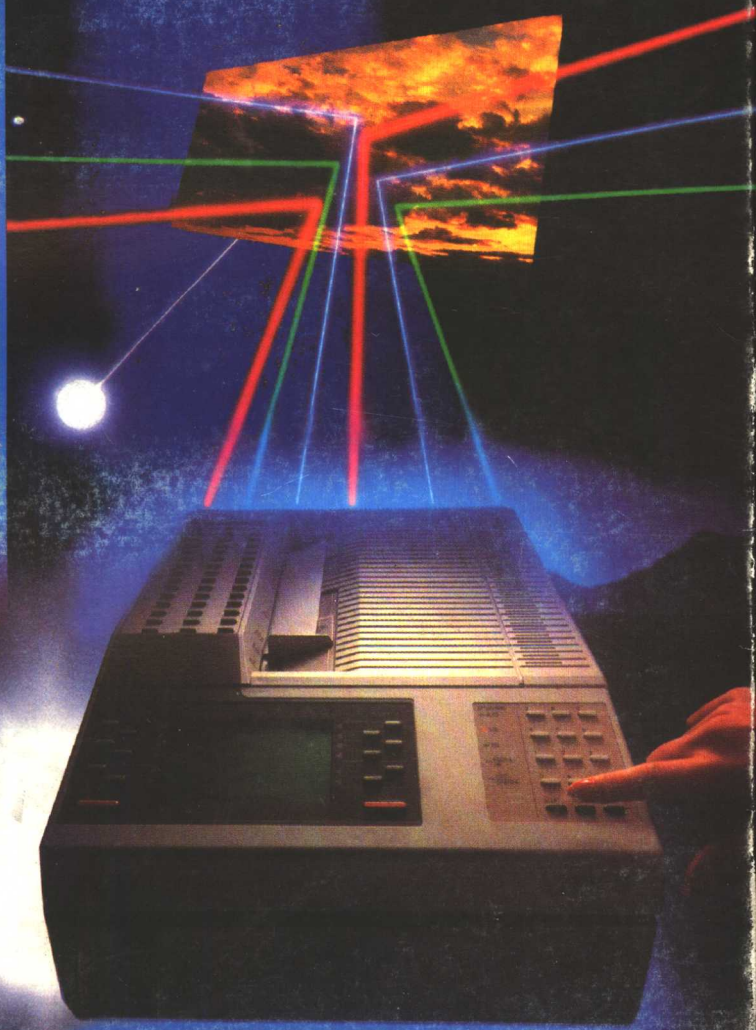
新世纪百科
知识金典
XINSHIJI
BAIKE ZHISHI
JINDIAN

计算机信息技术 2

傅世海 著

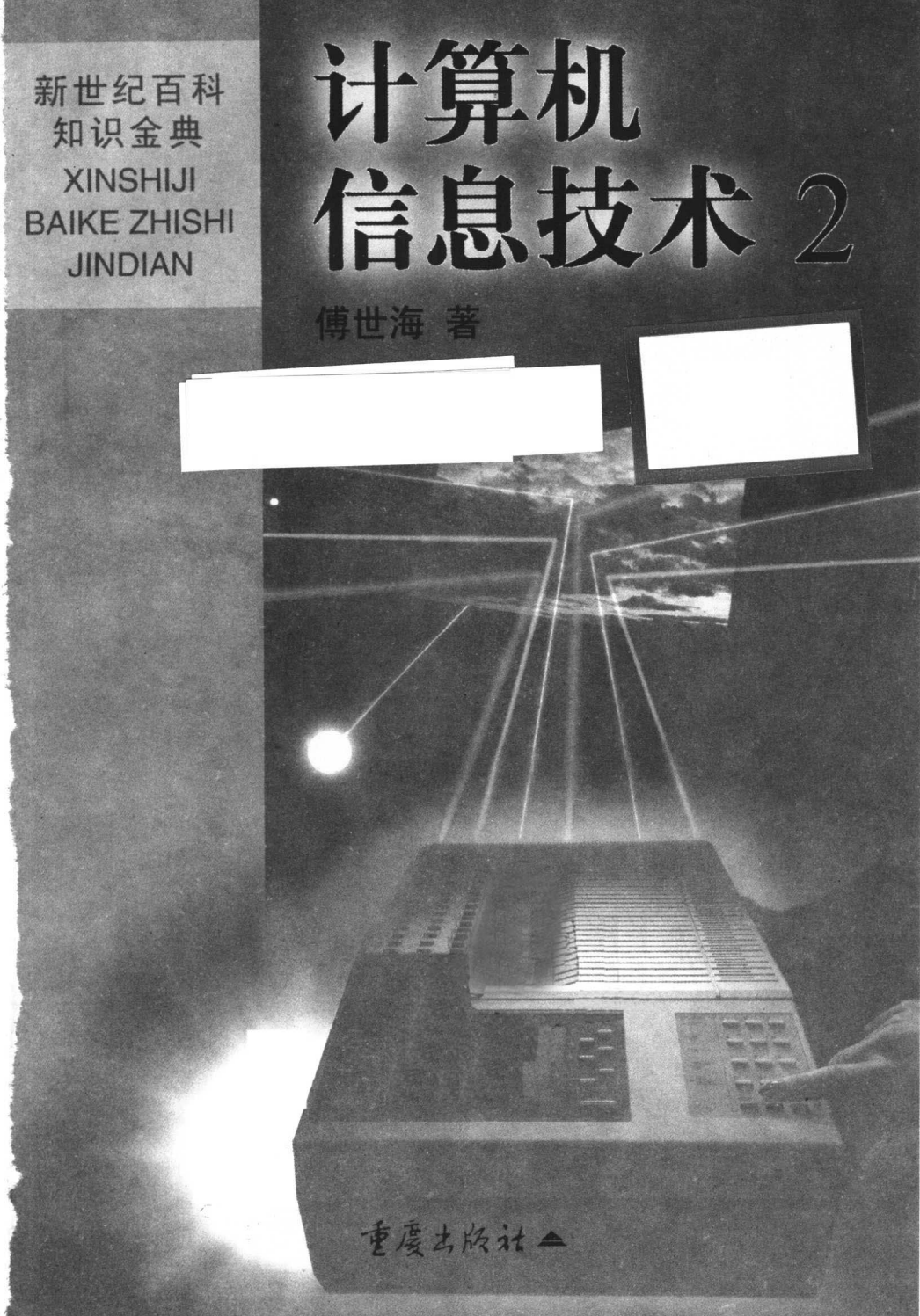
重庆出版社



新世纪百科
知识金典
XINSHIJI
BAIKE ZHISHI
JINDIAN

计算机信息技术 2

傅世海 著



重庆出版社

责任编辑　**刘庆丰**
封面设计　**金乔楠**
技术设计　**刘黎东**

新世纪百科知识金典
计算机信息技术 2
傅世海　著

重庆出版社出版、发行　（重庆长江二路205号）
新　华　书　店　经　销　重庆新华印刷厂印刷
*
开本 850×1168　1/32　印张 5.25　插页 4　字数 122 千
1999 年 4 月第一版　　1999 年 4 月第一版第一次印刷
印数：1—5,000
*
ISBN7－5366－4176－1/TP·44
定价：8.00 元

新世纪百科知识金典

目录

第七章　用语言设计程序

从前面的讨论中，我们已经知道，电脑是靠程序工作的，电脑中的 CPU 要不断地从内存中取得指令、并不断地解释所得到的指令并加以执行，指令形成的序列，就是电脑中的程序。为此，人们设计了编写程序的语言。

一、CPU 只认得指令系统

CPU 是电脑的核心部件，在它不断执行指令的过程中，电脑完成了程序给它规定的功能。CPU 就是我们通常说的 286、386、486、586 和 686，这是个人电脑中用得最多的一类 CPU。当然，也还有别的 CPU，比如在 Intel 公司的 80X86 系列外，有 Motorola 的 68000 系列、有 DEC 公司的 ALPHA 的系列、有 SUN 公司的 SPARC 系列等等，这些不同系列的 CPU 是用不同的指令系统。好比人的语言，各个不同国家的民族，所用的语言互不相同。因此，不同系统的 CPU 所用的指令不同。指令是用 0 或 1 的数位组合的，是计算机的机器语言，计算机可以执行的程序都是机器语言表示的指令序列。CPU 只认得属于它的一套指令系统编写的程序，其他 CPU 可用的程序，是不能运行的。CPU 的兼容性是指同一类型、同一系列的 CPU 之间，有向上兼容的性能。什

么是向上兼容呢？就是同一类 CPU，如 Intel 的 80X86 中，有 8086、80186、80286、80386、80486、以及俗称为 586 的 Pentium（奔腾）和俗称为 686 的 PentiumPro（超能奔腾）等。这一个系列是多年进化而来的。也就是说，随着技术的进步，80 × 86 系列 CPU 中后来出的 CPU 包含有前面 CPU 的指令，指令的功能增强，除了增强原有指令的功能，还增加了不少新的指令。所以，486 上可以运行的程序，586 上也可以运行，反过来就不一定了。这就叫做向上兼容。CPU 同一系列之间这种向上兼容的功能，是 CPU 设计者和生产厂商保障他们老用户的利益，也就是他们可以继续保持老用户使用他们产品的可能性。这样做是很好的。不这样做，会失去原来的用户。比如原先流行的苹果机是苹果Ⅱ型的，在七十年代末到 1982 年间，风靡全球，Apple Ⅱ 中用的 CPU 是 6502，可是 1983 年苹果公司推出的 Apple Ⅲ 有两款 Lisa 和 Macintach，用的 CPU 改了，是 68000，Motorola 公司的，这一改，把许多是苹果公司的老用户拱手让给了当时上市的 IBM-PC 了。这是因为新的苹果和原来的 Apple Ⅱ 完全不兼容，太贵，而 IBM-PC 选择的 Intel 系列 CPU，一上来就有诸如微软公司（Micro soft）等一大批公司为它开发软件。在 1983 年，苹果公司的 Macintach 一上来就有个好用的视窗操作系统，但也没有非常走俏。这是因为以前的程序一下子不能用了，以后为他开发软件的公司究竟有多少，当时前景不明。所以，号称“蓝色巨人”的 IBM 公司大旗一挥，个人电脑从此是 IBM 和 IBM-PC 兼容成了主流。从 1983 年到如今的十五年中，Intel 的 CPU 成为个人电脑上的主流。Intel 步步为营地推出的新的 CPU 总是兼容以往的指令，Microsoft亦步亦趋地与之相伴，DOS 改了许多个版本，十二个年头以后，才使 Windows95 隆重登场，这中间，就是让用户也步步跟得上的战略，使得个人电脑市场稳步地扩大。

二、从机器码到语言

电子计算机在问世时，根本没有高级语言。

最早出现的编程语言，只是一些符号，又称助记符，因为机器指令的0和1编码不好记，所以，用八进制，后来是十六进制来记忆指令代码。比如，在Intel80X86中，调用中断13去作磁盘管理的指令，我记得最牢，是CD13，译成机器码就是11001101 00010011，C对应着1100，D是1101，ABCDEF在十六进制中分别代表相当于十进制中的10，11，12，13，14，15，CD分别是12和13，二进制是1100和1101；十六进制的13是00010011，十六进制的一位代表四位二进制，所以，人们先是用八进制（一位代表三位二进制），后来用十六进制来辅助记忆机器指令。最后，又把很简短的英文缩略语引入，称为汇编语言，如加法用ADD，减法用SUB去表示指令功能。上面讲的中断13，它的机器码、十六进制形式和汇编语言分别是：

11001101 00010011（机器码）

CD　13　（十六进制）

INT　13H　（汇编语句）

汇编语言是最早的程序设计语言。

一句汇编语言，对应一条机器指令。尽管这样做，对于大多数用户来说，仍然难以接受，但是，在写程序时，至少可以有一些帮助记忆的英文缩略语，帮助我们看到一点指令的功能含义，单是0和1或十六进制，是一点含义都看不到的。

有了程序设计的语言，就得有翻译程序。

翻译程序的作用，是把程序设计语言写的程序，翻译成为CPU可以认得的机器码指令序列。程序设计语言写的程序和CPU可以识别的机器码程序不同，人规定的符号CPU不可能立

即当指令执行,所有用程序设计语言写的非机器码指令的程序,统称为源程序。

把汇编语言写的源程序翻译成机器能够认得的 CPU 指令,就是汇编程序,又称汇编语言翻译程序。

于是,就需要有两种实用的软件:编辑程序和翻译程序。

编辑程序是用来编写源程序的程序。人们使用编辑程序,可以把文字写入文件中去,文字先是写成汇编语言的源程序,后来,又演变成为写和修改各种文稿的文字处理程序。

翻译程序是用来把源程序转换为机器码指令序列的机器码程序。

汇编程序是所有翻译程序中最简单的,因为汇编语言写的程序(应当叫源程序),是一句对应一条指令的。当我们按汇编语言规定的格式写好源程序后,就可以逐句逐行地作翻译。

三、语言的高级和低级

汇编语言是对应着一条条指令而设计的,是面对机器的指令系统编写程序的,所以要运用和调度电脑中硬件的资源,就要说明用 CPU 中的哪一个寄存器,内存中的哪一个地址等等来写程序,这种方式称为面向机器的编程。

面向机器的编程,要学习许多硬件的知识,这不是所有电脑用户都能办到的。所以,我们必需另外找一条途径,来让用户编写他自己需要的源程序。

开始的用户大都在英语地区和国家,所以,采用了一种类似英语的语言来组成程序设计语言。英语是自然语言。这种语言和机器语言的差距就比汇编语言和机器语言的差距更大了。为了人们编程方便,为了要清楚地叙述人们解决问题的过程,高级语言的程序设计就要方便编程的人,把精力集中在所解决的问

题上，而不能再沿用面向机器的程序设计方法。高级语言被设计成面向问题的程序设计。编写源程序的人不再关心程序中用到 CPU 中哪个寄存器，数据事先该存放在内存中什么地址，中间结果和最终结果又应放在内存的哪个地方。这些机器中的问题，由翻译软件承包了。

高级语言由于它不再是面向机器的程序设计语言，所以，它最大的好处就是可以在写源程序时，做到和机器无关，和 CPU 是哪个指令系统无关。正是由于高级语言的这个优点，如今，高级语言设计的源程序，可以做到在不同 CPU 的机器中共享。我们已经知道，源程序并不是拿来就可以执行的程序。把源程序翻译成为本台电脑或同一类 CPU 的电脑可用的程序，还得由翻译程序去解决，翻译为机器码指令后，当然也只能在同一种 CPU 指令系统的电脑中通用。源程序为了要能够让各种不同 CPU 的翻译软件可以翻译，要求源程序有统一的格式，即统一的语法、词法和单词。

所以，任何一种编程语言的学习，都是类似于学一种外语。语法、词法和单词，均不可以随心所欲地创造发明，只有完全遵守规定的规则才行。程序中的创造性是指在遵守某一种编程语言的语法、词法的前提下，就程序本身的执行步骤、方法上的创造，而决非任意地创造语句，不然，翻译软件不认得你的意思，就无法工作，无法把源程序译成 CPU 认得的指令序列，即不能产生可执行的机器码程序。

四、解释和编译

有的青少年读者学过 BASIC 语言，他对我们关于源程序要先翻译成机器码的程序才能执行，持异议态度。他感到，在他写入 BASIC 程序后，打入一个运行命令 RUN，BASIC 的源程序似乎

马上被逐条执行了，并没有产生什么可执行的机器码程序。

感觉是一回事，事实又是另一回事。

这位读者一定用了某一个解释性的 BASIC 版本，如 BASIC 或 QBASIC。

翻译程序有两类，那就是编译和解释。开始有高级语言时，并没有解释性的翻译程序。解释性的翻译程序和编译不同，它是把源程序一句一句地作解释，产生出对应于某一句或某一段源程序的机器指令序列，也就是可执行的程序，CPU 指令系统中规定的机器码指令程序，并马上执行这一小段程序。用户在 RUN 命令下达后，看不到这个过程，这个过程被“透明”掉了。

解释性的翻译程序的优点在于，可以逐句地为你“执行”源程序。这样做有什么好处呢？好处就是让用户以为在执行源程序，于是，当某一句源程序写得不对时，马上会停下来指出错误。这种做法很受初学者的欢迎，编写程序后，可以在试运行中调试程序。可是，缺点也随着优点而来！正像外语翻译在做口译那样，要一句一句地翻译，若像笔译那样一气呵成，以后只需直接读翻译好的中文，不是更快些?！正因为解释性的翻译程序要一句句地翻译、再一句句地执行，运行速度就比另一种称为编译的翻译程序慢多了！

编译后，用不着源程序了，每次可以立即执行可执行的机器码程序，当然快了。但是，这个优点又产生了缺点。当你的程序，也就是源程序写错了，比如有语法错误，编译程序并不停下来告诉你哪句错了，它把整个源程序给你翻过来，所以前面错的地方，会影响后面，故最终报出一大堆出错信息。其实，前面某处改过后，后面并没有错，用不着改。由于编译给你报出一大堆错误，究竟哪些地方是真的错了，需要修改；哪些地方没有错，并不要修改，对初学者来说，不易搞明白。

所以，比较理想的高级语言是同时有解释和编译两种版本，

调试程序时，完全可利用解释的优点，一旦调试完毕，再把源程序做成可执行的机器码程序，把没有错误的源程序生成一个可执行的程序，不就两者的优点都有了，缺点也避免了。这才是一种聪明的办法。

解释 BASIC 实际上是在推广 FORTRAN 编译时有困难，把 FORTRAN 语言中的一些语句，搞了一个子集，做成了解释 BASIC，于是入门就有了交互式的翻译程序。

随着应用普及，BASIC 的解释版本不够用了，又有了 BASICA(高级的、扩充的 BASIC)，也有了 BASCOM(BASIC COMpiler)，即 BASIC 的编译版本，以后又在 QBASIC 中，把解释、编译做在一起，让用户在解释通过后，再做编译。

BASIC 现时还有了 Visual Basic，可视化的 BASIC，可以用作多媒体的编程，功能是很强大的。

类似兼有解释、编译的有 FoxBASE，人们称为大众数据库的流行软件。

最后还想讲一下不完全编译。一个东西是完全好，还是不完全好？人们自然以为是完全的好？那么，有了完全编译后，为什么要做不完全的编译呢？原来这种做法是为了减少最后所生成的机器码程序所占的空间。国内有一个搞 2.13 汉化的软件公司，现时已和别人合并了，当初搞 2.13 汉化是晓军电脑公司，我注意到他开始的产品中带有 BASRUN.EXE，这正是不完全编译所需的一个公共软件。如果为了完全编译，每个模块都要增加二十到三十千字节，当一个软件由几十个模块组成时，不如用一个公用的支持模块 BASRUN.EXE，它虽有四五十千字节，但是有三、四个以上模块时，对盘空间总的占用数会大大节省下来，这正是当年用软盘递交产品时所希望的。

五、各显神通的高级语言

在电脑中用于程序设计的高级语言有好几百种，各种高级语言都有一定的特点，随着电脑应用的普及，它们不断地推陈出新，以更新版本的方式，扩大功能，有的兼有其他高级语言的优点，还各自根据应用场合的不同保留自己的特点。开始，各种高级语言分别都在 DOS 操作系统、或在 UNIX（XENIX）操作系统下，改进着各自的版本。在 Windows 操作系统推出以后，一些常用的高级语言又为了适应视窗工作环境，推出了面向对象的可视化（Visual）的版本。

BASIC 语言的名称就是初学者通用和符号指令代码（Beginner's All-purpose Symbolic Instruction Code）第一个字母的缩写。它是目前国际上非电脑专业人员应用最广泛的语言，几乎所有的计算机，包括儿童启蒙用的学习机和掌上微电脑，都配有这种高级语言。几乎所有的学习电脑的人，都把它当作入门课程。BASIC 语言有简单、明了，容易掌握的特点。这是因为它是六十年代由电脑语言设计的专家们，把 FORTRAN 的一些基本语句作基础，推出的一种交互式的解释性翻译程序，开始只有十几种语句，比其他的语言更加方便。日后，由于应用面的扩大，应用场合的要求不同，BASIC 在 PC 机中出现了 BASICA 这一类的高级 BASIC，它是基础 BASIC 的扩展。微软公司的总裁比尔·盖茨，被人们称为电脑奇才。他和一群当时的年青伙伴一起，曾经把 BASIC 的功能作了充分扩展，使 BASIC 不仅有解释版本，而且有了编译版本。BASIC 不仅可以用于一般的计算，而且还加入了输入输出的函数和语句，去访问内存和接口。于是，又有了用于控制的 BASIC。比尔·盖茨毫不夸张地说，世界上所有其他程序可以实现的，他都可以用 BASIC 程序编写出来。这当然不是

早期那种十几条语句的低级 BASIC。BASIC 后来有了 True BASIC 和 Quick BASIC(QBASIC),成为 DOS 附带的一个实用软件。到了 Windows,现在又有了 Visual Basic,简称 VB,可以作多媒体的应用开发。是不是一开始就学最高级、最新版本的 VB 呢?据我个人的经验,对初学者,不必开始就找最高版本来学。任何一种软件的最新版本,往往内容过于丰富,反而会使初学的青少年感到丈二和尚摸不到头脑。电脑软件应用中,人们发现了一个 20%的规律,一个软件做出来以后,甚至到了它应用的鼎盛时期,直到有更新的软件代替它为过,往往也只有 20%的功能被充分利用,而其他 80%的功能则很少被用到。因而,为了避免初学时的困难过多,青少年读者在开头时,不一定要找一种什么最新版本的最高级语言,想来一个一通百通,一学就会的解决办法。目前,有许多可视化的高级语言,甚至在编程中可以用操作去代替,但是,在简单的背后,可能有许多术语、概念,不是一下子可以理解的,很多软件中的高级功能,需要我们打好基础,才能理解和运用。

PASCAL 是学习计算机专业的学生必需掌握的一种高级语言。它目前没有解释版本。PASCAL 是以一位数学家命名的,这种语言明确、严谨,所以,在电脑专业的书籍中,特别在描述算法的书籍、论文中,人们常常用一种类 PASCAL 的方法,用一段 PASCAL 程序说明一种算法,比用文字来叙述就清楚多了。

目前在高级语言中被公认为阳春白雪的是 C 语言,C 语言是用来写系统软件的语言,它的特点是有许多标准函数库,几百种标准的函数可供用户调用,使得它的关键词,也就是编写程序的语句类型只有二、三十个。C 语言很灵活,在用指针上很方便,这有利于系统软件来实现表处理技术。不同的 C 语言版本,差异在于可用的标准库函数,在 Visual 类语言出现前,人们首先在 C 语言中加入了绘图功能,因此,不少大型应用程序是

用C语言开发的。

不管各种语言有什么样的发展，许多工程计算的专家仍然看好FORTRAN语言，它仍然是国际上最流行的数值计算语言，FORTRAN是公式翻译的缩写(FORmula TRANslator)，从1956年以后，它发展很快，从FORTRAN Ⅰ、FORTRAN Ⅱ……一直到了FORTRAN77。有人统计过，在BASIC出现前，数值计算的科技人员，有90%是用FORTRAN来编写程序的。FORTRAN语言早期是在穿孔卡片上输入的，所以，这种语言的书写格式上有对应于卡片上各列位置的要求，比如1～5列用作编号，语句通常是写在第7列到第72列上。这种格式规定是它的一个特色。

英语国家中，许多商人、企业家对COBOL情有独钟，COBOL程序结构很像中国八股文的启承转合，任何一个COBOL程序必须有以下四个部分组成：

标识部分

环境部分

数据部分

过程部分

COBOL语言中大量使用英文，如：

ADD ALPHA,0.5 TO BATA [ROUNDED]

是要求把变量ALPHA加0.5和BATA，并把结果四舍五入后送入BATA中。

又如乘法A乘B并把结果送入B写为：

MULTIPLY A BY B

而MULTIPLAY A BY B GIVING C则结果送C.

于是，COBOL语言对于英语为母语的使用者很方便，看COBOL犹如看文章，更接近于自然语言。在进到中国的一些外资公司中，不少经理要求受聘员工懂COBOL语言。实际上COBOL是COmmon Business Oriented Language，是面向公用商业的语言，

它是一种商用和行政管理语言。

说到管理，人们就会想起流行的数据库语言，叫“大众数据库”，从 dBASE 到 FoxBASE，又发展到 Foxpro，最近也有了它的 Visual 版本。数据库语言的优势在于对于二维表格表示的数据信息的管理，有独到的优势。所以，在商业、财会人员中特别流行，政府公务员的考试，不少地方也要求懂数据库。管理上大量的信息可以用二维表格表示数据的关系，故人们又称为关系数据库。这种高级语言有解释版本，所以，对入门者，只须学几种命令，如 USE、CREATE、LIST，BROWSE，以及知道〈ESC〉和 ∧W 的作用，就可浏览，使用数据了。它和 BASIC 比较，有一句可代替 BASIC 好几句的感觉，因而被人们称为功能更强的高级语言。

ALGOL 语言曾在我国流行过一段，这与早期我们同东欧国家来往多有关。ALGOL 就是算法语言 ALGOrithm Langaage 的缩写。也是在我国最早流行的一种算法语言，它和 PASCAL 很相像，所以，不少用 ALGOL 语言编程的人，后来自然转向了 PASCAL。

高级语言程序设计是面向对象的。数值运算处理，使人们留心计算过程中变量的变化。所有开始学习高级语言的人，一开始都有一点容易混淆的地方，就是和数学不同，高级语言程序设计中有一个赋值的概念。如

I = I + 1

这是数字中视为荒谬的，如果把它当数学式子，把左右的 I 对销，就成了 0 = 1。计算机中把这个等号叫作赋值。就是把右边的表达式计算过后送到左边的变量中去。I = I + 1 是 I 增加 1 的增量运算。这就使我们看到，电脑高级语言虽在搞计算，它不同于数学。要弄清程序，不管它是哪一种语言编写的，都要弄懂赋值的含义。我们不是常常看到程序设计讲究计算方法，也就是算法。我以为算法中最基本的一点，和数学不同之处是，变量