IBM PC译丛

FORTRAN编译程序

刘熙明　朱玉珑　译

辽宁省电子计算机学会
《小型微型计算机系统》编辑部

序　　言

本书是IBM个人计算机的 FORTRAN语言参考手册。

用户预先应该有一些关于 FORTRAN 语言的知识。本手册并非教程，每节都只讲解FORTRAN语言有关题目的一部分。本手册由以下几部分组成：

- 第1章“引言”。为读者引入记法约定、 FORTRAN 程序结构、数据类型、表达式和 FORTRAN 名字。
- 第2章“编译一个 FORTRAN 程序”。描述编译、连接和执行 FORTRAN 程序的机制。
- 第3章“编译程序元命令”。描述处理 FORTRAN源文本的编译程序元命令。
- 第4章“语句”。介绍控制、程序函数和子程序、I/O 及说明语句。另外，也介绍了DATA 语句。
- 第5章“ I/O系统”。描述 FORTRAN I/O 系统，包括基本 FORTRAN 的I/O概念和 FORMAT 语句。
- 第6章“内部函数”。介绍在一 FORTRAN 程序中可使用的内部函数。
- 附录A“消息”。提供计算机可能发送给你的消息表。
- 附录B“IBM FORTRAN和 ANSI FORTRAN 77 之间的差别”。介绍 IBM FORTRAN与标准子集语言的差异。
- 附录C“连接程序（ LINK ）”。给读者提供连接程序的信息。
- 附录D“连接目标模块”。介绍如何将各种编译程序包连接到目标模块上。
- 附录E“程序举例”。举一个实例说明如何记入、执行、修正、再执行和运行一个IBM 个人计算机FORTRAN程序。

前　言

IBM / PC，XT &AT及其兼容机（长城0520等），是我国微型计算机普及率最高的机种，为促其开发和应用，辽宁省电子计算机学会曾于八四年出版过一套《IBM－PC译丛》（共十九本，350 万字，俗称黄皮书），深受用户欢迎。但由于印数有限，未能满足用户需要。为此，辽宁省电子计算机学会和《小型微型计算机系统》编辑部决定重新排印这套丛书。

再版过程中，我们对原书目录进行重新编选，淘汰一些对用户关系不甚密切的部分编译程序、专用软件，以及内容雷同的硬件技术手册，增加了广大用户迫切需要的dBASE Ⅱ、dBASE Ⅳ、IBM－PC通讯与网络等重要内容。对保留的内容，译者结合自己的实践经验，进行了重新的校订，这样便使这套丛书更为实用。

这次重新排印，时间很紧，尽管做了不少努力，更正一些错误，还可能会出现新的疏漏，敬请读者批评指正。

辽宁省电子计算机学会
《小型微型计算机系统》编辑部

目　录

目　　录

第四章 语句

第五章 I／O系统

第一章 引 言

记 法 约 定

本手册中使用的记法约定如下：

- 大写字母和特殊字符出现在程序中。（例如CONTINUE.）
- 小写斜体字母和单词是在正文中描述的具体语句中应提供的项。一旦一个小写项定义过后，就在整个程序的讨论中保持它的意义。（例如PROGRAM pname）。

例如，在描述整数编辑的格式中的大写和小写字母是Iw，其中w是一个非零的无符号整常数。

所以，在一个具体语句中，一个程序可以包含 I3 或 I44。描述实数编辑的格式记号是Fw.d，其中d为无符号整常数。在一具体语句中，F7.4或F22.0都是有效的。注意，黑圆点作为一个特殊字符取其字面意义。

- 括号指示任选项。例如，A〔w〕指示A或A12都是有效的（作为规定字符格式的方法）
- 省略号（…）指示在省略号之前可能出现一个以上的任选项。

例如，下面介绍的计算GOTO语句指示：用逗号隔开的、由几个s表示的语法项可以重复任意多次：

GOTO（s〔,s〕…）〔,〕i

- 空格在FORTRAN语句的介绍中通常是无意义的。在“FORTRAN 程序结构”一章中将讲解空格的一般使用规则，该章将对空格进行解释。

FORTRAN 程序结构

字符集

一个FORTRAN源程序是由一系列字符组成的，这些字符包括：

- 字母：从A到Z的52个大写和小写字母。
- 数字：0，1，2，3，4，5，6，7，8和9
- 特殊字符：ASCII字符集中可打印的其余字符。

除在字符常数和何勒内斯字段之外，FORTRAN 将所有上下文中的小写字母解释作大写。所以，如下的用户定义名，对于IBM FORTRAN系统来说都是不可区别的：

ABCDE abcde AbCdE aBcDe

IBM FORTRAN字符集的整理顺序是ASCII顺序。

行

一个FORTRAN源程序也可看成是行的序列。编译程序仅把一行中的前72个字符看成

是有意义的。编译程序忽略第72个字符以后的所有字符。

如果一行中的字符数少于72个，编译程序认为是有意义的（为了说明这一点，“字符表达式”这一章将讨论字符常数）。

列

一给定行中的字符都落在列中。例如，第一个字符是在列1中，第二个是在列2中，如此等等。

一个制表字符可放在一行的1-6列中，它使下一个字符像列7中的字符一样被解释。在列表文件中，这种制表将扩充为适当数目的空格。对于列表文件，所有其它制表都被不加改变地传给列表文件。

有字符存在其中的列在 FORTRAN中是有意义的：

列

1—5　语句标号和注释指示符
6　　延续指示符
7-72　源语句

初始行

初始行是在第6列中包含着一个空格或0的且不是注释行或元命令行的任意行。该行的前五列必须或者全是空格，或者包含着一个符号。只有跟在逻辑IF之后的语句例外。FORTRAN语句由初始行开始。

注： 初始行上延续列中的零（　0　）用来在初始行后使用延续行时改进易读性。

例

```
0IF     ((A.LT.0).(AND.)B.LT.0)
1       .AND.(C.LT.0))
2       THEN
```

空格

空格字符除下面注明的例外情形外，在FORTRAN 程序中是无意义的，可以用它来改进FORTRAN程序的易读性。这些例外是：

- 字符串常数中的空格
- 何勒内斯字段中的空格
- 列6中区别初始行和延续行的空格

注释行

注释行不以任何方式影响FORTRAN程序的执行。

当满足如下任意条件之一时，将一行看成是一个注释行：

- 在列1中是“C”（或c）。

• 在列1中是“*”。

• 一个只包含空格的行。

注释行后面必须直接跟着初始行或一个注释行。它们的后面一定不能跟着一个延续行。

注意： 在一个FORTRAN程序结尾处的额外空格行产生一个编译期间错，因为FORTRAN系统将它们解释为是注释行，但它们后面又没跟着一个初始行。

标号

一个语句标号是从一位到五位数字的序列，这个序列在每个程序单位中是唯一的。至少有一个数字必须为非零。一个标号可以放在初始行的列1到列5中的任何地方。空格和前导零是无意义的。

延续行

一个延续行是不为注释行和元命令行的任意行，在它的列6中包含着空格和0以外的任意字符。

延续行可以给出更多的地方来写语句。如果一个语句在一个初始行中写不下，可以最多用九个延续行来扩展它。

语句

一个FORTRAN 语句由一个初始行和可多达九个的延续行所组成。语句的字符是放在这些行的第7列到 72 列中的，可多达660个字符。END语句必须完整地写在初始行中，并且在该行中不能有任何其它的语句。

程序单位

子程序和主程序称为程序单位。

一个子程序由SUBROUTINE或一个FUNCTION语句开始，而以END语句结束。

一个主程序任选地由一PROGRAM语句开始。

FORTRAN语言在语句和构成一个FORTRAN编译的行中设定了某种次序。一般说来，一个编译由至多一个主程序和零或多个子程序所组成（关于程序单位和子程序的编译的更多信息请见附录D）。语句书写顺序的规则可在下面看到。

主程序和子程序

一个主程序由一PROGRAM语句开始，或由SUBROUTINE或 FUNCTION 语句以外的任何其它语句开始，而以END语句结束。

语句顺序

在一个程序单位中，不论是主程序还是子程序中，语句的出现顺序必须符合如下规则：

1. 若有一SUBROUTINE或FUNCTION语句或PROGRAM语句，必须作为该程序单位的第一个语句出现。

2．FORMAT语句可以出现在SUBROUTINE或FUNCTION语句或PROGRAM语之后的任何地方。

3．所有的说明语句都必须放在所有DATA 语句、语句函数语句和可执行语句的前面。

4．在说明语句内，IMPLICIT语句必须放在所有其它说明语句的前面。

5．所有的DATA 语句必须放在说明语句之后且在所有语句函数语句和可执行语句之前。

6．所有语句函数语句都必须放在所有可执行语句之前。

程序单位内语句的顺序

下表解释如下：

- 在一个语句的上面和下面给出的其它语句必须以指定的顺序出现。
- 注释语句或FORMAT语句可以散放在与它们交叉的其它语句中。

<table>
<tr><td rowspan="6">注释行</td><td colspan="2">PROGRAM、FUNCTION或SUBROUTINE语 句</td></tr>
<tr><td rowspan="5">FOMRAT
语 句</td><td>IMPLICIT语句</td></tr>
<tr><td>其它说明语句</td></tr>
<tr><td>DATA语句</td></tr>
<tr><td>语句函数语句</td></tr>
<tr><td>可执行语句</td></tr>
<tr><td colspan="2">END语句</td></tr>
</table>

数 据 类 型

在IBM FORTRAN 中存在四种基本数据类型：

1）整数 2）实数

3）逻辑 4）字符

这节介绍每种类型常数的性质、取值范围和形式。

IBM FORTRAN 数据类型对于无格式文件和随机访问存储器的存储要求是：

类 型	存储（字节）
LOGICAL	2或4
LOGICAL*2	2
LOGICAL*4	4
INTEGER	2或4
INTEGER*2	2
INTEGER*4	4
CHARACTER	1
CHARCTAER*n	n
REAL	4
REAL*4	4

整数

整数数据类型是正、负整数的一个子集。一个整数的值是对应整数的精确表示。一个整数型变量占用2或4个存储字节。

在IBM FORTRAN中，可将一个整数规定为 INTEGER*2、INTEGER*4或 INTEGER。前两种分别规定2和4字节整数。第三种规定或者为2或者为4字节整数，这由 $STORAGE元命令的设置所决定（缺省时为4字节）。

一个2字节整数可以包含在-32767到32767范围内的任何值。一个4字节整数可以包含在 -2147483647到2147483647范围内的任何值。

整常数是一个或多个十进制数字，前面带有任选的算术符号正（+）或负（-）。在整常数中不允许有小数点。下面是整常数的例子：

123	+123	- 123	0
00000123	32767	-32768	

实数

实数数据类型由实数的子集——单精度实数所组成。单精度实数值是所需求实数的近似值。单精度实数值占用四个字节存储器。该精度有六个十进制数字。

一个实常数或者是基本实常数，或者是后面跟着指数部分的基本实常数，或者是后面跟有指数部分的整常数。例如：

+1.000E-2	1.E- 2	1E-2
+0.01	100.0E-4	.0001E+2

以上所有的数都表示同一个实数：百分之一。

单精度实值的范围是

3.0E-39到1.7E+38（正的范围）

1.7E+38到-3.0E-39（负的范围）

一个实常数由一个选择性符号，后跟着一个整数部分，一个小数点，一个小数部分和一个选择性的指数部分所组成。整数部分和小数部分由一个或多个十进制数字组成，小数点是个实心圆点（.）。不论是整数还是小数部分都可以省略，但是决不能两者都没有。基本实常数的例子是：

-123.456	+123.456	123.456
-123.	+123	123.
-.456	+.456	.456

指数部分由字母“E”或“e”后跟着一个任选的一位或两位数字的有符号整常数所组成。指数部分指出：放在它前面的值要用10去乘指数部分的整数值那么多次。指数部分的例子是：

E12　　E-12　　E+12　　E0

逻辑数据类型

逻辑数据类型由真和假两个逻辑值组成。一个逻辑变量占用两个或四个存储字节。

在 IBM FORTRAN中，一个逻辑数据可规定为LOGICAL*2，LOGICAL*4或LOGICAL。前两者分别规定2-和4-字节逻辑数据。第三个按照$STORAGE元命令的设置或者规定2-字节逻辑数据或者规定4-字节逻辑数据（缺省时为4-字节）。

仅存在两种逻辑常数，.TRUE.(真) 和.FALSE.（假），它们表示两种对应的逻辑值。.FALSE.的内部表示一个全零（0）的字，而.TRUE的内部表示是除在最低有效位为1外其它各位均为零的字。若一个逻辑变量包含了任何其它类型的位值，则它的逻辑含意是不确定的。

逻辑变量的意义不受$STORAGE 元命令的影响，它主要是为了能和对于同样长度的LOGICAL、REAL和INTEGER变量的ANSI要求相一致。

字符数据类型

字符数据类型是ASCII 字符的序列。一个字符值的长度等于该序列中字符的个数。一个特定的常数或变量的长度是固定的，并且必须是在1到127个字符之间。

一个字符变量在存储器中是每个字符占用一个存储字节，若长度为奇数时，还要加上一个字节。字符变量总是在字的边界上对齐的。在字符值中允许有空白字符，并且是有意义的。

字符常数是由一对撇引号括起来的一个或多个字符。在字符常数中允许空白字符。每个字符都按一个计数。在一字符常数内的撇引号使用中间不带空格的两个相继的撇引号表示。字符常数的长度等于撇引号之间的字符的个数，两个撇引号作为一个单个撇引号字符来计数。字符常数的例子是：

```
'A'''                    'Help'
'A very long CHARACTER constant'          '''
```

在最后的例子中，' ' ' ' ，表示一个单个的撇引号。

FORTRAN系统允许源程序行有多达72个列。短一些的行由空格补足到72列；因此，当一个字符常数超过了行的边界时，它的值恰如由空格补足到72列那样来计算，并且放在相继行的前面。这样，字符常数：

```
200    CH=' ABC
      XDEF'
```

的长度为63。

表 达 式

FORTRAN有四类表达式：

1) 算术　　2) 字符

3) 关系　　4) 逻辑

算术表达式

一个算术表达式产生一个或者为整型或者为实型的值。算术表达式最简单的形式是：

· 无符号整数或实常数

• 整型或实型变量引用

• 整型或实型数组元素引用

• 整型或实型函数引用

一个变量引用或数组元素引用的值出现在算术表达式中时必须是定义过的。并且，整形变量的值必须是用算术值定义过的，而不能是用放在 ASSIGN 语句前面的语句标号值定义的。

其它算术表达式可以使用括号和如下的算术运算符从上面的简单形式构造出来：

算术运算符

运算符	操作	优先次序
**	指数	最高
\	除	中间
*	乘	中间
-	减或负	最低
+	加	最低

所有这些运算符都是二元运算符，放在表达式中的两个操作数之间。

优先次序相等的操作是左结合的，只有指数操作例外，它是右结合的。为此，

A/B*C　　与　　(A/B) *C　　相同

A**B**C　　与　　A** (B**C)　　相同

象大多数程序设计语言一样，可以按照通常数学意义形成算术表达式，例外的是 FORTRAN禁止两个运算符紧挨着出现。所以，

A** - B

是禁止的，然而允许

A * * (- B)

注意，一元运算符求负也是最低优先次序的，所以

- A*B

解释为：

- (A*B)

在表达式中可以使用括号，以控制求值运算的结合及次序。

整数除法

两个整数的除法产生两个值的数学商，并截断为一个整数，于是7/3求得值2，(-7/3) 求得值-2，9/10求得值0，9／(-10) 求得值0。

类型转换和结果类型

当算术表达式的所有操作数都是相同类型的时，由该表达式所产生的值也是这种类型的。当操作数是不同的数据类型时，由该表达式所产生的值的数据类型按以下等级确定：

算术数据类型	等级
REAL	最高
INTEGER*4	中间
INTEGER*2	最低

当一个操作有两个不同数据类型的算术操作数时，产生值的数据类型是最高等级操作数的数据类型。例如，对于整数和实数组元素的一个操作产生实型数据值。

一个表达式的数据类型是在求值该表达式中所实现的最后操作的结果的数据类型。操作的数据类型被分类为INTEGER*2、INTEGER*4或REAL。

整数操作仅能在整型操作数上实现。一个由除法所产生的小数被截断为整数，而不是舍入。所以如下的例子求得值0，而不是1。

```
1/4 + 1/4 + 1/4 + 1/4
```

注意：对于类型INTEGER（没有*2或*4扩展）的存储依赖于$STORAGE元命令的用法。详见第三章。

实操作是在实操作数和整操作数的组合上执行的操作。当一个操作有一实操作和一整操作数时，先将整操作数用给出等于零的小数部分的办法转换为实数据类型，然后使用实算算术运算求值该表达式。

例如：

```
A = N/B
```

N转换为一实数据类型，然后在N和B上实现实数除法。

当一表达式中包含着混合数据类型时，按照运算符的优先次序实现整型和实型操作的求值。例如：

```
Y = X* (I + J)
```

在I和J上实现整数加法，得到的和转换为实数据类型，然后在结果和X上执行实型乘法。

注意：IBM FORTRAN 不检查整数溢出。如果超出了整数值的界限，将会出现不可预料的结果。

字符表达式

字符表达式产生一个字符类型的值。字符表达式的形式是：

- 字符常数
- 字符变量引用
- 字符数组元素引用
- 由括号括起来的任何字符表达式。

运算符在字符表达式中不起运算符的作用

关系表达式

关系表达式比较两个算术或两个字符表达式的值。

一个算术值不能与一字符值相比较。关系表达式的结果是逻辑类型的。

关系表达式可使用下面列出的进行值比较的任何运算符。

关系运算符

运算符	所表示的操作
.LT.	小于

.LE.	小于或等于
.EQ.	等于
.NE.	不等于
.GT.	大于
.GE.	大于或等于

所有这些运算符都是出现在它们的操作数之间的二元字符。例如

A .GT. B

X .EQ. 7

上例是正确的表达式。在关系操作数间没有相对地优先次序或结合问题；因此

A .LT. B .NE. C

是不正确的。上例违反了操作数的类型规则。关系表达式仅能出现在逻辑表达式中。

具有算术操作数的关系表达式可以有一个整型和一个实型的操作数。在这种情况下，在求值关系表达式之前，将整操作数转换为实型操作数。

具有字符操作数的关系表达式比较它们的操作数在 ASCII 整理顺序中的位置。如果一个操作数在该整理顺序中出现较早，则它就比另一个小。当要比较不等长的操作数时，将较短的操作数看成是用空格扩展成和较长操作数一样长了。

逻辑表达式

一个逻辑表达式产生一个逻辑类型的值。逻辑表达式的最简形式是：

- 逻辑常数
- 逻辑变量引用
- 逻辑数组元素引用
- 逻辑函数引用
- 关系表达式

其它的逻辑表达式可以从上面的最简形式使用括号和下表中列出的逻辑运算符构造出来：

逻辑运算符	操作	优先次序
.NOT.	非	最高
.AND.	与	中间
.OR.	或	最低

AND和OR是二元运算符，它们出现在它们的逻辑表达式操作数之间。NOT运算符是一元的，放在操作数的前面。等优先次序的操作是左结合的，例如：

A.AND.B.AND.C

等价于

(A.AND.B) .AND.C

作为优先次序规则的例子，

.NOT.A.OR.B.AND.C

被解释为和下面表达式相同：

(.NOT.A) .OR. (B.AND.C)

两个NOT运算符不可以彼此相邻，尽管

```
A  .AND. .NOT. B
```

是可允许的具有两个相邻运算符的表达式的例子。

逻辑运算符的意义是它们的标准的数学语意，.OR.是“非互斥的”，即

```
.TRUE. .OR. .TRUE.
```

等价于值

```
.TRUE.
```

数组元素名

数组元素名用来引用一数组的一个元素。

```
arr  (sub[,  sub]…)
```

arr项是数组的名字。sub是下标表达式。

```
C     ASSIGN THE 4TH ELEMENT OF
C     ARRAY A THE VALUE 3.8
      A (4,1,1) =3.8
```

下标表达式是一个用来选择数组的一个确定元素的整型表达式。下标表达式的个数必须与数组说明符中维数的个数相匹配。下标表达式的值必须介于1和它所表示的维数的上界之间。

函数引用

函数引用可出现在算术或逻辑表达式中。函数引用的执行使该函数求值，并且产生的值在包含它的表达式中作为一个操作数。

```
fname ([arg[, arg]]…)
```

fname项是外部、内在或语句函数的用户定义或系统定义名。

arg项是实在参数。实在参数可以是算术表达式、数组或用户定义或系统定义的函数或子程序。下面是表达式中函数引用的例子：

```
1+SIN(.5)
A+B+USERFN(9)
.TRUE..AND.BITFN(3)
VAR1/XYZ(3, A)
```

有关形式参数的更多知识请见第四章。实在参数的个数必须和函数定义中的形式参数个数相同，并且对应的数据类型也必须一致。

表达式的优先次序

当在同一表达式中出现算术、关系和逻辑运算符时，它们的相对优先次序如下：

运算符类的相对优先次序

运算符	优先次序
算术	最高
关系	中间
逻辑	最低