Microsoft FORTRAN 4.0版

优化编译程序语言参考手册

中国中文信息学会咨询服务中心
中国科学院计算技术研究所十一室
中国科学院希望高级电脑技术公司

一九八八年十月

目　录

第一章 引 言

1.1 概况

本章介绍《Microsoft RFORTRAN优化编译程序语言参考手册》中的内容，描述手册中使用的记号约定；并告知深入了解FORTRAN的途径。

如何在系统上编译和连接FORTAN程序，以及有关Microsoft FORTRAN扩展的讨论，请参阅《Microsoft FORTRAN编译程序用户指南》。关于如何使用Microsoft CodeView™面向窗口的调试程序进行程序调试，请见Microsoft CodeView手册。

1.2 关于这本手册

《Microsoft FORTRAN优化编译程序语言参考手册》解释由Microsoft FORTRAN优化编译程序4.0版所实现的FORTRAN语言，本手册打算作为在FORTRAN语言方面有一定经验的程序员的参考手册。它不教你如何用FORTRAN进行程序设计，有关FORTRAN的推荐书籍表可见1.4节："有关FORTRAN的书籍"。

注意

Microsoft FORTRAN对全ANSI标准作了许多扩展。在这本手册中，所有有关Microsoft扩展的信息都以黑体字印出。

手册的第二、三、四章讨论Microsoft FORTRAN语言的特殊元素。第五、六章和附录B分别给出按字母顺序排列的语句表、元命令表和内部函数表。下表展示了查找特定专题的有关信息的章节：

有关信息	请参见
Microsoft FORTRAN中的字符，名字，数据类型，属性和表达式。	第二章，"FORTRAN的元素"
源程序中行的格式	第三章，"程序结构"
Microsoft FORTRAN程序的结构化	第三章，"程序结构"
子程序，函数和参数	第三章，"程序结构"
Microsoft FORTRAN的输入和输出	第四章，"输入／输出（I/O）系统"
Microsoft FORTRAN的语句，按字母顺序列出	第五章，"语句"
称为元命令的编译程序伪指令，按字母顺序列出	第六章，"元命令"
美国标准信息交换码（ASCII）字符集	附录A，"ASCII字符代码"
内部函数，按字母顺序列出	附录B，"内部函数"

本手册中使用的部分术语　　“术语字典”

1.3 记号约定

这本手册使用下列记号。注意，在大多数情况下，FORTRAN中的空格是没有什么意义的。详见2.2.1节：“空格”。

约定的例子	约定的描述
对ANSI标准扩展	手册中黑体字描述ANSI FORTRAN 77 完全语言标准的扩展特性。这些扩展可以是，也可以不是由符合完全语言标准的其它编译程序实现的。 例如，句子“COMPLEX或COMPLEX*8数据类型是一个单精度实数的有序对。”在该句子中，词“或COMPLEX*8”为黑体字，因为COMPLEX*8数据类型是对ANSI FORTRAN 77 完全语言标准的一项扩展。
FORTRAN 关键字	黑体的大写字母表示FORTRAN关键字。正如下面解释的一样，除非这些关键字用二个圆括号括起来，否则它们是语句语法要求的一部分。在你写的程序中，这些FORTRAN关键字可以大写或小写或以它们的组合键入。 在下列语句中，IF和THEN为FORTRAN关键字： IF (expression) THEN
其它关键字	黑体小写字母表示其它语言的关键字。 在句子“由LOCNEAR返回的值等价于Microsoft C 中的near函数或数据指针，或Microsoft Pascat中的adr类型”中，词LOCNEAR是FORTRAN关键字，而词near和adr分别为Microsoft C 和Microsoft Pascal的关键字。
〔(, */=)〕	黑体表示按要求准确键入的任何标点或符号（如逗号、括号、分号、连字号、等于号和操作符）。 例如，内部函数CHAR的语法表示为CHAR(int)，你必须输入词CHAR，后面跟一个左括号，然后是一个整数，最后为一个右括号。注意，因为CHAR是FORTRAN关键字，你可以大写或小写形式键入。
省字号：’’’	省字号作为右单引号（’）而不是左单引号（‘）输入。注意，例子中使用的字体，如’string’，省字号看上去为：’。
替代符	写成斜体的字是你所提供的信息类型的替代符。文件名就是这类信息的一个例子。 在下列语句中，label印成斜体字，表示了GOTO语句的通用形式： GOTO *label* 在实际的程序语句中，替代符label必须由一个特定的行标号替代，如下面的例子所示：

GOTO 150

斜体字偶尔也用于正文，表示强调。

〔〔可选择项〕〕

双重方括号内的内容是可选择的。

例如，下列语句表示在STOP语句中可选择输入一个message。

STOP 〔〔message〕〕

这样，下述STOP语句的任一条都是可接受的：

STOP

STOP 'End of Program'

注意

双重方括号（〔〔 〕〕）是手册中使用的语法约定，以表示可选择项。单方括号（〔 〕）是应该在出现的地方键入的标点符号。

{choice1 | choice2}

花括号和竖杆表示可在二个或二个以上项之间作选择。花括号括住那些选择，而竖杆分离那些选择。除非所有项同时都包括在双重方括号内，否则你必须选择其中一项。

例如，INTERFACE语句有如下语法：

INTERFACE TO {fwnction | swbrowtine}

这条语句表示字INTERFACE TO之后必须输入一个fwnction（函数）或一个子程序（swbrowtine）。

重复元素

跟在一项后的三个点表示可以输入具有相同形式的更多项。

例如，内部函数MAX的语法：

MAX(genA, genB 〔〔, genc〕〕 …)

跟在〔〔, genc〕〕后的点表示，只要将参数用逗号分开，你就可以输入更多的参数。

垂直省略号

语法行和程序例子中的垂直省略号表示省略了该程序的一部分。

例如，在下列程序段中，仅显示了二行，它们之间的行被省略了：

```
Call  getnum( I, *10 )
  ⋮
SUBROUTINE getnum( I, * )
```

键名

较小的大写字母用于表示你必须按下的键和键系列。例如ENTER和CONTROL-C。

■ 例子

下例展示了本手册的记号约定如何用于表示EXTERNAL语句的语法：

EXTERNAL name〔〔 〔attrs〕 〕〕 〔〔, name 〔〔 〔attrs〕 〕〕 〕〕

这个语法表示，使用EXTERNAL语句时必须首先输入字EXTERNAL，后面紧跟一个你说明的name（名字）。然后，你可以有选择地输入一个左方括号（〔），紧跟一个你给定的属性（attrs），再接一个右方括号（〕）。如果想给出更多有选取属性（att-

rs）的名字，则必须输入一个逗号，紧跟一个名字，紧跟一个左方括号，属性和一个右方括号。因为〔(, name〔〔 〔attrs〕 〕〕 〕〕串后有三个点（…），所以按需要输入更多的这样的串（一个逗号，跟着是一个名字，可选择地紧跟方括号内的属性）。

1.4 有关FORTRAN的书籍

下列书籍含有FORTRAN程序设计方面的信息：

Agelhoff, Roy, and Richard Mojena. *Applied FORTRAN 77, Featuring Structured Programming*. Belmont, Calif.: Wadsworth, 1981.

Ashcroft, J., R. H. Eldridge, R. W. Paulson, and G. A. Wilson. *Programming with FORTRAN 77*. Dobbs Ferry, N.Y.: Sheridan House, Inc., 1981.

Friedman, Frank and E. Koffman. *Problem Solving and Structured Programming in FORTRAN*. 2d ed. Reading, Mass.: Addison-Wesley, 1981.

Kernighan, Brian W. and P. J. Plauger. *The Elements of Programming Style*. New York, N.Y.: McGraw-Hill, 1978.

Wagener, Jerrold L. *FORTRAN 77: Principles of Programming*. New York, N.Y.: John Wiley and Sons, Inc., 1980.

第二章　FORTRAN 的元素

2.1 引言

FORTRAN源文件由字符、名字和数值组成。本章解释这些元素并讨论它们在表达式中如何组合。

2.2 字符

Microsoft FORTRAN源文件可包含ASCII字符集中任何可显示的字符。在附录A，“ASCII字符代码”中列出的ASCII字符集包括下列字符：

- 52个大写和小写字母（从A到Z和从a到z）。Microsoft FORTRAN将美元符号（$）也看作字母。进行名字校对时，美元符号立即跟在大写的Z后面。

除之字符常量和Hollerith域（Hollerith域的解释见4.8.1.2 节）外，Microsoft FORTRAN编译程序将所有上下文中的小写字母解释为大写字母，在字符常量和Hollerith域中，大小写是有区别的。例如，语句WRITE（*，*）和write（*，*）是相同的，但字符常量'ijR'和' IJK' 是不同的。

在字符常量中，对大小写的敏感性有个例外。除了说明 $STRICT元命令外，以第五章“语句”中列出的FORTRAN语句所说明的字符常量对大小写是不敏感的。如在CLOSE语句中，你可以输入一个字符常量说明是保留还是删除文件。其语法为〔〔,STATUS=Status〕〕，而且Status（状态）可接受的值为 'KEEP '和 'DELETE '。只要没有设置 $STRICT元命令，将STATUS置为' KEEP' 等价于将STATUS置为' Reep ' 或 ' KeEp ' 。注意，由于STATUS是FORTRAN关键字，它可以大写或小写或是任意组合拼出。

- 10个数字（从0到9）。
- ASCII 字符集中所有可打印的其它字符：

```
! # % & ' ( ) * + , - . / : ;
< = > ? @ [ ] ^ _ ` { | } ~
```

- 空字符。
- 制表字符。

Microsoft FORTRAN字符集的校验系列就是ASCII系列。

2.2.1 空格

除了下面列出的以外，在Microsoft FORTRAN源程序中，空格字符没有什么意义。所以，你可使用空格使程序易于阅读。例外情况如下：

- 在字符常量或Hollerith域中，空格是有意义的。
- 第六列的空格或0表示初始行（初始行的解释见3.2节：“行”）。
- BZ 编辑描述符有效时，数值输入域中的空格（不是开头的空格）解释为零。用BN和BZ编辑描述符（在4.8.1.10节：“空格解释”中描述），或用OPEN语句中的BLANK

关键字（在5.3.38节、“OPEN语句”中描述）可以改变这种状态。

2.2.2 制表字符（Tab）

制表字符的解释取决于制表字符所在的列：

列	解释
1—5	在源程序行中，跟在制表字符后的字符，如同第七列中字符一样被解释。
6—72	除非在字符或Hollerith常量（在4.8.1.2节描述）中．制表字符解释为空格。字符或Hollerith常量中的制表字符解释为制表字符。

2.3 名字

变量、数组、子程序和程序都由名字标识。Microsoft FORTRAN定义一些名字，用户定义其它的名字。名字是字母数字字符所组成，而且必须遵守下列规则：

- 名字中的第一个字符必须是字母；其余字符必须为字母数字。注意，Microsoft FORTRAN允许美元符号在名字的校验系列中看作Z后面的一个字母字符。
- 编译程序忽略空格。这样，象low voltage和lowvoltage这样的变量名对编译程序来说是相同的。
- 除非设置了$NOTRUNCATE元命令，否则只有前六个字母数字字符是有意义的，其余部分被忽略。空格字符不算在内：除非设置了$NOTRUNCATE元命令，delcate和d e l cate都解释为delcate。
- 编译程序限制名字为31个字符。你的操作系统或连接程序可能对名字长度有其它限制。有关系统的特定信息请参阅《Microsoft FORTRAN编译程序用户指南》。

与其它语言不同，FORTRAN中的关键字不是保留字。编译程序通过上下文识别关键字。例如，程序可有名为IF，read或Goto的数组。然而，使用这些名字会使程序较难阅读和理解。为了可读性，程序员应该避免使用与FORTRAN语句部分相似的名字。请看下面二条语句：

```
DO S INC=1.20
DO S INC=1,20
```

第一条语句将值1.20赋给名为DOSINC的变量。第二条语句是一个DO循环的第一条语句。注意，两条语句之间仅有的不同之处是第一条包含一个句点而第二条包含一个逗号。

下列三个预定义的名字不能使用：

1.-main，这是主程序的外部名字。（“main”允许在一定的条件下使用，但建议不要使用。更多的信息见3.8节：“主程序”。）

2. COMMQQ，这是空的公共块的名字。

3. BLKDQQ，这是数据块子程序的缺省名。

此外，所有以二个下划线字符（__）开头或以QQ结尾的名字，如__main或MAINQQ均由编译程序保留。如果你需要使用以二个下划线字符开头或以QQ结尾的名字，请用ALIAS属性（在2.6.1节描述）。

2.3.1 全程名和局部名

名字有二种基本类型：

类型	描述
全程名	在某个给定程序的任何地方都可识别全程名，所以它们只在该程序的任一地方作了一次全程定义。所有子程序，函数，公共块和程序名都是全程的。例如，你若在一个程序中使用名为Sort的子程序，就不能在该程序中将一个函数命名为Sort。 但是，只要你不在不同的程序单位内引用全程名Sort。你可以将名字Sort用作一个局部名（下面描述）。例如，只要说明变量Sort的子程序不调用函数Sort，含有函数Sort的程序就可以包含该子程序。 公共块名是全程名的一个特殊情况。程序中的公共块名也可用作同一程序的一个局部名。之所以允许这样做，是因为公共块名总是包括在斜杠之间，因而可与其它名字区分开。例如，如果你的程序含有名为／distance／的公共块，你也可以将该程序中的一个数组命名为distance（数组具有局部名）。
局部名	局部名仅在一个程序单位里定义。在同一程序的另一个程序单位中，相同的名字可再次作相同或不同的定义。 所有变量，数组，参数和语句函数都有局部名。 语句函数的参数是局部名的一个特殊情况。这些参数仅在语句函数的语句中定义。但是，如果在语句函数的语句之外使用参数名，那么包含在子程序中的局部变量必须与有相同名字的语句函数参数有相同的数据类型。更多的信息请参见5.3.48节：“语句函数语句”。

2.3.2 未说明的名字

如果一个名字没有给出明确的定义，编译程序按首次遇到的上下文将该名字分类。如果已经说明了 $DECLARE元命令，那么在首次使用说明语句中尚未说明的任何变量就会产生警告信息，下表解释来说明的名字的分类方法：

名字的使用	分类
作为变量，或在函数调用中	变量或函数的返回值的类型由名字的第一个字母确定。按缺省规则，以字母I，J，K，L，M或N（大写或小写）开头的变量名为INTEGER（整数）型。以任何其它字母或美元符号开头的变量名为REAL（实数）型。使用IMPLICIT语句可改变类型和第一个字母字符（包括美元符号）之间的对应关系。更多的信息请参见5.3.31节：“IMPLICIT语句”。
作为CALL语句的目标	编译程序假定该名字为子程序名。 如果子程序的定义在调用该子程序的源文件中的CALL语句之前，那么编译程序就校验在CALL语句中实际参数的个数和类型是否与在相应的SUBROUTINE语句中说明的那些参数个数和类型相一致。仍要注意的是，使用INTERFACE语句（在5.3.34节描述）

可确保子程序的调用具有正确的参数个数和类型。

在函数引用中

编译程序假定该名字为一个函数名。

如果函数的定义在引用该函数的源文件中的函数之前，那么编译程序就校验函数引用中实际参数的个数和类型是否与在相应的FUNCTION语句中说明的那些参数个数和类型相一致。仍要注意的是，INTERFACE语句（在5.3.34节描述）可确保子程序的调用具有正确参数个数和类型。

2.4 数据类型

Microsoft FORTRAN中的数据有五个基本类型：

1.整数（INTEGER，INTEGER*1，INTEGER*2和INTEGER*4）

2.实数（REAL，REAL*4，DOUBLE PRECISION或REAL*8）

3.复数（COMPLEX，COMPLEX*8和COMPLEX*16）

4.逻辑（LOGICAL，LOGICAL*1，LOGICAL*2和LOGICAL*4）

5.字符（CHARACTER[(*n)]，其中1<=n<=32，767）

变量，数组，数组元素，有符号名的常量，或函数的数据类型可在说明语句中说明。如如还没有说明数据类型，编译程序就根据名字的第一个字母确定其数据类型。（正如2.3.2节："未说明的名字"所描述的一样）。类型语句也可以包括维数信息，而且可用于变量和数组初始化。类型语句的详细描述见第五章："语句"。

下列各小节（2.4.1至2.4.6节）描述各个数据类型。内存要求如表2.1所示。

表2.1 内存要求

Table 2.1

Memory Requirements

Type	Bytes	Notes
INTEGER	2 or 4	Defaults to 4 bytes. The setting of the $STORAGE metacommand determines the size of INTEGER and LOGICAL values.
INTEGER*1	1	
INTEGER*2	2	
INTEGER*4	4	
REAL	4	Same as REAL*4.
REAL*4	4	
DOUBLE PRECISION	8	Same as REAL*8.
REAL*8	8	
COMPLEX	8	Same as COMPLEX*8.
COMPLEX*8	8	
COMPLEX*16	16	
LOGICAL	2 or 4	Defaults to 4 bytes. The setting of the $STORAGE metacommand determines the size of INTEGER and LOGICAL values.

LOGICAL*1	1	
LOGICAL*2	2	
LOGICAL*4	4	
CHARACTER	1	CHARACTER and CHARACTER*1 are the same.
CHARACTER*n	n	Maximum n is 32,767

2.4.1 整数数据类型

整数数据类型是整数一个子集。整数值是相应整数的精确表示。表2.2展示了整数的不同类型，每种类型占用内存的字节数及其范围。注意，除非使用 $STORAGE元命令说明字节内存分配，否则以INTEGER说明的变量和函数内存分配与INTEGER*4相同。$STORAGE元命令也确定整数常量的隐含存储大小。例如，如果说明了 $STORAGE：2元命令，按缺省，整数常量为2字节长。但是，如果常量在INTEGER*2 范围之外，该常量就给予4字节存储。

表2.2 整数

Table 2.2

Integers

Data Type	Bytes	Range
INTEGER*1	1	−127 to 127
INTEGER*2	2	−32,767 to 32,767
INTEGER*4	4	−2,147,483,647 to 2,147,483,647
INTEGER	2 or 4	Depends on setting of $STORAGE

注意，值的范围不包括在给定字节所表示的最小负数。数−128超出了INTEGER*1数据类型的范围；−32，768超出了INTEGER*2数据类型的范围；−2，147，483，648超出了INTEGER*4数据类型的范围。这些数被看作未定义的数，由编译程序用于错误检验。

■ 语法

按缺省规则，常量均解释为10进制数。为了说明不是10进制的常量，可使用下列语法：

〔{Sign}〕 〔{〔{base}〕#}〕consfant

其中Sign（符号）是一个可选择的正号或负号。base（基）可以从2至36之间的任意整数是。如果省略了base但说明了#，该整数就解释为16进制（基为16）。如果base和#均有省略，该整数就解释为10进制。对11至36进制，字母A到Z代表大于10的10进制数。例如，对36进制，A代表10，B代表11，C代表12，等等，直到代表35的Z。注意，字母的大小写没有意义。

■ 例子

例如，下列七个整数都赋予等于十进制值3,994，575的一个值：

```
I = 2#111100111001111001111
m = 7#45644664
J = +8#17171717
K = #3CF3CF
n = +17#2DE110
L = 3994575
index = 36#2DM8F
```

整数常量中小数点是不允许的。

整数常量必须在前面说明的范围之内。但是，对基数不为10的数，编译程序可读大至2^{32}的超范围数。这些数被解释为相应内部表示的负数。例如，16#FFFFFFFF结果变为-1的算术值。

2.4.2 单精度IEEE实数数据类型

单精度IEEE（Institute of Elecfrical and Electronics Engineers,Inc·）实数数据类型（REAL或REAL*4）是实数的一个子集。单精度实数值通常希望为实数的近似值，占用内存4个字节。这一数据类型的精度在六至七位小数之间。注意，你可以说明大于六位的数字，但是，对单精度实数仅仅小数点后的前六位有效。单精度实数值的范围包括负数：从约：-3.4028235E+38至-1.1754944E-38，正数：从约+1.1754944E-38至3.4028235E+38。

■ 语法

实数常量的格式如下：

〔{Sign}〕〔{integer}〕〔{.}〕〔{fraction}〕〔{Eexponent}〕

参数	值
sign	一个符号（+或-）
integer	一个整数。可省略integer或fractien，但两者不能同时省略。
	一个小数点
fraction	小数部分，由一个或多个十进制数字组成。可省略integer或fraction，但两者不能同时省略。
Eexponent	指数部分，由选择性的带符号的一位或二位数字的整数常量组成。指数表示，指数之前的值要乘以10的exponent的值的次方。

■ 例子

下列实数常数量均代表同一个实数（1.2300）：

+1.2300E0	·12300E2	1.23E0
123E-2	+1.2300	123.0E-2
·000123E+4	1230E-3	

2.4.3 双精度IEEE实数数据类型

双精度数数据类型（REAL*8或DOUBLE PRECISION)是实数的一个子集。这个子集比单精度实数数据类型所对应的子集大。双精度实数值通常为期望实数的近似值，占用8字节内存，其精度大于15位十进制数字。注意，你可以说明更多的数字位，但只有前15位

有效。双精度实数值的范围包括负数：从约－1.797693134862316D＋308至－2.225073858507201D－308，和数0。也包括正数：从约＋2.225073858507201D＋308至＋1.797693134862316D＋308。

双精度实数常量的格式与单精度实数常量相同，只是以字母D替代字母E来表示指数，而且指数部分是必须的。如果省略了指数，则该数就被解释为单精度常量。下列双精度实常量均代表千分之五十二：

5.2D－2　　＋.00052D＋2　　.025D0　　52.000D－3　　52D－3

注意，由于没有说明指数，象。052这样的常数被看作是单精度值。

2.4.4　复数数据类型

COMPLEX或COMPLEX＊8 数据类型是单精度实数的一个有序对。COMPLEX＊16是一个有序的双精度实数对。有序对中的第一个数代表复数中的实数部分，而有序对中的第二个数代表虚数部分。COMPLEX或COMPLEX＊8的实数和虚数部分都是REAL＊4数。所以COMPLEX或COMPLEX＊8数占用8个字节内存。COMPLEX＊16数的实数和虚数部分都是REAL＊8数，所以COMPLEX＊16占用16字节内存。

■ 语法

〔〔sign〕〕（real，imag）

参数	值
sign	一个符号（＋或－）。一旦说明，sign对real和image两者都有效。
real	一个整数或实数，代表实数分。
imag	一个整数或实数，代表虚数部分。

例如，复数（7,3.2）代表7.0＋3.2i。数－（－.11E2，＃5F）代表11.0－95.0i。

.4.5　逻辑数据类型

逻辑数据类型包括两个逻辑值：·TRUE·和·FALSE·。一个LOGICAL变量占用2个或4个字节内存，这取决于$STORAGE元命令的设置。缺省为4个字节。逻辑型变量的意义不受$STORAGE元命令的影响。该元命令的根本作用在实现与ANSI要求的兼容性，即逻辑型、单精度实数和整数变量大小相同。

LOGICAL＊1的值占用一个字节，要么是0（·FALSE·），要么是1（·TRUE·）。LOGICAL＊2的值占用二个字节：低有效（第一个）字节为LOGICAL＊1的值；高有效字节未定义。LOGICAL＊4变量占用二个字：低有效（第一个）字为LOGICAL＊2的值；高有效字未定义。

2.4.6　字符数据类型

字符的值是一串有省字号（’）括起来的一个或更多的可显示的ASCII字符。

注意

省字号象单右引号（’）输入，但不是单左引号（‘）。注意，在例子中使用的字体，如’string’，省字号看上去为‘。

把串括起来的省字号不与该串一起存储。为了在串中表示一个省字号，可说明两个之间

没有空格的连续省字号。在字符型常量中，空格字符和制表字符是允许的，而且有意义。字母字符的大小写是有意义的。串可包含ASCII字符集中任何可显示的字符。你可使用C串（如2.4.6.1节所描述）去定义不可显示的字符串，或说明空串。

字符的值的长度等于省字号之间字符的个数。一对省字号算作一个字符。具有符号名的字符变量、字符数组元素、字符函数或字符常量的长度必须在1和32，767之间。其长度可由用下列任一方法说明：

- 在1—32，767范围内的无符号整数常量。
- 圆括号内的整数常量表达式。
- 圆括号内的星号：（*）。

更多的信息，参见5.3.6节：“CHARACTER语句”。

下面列出了一些实例字符常量。

String	Constant
'String'	String
'1234!@#$'	1234!@#$
'Blanks count'	Blanks count
''''''	''
'Case Is Significant'	Case Is Significant
'"Double" quotes count'	"Double" quotes count

注意，FORTRAN源程序行长度为72个字符（编译程序忽略73—80列内的字符），而且，不足72个字符的行则填满空格。因此，当字符常量超出一行的界线时，其值就包括填入该格的所有空格。请看下列FORTRAN语句：

```
heading(second column)='Acceleration of Particles
$from Group A'
```

该语句将数组元素heading(secondcolumn)置为'Acceleration of Particles from Group A'。Particles和from之间有14个空格，因为字Particles在赋值语句的58列结束，添加了14个空格填满该行。

在公共块中进行内存分配时，字符变量为串中的每一个字符占一个字节内存。字符变量不管字界而分配给连续的字节。但是，当字符和非字符变量被迫分配在同一个公共块时，编译程序假定，紧跟在字符变量后的非字符变量总是在字界上开始。有关公共块中字符变量的更多信息，请参见5.3.8节：“COMMON语句”。

2.4.6.1 C串

C语言中的串值中止于空字符（CHAR(0)），而且可能包含不可显示字符，（比如新行和回退）。使用反斜杠作为替换字符，后跟表示期望的不可显示字符的一个字符，就能说明这些字符。在Microsoft FORTRAN中，这类串可用标准串常量后跟字符C来说明。这样，标准串常量就解释为C语言常量。反斜杠被看作转义标志，而且在串结尾自动添加一个空字符（即使该串已经以空字符结束）。表2.3列出了合法的转义系列。如果串中含有不在表中的转义系列（如\Z），编译程序忽略反斜杠。

表2.3　C串转义系列

系列	字　符
\n	新行
\t	水平的制表符
\v	垂直的制表符
\b	回退符
\r	回车
\f	换页
\'	单引号'
\"	双引号
\\	反斜杠
\000	八进制位模式
\xhh	十六进制位模式
\a	振铃

" 在FORTRAN中，必须打入\" 以表示这个转义系列。

串必须是如2.4.6节："字符数据类型"所描述的合法FORTRAN串。注意，因为C串转义系列开始时被看作FORTRAN串，所以串本身中的所有引号必须是由击打两次单引号键而产生的双引号（" "）。转义系列\'a会产生语法错，这是因为FORTRAN将该引号解释为一个串的结束。正确的形式为\" a。C串和普通串仅在说明该串的值的方法上不同。例如，当分配填加或截除一个串时，编译程序对两种串类型作相同处理。

系列\000和\xhh允许将ASCII字符集中的任意字符作为一至三位八进制或一至二位十六进制字符代码给出。0数字必须在0—7的范围内，h数字必须在0—F范围内。例如，C串'\010'C和'\x08'C均代表由空字符跟着的一个回退字符。

C串'\\abcd'C等价于串'\abcd'后再附加一个空字符。串" C代表ASCII空字符。注意，字符常量" 是非法的，因为其长度为0，但" C是合法的，因为其长度为1 。

2.4.6.2　字符子串

从类型上讲，子串为字符类型，用于访问字符变量的一个连续部分。

■ 语法

variable ({{first}}:{{last}})

参数	描　述
variadle	一个字符变量或字符数组中的一个元素。
first	定义子串中第一个（最左边）字符的一个算术表达式。编译程序将first值截除为一个整数值。first的值缺省为1，所以，如果没有说明first，则子串开始于串中的第一个字符。
last	定义子串中最后一个（最右边）字符的一个算术表达式。编译程序将last值截除为一个整数值。last的值缺省为串长度，所以，如果没有说明

last，则子串结束于串中的最后一个字符。

如果出现$STRICT元命令，对first和last使用非整数表达式会造成错误。注意，variable(:)等价于variable。子串的长度为last－first＋1。例如，对包含串'Jane Doe'的10字节字符变量name，name(:5)为'Jane'，name(5＋1:)为'Doe'，name(:)为'Jane Doe'。

警告

假设length为字符变量的长度，则必须满足下列关系：

●first<=last

例如，对10字节字符变量name，name(6:5)是不允许的。

●1<=first<=length

例如，name(0:4)和name(11:12)是不允许的。

●1<=last<=length

例如，name(:0)和name(:11)是不允许的。

如果$DEBUG元命令处于开启状态，则一旦不满足这些关系就会产生错误。如果$DEBUG不是处于开启状态，其结果为未定义。

■ 例子

```
C This program writes the second half of
C the alphabet, followed by the first half
      CHARACTER alpha*26
      alpha='abcdefghijklmnopqrstuvwxyz'
      WRITE(*,*)alpha(14:),alpha(:13)
      END
```

2.5 数组

数组中元素的个数仅由变量内存所限制。但是，如果说明了$STRICT元命令，则一旦使用超过七维的数组，就会出现警告信息。为了引用数组元素，须使用下列语法：

■ 语法

array(subscripts)

参数	值
array	该数组的名字。如果数组类型没有在类型语句中说明，则数组元素的类型由array的首字母表示。
subscripts	下标表达式。如果下标表达式多于一个，则必须由逗号分开。下标表达式的个数必须与所说明的数组维数相同。有关说明数组的更多信息，请见5.3.12节，“DIMENSION语句”。 每个下标都必须是算术表达式。表达式的结果用截除法取整。如果说明了$STRICT元命令，则每个下标都必须是整数表达式。函数引用和数组元素引用都是允许的。subscripts的值可以是正数，负数，或0。

■ 例子