IBM-PC-06

FORTRAN 编译程序

中国科学院科理高技术公司

一九八五年一月

前　言

这是IBM个人计算机Fortran语言的参考说明书。

读者应该具备初步的Fortran知识。因为本书不是一本指导性的说明书，而是在每一章中较全面地解释一部分Fortran语言。本书内容如下：

· 第一章是“引言”，介绍符号约定，Fortran程序结构，数据类型，表达式和Fortran名。

· 第二章是“编译Fortran程序”，说明编译，连接和执行Fortran程序的原理。

· 第三章是“编译程序元命令，描述处理Fortran源文本的编译程序元命令。

· 第四章是“语句”，描述控制，程序函数和子程序，I/O以及说明语句。此外，还说明DATA语句。

· 第五章是“I/O系统”，描述Fortran I/O系统，包括基本的Fortran I/O概念和FORMAT（格式）语句。

· 第六章是“内部函数”，描述Fortran程序使用的各种内部函数。

· 附录A是“信息”，提供计算机给出的信息表。

· 附录B是“IBM Fortran和ANSI Fortran的差别”，描述IBM Fortran和标准基本级语言的差别。

· 附录C是“连接（LINK）程序”，提供能连接用户程序的信息。

· 附录D是“连接目标模块”，描述是如何将目标模块与不同的编译程序包连接。

· 附录E是“实例对话”，演示一个实例，说明如何送入，执行，修改、再执行和运行IBM个计算机的Fortran程序。

目　录

第1章　引　言

IBM个人计算机Fortran符合ANSI×3.9—1978标准（FORTRAN 77）的基本级，在本说明书中称为IBM Fortran。它也包含ANSI×3.9—1978完备级的部分特性。

IBM Fortran的功能已得到增强，容易同已有的Fortran 66 程序实现转换，例如DO循环的Fortran 66语义是编译程序的一个选择项。

IBM个人计算机连接程序允许把Fortran目标模块和其它语言，IBM个人计算机MACRO汇编程序，以及IBM个人计算机Pascal编译程序的目标模块结合在一起，使程序的不同部分需采用不同语言的各种应用程序易于编写。

IBM Fortran由Fortan编译程序和目标模块库组成，后者构成Fortran运行时间库。建立可执行的Fortran程序的第一步是编译它的（一个或多个）源模块，然后将编译产生的目标模块与Fortran运行时间库相连接，得到一个能被IBM个人计算机DOS调用的运行文件。

待连接的目标模块和库，除了包括IBM Fortran编译程序的输出以外，还可以包括IBM个人计算机MACRO汇编程序和IBM个人计算机Pascal编译程序的输出。

1.1　符号约定

本说明书中所用的符号约定如下：

（1）程序中送入的大写字母和特殊字符与所示的一样。（例如，CONTINUE）

（2）小写斜体字母和字词都是一些项目，即应该在文本所说明的实际语句中提供的项目。一旦定义一个小写的项目，它在程序的全部讨论过程中保持它的意义。（例如，PROGRAM Pname）

例如，在说明整数编辑格式中，大写和小写是用Iw表示的，其中W是一个非零的不带符号的整常数。

这样，在实际语句中，程序可以包括I3或者I44。描述实数编辑的格式是Fw.d，其中d是不带符号的整常数。在实际语句中，F7.4或者F22.0都是正确的。注意，句号作为一个特殊字符，它要占一个字符的位置。

（3）方括号表明是任选项目，例如，A〔w〕表示A或A12都是正确的（作为说明字符格式的方法）。

（4）省略号（…）表示在省略号前面的选择项目可以出现一次以上。

例如，在下面说明的所计算的GOTO语句表明，用句号分开的各个用S表示的语句项可以重复任意多次。

GOTO（S〔，S〕…）〔，〕i

（5）空格在Fortran语句的说明中通常是无关紧要的。本章“Fortran程序结构”一节所描述的空格的一般规则，适用于对所有空格的解释。

1.2 Fortran程序结构

1.2.1 字符集

Fortran源程序是由一系列字符所组成的，这些字符包括：

（1）字母：从A到Z，52个大写和小写字母；

（2）数字：0，1，2，3，4，5，6，7，8，和 9；

（3）特殊字符：ASCII字符集的其余的可打印字符。

除了在字符常数和Hollerith字段中以外，Fortran在所有场合都把小写字母解释为大写字母。这样，以下用户定义的名对IBM Fortran系统来说，都是难以区分的。

ABCDE abcde AbCdE aBcDe

IBM Fortran字符集的排列顺序是ASC II的顺序。

1.2.2 行

Fortran源程序也可认为是一系列的行。编译程序只处理每行的前72个字符。编译程序将第72列以后的所有字符都略去不管。

若一行少于72个字符，编译程序认为它的长度是有效的（有关这方面的说明，见本章后面的“字符表达式”一节所说明的字符常数）。

1.2.3 列

在一给定行中的各个字符都是按列排的。例如，第一个字符是在第1列上，第二个字符是在第2列上等等。

Tab字符占有一行的第1—6列，它使下一个字符解释为正好在第7列上，这个tab字符在列表文件中将扩大到适当的空格数。所有其它方式传送的tab与列表文件中的tab一样。

在Fortran中，字符所在的列是有其一定意义的：

列	用　途
1—5	语句标号和注释说明符
6	续续说明符
7—72	源语句

1.2.4 初始行

初始行是在第6列为空格或零的任意行，它不是注释行，也不是元命令行。这种行的前5列一定要都是空格或者包含一个标号。除了语句跟在逻辑IF后面是个例外以外，所有Fortan语句都是以初始行开始的。

注意：当初始行后还有继续行时，在初始行的继续列位置上写零（0），能使程序更清晰易读。

例如：

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

2 THEN

1.2.5 空格

空格符除下面所示的例外以外，它在Fortran源程序中是无关紧要的，但是用一定的空格可以使Fortran程序便于阅读。这些例外是，

（1）串常数中的空格；

（2）Hollerith字段中的空格；

（3）第6列上的空格用以区分初始行和继续行。

1.2.6 注释行

注释行不以任何方式影响Fortran程序的执行。

凡是遇到下列情况之一，就作为注释行处理：

（1）第1列为“C”（或“c”）；

（2）第1列为“*”；

（3）该行全部都是空格。

注释行的后面必须紧跟一个初始行或另一注释行，它的后面不能跟继续行。

注意：Fortran程序结束处的多余空格行，尽管它们的后面不是初始行，由于Fortran将它们解释成注释行，往往产生编译时间错误。

1.2.7 标号

语句标号是一个1～5位数字的序列，在每个程序单元中，它们是唯一的。至少有1位数字是非零的。标号可以放在初始行1至5列的任何位置上，空格和前导零都是无所谓的。

1.2.8 继续行

继续行不是注释行，也不是在第6列上是除空格或零以外的任何字符的元命令行。继续行的前5列一定要是空格。

继续行让用户能把一个语句写得更长些。若语句在初始行内装不下，可以扩展为最多有9行的继续行。

1.2.9 语句

Fortran语句由初始行和最多有9行的继续行所组成。语句的字符写在这些行的第1至第72列上，最多可以有660个字符。END语句必须全部写在初始行上，并且没有一个其它语句可以出现END语句的初始行，

1.2.10 程序单元

子程序和主程序都称为程序单元。

子程序以SUBROUTINE或FUNCTION语句开始，以END语句结束。

主程序可以选择以PROGRAM语句开始。

Fortran语句在组成Fortron编译的语句和行中间要求有一定的顺序。一般地说，编译至

多由一个主程序和零个或多个子程序所组成（详见附录D的程序单元和子程序编译）。对语句进行排序的规则在下面说明。

1.2.11 主程序和子程序

主程序从一个PROGRAM语句，或者任何一个不是SUBROUTINE或FUNCTION语句的语句开始，而且以一个END语句作为结束。

1.2.12 语句次序

在一个程序单元中，无论是主程序还是子程序，各个语句必须按次序出现，其规则如下：

（1）若存在SUBROUTINE或FUNCTION语句，或者PROGRAM语句，它们必须作为程序单元的第一50语句出现；

（2）如果存在FORMAT语句，则它可以出现在SUBROUTINE或FUNCTION语句，或PROGRAM语句后面的任何地方。

（3）所有说明语句都必须在全部DATA语句，语句函数语句和可执行语句之前。

（4）在说明语句中，IMPLICIT语句必须在所有其它说明语句之前。

（5）全部DATA语句必须出现在说明语句之后，在全部语句函数语句和可执行语句之前。

（6）所有语句函数语句必须在全部可执行语句之前。

1.2.13 程序单元中的语句次序

对下表说明如下：

（1）上下语句之间的关系必须按规定的次序出现；

（2）注释行或FORMAT语句可以插入右边其它各语句的中间。

<table>
<tr><td rowspan="6">注释行</td><td colspan="2">PROGRAM，FUNCTION或SUBROUTINE语句</td></tr>
<tr><td rowspan="5">FORMAT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="3">END语句</td></tr>
</table>

1.3 数据类型

*IBM Fortran*有四种基本的数据类型：

1）整数型

2）实数型

3）逻辑型

4）字符型

本节要说明每一种类型的性质，取值范围，以及常数形式。

对于无格式文件和随机存取存贮来说，IBM Fortran数据类型的存贮要求是：

类　型	存贮量(字节)
LOGICAL	2或4
LOGICAL*2	2
LOGICAL*4	4
INTEGER	2或4
INTEGER*2	2
INTEGER*4	4
CHARACTER	1
CHARACTER*n	n
REAL	4
REAL*4	4

1.3.1 整数型

整数数据类型是包括负数和正数的一个子集，整数值是对应整数的精确表示。一个整数变量占据2个或4个字节的存贮量。

在IBM Fortran中，一个整数可以规定为INTEGER*2，INTEGER*4或INTEGER。前二个分别规定2字节或4字节的整数。而第三个则根据$STORAG元命令规定为2字节或4字节的整数。（缺省是4字节的）。

2字节整数包括从－32767到32767之间的任何整数。4字节整数包括－2,147,483,647，到2,147,483,647之间的任何整数。

整常数是带任意算术符号加（＋）或（－）的一位或多位十进制数字。在整常数中，不允许有十进制小数点。下面是整常数的例子：

```
    123     +123     -123
00000123    32767    -32767
```

1.3.2 实数型

实数数据类型由实数子集，即单精度实数所组成。一个单精度实数值是所需实数的近似表示。单精度实数值占据4个字节的存贮单元，其精度为6位十进制数字。

实常数是基本实常数，后面带有指数部分的基本实常数，或者是后面带有指数部分的整常数。例如：

```
+1.000E-2    1.E-2    1E-2
+0.01        100.0E-4 0.0001E+2
```

以上各例都表示同一实数：1/100。

单精度实数值的范围是：

3.0E−39到1.7E+38（正的范围）

1.7E+38到−3.0E−39（负的范围）

一个实常数由带任意符号的整数，小数点，小数部分和任选的指数部分所组成。整数和小数部分通常是一位或几位的十进制数字，小数点是一个句号（.），整数或小数部分可以省略，但不能都省略。基本实常数的几个例子如下：

```
−123.456     +123.456     123.456
−123.        +123         123.
−.456        +.456        .456
```

指数部分由字母“E”或“e”及其后面有任选的带符号的1位或2位数字的整常数所组成。指数表示前面的基本实常数和以指数部分的整数为幂次以10为底的数相乘。指数部分的几个例子是：

```
E12    E−12    E+12    E0
```

1.3.3 逻辑型

逻辑数据类型由两种逻辑值真和假所组成。一个逻辑变量占据2个或4个字节的存贮单元。

IBM Fortran的逻辑变量或数据用LOGICAL*2，LOGICAL*4或者LOGICAL来说明，前二个分别规定2个或4个字节的逻辑值，而第三个则根据$STORAGE元命令的设定情况是2字节或4字节的逻辑值，（缺省为1字节）。

只有两种逻辑常数：.TRUE.和.FALSE.，它们表示两个相应的逻辑值。.FALSE.的内部表示是一个全部为零的字，而.TRUE.的内部表示是最低位为1其余为零的字。若逻辑变量包含任何其它位值，它的逻辑意义是不确定的。

逻辑变量的有效性不受$STORAGE元命令的影响，这种元命令是事先提供的，其目的是允许与ANSI要求相容，即LOGICAL，REAL和INTEGER变量有同样大小。

1.3.4 字符型

字符数据类型是一组ASCII字符的序列，字符值的长度等于序列的字符个数。特定常数或变量的长度是固定的，而且必须在1到127个字符之间。

对于序列中的每个字符来说，一个字符占据1个字节的存贮单元，如果长度是奇数，要加1个字节。字符变量总是以字界排列的。在字符值中允许空格字符，而且空格符是有意义的。

字符常数是括在一对撇号内的一个或多个字符。在字符常数中容许有空格符，每个空格计作一个字符。在字符常数中的撇号由中间没有空格的两个相邻的撇号所代表，字符常数的长度等于撇号之间的字符数，而双撇号算作单撇号字符。字符常数的几个例子为：

```
'A' ' '        'Help'
'A very long CHARACTER Constant'  ''''
```

最后例（''''）表示为一个单撇号（'）。

Fortran允许源程序行最多为72列。短行用空格添补至72列。这样，当一个字符常数超

过行边界时，赋给它的值好象该行添许多空格至72列，且放在继续行之前。这样，字符常数

```
200  CH='AB
     XDEF'
```

其长度为63。

1.4 表达式

Fortran有四种表示式，即为：

（1）算术表达式；

（2）字符表达式；

（3）关系表达式；

（4）逻辑表达式；

1.4.1 算术表达式

算术表达式产生的值不是整型值就是实型值。算术表达式的最简单形式为：

（1）不带符号的整常数或实常数；

（2）整数或实数变量的引用；

（3）整数或实数数组元素的引用；

（4）整数或实数函数的引用。

变量是引用值或数组元素引用值要出现在算术表达式中，就一定要定义。此外，定义整数变量值必须用算术值，而不是用在赋值语句（ASSIGN）中预先设定的语句标号值。

根据上述诸简单形式的算术表达式，用括号和算术运算符建立其它算术表达式。

算术运算符

运算符	运　算	优先级
**	指数	最高
/	除法	中等
*	乘法	中等
−	减法或变负	最低
+	加法或同号	最低

全部算术运算符都是二元运算符，它们出现在算术表达式诸运算数之间。在运算数前的+或−也可以是一元运算符。

同等优先级的运算是向左结合的，只有指数运算是向右结合的。因此，A/B*C与(A/B)*C是一样的；A**B**C与A**(B**C)是一样的。

除Fortran禁止两个运算符相连出现在一起以外，算术表达式按通常的数学意义来构成，这与大多数编程语言是一样的。

这样，尽管允许A**(−B)，但是A***−B是禁止的。注意一元运算符的负号也属最低

优先级，所以把－A*B解释成－（A*B）。

表达式中可以用括号，以便控制运算的结合方式和计算次序。

整数除法

两个整数的除法得到的值是它们数学商取整。于是，7/3计算得2，（－7）/3得－2，9/10得0，9/（－10）也得0。

类型转换和结果的类型

当算术表达式中全部运算数都是同一类型时，表达式计算得到的值也是同类型的；当运算数具有不同数据类型时，表达式产生的值有下列等级所确定的数据类型：

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

当运算是在两个不同数据类型的算术运算数之间进行的时候，产生的数据类型是最高等级运算数的数据类型。例如，整数和实数元素的运算产生实型数据值。

表达式的数据类型是计算表达式的最后一次运算结果的数据类型。运算的数据类型分成INTEGER*2，INTEGER*4或REAL。

整数运算仅在整数运算数之间进行。除法最后产生的小数部分按整数运算方式被截去，而不用四舍五入的方式。这样以下例子计算得到的是0而不是1。

$$\frac{1}{4}+\frac{1}{4}+\frac{1}{4}+\frac{1}{4}$$

注意：类型INTEGER（不带*2或*4的扩展说明）的存贮量取决于使用$STORAGE元命令的情况，详见第三章。

实数运算是在实数运算数之间或者对实数和整数运算数的组合进行的。当运算中出现实数和整数运算数时，先把整数运算数转换成实数数据类型，即使实数的小数部分为零，然后用实数算术方式来计算表达式。

被如，A=N/B，先把N转换成实数数据类型，再对N和B进行实数除法。

当表达式包含不同类型的数据时，整数和实数运算的进行是按运算优先级次序来计算的。例如：Y=X*(I+J)，首先进行I和J的整数加法，然后把它们的和变成实数数据类型，最后将此结果和X进行实数乘法。

注意：IBM Fortran并不检验整数溢出，如果超出整数值的限制范围，会产生无法预计的结果。

1.4.2 字符表式达

字符表达式产生字符型的值，字符表达式的形式有：

（1）字符常数；

（2）字符变量引用；
（3）字符数组元素引用；
（4）在括号内的任何字符表达式。
不能用算符构成字符表达式。

1.4.3 关系表达式

关系表达式比较两个算术或字符表达式的值。算术值不可以与字符值进行比较。关系表达式的结果是一个逻辑值。

关系表达式能使用下列所示的运算符中的任何一个来比较数值。

关系运算符

运算符	代表的运算
.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

所有的运算符都是两个字符，并出现在它们的运算数之间。例如：

```
A .GT. B
X .EQ. 7
```

以上两例都是正确的表达式。在关系运算数之间没有相对的优先级或者结合规则。因此

```
A .LT. B .NE. C
```

是不正确的，它违反运算数的类型规则。关系表达式只出现在逻辑表达式中。

有算术运算数的关系表达式可以有一个整型运算数和一个实型运算数。这时，在关系表达式计算之前，整数运算数被转换成实型运算数。

字符运算数的关系表达式按ASC II排列次序来比较它们的运算数位置。一个字符运算数小于另一个，则它的排列位置一定在前；如果长度不等的两运算数进行比较时，先把短的操作数用空格来填补，延长它的长度，使它的长度与较长的运算数一样。

1.4.4 逻辑表达式

逻辑表达式产生一个逻辑型的值，逻辑表达式最简单的形式是：

（1）逻辑常数；
（2）逻辑变量引用；
（3）逻辑数组元素引用；
（4）逻辑函数引用；
（5）关系表达式。

其它逻辑表达式是通过以上简单形式用括号和逻辑运算符建立起来的。逻辑运算符如如下表所示：