基础工程结构分析及程序

J.E.鲍尔斯 著

中国铁道出版社

内 容 简 介

本书内容比较新颖，提供了各种类型基础工程结构的分析计算方法及计算机程序，对工程实践及教学都有重要参考价值。

本书的计算机程序，使用的是FORTRAN Ⅳ语言。

读者对象：土建工程技术人员，大专院校师生。

JOSEPH E.BOWLES
Analytical and Computer Methods in Foundation Engineering
McGRAW-HILL BOOK COMPANY 1974

基础工程结构分析及程序

J.E.鲍尔斯著
胡人礼
陈太平 译
林亚超
邹守简 校

中国铁道出版社出版
责任编辑 陈保兴
新华书店北京发行所发行
各地新华书店经售
中国铁道出版社印刷厂印

开本：787×1092 1/16 印张：21.5 字数：421千
1982年2月第1版 1982年2月第1次印刷
印数：0001—5,500册 定价：2.65元

目　录

第一章　采用FORTRAN IV语言的计算机程序设计

1—1　编写计算机程序简述

计算机程序是一种通过一系列指令使计算机完成所要求的一组运算的方法。本书中的讨论限于诸如乘、除、自乘幂、求根以及求三角函数等各种数学运算。

由于假设读者已学过计算机程序设计，所以本章只作为计算机方法的简要参考资料，至于更完整的讨论和其它技巧见Bradley（1969）和IBM（1968）。

1—2　可采用的计算变量类型

在算术运算中，计算机采用两种类型的数和变量标识符。变量通常总是用字母-数字来标识。

1．定点数或整数，这种类型的数不带小数点，如1、5、6、8、200。

2．浮点数或带有小数点的数，如1.00、1.41416、3.1416、500.、500.00、500.001。

字母-数字的标识符是以字母作为字首后面带有数字或不带数字来标识变量。如

A	SLOPE2	STRESS	INERTA
K	CAT	$MIN	MODEL
ABLE	PHI1	MOM	G4

定点变量通常采用下面诸字母或用它们作为字首，由计算机自动标识：

I、J、K、L、M、N

因而，前段中的K、MOM、INERTA、MODEL均为定点变量。

浮点变量则采用其它字母和货币符号$，因而

A、ABLE、CAT、STRESS、$MIN

则为浮点变量。从书写变量的方式看出，在变量的使用上，编制程序人员有很大的活动余地，因而，要力求应用能够进行区别的变量名。第1—5节提出了有关六个字母-数字字符的变量的长度限制。

定点数与浮点数可作为程序规定指令的一部分进行互换（见第1—12节）。

一般不应在任何给定的计算中混淆了定点变量和浮点变量，虽然有些计算机容许这样做。将一浮点运算（或数）或一定点计算表示为一整型变量时总是引起小数位截除成最近整数。

1—3　单精度和双精度

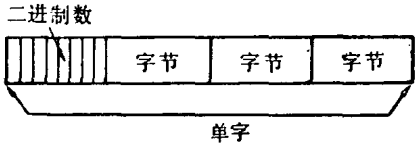
通常IBM计算机按单精度进行算术运算。单精度计算应用七位数值（如果不是七位，则用零来填充或截尾）加上正负号和小数点的位置，如

−9998887.　　或　　.0000123

规定的双精度计算允许应用16位数值，带有正负号和小数点，例如

−56789.83542155774

单精度数为两个“半单字”组成的单字，一个“半单字”为两个字节，一个字节由8个二进制数组成。字节用来描述计算机的内存容量，这种容量以2^{10}个字节计。为了简化讨论，计算机内存的能力以K来描述，其中$K=2^{10}$字节。计算机的内存大小可为



$$16K=16(1,024)=16,384字节$$

和

$$128K=131,072字节$$

一定数量的字节分配给计算机的簿记存贮。具有128K内存的360/40系统，簿记存贮约为14K。因此，在此系统中，对于所有实际问题，程序员在具有这种内存能力的计算机中则可用$114K/4\approx29,000$单字（或七位数）。应用双精度数很快地消耗计算机内存，因为按全部双精度，容量约为14500个单字。不应主观地认为双精度提供较高的计算精确度。实际上，大小为60×60的矩阵在IBM360上以单精度进行求逆时，精度损失可以很小；一个90×90的矩阵以单精度求逆可以达到输入数据的精度。

1—4 加、减、乘、除（+、−、*、/）

研究A、B、C三个变量。为了使A加B，可按下面用计算机语言（FORTRAN Ⅳ）写出其和为一个新的变量D：

$$D=A+B$$

为了使A减C和D减C，引进新的变量F和G：

$$F=A-C$$

$$G=D-C$$

为了求A乘B并以其积 $AB=H$ 去乘F，则

$$H=A*B$$

$$P=A*B*F$$

或

$$P=H*F$$

显然，在后一种情况，如果在$P=H*F$之前不给出$H=A*B$，那么会得出错误信息，因为直到相乘运算进行之前H没有被赋值。

为了用P除F，用

$$Q=F/P \quad 或 \quad Q=(A-C)/P$$

而不是$Q=A-C/P$，因为这样会成为A减去C/P值，这不是所想求的。

还应注意，如果$A=20.$，$C=4.$和$P=2.$，

$$(A-C)/P=(20.-4.)/2.=8.0$$

因而

$$(A-C)/P+1.0=9.0$$

但是

$$(A-C)/(P+1.0)=5.333$$

1—5 变量长度、下标变量

除去标识符外变量长度不允许多于六个字母-数字字符。

如

AA3	3个字符长度
ALINE	5个字符长度
ASLOPE	6个字符（最大长度）
ASLOP1	6个字符（最大长度）
ASLOPE(I,J)	6个字符（最大长度）
ELASTIC	错误的（7个字符长度）

变量可以注以下标，以使它们可为后面计算应用存储起来，或写成特殊的输出形式（FORMAT）。所有注有下标的变量必须经过DIMENSION（数组说明），并应放在程序的开始处（见第1—9节）。数组说明规格（语句）可以大于或等于下标的最大值，但决不能小于它。数组说明语句是一种对计算机内存预约或分配数组说明中所指定的单元个数的指令。变量可注以一个下标、两个下标或多个下标。例如

A(I)　　一个下标：A(20)预约20个单元

A(I,J)　两个下标：A(20,20)预约400个单元

A(I,J,K) 三个下标：A(2,8,12)预约192个单元

预约的单元均按单精度考虑。如果A(20)为双精度变数，那么实际上要分配40个单元。

下标计数符（I，J），或用任何其它变量，必须规定为定点者。

1—6 指数和根

为了使一变量形成任何次幂（自然，在IBM360上，计算机限制近似地为±75），研究如下：

已知：A、B、C

求　：$D=A^2$　$F=AB^3$　$G=A(BC)^4$　$D=(ABC)^2$

解　：D=A**2　　G=A*(B*C)**4

　　F=A*B**3　D=(A*B*C)**2

但是要注意A*B*C**2 = ABC^2。

为了得出$\sqrt{A}$、$\sqrt[3]{C}$、$\sqrt[4]{D}$和$\sqrt{A^2+B^2}$，可以引进变量，求解如下

G=A**.5	$\sqrt{A}$
H=C**.333	$\sqrt[3]{C}$
P=D**.25	$\sqrt[4]{D}$
Q=(A**2+B**2)**.5	$\sqrt{A^2+B^2}$

对于第一个和最后的开方根，可以应用计算机软件系统配备的子程序：

```
G=SQRT(A)
Q=SQRT(A**2+B**2)
```

或

```
Q=DSQRT(A**2+B**2)
```

这里DSQRT中的字头“D”是指双精度。对这种特定子程序，自变量A^2+B^2不可能是负的，但是其开方可能会是负的，例如，在求解一般形式的二次方程根时就会这样：

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

如果平方根为负的或为复数，那么可以采用两种别的子程序（CSQRT和DCSQRT）。

在求幂时，可以应用定点或浮点指数值。而在求根时，所有运算需要应用浮点指数值。

为了对 e 自乘到任何指数，或求数字的对数，可以采用计算机子程序：

e^x = EXP(X)　　或用双精度DEXP(X)

logX = ALOG10(X)　　或DLOG10(X)

lnX = ALOG(X)　　或DLOG(X)

1—7 三角函数和双曲函数

三角函数可以应用计算机程序系统中的下列子程序得到：

函数	计算机单精度子程序	函数	计算机单精度子程序
正弦X	SIN (X)	反余弦C	ARCOS (C)
余弦Y	COS (Y)	反正切D	ATAN (D)
正切Z	TAN (Z)	双曲正弦F	SINH (F)
余切A	COTAN(A)	双曲余弦G	COSH (G)
反正弦B	ARSIN (B)	双曲正切P	TANH (P)

对于双精度，上面子程序名的前面加一个字母*D*，例如，DSIN(X)，DTANH(P)。

三角函数子程序中所用自变量必须以弧度计，反函数所计算得到的角也以弧度计。一角度按下面换算成弧度：

$$\text{以弧度计的角} = \frac{\text{以度数计的角}}{57.2957795131}$$

$$1\text{ 弧度} = \frac{180}{3.1415926535}$$

1—8 正负号、求一数组中的最大值或最小值

正负号可藉引进一个负号或者乘以一个−1.0来得到变换。

如果需要比较（或检查）一个值，而且事先不知道它的正负号，那么可以采用绝对值如下

*ABS(arg)　　或　　DABS(arg)

取决于采用单精度或双精度（DABS）。在双精度时，自变量与进行比较的数必须为同样的精度。应用此子程序有效地将用作自变量的任一值或一个数组变换成正号。

为了求出一群数中或一数组中变量的最大值或最小值，还可以应用另外的计算机子程序。X、Y、Z、W、U、V诸值中的最大值为

AMAXO(I,J,K,L,M,M1)

AMAX1(−X,Y,−Z,W,A,…)**

MAXO(…,F,G)

MAX1(…,A,B,C,…)

DMAX1(A1,B1,C2,F2,…)

当包括正负号时，对于绝对最大值采用 AMAX，当不考虑正负号时，采用 MAX。另外，

---

*ABS(arg)和DABS(arg)中的arg为argument（自变量）的缩写。——译者注

**见第 5 章中的计算机程序。

零（0）或（1）区别变量为定点还是浮点（AMAX1 和 MAX1 均为浮点）。也可以另写一系列语句来代替这些子程序。

一群数中或一数组中的最小值可用下面子程序得到

AMIN0(N,N1,N2, NN)
AMIN1(X,Y,Z,A)
MIN0(N,N1,N2,……)
MIN1(A,B,C,……)
DMIN1 (A,B,C,……)

这里A、0 和 1 分别为绝对最小、定点或浮点。在最大和最小值的子程序中，字头 D 为双精度，如果自变量不是双精度，那么将显示出错误信息。

1—9 计算机技术

本节归纳了作者认为写计算程序时容易接触到的许多技术中的一小部分。

参见第 5 章的程序编排，可以说任何程序由下面几部分组成：

a、说明语句，按下面一定的次序编排（注意，在实际的程序中，下面五种语句可能不全都需要）：

```
1   DIMENSION TITLE(20), X(40, 2) , Y(40, 40) , E(30, 30)
2   COMMON  D,W
3   EQUIVALENCE (E(1,1) , A(1,1) )
4   IMPLICIT INTEGER(A,BX) (见第1-12节)
    IMPLICIT REAL*4(N2, LK)
    IMPLICIT REAL*8(N3, LKK)
    INTEGER C,CX
5   DOUBLE PRECISION X,Y
```

b、运算语句（程序），其中嵌入下列语句：

```
1   READ(1,200) K,Z1,Z2,D (输入语句)
2   WRITE(3,1008)Z1,Z2,D(输出语句)
3   FORMAT语句（输入或输出的规格）
```

READ语句容许按下面所示规格输入计算数据

```
200  FORMAT(I5,3F10.4)
```

WRITE语句可按下面所示规格输出所标识的变量（例如Z1、Z2和D）

```
1008  FORMAT('1',/, T5, 'Z1=' F6.2, 5X, 'Z2=F6.3, 5X,'D= ' F8.2,
      'LB', /)
```

输出要便于解释是非常重要的。撇号在输出中是非常有用的工具。左边括号处，如果撇号围住一个 1（即'1'），那么开始新的一页；如果围住一个零（即'0'），则跳过两行；若留下一个空白（即' '）便跳过一行。括号内别处的撇号围住打印机要打印出的任何东西。如果一个字中包含一个撇号，例如don't，那么写成'DON"T'，亦即采用内部双撇号。具体地说，参见刚才所给运算语句中输入/输出（I/O）的例子，那么语句

```
200  FORMAT(I5,3F10.4)
```

说明K为一个置放于数据卡片中前五个字格内靠右对齐的定点数。一位数采用第五个字格，

两位数采用第四和第五个字格，等等。在后面30个卡片空格中每10个格用于浮点变量Z1、Z2、D中的一个（不需要用小数点）。但是，如果不包括小数点，右边四个空格由计算机自动地填上，以满足F10.4字段规定。

现在假设

Z1=200.02

Z2=25.0

D=3255.6

输出规格

```
1008  FORMAT('1', /, T5, 'Z1=  ', ···)
```

说明如下：

'1'，建立新的一页。

/，为了打印第一行，将纸从该页顶上提前两个空行。

T5，在第五列处打印机开始打出Z1=，它的一般形式为T_w。

'Z1= 'F6.2使之打印出：Z1=200.02。

5X，跳过五个空格，在同一行上开始下一个打印，它的一般形式为wX。注意，在下面格式中计算机将加上一些零。

'Z2= 'F6.2打印出：Z2=25.00，示于与Z1相同的行上。

5X，跳过五个空格。

'D= 'F8.2, ' LB'，打印出"D=3255.60LB"。注意，' LB'内的跳格也打印出来。

/为了打印下一输出将纸提前两个空行。

诸逗号按次序将各个指令和信息块分隔开。

包括H（或Hollerith字段规格）和T规格详细描述的其它方法，可以用来将输出变成任何形式。这里所示T_w和wX连同撇号的应用，将可处理大多数输出格式。由于这个原因就不需要详细说明另一种字段规格。现在考虑

```
      WRITE(3,XYZ)X
XYZ   FORMAT('1', T5, 'THE X VALUES ARE', /, T5, F10.4)
```

括号里边第一个内容仍为'1'，后面跟着一个逗号。由于开头要从左边缘起第五个打印空格开始，所以我们用T5。其中/将纸提前两个空行，F10.4为需要输出的X的最大尺寸，一行中一个数，起始于距边缘五个空格，而结束于第十五列。另一种输出格式是早已开始了一页，这种输出格式如下。如果我们想在第二行上输出下列带有有关单位的基础尺寸和弹性模量：

长度=EL

宽度=B

弹性模量=EC

可以写成如下：

```
      WRITE(3,104)EL, B,EC
104 FORMAT(/, T5, 'FTG LENGTH=  ', F6.2,'FT', 5X, 'FTG
    WIDTH=  ',F6.2,'  FT', /, T5, 'MODULUS OF ELAS=
    ',F8.2, '  KSF', /)
```

解释如下：

//，跳过两行。

T5，在第五列开始“FTG”的F。

‘FTG LENGTH= ’被打印出来，因为它括在撇号内。

F6.2为基础长度最多占六个格，可写成带有两位小数（999.99），它不包括正负号，因为基础长度不可能是负的。

‘FT’，跳过两撇号内所括的空格写出单位。

5X，跳过五个空格开始下一输出信息。

‘FTG WIDTH= ’，F6.2，打印出最多为六个字格的数。

F6.2如上面所述，其单位为‘ FT’。

/，打印机向下走一行。

T5，在第5列中开始写出‘MODULUS OF ELAS= ’。

F8.2，采用具有两位小数的8位最大数，因为E通常大于基础长度。

‘KSF’为E的单位，在KSF中K的前面留有一空格，以使数码与单位之间借一空格来分开。

//）在要打印出下一数据之前将打印机向下走两行。

有时，特别是如果数字字段的大小事先不知道时，最好按E－FORMAT规定输出，举一个例子

×××××　FORMAT(……，aEw.d,……)

这里a＝代表希望输出重复次数的整数。

E＝带有指数的单精度输出说明。

w＝说明所需总格数的整数。

.d＝为小数（或分数）所保留的格数。

值	FORMAT	打印出的值
-0.004	E10.3	-0.400E-02
250.2	E10.2	0.25E+03

1—10 数据卡片

注解卡片（COMMENT卡片）

将C置于第1列内，任何想要的字母-数字信息置于其余79个空格中，可以采用连续卡片，但是在每一卡片的第1列中置放一个C。

说明卡片（SPECIFICATION卡片）

它们从第7列开始，可以应用到第72列。

运算、输入、输出、格式

列	信息
1—5	语句标号
6	除非延续外，此列为空白。如果延续，则在该列置放任何字母符号或数字符号，但是此时该格必须填充。
7—72	语句
73—80	数据处理标识符（如果要求的话）。

数据卡片（输入或输出）

空格 1 至80可按格式说明来使用，例如3I5采用前15个字格。

2I5，6F10.5采用70个字格，带有八个数据项，其中前两个数据项为定点。

8F10.4采用所有80个字格，带有八个数据项，每一项为10个字格宽。

1—11 字母-数字数据

有时要求将字母-数字数据读入（或写进）程序中。第 7 章中所示计算机程序就有这种情况。

这可应用A格式说明来做到，该格式说明的一般形式为

$$A_w$$

式中w 为表示所需处理字符个数的正整数（4 个或 8 个）。如果知道需要多少个字符（譬如需要读入和写出下列标识K

$$K = 4X + 6Y$$

它包括空白、=、+等，为11个字符），那么

```
      READ (1, 1000)TITLE
1000  FORMAT (3A4)
      WRITE (3, 1001)TITLE
1001  FORMAT (3A4)
```

本书所列程序中有一些程序应用了FORMAT(20A4)，它为字母-数字数据占用了一张数据卡片，因为它占用了 4 ×20＝80个字符格。

一个DIMENSION语句必须配合此种输出模式，否则输出时数据中将只保存一个字符，此语句可为

```
      DIMENSION TITLE(20)
```

自然，如果20没有预定足够的空格，那么不是所有字母-数字得到存储。

1—12 定点转换为浮点

当希望将某些正常情况专用于定点变量的字母段（I至N）作为浮点变量（譬如 M2，N）时，可以应用：

```
REAL*4 M2,N 4是转换为单精度
REAL*8 M2,N 8是转换为双精度
```

或

```
DOUBLE PRECISION M2,N
```

由此可见，DOUBLE PRECISION 和 REAL*8功用相同，并且可以互换。

为了将以M、J和L等为开头的任何变量转换成浮点变量，可以采用

```
IMPLICIT REAL*4(M,J,L)
```

作为一个例子，如果变量MCOL,J,LCM用于程序中，那么它们将按单精度浮点变量来应用。如果采用了REAL*8，那么则得出双精度。

浮点转换为定点

为了将部分字母（不是I至N）转换为定点，可以采用

IMPLICIT INTEGER(A,G)

它将以A或G开始的任何变量（如A，AC，ABLE，G1，GMX，G）转换为定点。为了将字母串（譬如A至G）转换为定点，那么采用

IMPLICIT INTEGER(A-G)

为了只转换某些变量，采用

INTEGER A,B,RAT

这只将以A、B和RAT所标识的变量转换为定点。

1—13 IF 语句（如果语句）

分为两类，即逻辑IF语句和计算IF语句。

逻辑IF语句

IF(A.GE.B)X＝Y表示如果A≥B，则X＝Y

IF(A.LT.B)L＝4表示如果A＜B，则L＝4

IF(A.EQ.C)W＝0.0表示如果A＝C，则W＝0

IF(A.NE.B)GO TO 7表示如果A≠B，则转向执行标号7的语句

IF(A.LE.D)W1＝W2表示如果A≤D，则W1＝W2，

IF(A.GT.D)Z＝6.表示如果A＞D，则Z＝6。

也可以采用其它程序语句，例如

READ(1,XYZ) 或WRITE(3, XXX)

代替前面例子中的Z＝6.，L＝4，X＝Y.

计算IF语句

它们的形式为

IF(A-B)67, 68, 69

67 DO……

68 Z＝K＋1

69 AA＝B＋C/D

它们表示，如果A-B为（见右）

正负号	执行
-	GO TO 67
0	GO TO 68
+	GO TO 69

跟在此IF语句后面的第一个语句必须具有这三个控制转移标号（或一语句标号）中的一个，以避免出现错误信息。

1—14 GO TO 语句（转向语句）

分为两种，即程序GO TO语句和计算GO TO语句

程序GO TO语句

当碰到像

GO TO 60

的程序语句时，它将控制转移到标号为60的语句。注意，如果标号60的语句放在程序中此语句之前，那么它会引起计算机无限地循环。

计算GO TO语句

像下面

GO TO (61, 62, 63, ……) , LL

的计算GO TO语句将控制转到下列标号语句

61 当LL= 1 时

62 当LL= 2 时

63 当LL= 3 时

……

显然，必须先对以LL命名的变量进行赋值（必须应用定点变量）并在LL赋值之后回到此GO TO语句。

1—15 子程序和磁盘存储器

程序中可以应用子程序，例如

SUBROUTINE INVERT

和

SUBROUTINE (B,IX)

可以用于具有下列语句的主程序中的适当位置处：

CALL INVERT

如果需要用的变量为SUBROUTINE (B,IX)，那么这个子程序用于具有CALL(W, 5)语句的主程序的适当位置处。在这个例子中，用作该子程序中的变量为W（对应于B）和5（对应于IX）。换句话说，被调用的子程序有一数组B和标识符IX，当调用该子程序时，所有变量B成为主程序中的变量W，而IX将具有整数值5。

程序还可以使用磁盘暂存器，随着不同的计算中心，其方法有所不同。应用暂存器时，要求在－DO循环中用下面典型语句

WRITE(5) W(I,J)

将信息存进磁盘。

下面语句

REWIND 5

将磁盘定位器返回到W(I,J) 开始的起始点处。

语句

READ(5) (W(I,J), J=1,M)

将调出磁盘中所存储的W(I,J) 信息。必须特别注意，WRITE(5)，REWIND5和READ(5)采取正确的顺序，例如，如果应用

WRITE(5) (W(I,J), J= 1, M)

之后，立即开始用下面语句写入矩阵A(I, J)

WRITE(5) (A(I,J), J=1, N)

那么A(I,J)的数据跟随在W(I, J) 的后面。现在，如果紧接下一计算涉及调出A矩阵，那么语句

REWIND 5

仍将磁盘回到W(I,J) 的开始点，而不是回到A(I,J) 的开始点。如果紧跟在W(I, J) 之后，使用了REWIND 5，那么A(I, J) 就会写在W(I,J) 上。大多数计算机具有一个以上暂存器，例如编号为4，5，6。如果A(I, J) 用于W(I, J) 的前面，那么宜用

```
      WRITE(5) 对于A(I, J)
      WRITE(6) 对于W(I, J)
```

而不发生顺序问题，因为A(I, J) 的应用简单地要求

```
      REWIND 5
      READ(5) A(I, J)
```

写入磁盘内的顺序与从磁盘调出所用的顺序相同，这是绝对需要的。例如，如果写入的次序为

```
      DO 60 I=1, L
   60 WRITE (6) (A(I, J), J=1, N)
```

那么不能用

```
      DO 61 J=1, N
   61 READ (6) (A(I, J), I=1, L)
```

但是可以用

```
      DO 61 K=1,L
   61 READ(6) (A(K,KK), KK=1, N)
```

如果需要回退一个记录，那么应该使用BACKSPACE指令*，例如

```
      DO 62 K=1, L
   62 READ(6) A(I, K)
      BACKSPACE 6
```

则将位于磁盘或磁带上中间结果存储区 6 中的A(I, K) 回退一个记录。但这不能用于重写或重新定义别的记录来占用这同一存储器。

COMMON语句用来说明某些变量对于一个以上的程序或子程序是公用的（应用的是同一变量）。应用COMMON语句时，这些变量的次序必须保持不变。应用 COMMON 语句时，紧跟在程序正文后面的程序内存分配表将包含一公用块。对应用了公用变量的程序的公用块所进行的检查将指明公用存储区使用得是否恰当。

例如，对于两程序的语句COMMON I,J,W有如下的内存分配表

	I	J	W
主程序	0	4	8
子程序	0	4	8

则说明诸变量都是公用的。如果内存分配表为

	I	J	W
主程序	0	4	8
子程序	0	4	2C

那么在子程序中变量*W*就不在公用区内（亦即不在同一磁心位置中，因为2*C*和 8 为不同的磁心存储位置），当子程序用*W*开始时，则无疑会发生异常情况，因为被用到的将是存在于地址码为2*C* 单元（而不是地址码为 8 的单元）中的任何内容。当应用 COMMON 时，照例应当检查公用块内存分配表，因为可能没有出错信息；不然首次警告可能是一组迷乱的输出表格。在所有公用程序中必须以同一方式标出公用下标变量。

* BACKSPACE即退一个记录的意思。
——译者注