中，表示一个数的方法是把它的整数部分和小数部分分开表示，即将任一数N表示成下式：

$$N = \pm d \times 2^{\pm P}$$

式中，d称为N的尾数；P称为数N的阶。所谓浮点制，即数的阶P是变化的，亦即随着不同的阶，小数点的位置是变动的。

浮动程序　relocatable program

浮动程序是指在一定意义下可以浮动（俗称“搬家”）的程序。其有关指令的地址不是绝对地址（即真地址），而是相对地址。使用时，只需根据需要更改基准地址（也称基准量），程序便可浮动。

通用语言　all—purpose language; general purpose language

由于现在计算机使用面越来越广，各种专用语言也越来越多，因此自然地产生一种要求：设计一种通用语言，它能够具备各种专用语言所要求的主要特征和功能。为满足这种要求。目前从两个途径来实现，一是设计一种汇集型语言，二是研究一种可扩充语言。

通道状态　channel status

它反映出通道或外部设备的工作状态，如通道忙、设备空、出错等。

通道状态字　channel status word

将有关通道状态按约定的次序排列起来，存入主存储器的固定单元，称为通道状态字。

通道程序　channel program

通道程序是控制通道工作的一串按序执行的指令。它一般放在主存储器中，由中央处理器给出开始地址（即通道地址字）和启动它工作，而由通道执行之。

通道指令　channel instruction

使通道作某种操作（如输入或输出）的命令称为通道指令。它除了操作本身所需要的操作码以外，还有该操作需要的其它参数，如输入输出命令的内存开始地址和交换长度等。

原码　true form

原码的表示方法就是数的一般表示方法。亦即原码形式与原来数的形式一样，但数的正负符号分别用“0”和“1”表示。在定点制中，数的绝对值都小于1，所以均把整数的那一位作为符号位。例如对于数x = +0.1011，y = −0.1101，则两数的原码表示法分别为：

$$[x]_{原}=0.1011,\ [y]_{原}=1.1101$$

高级语言和初级语言　high level language and low level language

接近于人所懂的语言为高级语言，接近于计算机所懂的语言则为初级语言。语言系统就是把高级语言变换成初级语言的系统。

容错技术　fault-tolerant technique

容错技术是一种便于在系统内发现错误和纠正错误的技术。它包括编码技术（如冗余编码、纠错编码）、诊断技术、指令复执、程序复执与待命系统以及多机协同操作等。

十 一 划

接口 interface

在计算机各部分（如中央处理器与通道、通道与外部设备控制器、中央处理器与主存储器等）之间，计算机与计算机之间，计算机与通信系统之间的连接设备。它包括许多信息传输线及其逻辑控制电路。

接口逻辑 interface logic

在数字信息处理系统中，连接不同的系统或设备的逻辑网络称为接口逻辑。

接口程序 interface routine

接口程序在系统内向所有处理机提供一个简单的标准接口，它最适用于源语言语句的输入和二进制浮动码结果的输出。

强迫中断 involuntary interrupt

程序在运行过程中，由于外部设备结束、实时信息、故障和程序出错等原因迫使中断运行，待管理程序处理完后再返回来继续运行。这种中断并非程序设计人员事先安排的，而是随机发生的，称之为强迫中断。

脱机操作 off-line operation

凡某一设备的操作并非直接接收系统的中央处理机或中央控制设备控制的就称为脱机操作，也称离机操作。

停机状态 stopped status

计算机各部分停止工作，指令不执行，中断不响应，称为计算机处于停机状态。

逻辑操作 logical operation

完成逻辑功能的操作叫逻辑操作。它包括逻辑乘、逻辑

加和按位加等。

符号语言　symbolic language

是机器指令系统的符号化。它用记忆码来表示操作码，用符号地址表示地址码。

符号程序　symbolic program

即指用符号形式编写的程序。

手编程序时，可以作为过度的形式。先用文字符号代替操作码及地址码，写出符号程序（也称为文字程序），然后再分配内存、代真、将符号程序转换为机器代码程序。符号程序便于修改和检查。

控制器　control unit

翻译指令代码、安排操作次序，并发出适当的命令到计算机各部分线路，以执行指令的部件叫做控制器。它是中央处理器的重要组成部分之一。控制器的控制方式有同步和异步两种。

十　二　划

硬件　hardware

组成计算机的任何机械的、磁性的、电子的装置或部件均称为硬件，也称为硬设备。实际上除信息数据和程序以外，构成一台计算机的各个部件或设备，如运算器、控制器、存储器、输入输出设备等均称为硬件。

“智能”终端　intelligent terminal

“智能”终端是指自身带一个处理器的终端，例带处理器的电传打字机或带处理器的键盘显示装置。

装配程序　linkage editor

这种程序是由代真程序发展起来的。它的主要功能是把

在编译时某些不能确定的成分和在外部设备上的程序段或标准程序装配为一个完整的解题程序。

缓冲 buffer

在操作系统中，缓冲是用来在两种不同速度的设备之间传输信息时平滑传输过程的常用手段，它是用来临时存放输入输出数据一块存储区域。

等待状态 waiting status

表示机器处于不执行指令，但可以响应中断的一种程序状态。

程序性中断 program interrupt

在现行程序执行过程中，发现了程序性质的错误或出现了某些特定状态而产生的中断，称为程序性中断。有时也称“陷井”。

程序 program

为了使计算机实现所预期的目的（如解某一算题或控制某一过程）而编排的一系列步骤称为程序。程序可以用机器指令来编写，也可以用程序设计语言来编写。

程序包 routine package

一般说来，程序包指的是具有一定功能、并满足一定要求的一组程序。

程序库 routine library

把常用的各种标准程序，服务程序，专用程序、通用程序及专门设计的软件放在一起，就组成了一个程序库。有了程序库可以减少大量重复性劳动。

编译系统 compiling system

编译系统就是各种程序设计语言的编译程序的组合。也有人把程序设计语言作为编译系统的组成部分。

编辑程序　editor

编辑程序的功能是把多个模块程序联结成一个程序。它可以增加、删除或替换程序中的某些段落。功能较全的编辑程序，还可以把不同类型语言写出的程序编辑在一起。

编译程序　compiler

编译程序能把输入的源程序翻译成等价的目标程序。

编译语言　compiler language

这是把接近于英语的高级语言书写的程序变换成计算机能够执行的机器语言所用的系统。根据高级语言写法的不同，编译语言分成科学技术运算用和事务处理用等各种类型。

十　三　划

微处理器　micro processor (μp)

由一块或几块芯片组成的CPU，称为微处理器。

微型计算机　micro computer (μC)

由微处理器加上存储器片，I/O接口片等支持片组成微型计算机。

微型计算机系统　micro computer system

微型计算机配上成套的外围设备、系统软件及电源后称微型计算机系统。

源程序和源语言　source program and source language

源程序是用近似数学语言的程序设计语言书写的程序。它是翻译（或解释）程序处理的对象。

编写源程序的语言称为源语言，因此各种程序设计语言都可以作为源语言。

解释程序　interpreter

有的程序不能在一台指定的计算机上直接执行，例如，在定点机上进行浮点运算、要求双倍精度等等，这就要求使用解释程序。解释程序可以逐条地取出源程序中的语句（或指令），边解释边执行。

键盘　key board

键盘是计算机的输入装置。其上排列若干文字和符号。输入时，将键按下，即可把该处的文字或符号变为代码。编码方法分为ASCII和JIS两种。键盘上文字的排列与普通打字机大致相同。

溢出　overflow

在计算机中，如果在计算过程中产生的数超过机器所能表示的范围，就称为“溢出”。

数据管理　data management

操作系统中负责管理数据——包括文件系统、程序库……的那一部分。

数据通路　data path

计算机中各种信息代码，如数据、指令、地址等在机器内传输的路径，连同暂时存放它的寄存器，锁存器系统称为数据通路。它一般在逻辑总框图中示出。

输入输出操作　input-output operation

指计算机的内存储器和输入输出设备之间的信息传送操作。

输入输出通道　input-output channel

它是用于外部设备与主存储器之间作信息和数据的传输通道。输入输出通道通常包括选择通道，字节多路通道和数组多路通道。

十 四 划

模拟程序 simulator

在一台计算机上解释执行另一台计算机的程序，这个解释程序称为模拟程序。模拟程序可以用来模拟各种程序设计语言，也可以模拟计算机整机功能或某部件的逻辑结构。

模块 module

在操作系统中，模块是指为了完成某一功能所需的一段程序。操作系统所需执行的功能可以分门别类地交给各个单独的、为完成一定功能所预先设计的模块去完成。模块结构是设计操作系统的一种方法。

模块化程序 modularized program

模块化程序也称为积木式程序。它的基本思想是把一个大型的程序从结构上分解成许多相对独立的模块程序。即每一块模块程序具有特定功能，它既有信息入口，又有信息出口。这样，就可以使大型程序结构清楚，调试时可以分块进行。

管理程序 supervisor

有时亦称之为监督程序或执行程序等。它是为提高计算机的使用效率、合理使用资源、方便用户而设计的一种程序。

算法语言 algorithmic language

是一种接近数学描述的程序设计语言。

随机性故障 random fault

由于外界电网电压的跳动、外部电磁性干扰信号或某连接处的虚焊等原因所引起的偶然性故障叫做随机性故障。它往往不待人工干预，系统又能自动恢复正常工作。

随机文件和顺序文件　random and sequential file

当在磁带上读写文件时，由于磁带本身形状的限制，只能从头开始按序进行。无论选取哪一段数据都须从头来，既浪费时间又不方便。在使用软盘时，根据程序的指示，可以直接读写任一道的数据，因而能进行高速数据处理。前者称为顺序文件，后者称为随机文件。

随机存储器　RAM

既能读出也能写入的存储器。因此也称为读写存储器(RWM)。

十六划

操作系统　operating system (OS)

在机器硬件的基础上设计一套程序，使机器的操作、各种资源的调度与管理自动化，尽量减少人工干预，并实现计算机资源多个用户共享，这就是“操作系统”。广义的操作系统一词还把各种语言处理程序（编译程序、汇编程序、编辑程序等等）也包括在内。

操作码　operation code

操作码是指令的一个组成部分，表示要求计算机执行一个什么样的操作；例如加法、减法、移位等等。通常操作码是用机器代码表示。在符号语言中，操作码用记忆码表示。

十八划

翻译程序　translator

翻译程序的功能是把一种语言写成的语句转换成另一种语言的语句，翻译前后的语句功能是等价的。如汇编程序，编译程序等都是具体的翻译程序。

翻译语言　translator Language

同编译语言类似的一种语言，二者是同级的。但是翻译语言不象编译语言那样完全变换成机器语言，而是暂且变换成中间语言程序并借助执行程序来执行它。