DJS—21

算法语言讲义

湘潭大學

数理系计祘数学专业编

1979年10月

序

随着国民经济的迅速发展，电子数字计算机将更加广泛地应用在生产与科学研究的各个部门。普及与推广使用电子计算机的工作是十分必要的。在使用电子数字计算机都要遇到一项重要的工作，就是编写计算问题的程序。目前在科学和工程计算中最常用的程序语言有三种：ALGOL60，FORTRAN，BASiC，国内多数电子计算机配有ALGOL60的算法语言。

本讲义是在76年为工人，工程技术人员掌握使用ALGOL60算法语言而编写的，为了便于自学，力求通俗易懂，因此在讲义中尽量少用专门的数学知识，在例子中也未采用一些常用的算法。在编写程序时可以参考沈阳计算技术研究所编写的“电子计算机常用算法”一书（科学出版社1976年）。

目前国内DJS—21机（即121机）与DJS—6机使用较多，因此以DJS—21机为主来编写。这并不影响掌握ALGOL60与DJS—6机的使用。在附录中指出了它们之间的区别，与DJS—6机的操作使用。

本讲义主要参考了中山大学《怎样使用121机算法语言》、北京工业大学《怎样使用计算机》、吉林大学《算法语言ALGOL60入门》。

由于水平有限，难免存在错误和不足之处，请批评指正。

符启宇

一九七六年四月

目　录

第一章　基本概念与源程序结构

第1节　引　　言

随着国民经济的发展。我国生产与科研的各个部门日益广泛地使用电子数字计算机，以解决本专业的大量计算问题，并取得了很好的成效。

使用电子数字计算机都要遇到一项重要的工作，就是编写计算问题的程序。通常编写程序的方法，一种是用数字代码形式的机器语言来编写，即所谓手编程序的方法，用这种方法进行程序设计一般的步骤为：画出框图，编写文字（符号）地址程序，分配单元，将文字地址程序进行代真等。这些工作的工作量很大，十分烦琐而且容易出错，又不便于检查就是检查出错误修改起来也非常不便，对于一个大型复杂的问题，工作量相当大有时要用很长的时间。而电子数字计算机进行运算的速度很快，因此，提高机器的使用效率与落后的手编程序设计之间的矛盾逐渐突出，随着大力推广和广泛使用电子数字计算机广大工农兵和工程技术人员直接掌握使用计算机与使用计算机人员必须掌握繁难的手编程序设计的知识之间的矛盾也逐渐突出，矛盾的主要方面是计算机只认识二进制数表示的指令与数构成的程序。这种程序的编制与数学公式和日常用语相差很大。

例如，　　当计算 $S=\dfrac{(X+0.5)(X-0.5)}{0.5\times 0.31-X}$ 时，

比较两种语言的特点。

机器语言

```
0300        X
0301   000  0  500000000
0302   000  0  310000000
0303        S
```

数学语言

计算

$$S=\frac{(X+0.5)(X-0.5)}{0.5\times 0.31-X}$$

程序（“10”换“2”的程序省略）

0310	002	0301
	$00\bar{2}$	0302
0311	009	0300
	004	0303
0312	002	0300
	008	0301
0313	004	0302
	002	0300
0314	009	0301
	$00\bar{2}$	0302
0315	$00\bar{4}$	0303
	004	0303
0316	034	$01\bar{0}9$
	$03\bar{5}$	0000

从这个例子中很清楚地看到它们的特点为：

优点：（1）机器能接受
（2）能描述数值计算的细节
缺点：（3）一般人不习惯
（4）难读

缺点：（1）机器不能接受
（2）不能描述数值计算的细节
优点：（3）一般人习惯易接受
（4）易读

算法语言是吸取前面两种语言各自的优点，剔除其缺点而设计的程序设计语言。对于上例若用算法语言编写程序，则为：

Y（源程序）

‘BEGIN’（开始）
‘REAL’ S,X;（S,X为实型数）X:=READR;
S:=(X+0.5)×(X-0.5)/(0.5×0.31-X);
PRINT(S);（打印结果）
‘END’（结束） ××××

从这个例子中可以看到用算法语言编制程序有以下的特点，

（1）比较接近数学语言与日常用语，又能描述计算过程。

（2）其书写与数学公式比较接近，形式直观，容易掌握。

（3）编制方便、检查方便、修改方便，节省编制程序的时间。

（4）只要各计算机所配的为同一算法语言，则不同型号的计算机都可以使用同一程序。

由于上述特点，普及用算法语言编写程序的方法是推广应用电子数字计算机的一个重要环节。

用算法语言编写的程序，机器如何理解并去执行，是通过“编译系统”来实现的，

也就是在机器内先输入编译系统，当输入用算法语言编制的程序后，机器通过“编译系统”将其编译成机器语言（即数字代码程序，我们称为目的程序）进行计算。

用语言来编制程序，其语言形式很多，这里是介绍121机所用的算法语言，它是在ALGOL——60的基础上设计而成的。

算法语言即是Algorithmic lanquage简称ALGOL。这种语言是1960年在法国巴黎召开的国际会议上接受为国际语言故称ALGOL——60。

目前121机，108乙，X－2，709等都配有以ALGOL——60为基础的算法语言，基本上大同小异。

第2节 基本符号

一切算法语言编制程序，都是由基本符号严格按照语法规定写成的，在程序的任何地方都不允许出现非基本符号或违反语法规则的情况，否则机器将不能进行正确的计算。

121机的算法语言中基本符号包括英文大写字母，数字。运算符号和一些特定的符号。由于电传机上的符号不能满足习惯上常用的一些关系运算符和逻辑运算符号，所以就采用字母并起来表示的办法，而括弧中是习惯的写法与意思。

（1）字母：A、B、C、D、E、F、G、H、I、J、K、L、M、N、O、P、Q、R、S、T、U、V、W、X、Y。Z。

字母共26个不分大写小写，字母在源程序中用作标识符。

（2）数字：1、2、3、4、5、6、7、8、9、0。

数字在源程序中组成数与标识符。

（3）两个逻辑值　TRUE（真）　FALSE（假）。

逻辑值用来表示逻辑运算中布尔变量的取值情况的。

（4）括号 ..

（ ）园括号，〔 〕方括号，‘ ’定义符括号，“ ”行括号，‘BEGIN’ ‘END’语句括号。

这里要注意，在算法语言中方括号〔 〕的使用不同于数学语言中的方括号，在算法语言中方括号只用于下标表达式。

（5）运算符：　(i) 算术运算符

+（加）、－（减）、×（乘）、/（浮点除）、

//（定点除，先用浮点除，再4舍5入取整数）

××（乘方）。

(ii) 关系运算符：

‘LS’（小于<）、‘LQ’（小于等于≤）、=（等于）

‘GR’（大于>）、‘GQ’（大于等于≥）、‘NQ’（不等于≠）。

(iii) 逻辑运算符：

‘NOT’（非¬）、‘AND’（与∧）、‘OR’（或∨）、

‘IMR’（蕴含⊃）、‘EQV’（等价≡）。

(iV) 顺序运算符：

'GOTO' (转向)、'IF' (如果)、'THEN' (则)、ELSE' (否则)、'FOR' (对于)、'DO' (做)。

(6) 分隔符：

.(小数点)、, (逗号)、; (分号)、: (冒号)

:=(赋值号)、::=(双点赋值号)、10 (小拾)、

'STEP' (步长)、'UNTIL' (直到)、'WHILE' (当)

† (消错符号)。

(7) 说明符：

'OWN' (固有)、'BOOLE' (布尔型)、'INTEG' (整型)

'REAL' (实型)、'ARRAY' (数组)、'SWITCH' (开关)

'PROC' (过程)、'DRUM' (鼓)、'CODE' (代码)。

(8) 分类符：

'VALUE' (值)、'LABEL' (标号)。

第3节 一些基本概念

(一) 数：

在科学计算与数据处理中，总离不开数，在121机的算法语言中，对数的表示形式提出相应的要求。

在121机的算法语言中，数的表示形式与通常十进制的数的形式相近，但在书写时，作如下几点规定。

数按书写形式分整型数('INTEG')与实型数('REAL')。

(1) 整型数 为由数符(+)或(-)与0～9组成的数。

例如：+1437，-51，-1437，-1200，+700，等都是整型数。当数符为(+)时，与数学语言中的习惯一样，可以省略不写，例如+1347，与+700，可写成1347与700，根据规定可以看出，带有小数点的数，如-51.0虽与-51相等但不能定为整型数，带有乘方的数 7×10^2 虽与700相等也不能定为整型数。

(2) 实型数 即非整型数，其书写一般形式为：

$$S_fC.C_{10}S_fC$$

其中 S_f 表示数符，C表示自然数和0，例如 $19.8_{10}11$，$-2_{10}-4$，5.0，0.5，0.032，$1_{10}8$，.32，都是为实型数。

在书写数时要注意以下几点：

(I) 小拾10它本身并不表示一个数，是一个基本符号，并要求10后面必须是整型数。例如 $1.2_{10}3.0$，是不正确的。

(II) 数只指单个的数，算术表达式不能作为数来使用，例如 $\frac{1}{2}$ 是个表达式不能作为数来使用，因为用了运算符，若要作为数来使用应写成0.50或.5

(III) 在书写时，不允许按照日常习惯添加逗号，例如21478956不能写成21,478,956的形式。

（Ⅳ）如果有效数字为1，则可允许省略，即 $1_{10}4$ 可写成 $_{10}4$，数中不必要的数符与0可以省略，例如003可写成3，+0.050可写成.50。

（Ⅴ）对于实型数　小数点之后至少有一位，即3.0中的0不能省略。

（Ⅵ）书写数时　数与数之间必须用分号（；）隔开不允许用别的符号。

（Ⅶ）任何电子数字计算机，由于只有一定位数的字长，所以所能表示的数的范围是有限制的。对121机。

整型数绝对值不能超过 2×10^{10}

实型数其绝对值不能超过 3×10^{19}

从这里可以看出，整型数的取值范围远远小于实型数的取值范围，但是整型数是完全精确地表示的，而实型数则往往带有一些误差，只能达到一定的精度，当要求需要用完全精确地表示数（例如控制循环的次数）用整型数是方便的。

（Ⅷ）在计算中用到的数有直接写在程序中，有的另外给出放在数据部份，不管什么地方出现，都按上述形式书写。

关于数的其他说明将在有关章节中叙述。

（二）变量与标识符：

在121机的算法语言中的变量概念与数学中的变量概念是一致的，即变量是指可以改变其取值的量。在数学中常用 x，y 等来表示。而在算法语中规定用标识符来表示变量。

在算法语言中规定，标识符是由字母串或以字母开头的字母数字串组成，不能再含有别的符号与其他基本符号。在121机中限定。标识符长度小于或等于7个字符。

例如：X，Y，X_4，B2K5，ALPFA，等都是标识符。

而$4X$，B－Y，A.B，β，α，Bπ，等都不能作标识符。

标识符除了作变量标识外，还作为给数组，过程及标号取名用，（有关概念在以后的章节中介绍）在标识符的选取上，并没有什么限制，一般以选取符合于习惯或有助于理解的名称为佳。

例如，时间用T，和用SUM，高度用H，项用TERM等。

在121机算法语言中变量可分算术变量与布尔变量，在每类变量中又可分为简单变量与下标变量两种。

（1）简单变量与类型说明：

简单变量就是取值只能是一个数值或一个逻辑值的变量，如果简单变量的取值是整型数，则称为整型简单变量，简称整型量。如果简单变量的取值是实型数，则称为实型简单变量简称实型量。如果简单变量的取值是逻辑值，则称为布尔型简单变量简称布尔型量。

例如我们对于关系式的取值就是逻辑值，当关系式成立为TRUE（真）否则为FALSE（假）

若：　　X＜5　当X≥5　时取值　FALSE

　　当X＜5　时取值　TRUE

由于简单变量分三种：实型量，整型量，布尔型量而它们在进行运算时运算方式不

同，在编译系统分配单元时需要加以区分以便以后运算时方便。所以在编写源程序（即用算法语言编写的程序）时，应加以说明，因为简单变量直接用标识符来表示，在书写时从标识符本身无法区别。在源程序中，这种说明标为类型说明，其书写方式为：

(i) 首先书写说明符，对整型量用 'INTEG' 对实型量用 'REAL' 对布尔型量用 'BOOLE'

(ii) 把相应类型的简单变量的标识符写在说明符的后面，对同类型的各个变量可以分开说明也可以合起来说明，当合起来说明时，每个变量间用逗号（，）隔开，在写完后用分号（；）结束。

例如：

'INTEG' I, J, N, TIOM, HK5;
'REAL' H, K, SUM, B5H;
'BOOLE' B, BOH, POD;

这表示说明 变量 I，J，N，TIOM，HK5为整型量
变量 H，K，SUM，B5H为实型量
变量 B，BOH，POD为布尔型量

(iii) 在源程序中，对简单变量的说明，至于先说明那一种类型变量并无要求，但对所有用到的变量都要说明。

（2）下标变量与数组说明：

在解决实际问题时，常常遇到按一定顺序排列的一组量，其中每个量在计算过程中可以取不同的取，因此这些量本身一般讲都是变量，我们称这些量的全体为一个数组。

例如：n 阶多项式的系数

$$P(X)=C_0x^n+C_1x^{n-1}+\cdots\cdots C_{n-1}x+C_n$$

其中系数C_0，C_1，$C_2\cdots\cdots C_{n-1}$，C_n就是一个数组。

在齿轮计算中常会遇到摆线公式

$$\begin{cases} x=t-\sin t \\ y=1-\cos t \end{cases}$$

当 t 取一组不同的数值时，t_1，t_2，t_3，$\cdots\cdots t_{n-1}$，t_n，

则对应有一组 x_1，x_2，x_3，$\cdots\cdots x_{n-1}$，x_n

对应有一组 y_1，y_2，$y_3\cdots\cdots y_{n-1}$，y_n，

那么我们把 t 看成一个数组 x 看成一个数组，y 看成一个数组。

对于一个线性方程组

$$a_{11}x_1+a_{12}x_2+\cdots\cdots a_{1n}x_n=b_1$$
$$a_{21}x_1+a_{22}x_2+\cdots\cdots a_{2n}x_n=b_2$$
$$\cdots\cdots\cdots\cdots$$
$$\cdots\cdots\cdots\cdots$$
$$a_{n1}x_1+a_{n2}x_2+\cdots\cdots a_{nn}x_n=b_n$$

我们把它的解（x_1，$x_2\cdots\cdots x_n$）作为一个数组

把常数项（b_1，$b_2\cdots\cdots b_n$）作为一个数组

系数 a_{11}，$a_{12}\cdots\cdots a_{1n}$

a_{21}，a_{22}……a_{2n}

……

……

……

a_{n1}，a_{n2}……a_{nn}

也作为一个数组。

使用数组概念之后，在书写与表达形式上带来很大的方便，例如对于100阶的线性方程组要写出10200个变量，既烦琐工作量也大。使用数组概念后，我们并不需要对其中的每一个变量取名而是对整个数组命名。例如解的数组为数组x，

常数项的数组为　B

系数的数组为　　A

而对数组中的每一个变量，通过标明它在数组中的位置来表示，即用下标值来表示。例如X_5表示数组x中的第5个变量。b_4表示数组B中的第4个变量。$a_{8,9}$表示数组A中第8行，第9列的变量在书写时其一般形式为x_i，b_i，a_{ij}。

由于数组中每一个变量都是通过下标来区分的，所以叫下标变量。

如果一个数组中的每一个变量（又称分量）用一个下标值就能表示，如数组x，B中的分量，则称为一维数组，如果需要2个下标值才能表示的，如数组A中的分量称为二维数组。同理，若用n个下标值才能表示的称为n维数组。

数组中的下标变量（即分量）在算法语言中的表示方式原则下与数学中的一致，由于在语言中必须用其基本符号，同时不允许产生二义性，故作了一些规定。

数组的命名用标识符。

下标值的书写，其基本符号中没有小1，2……9的符号，故改用基本符号中的数字0，1……9。如果把数字直接写在数组标识符的后面，例如X_5此时产生二义性，既可理解为X数中的第5个分量又可理解为标识符X_5，因此用加基本符号中的方括号来解决即x〔5〕表示数组x中的第5个分量，而X_5表示标识符。

对于多维数组，例如数组A中的下标变量a_{89}若写成A〔89〕则又产生二义性，即可理解为二维数组中的第8行第9列的分量，又可理解为一维数组A中的第89个分量。因此用基本符号逗号（，）加以隔开即$a_{8,9}$应写成A〔8,9〕。

由于数组中下标变量是按一定顺序一个一个排列的，而下标值是指明所在的位置，所以下标值必须是整数。这里需要说明的下标值除了用整数外，还可以为表达式。

例如　A〔X＋6，X＋8〕

当　X＝1时　则为A〔7，9〕

当　X＝5时　则为A〔11，13〕

有关表达式的概念以后介绍

如果表达式计算出的数值不为整数，则用4舍5入的原则取整数（此在机器内自动进行）。当我们把整数作为表达式的特殊情况则对于数组中的下标变量其表示形式为

标识符〔E_1，E_2……E_K〕（为K维数组）

其中E_1，E_2……E_K为表达式。

我们通常把方括号内的内容称为下标表，则下标变量的表示形式为

标识符〔下标表〕

对于每个下标变量是相当于一个简单变量，在运算与语言中的作用与一般简单变量是相同的，仅仅书写方式不同。

下标变量与简单变量一样，在源程序中都要加以说明，以便于计算机分配单元。由于采用数组，在说明时并不需要把所有的下标变量逐个加以说明，而是对数组整体加以说明。这对编写源程序带来很大的方便。

由于数组标识符本身不能说明是简单变量还是数组，因此在数组标识符前面必须冠以数组说明符‘ARRAY’。再对数组加以类型说明。在下标表内每个下标值取值范围用上下界（或称界偶）代替，上、下界用冒号隔开。一维的数组用一对上下界，n维数组用n对上下界，其一般形式为〈类型说明〉‘ARRAY’〈标识符〉〔下界：上界，……下界：上界〕

例如：

‘INTEG’ ‘ARRAY’ A〔1:10, 2:15〕;

此表示数组A内的下标变量皆为整型量，是二维数组，它的下标变量为：

A〔1,2〕, A〔1,3〕……A〔1,15〕,

A〔2,2〕, A〔2,3〕……A〔2,15〕,

……

A〔10,2〕, A〔10,3〕……A〔10,15〕,

即相当于数组a_{ij} i=1,2……10, j=2,3……15。

在数组说明中作如下的规定

(i) 上下界必须是整数，上界必须大于下界。

即A〔10:1, 2:8〕, B〔1.5:8〕等都是不正确的。

在数组说明中的上下界可以是表达式，当表达式所计算出的数值不为整数时，则4舍5入取整数，例如当计算出的结果为A〔1.5:8.4〕则机器自动地取A〔2:8〕。

在规定中只要求上界的数值大于下界的数值，而上下界仅表示下标值取值的个数，即要求（上界－下界＋1）这个整数值需要保持恒定，因此上下界的起讫可以“浮动”。

例如有一些数组为3；5；7；2；9；若用B为其标识符则在说明中可写成：

‘INTEG’ ‘ARRAY’ B〔1:5〕也可写成

‘INTEG’ ‘ARRAY’ B〔－2:2〕或

‘INTEG’ ‘ARRAY’ B〔15:19〕

但当数组说明确定之后，在书写下标变量的下标值时就必须与说明中的上下界对应。

当上例数组B的上下界为B〔－2:2〕时，则相应的下标变量应为B〔－2〕, B〔－1〕, B〔0〕, B〔1〕, B〔2〕。

当上下界定为B〔15:19〕时，则相应的下标变量为

B〔15〕, B〔16〕, B〔17〕, B〔18〕, B〔19〕。

(ii) 对于下标变量为实型数时，说明符‘REAL’可以不加，对布尔型则加‘BOOTE’

(iii) 同类型的几个数组，即维数相同，上下界相同，可以作一次说明。

例如：E〔1:8〕，D〔1:8〕，A〔1:10，1:15〕，B〔1:10，1:15〕，C〔1:10，1:15〕则在数组说明时可以写成

'INTEG' 'ARRAY' E，D〔1:8〕，A，B，C〔1:10，1:15〕；

在这种说明中，每个数组之间必须加逗号，在每类型的数组说明的结尾要加分号（；）结束。

在源程序中数组被说明后，机器在编译时按不同类型的数在内存分配单元。对于数组其下标变量的排列次序是先按最后一个下标的变化范围依次排列，然后再按前一个下标排列，因此，我们抄写数组时，也应如此排列。

（iV）鼓数组说明：

当数据太多，内存紧张时可以把一部分数组存入磁鼓，需要时再调用，这样的数组称为鼓数组。编译系统对鼓数组在鼓上分部单元，程序中用相应的记鼓，调鼓语句进行记入与调用。

鼓数组说明不写类型说明符，而加上'DRUM'以和内存数组相区别。即写成

'DRUM' 'ARRAY' A〔1:100，1:50〕；

鼓数组的类型决定于相应的与之交换的内存数组，其界偶应和相应的内存数组相同。

121机算法语言规定不允许使用'OWN'（固有）数组。

（三）标准函数符

在科技的计算中，经常遇到对初等函数的计算，为了使用的方便，在语言中备有一些标准函数，使用时只要写出它的名字与自变量就可以了，对于这些函数的名字（即标识符）使用时无需（也不允许）在程序中预先说明，这些函数叫标准函数。

121机算法语言可供应用的标准函数有14个

ABS（E） 求E的绝对值 （E为算术表达式，以下同）

SIGN(E) 求E的符号 $\mathrm{SIGN}(E)=\begin{cases}1 & \text{若 } E>0\\ 0 & \text{若 } E=0\\ -1 & \text{若 } E<0\end{cases}$

GN2（E） 求平方根 即$\sqrt{E}$

GN3（E） 求立方根 即$\sqrt{E}$

SIN（E） 正弦运算

COS(E) 余弦运算

TAN(E) 正切运算

ARCSIN（E）反正弦运算

ARCTAN(E)反正切运算

LN（E） 求E的自然对数 $\ln E$

EXP(E) 求指数函数 e^E

TOINTG(E)求与E最接近的整数，即对E 4舍5入取整数。

ENTIER(E) 求不大于E的最大整数

TOREAL(E) 把E化为浮点数

在机器中可以根据实际需要进行增加，在这里需要注意几点：

(i) 标准函数也是用标识符表示，在机器内已作固定的函数运算使用，此种称为标准函数符（又称函数命名符）这里要注意应包括园括号与自变量E，为了不产生二义性，规定标准函数符不能表示别的变量，而是某特定的变量，它与一般的变量一样参加运算。

(ii) 虽然E可以是任意的算术表达式，121机算法语言规定不允许出现标准函数自身相套的情况，即SIN (SIN(E)) 是不允许的。

为了下面叙述方便，先介绍121机的几个常用的标准过程

READR	从数据区读入一个实型数
READI	从数据区读入一个整型数
READB	从数据区读入一个布尔量
READ (A)	从数据区顺序读入该数组的全部元素
RRINT (E)	打印表达式E的值
APRINT(A)	顺序打印数组A的全部元素的值

第4节 源程序的结构

用算法语言编写的程序称为源程序，我们举一个例子来说明源程序的结构。

考察自由落体运动，设一物体自由向下落的过程中，这物体到地面的距离为H，与时间T的关系为

$$H = 19.6 - \frac{1}{2}GT^2$$

已知T值求H的值。

在进行计算时，一般总先看G，T是什么数，然后按表达式$19.6-\frac{1}{2}GT^2$计算出H的值。并记录下来。在用电子计算机计算时，我们编程序时也按这个思路进行。即应该有这样几步。

(i) 告诉机器G，T的值是多少，写成

G:=READR, T:=READR

这里READ表示输入的意思。

(ii) 进行求H的运算写成

H:=19.6-G×T×T/2

(iii) 将计算结果输出写成

PRINT (H)

(iV) 还要对机器说明G，T，H都是实型量，写成

'REAL' G, T, H;

以便机器分配内存单元，因此这部分应写在程序的开头部分。

另外我们用语句括号 'BEGIN' 与 'END' 将整个程序括起来，并在最开头加Y表示为程序以便与数据区别，则计算程序写成：

Y

```
'BEGIN'  'REAL' G, T, H;  (说明部分)
        G:=READR;          T:=READR;   } 语句部分
        H:=19.6G×T×T/2;
            PRINT (H) ;
'END'       ×××××
```

从这程序中可以看出，用算法语言编写的程序不管怎样复杂，它总是由说明部分和语句部分组成，在每个说明与各语句之间都用分号隔开，前后用'BECIN'与'END'括起来。

说明部分起介绍作用，使编译系统了解所用的量具有什么性质，以便对不同的量作不同的处理，分配相应的存贮单元。

语句部分则是程序的主体，在语言程序中是完成计算过程的，它由具有各种功能的语句，如输入、输出语句，条件语句，赋值语句……等组成。对这些语句在第二章介绍。

在算法语言中还会遇到分程序与复合语句的概念。

所谓分程序，就是由一串语句在其前面带有说明，它们一起包含在开括号'BEGIN'和语句闭括号'END'之中，成为一个整体，它在程序中可以作为一个语句使用

所谓复合语句就是不带说明部分的一串语句用'BEGIN'与'END'括起来，成为一个整体。作为一个语句。

当引进分程序概念之后，使编制程序带来很多的方便，使其成为嵌套结构，这些在以后还要介绍则不再举例。

第二章　基本语句

第1节　算术表达式与赋值语句

（一）算术表达式：

科技的计算中，总要遇到按一定规则运算的公式，例如：

$$H = 19.6 - \frac{1}{2}GT^2$$

$$X = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$$

我们称这种按照一定规则进行运算的公式叫表达式。

表达式可以分为进行算术运算的算术表达式（如上两例）和进行逻辑运算的布尔表达式。在这里先只讨论算术表达式，布尔表达式到有关的语句中再去讨论。

（1）算术运算符及其优先级

一个算术表达式都由变量与算术运算符（+、-、×、/、//、××）所构成，这里讨论在编写源程序时有那些规定与限制。

+和-与通常数学中的加和减一样，只是当参加运算的数皆为整型数时其结果才为整型数，否规为实型数。

×即数学中的相乘，在算法语言中规定乘号不能省略，即5×B不能写成5 B。

在运算时只有当参加运算的量都为整型量时，其结果才为整型量。

／为数学中的（÷），在运算时不管参加运算的量是整型还是实型其结果皆为实型量。

//在数学中无此符号，此为整除运算，即不管参加运算的量是整型还是实型，皆作浮点除，其结果是4舍5入取整数。

××是数学中的乘幂记号，在算法语言中没有a^b的书写形式，对于a^b在算法语言中定写成a××b，乘方的定义规则如下：

以i表示整型数，以r表示实型数，以a表示整型或实型数，对于a××i而言，若i＞0它是a×a×……×a（i次）

若i＝0	且a≠0	a××i＝1
i＝0	a＝0	a××i无意义
i＜0	a≠0	它是1/（a×a×……×a）（i次）
i＜0	a＝0	a××i无意义

对于a××r而言：

若 a＞0　　a××r＝EXP（R×LN（A））（即$e^{r\ln a}$）

a＝0　且 r≤0　　它没有意义

a＝0　　r＞0　　它是0

对于A××B，在运算时只有当A与B皆为整型量时其结果才为整型量。

（ ）此为园括号和数学中的园括号使用相同，但在算法语言只有园括号一种（方括号已作它用，无花括号{ }）全部用园括号即园括号套园括号，在计算时规定从最内层先计算，在使用时与数学中一样，园括号必须成对使用，在书写时应当注意。

在算法语言中运算符的优先级与数学中的一样。

第一：××

第二：×，/，//

第三：+，-

在同一级运算符中是从左到右，凡有园括号时，园括号内的先做。

在算法语言中有一条重要的规定，算术运算符间不允许直接相联接，如果必须相联时须用园括号分开，例如A××-B是不允许的应写成A××（-B），A/-B也是不允许的，应写成A/（-B）。

在书写除法时要注意适当地加园括号，否则引起错误。

例如$\frac{a+b}{c}$与$\frac{a}{c+d}$应写成（A+B）/C与A/（c+D）

在编写源程序时应按上述要求计算公式写成表达式

例如：

$3\ (a^3+c_2)$　　应写成 3×（A××3+C〔2〕）

$a\sin\ (bt+e_{ij})$　　应写成A×SIN（B×T+E〔I，J〕）

$\frac{-b+\sqrt{b^2-4ac}}{2a}$　　应写成（-B+GN2（B×B-4×A×C））/（2×A）

$\sqrt[3]{a+\frac{b}{e^{\frac{1}{100}}}}$　　应写成 GN3（A+EXP（B/100））

算术表达式分简单算术表达式与条件算术表达式（在条件语句中介绍）简单算术表达式就是由数、变量、和函数符用算术运算符以及园括号联接而成的式子。这里需注意的在算法语言中规定任何一个算术表达式当它用园括号括起来之后，也就是一个简单算术表达式。而常数与任何一个其他运算分量作为表达式的特例。

（二）赋值语句

用算法语言编制程序时，其运算部分是通过一系列语句组成的。语句的排列形式是……语句；语句；……语句；在语句与语句之间用分号（；）隔开。一个大型题目的计算有时会用到上百条的语句。虽然语句很多，但都是由于一些基本语句；赋值语句、转向语句、条件语句、循环语句、输入输出语句，过程语句等组成。只要掌握了这些基本语句的使用，就能方便地用算法语言来进行程序设计。

算法语言的一个重要特点，就是十分接近人们已习惯了的数学语言，在科技中常常遇到大量的计算，这些计算是通过数学公式的形式出现的。

例如计算圆台的体积其计算公式为

$$V=\frac{1}{3}\pi(r_1^2+r_1r_2+r_2^2)h$$

在计算自由落体运动时要用到

$$H=h_1-\frac{1}{2}gt^2 \quad (h_1\text{为起始高度})$$

在算法语言中是用赋值语句来执行表达式的求值过程，求计算V，H值时写成赋值语句为

V:=3.1416×H (R〔1〕××2+R〔1〕×R〔2〕+R〔2〕××2) /3;

H:=H〔1〕-G×T×T/2;

赋值语句就是把一个表达式的当前值赋给一个变量或多个变量的标识符的语句。其形式为

V1:=V2:=V3……:=VK;

在书写形式上看，只是把数学公式中的等号换成了赋值号（:=）在形式上与数学公式很相近，但是它们间却有重大的差别，赋值号（:=）与等号（=）是完全不同的概念，等号是表示等号两边的量相等。而赋值号是表示把赋值号右边的表达式计算的结果赋给左边的变量，实际上赋值号是一种操作而不是一个简单的符号。同时赋值号是有方向性的，是把赋值号右边的内容赋给左边的变量。

例如：X:=X+1;

在数学中是不能有X=X+1的，而赋值语句中却常常会遇到X:=X+1，这表示把X的当前值加1后再赋给X若X原来是5（即当前值）当机器执行完X:=X+1，的赋值语句后，则把5+1=6的值赋给X。

又如X:=y，与y:=X；是两个完全不同的含义。X:=Y，表示把Y的值赋给X，而Y=X；则为把X的值赋给Y。

赋值语句的一般形式为：

变量:=表达式;

在书写赋值语句时我们要注意以下几点：

(i) 赋值号的左边只能是变量（简单变量与下标量）不能是表达式。

例如-S:=5，与A+B:=X-Y；都是错误的。

(ii) 赋值号（:=）左边的变量与右边表达式当前值的类型不一定要求相同，如果类型不同，则机器会自动地把表达式当前值转换成左边变量所说明的类型，然后再赋给左边的变量。当实型转换成整型时是4舍5入取整数。

例如：当N为整型量

N:=3.2; 实际结果N的值为5

N:=5.6; ” ” N的值为6

(iii) 在执行赋值语句时，凡是参加运算的各个运算分量当前值必须已有确定值。

(iv) 在赋值语句中允许多重赋值，即：

A:=B:=C〔1〕:=D〔4,8〕:=E; （表达式）

其中A，B，C〔1〕，D〔4，8〕是同类型量，若不为同类型量则分开来写。