关于FORTRAN的几篇译文

韩淑娟
姚兆炜
译校

中国人民解放军京字一一六部队

1974.9.

总 目 录

译 校 者 的 话

FORTRAN语言是国外较为广泛使用的程序设计语言之一。这里翻译了三篇有关FORTRAN语言的文章：

（一）美国标准协会FORTRAN语言标准化发展的历史概况

（二）自动数据处理系统信息加工的程序设计语言

（三）美国标准FORTRAN

美国标准基本FORTRAN

其中的(二)，是以FORTRAN语言与基本FORTRAN语言对照方式列出的。为了减少篇幅和易于了解文本的异同，我们翻译了(二)的全文，而对(三)仅以对照方式列出其与(二)的不同之处。(三)中的各附录是(二)里所没有的，故此处全部译出。

在译文最后附有部分英一中名词对照表，以供参阅。

由于译校者水平所限，译文中错误和缺点在所难免，敬请同志们给予批评指正。

一、美国标准协会FORTRAN语言标准化发展的历史概况*

W.P.Heising

1960年美国标准协会（简称ASA）成立了计算机和信息加工部分委员会X3，它隶属于商业装备制造商协会。接着ASAX3又成立了X3.4分委员会，从事一般程序设计语言标准化方面的工作。1962年5月17日X3.4根据决议成立了一个工作组，即X3.4.3—FORTRAN，以开展有关美国标准FORTRAN语言的一些规划工作。

决定：

由X3.4组成一个FORTRAN工作组，即大家所熟知的X3.4.3-FORTRAN，并规定：

范围. 提出FORTRAN语言的一些标准。

组织. 准备设立一个政策委员会和一个技术委员会。政策委员会对X3.4的工作组所要完成的任务负应有的责任。它决定总政策。以及诸如语言的内容、对技术委员会的指导等。

政策委员会会员职责. 根据X3.4领导委员会的决定制订指导方针(以后可能修改)，并包括下列几个方面：

a. 在实际研究或使用中，对每一个FORTRAN语言的实现，委任一名承担者选出的代表和一名使用者选出的代表。

b. 不参加活动的代表可被除名。

c. 委任协会的成员（无表决权，但可参加讨论）。

技术委员会. 在政策委员会的领导下，将研究并提出FORTRAN语言的标准文本。技术委员会将进行实际调查研究，并向政策委员会提出报告。

1962年6月25日，X3.4.3邀请对FORTRAN语言标准化感兴趣的有关制造商和各使用团体参加会议。第一次会议是在1962年8月13—14日在纽约城召开。决定X3.4.3继续进行工作，因为：

(1) FORTRAN语言标准化是需要搞的；(2) 过去已有相当广泛有关这方面人员的

---

* 本文译自《Communications of the Association for Computing Machinery》*vol.* 7，№.10，1964，*P.*590.

代表参与这项工作。

1962年8月14日一致通过一项决议。

“ASA的X3.4.3工作组的任务是：产生一个文本或若干文本，这种文本定出ASA的标准或者是规定FORTRAN语言的标准。所得的标准语言对过去所谓的FORTRAN语言以及它的各种变形来说，应是清晰、可认的。作为考虑的准则和评价各种语言成分的因素有以下几个方面（这里所列的先后顺序和本身的重要性是没有关系的）：

a。 应对人们使用方便；

b。 与以往FORTRAN语言的使用并不矛盾；

c。 应用的范围；

d。 扩充的潜力；

e。 实现简易，即指编译和执行的效率而言。

FORTRAN语言的标准文本将易于各种程序(用ASA的标准FORTRAN语言写的在不同机器上运转。这种标准语言为引用本文件的使用者(他们是希望达到这一目的的)和制造商(其程序设计的产品就成为可能)服务。标准文本提出的内容和方法将辨认这种目的。”

工作组一致赞同下列意见：(1)对应于大家了解的所谓FORTRAN Ⅳ确定一种标准文本；(2)为小型或中型计算机制订一种接近于FORTRAN Ⅰ的FORTRAN语言的标准文本，但宜于修改，以便与FORTRAN Ⅳ相适应。因此，分别成立了两个技术委员会：X3.4.3-Ⅳ和X3.4.3-Ⅰ去制订草案。在制订草案过程中，这两个技术委员会完成了大量细致工作。

1963年5月，X3.4.3-Ⅰ技术委员会完成并提出了一项草案。1964年8月，指定向技术成果审定委员会提出报告：“关于X3.4.3-Ⅰ所定的草案和X3.4.3-Ⅳ技术委员会所定的工作草案的比较。”过去曾暴露出在文体、术语和内容等方面不相一致的地方，以及好多矛盾之处。在1964年4月，X3.4.3-Ⅳ技术委员会完成了FORTRAN语言的一项草案。1964年6月，X3.4.3收到并比较了这两个草案，同时(1)解决了内容上的矛盾和(2)解决了文体和术语方面的不一致。这是由一方面保留原来内容，同时对照X3.4.3-Ⅳ文件的文体，并改写X3.4.3-Ⅰ的文件完成的。为了减少混淆起见，X3.4.3决定：分别称这两种语言为**基本FORTRAN**和**FORTRAN**。

二、自动数据处理系统信息加工的程序设计语言*

FORTRAN 对照 基本FORTRAN

目　录

* 本文译自《Communications of the Association for Computing Machinery》*vol.* 7，№.10，1964，*P.*591—625。

* 号表示在基本FORTRAN中为空白。

美 国 标 准 FORTRAN

1. 引言

1.1 目的。本方案建立用FORTRAN语言所表达的程序形式，并解释用这种语言所写的程序，目的是：促使这种程序可高度互换，以供各种自动数据处理系统之用。倘若加工程序接受并如规定那样解释这里所描述的形式和关系的话，那末，它应遵从本方案。

就不影响所描述的形式和一些关系的解释这一点来说，任何必要的陈述，如果不满足该要求时，本方案并不提供解释。同时，任何禁止的陈述，当违背了这种禁止时，本方案也不提供解释。

1.2 范围。本方案建立：

(1) 用FORTRAN语言写的程序形式。

(2) 写输入数据（即在各种自动数据处理系统上运算的并为这种程序所加工的数据）的形式。

(3) 解释这种程序含义的规则。

(4) 各种自动数据处理系统（从这些系统的若干解释规则中规定一种解释规则）使用这种程序所得结果的输出数据形式。

本方案并不规定下列诸项：

(1) 供数据处理系统用的把程序变形的技巧（这种技巧和数据处理系统合在一起称为加工程序）。

(2) 转录这种程序的方法，以及它们的输入数据如何记于数据处理介质，或如何从数据处理介质输出它们的输出数据的

美国标准基本 FORTRAN

1. 引言

1.1 目的。本方案建立用FORTRAN语言所表达的程秩形式，并解释用这种语言所写的程序，目的是：促使这种程序可高度互换，以供各种自动数据处理系统之用。倘若加工程序接受并如规定那样解释这里所描述的形式和关系的话，那末，它应遵从本方案。

就不影响所描述的形式和一些关系的解释这一点来说，任何必要的陈述，如果不满足该要求时，本方案并不提供解释。同时，任何禁止的陈述，当违背了这种禁止时，本方案也不提供解释。

1.2 范围。本方案建立：

(1) 用FORTRAN语言写的程序形式。

(2) 写输入数据（即在各种自动数据处理系统上运算的、并为这种程序所加工的数据）的形式。

(3) 解释这种程序含义的规则。

(4) 各种自动数据处理系统（从这些系统的若干解释规则中规定一种解释规则）使用这种程序所得结果的输出数据形式。

本方案并不规定下列诸项：

(1) 供数据处理系统用的把程序变形的技巧（这种技巧和数据处理系统合在一起称为加工程序）。

(2) 转录这种程序的方法，以及它们的输入数据如何记于数据处理介质，或如何从数据处理介质输出它们的输出数据的

方法。

（3）在数据处理设备上当使用这种程序时如何配置所要求的手动操作，以及在这种设备上如何使用这种程序控制。

（4）当那些解释规则都不能阐明这种程序的一种明确解释时所得到的一些结果。

（5）程序的大小或复杂程度超过具体数据处理系统的容量或具体加工程序的能力。

（6）数量的范围或精确度。

2．基本术语

本节先引进一些基本术语和若干基本概念。在后面几节中再分别给以严格的论述。这里只列举有关语法形式及一些具体词含义的某些假定。

能用作为完整计算过程的一个程序称为**可执行程序**(9.1.6)。

可执行程序恒由一个主程序（main program)并可能有一个或几个子程序(subrogram)组成的(9.1.6)。

一个**主程序**就是一组语句和注解，但它不包含函数语句、子例行程序（简称子例程subvoutine)语句或数据块语句（9.1.5)。

子程序类似于主程序，但用数据块语句、函数语句或子例程语句放在头上。子程序前面列有数据块语句的称为说明子程序。子程序前面列有函数语句或子例程语句的称为过程子程序(9.1.3,9.1.4）。

术语**程序单位**(program unit）是指主程序或子程序(9.1.7）。

除说明子程序外，任何程序单位都可引用**外部过程**（第9节）。

用FORTRAN语句定义的外部过程

方法。

（3）在数据处理设备上当使用这种程序时如何配置所要求的手动操作，以及在这种设备上如何用这种程序控制。

（4）当那些解释规则都不能阐明这种程序的一种明确解释时所得到的一些结果。

（5）程序的大小或复杂程度超过具体数据处理系统的容量或具体加工程序的能力。

（6）数量的范围或精确度。

2．基本术语

本节先引进一些基本术语和若干基本概念。在后面几节中再分别给以严格论述。这里只列举有关语法形式及一些具体词含义的某些习惯用法。

能用作为完整计算过程的一个程序称为**可执行程序**（9.1.6)。

可执行程序恒由一个主程序（main program）并可能有一个或几个子程序(subprogram)组成的（9.1.6）。

一个**主程序**就是一组语句和注解，但它不包含函数语句或子例行程序（简称子例程subroutine）语句（9.1.5）

一个**过程子程序**类似于主程序，但前面列有函数语句或子例程语句。过程子程序有时被认为是和子程序一样的(9.1.3)子程序。

术语**程序单位**（program unit)是指主程序或子程序（9.1.7）。

任何程序单位可引用**外部过程**（第9节）。

用FORTRAN语句定义的外部过程

称为**过程子程序**。外部过程也可用其它方法定义。外部过程可以是一个外部函数或外部子例程。前面列有函数语句用 FORTRAN语句定义的外部函数称为**函数子程序**。前面列有子例程语句用FORTRAN语句定义的外部子例程称为**子例程子程序**（第 8 节和第 9 节）。

任一程序单位都是由若干**语句**和**注解**组成的。一个语句可分成若干称为**行**的自然段，第一行称为**初始行**，其余的都称为**继续行**（3.2）

有一种称为注解的行，它不是语句，而只为文件的用途含义提供信息（3.2）。

FORTRAN中的语句分为两大类，即可执行的语句和不可执行的语句。可执行语句规定程序要实现的动作，不可执行语句描述程序的使用、运算对象的特点、编辑信息、语句的功能或数据的安排（7.1,7.2）等。

语句的语法元素是**名字**和**运算符**。名字被用来指称一些对象，诸如数据或一些过程。运算符（包括命令动词）规定对命名了的对象所施的动作。

象**数组名字**这样一类名字应专门讨论。由数组名字所表示的数值数组的名字和维数应在用之前就说明。可用一个数组名字代表整个数组。带有下标的数组名字可用来表示该数组的(5.1.3) 某个具体元素。

数据名字和算术（或逻辑）运算符可组成算术（或逻辑）表达式以求值。这些值是由对命名了的数据执行指定的运算得到的（第 6 节）。

用在FORTRAN语言中的标识符是一些名字和数。数据、过程都给以命名。语句用数来标记。输入/输出部件都编以号码，或用一个名字指称，该名字的值就

称为过程子程序。外部过程也可用其它方法定义。外部过程可以是一个外部函数或外部子例程。前面列有函数语句用 FORTRAN语句定义的外部函数称为函数子程序。前面列有子例程语句用 FORTRAN语句定义的外部子例程称为子例程子程序（第 8 节和第 9 节）。

任一程序单位都是由若干语句和注解组成的。一个语句可分成若干称为行的自然段，第一行称为初始行，其余的都称为继续行（3.2）。

有一种称为注解的行，它不是语句，而只为文件的用途含义提供信息（3.2）。

FORTRAN中的语句分为两大类，即可执行的语句和不可执行的语句。可执行语句规定程序要实现的动作，不可执行语句描述程序的使用、运算对象的特点、编辑信息、语句的功能或数据的安排（7.1,7.2）等

语句的语法元素是名字和运算符。名字被用来指称一些对象，诸如数据或一些过程。运算符（包括命令动词）规定对命名了的对象所施的动作。

象数组名字这样一类名字应专门讨论。由数组名字所表示的数值数组的名字和维数应在用之前就说明。可用一个数组名字代表整个数组。带有下标的数组名字可用来表示该数组的（5.1.3)某个具体元素。

数据名字和算术运算符可组成算术表达式以求值。这些值是由对命名了的数据执行指定的运算得到的（第 6 节）。

用在FORTRAN语言中的标识符是一些名字和数。数据、过程都给以命名。语句用数来标记。输入/输出部件都编以号码，或用一个名字指称，该名字的值就

是那个部件的数字表示（见第3、6、7节）。

在本文件的好几个地方，有一些语句带有若干与实体结合的表。除明显的例外之外，在所有这种情形里，都假定该表至少含有一个实体。例如，在语句：

SUBROUTINEs $(a_1, a_2, \cdots, a_n)$

中，假定在括弧内至少有一个符号名字是列在表里的。所谓一个**表**可看作为相同的一些元素的集合，每个元素和它的后继元素之间用逗号分开。而且在一句句子中，在专用的场合名词的复数形式，将也被假定记为该名词的单数形式，只要上下文允许这种解释的话。

术语**引用**是用作具有特殊意义的动词（见第5节中的定义）。

3．程序形式

每个程序单位是由组成行和语句的一些字符构成的。

3.1 **FORTRAN字符集**。写一个程序单位用到的字符有：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、0、1、2、3、4、5、6、7、8、9以及：

字　符	字符的名称
	空白
=	等号
+	加号
−	减号
*	星号
/	斜线
(	左括弧
)	右括弧
,	逗号
.	十进小数点
$	币符

是那个部件的数字表示）。

在本文件的好几个地方，有一些语句带有若干与实体结合的表。除明显的例外之外，在所有这种情形里，都假定该表至少含有一个实体。例如，在语句：

SUBROUTINEs $(a_1, a_2, \cdots, a_n)$

中，假定在括弧内至少有一个符号名字是列在表里的。所谓一个**表**是可看作为相同的一些元素的集合，每个元素和它的后继元素之间用逗号分开。而且在一句句子中，在专用的场合名词的复数形式将也被假定记为该名词的单数形式，只要上下文允许这种解释的话。

术语**引用**是用作具有特殊意义的动词（见第5节中的定义）。

3．程序形式

每个程序单位是由组成行和语句的一些字符构成的。

3.1 **FORTRAN字符集**。写一个程序单位用到的字符有：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、0、1、2、3、4、5、6、7、8、9以及：

字　符	字符的名称
	空白
=	等号
+	加号
−	减号
*	星号
/	斜线
(	左括弧
)	右括弧
,	逗号
.	十进小数点

上述各字符的顺序是任意的。

3.1.1 **数字**。数字指的是下列10个字符之一：0,1,2,3,4,5,6,7,8,9。除非另有规定外，一个数字串将被解释为十进制中的一个数。

八进数字指的是下列8个字符之一：0,1,2,3,4,5,6,7。它们只用在停语句(7.1.2.7.1)和暂停语句(7.1.2.7.2)中。

3.1.2 **字母**。字母指的是下列26个字符之一：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。

3.1.3 **字母数字字符**。字母数字字符指的是字母或数字。

3.1.4 **特殊字符**。特殊字符指的是下列11个字符之一：空白、等号、加号、减号、星号、斜线、左括弧、右括弧、逗号、十进小数点和币符。

3.1.4.1 **空白字符**。除规定使用外(3.2.2, 3.2.3, 3.2.4, 4.2.6, 5.1.1.6, 7.2.3.6和7.2.3.8)，空白字符是没有意义的，并只要遵守3.3中对继续行的限制，可任意使用之以改善程序的外形。

3.2 **行**。一行是由72个字符组成的字符串。除5.1.1.6和7.2.3.8中所讲的外，所有字符必须为FORTRAN字符集里的字符。

在每一行中，各字符所在的位置称为列，并顺次记为1,2,3,…,72。这些数字表示在该行中从左到右各字符的顺序位置。

3.2.1 **注解行**。在某一行的第1列中有字母C表示该行为注解行。注解行后必须紧跟初始行、另一注解行或结束行。

注解行丝毫不影响程序的原义，且对程序设计者提供方便是有用的。

3.2.2 **结束行**。结束行是在1至6

上述各字符的顺序是任意的。

3.1.1 **数字**。数字指的是下列10个字符之一：0,1,2,3,4,5,6,7,8,9。除非另有规定外，一个数字串将被解释为十进制中的一个数。

八进数字指的是下列8个字符之一：0,1,2,3,4,5,6,7。它们只用在停语句(7.1.2.7.1)和暂停语句(7.1.2.7.2)中。

3.1.2 **字母**。字母指的是下列26个字符之一：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。

3.1.3 **字母数字字符**。字母数字字符指的是字母或数字。

3.1.4 **特殊字符**。特殊字符指的是下列10个字符之一：空白、等号、加号、减号、星号、斜线、左括弧、右括弧、逗号和十进小数点。

3.1.4.1 **空白字符**。除规定使用外(3.2.2, 3.2.3, 3.2.4, 7.2.3.6和7.2.3.8)，空白字符是没有意义的，并只要遵守3.3中对继续行的限制，可任意使用之以改善程序的外形。

3.2 **行**。一行是由72个字符组成的字符串。除7.2.3.8中所讲的外，所有字符必须为FORTRAN字符集里的字符。

在每一行中，各字符所在的位置称为列，并顺次记为1,2,3,…,72。这些数字表示在该行中从左到右各字符的顺序位置。

3.2.1 **注解行**。在某一行的第1列中有字母C表示该行为注解行。注解行后必须紧跟初始行、另一注解行或结束行。

注解行丝毫不影响程序的原义，且对程序设计者提供方便是有用的。

3.2.2 **结束行**。结束行是在1至6

列中为空白字符的行，而在7至72列中，字符E、N、D按顺序出现，在E、N、D前，或在E、N、D间，或在E、N、D后可有空白字符出现。结束行指示加工程序：所编的程序单位（9.1.7）结束。每个程序单位必须用一结束行以示终止。

3.2.3 **初始行**。初始行既非注解行又非结束行，而是在第6列中出现数字0或空白字符的行。1至5列写语句标号，或全为空白字符。

3.2.4 **继续行**。继续行是在第6列中除数字0或空白字符外，可为任何字符，且在第1列中不为字符C的行。

继续行只可跟在某一初始行或另一继续行后。

3.3 **语句**。语句是由初始行其后顺序跟最多为19个继续行所组成的。语句写在各行的7至72列中。语句中字符的顺序是初始行的7至72列，当需要时，后跟第一继续行的7至72列，再跟下一继续行的7至72列等等。

3.4 **语句标号**。语句可任意给以标记，使得能在其它语句中引用它。语句标号是由1至5个数字组成。所表示的整数值无数值意义，但它必须大于0。语句标号可写在该语句初始行的1至5列的随便哪个位置上。在一个程序单位中，同一语句标号不能标记一个以上的语句。在区分语句标号时，左0不起作用。

3.5 **符号名字**。一个符号名字是由1至6个字母数字字符组成，但为首的必须是字母。参见10.1至10.1.10关于符号名字分类的讨论以及它们使用的若干限制。

3.6 **字符的排列**。在一个程序单位中给出各字符的排列。于是，各字符（它组成名字、行和语句）的任何有意义的集

列中为空白字符的行，而在7至72列中，字符E、N、D、按顺序出现，在E、N、D前，或在E、N、D间，或在E、N、D后可有空白字符出现。结束行指示加工程序：所编的程序单位（9.1.7）结束。每个程序单位必须用一结束行以示终止。

3.2.3 **初始行**。初始行既非注解行又非结束行，而是在第6列中出现数字0或空白字符的行。1至5列写语句标号，或全为空白字符。

3.2.4 **继续行**。继续行是在第6列中除数字0或空白字符外，可为任何字符，且在第1列中不为字符C的行。

继续行只可跟在某一初始行或另一继续行后。

3.3 **语句**。语句是由初始行其后顺序跟最多为5个继续行所组成的。语句写在各行的7至72列中。语句中字符的顺序是初始行的7至72列，当需要时，后跟第一继续行的7至72列，再跟下一继续行的7至72列等等。

3.4 **语句标号**。语句可任意给以标记，使得能在其它语句中引用它。语句标号是由1至4个数字组成。所表示的整数值无数值意义，但它必须大于0。语句标号可写在该语句初始行的1至5列的随便哪个位置上。在一个程序单位中，同一语句标号不能标记一个以上的语句。在区分语句标号时，左0不起作用。

3.5 **符号名字**。一个符号名字是由1至5个字母数字字符组成，但为首的必须是字母。参见10.1至10.1.10关于符号名字分类的讨论以及它们使用的若干限制。

3.6 **字符的排列**。在一个程序单位中给出各字符的排列。于是，各字符（它组成名字、行和语句）的任何有意义的集

成是作为一个完整的有序集出现。这种排列受3.2节（该节规定每行中各字符的排列）字符位置规则所制约，而且，为了加工起见，就以这种顺序列出各行。

4. 数据的各种类型

定义6种不同的数据类型。即整数型、实数型、双倍精确度型、复数型、逻辑型和Hollerith型。每型有不同的数学意义和相异的内部表示。于是，在解释和数据有关的运算时，数据类型就有意义了，函数的数据类型定义为：出现在表达式中的数的类型。

4.1 **数据类型的结合。**用来标识一个数或函数的名字附有数据类型的结合。表示一个常数的字符串既定义值又定义数据类型。

对每个程序单位来说，表示函数、变量、或数组的符号名字必须只有一种数据类型结合。一个特定的名字，一旦结合以某种具体数据类型后，那末，不管怎样使用它，总为该类型，即要求在它有定义的程序单位里，整个程序单位对它只定义一种数据类型结合。

对符号名字可用类型语句(7.2.1.6)中对整数型、实数型、双倍精确度型、复数型和逻辑型的说明建立数据类型。这种专门的说明就不管对整数型和实数型所用的隐含结合了（见5.3）。

没有一种手段可把符合名字和Hollerith数据型相结合。于是，和常数不一样，是假借其它类型的名字来标识这种类型的数据。

4.2 **数据类型的特性。**在下面几节中定义每种数据类型的数学性质和表示特点。对实数型、双倍精确度型和整数型的

成是作为一个完整的有序集出现。这种排列受3.2节（该节规定每行中各字符的排列）字符位置规则所制约，而且，为了加工起见，就以这种顺序列出各行。

4. 数据的各种类型

定义2种不同的数据类型。即整数型和实数型。每型有不同的数学意义和相异的内部表示。于是，在解释和数据有关的运算时，数据类型就有意义了。函数的数据类型定义为：出现在表达式中的数的类型。

4.1 **数据类型的结合。**用来标识一个数或函数的名字附有数据类型的结合。表示一个常数的字符串既定义值又定义数据类型。

对每个程序单位来说，表示函数、变量、或数组的符号名字必须只有一种数据类型结合。一个特定的名字，一旦结合以某种具体数据类型后，那末，不管怎样使用它，总为该类型，即要求在它有定义的程序单位里，整个程序单位对它只定义一种数据类型结合。

对符号名字用该名字的第一个字符建立数据类型（5.3）。

4.2 **数据类型的特性。**在下面几节中定义每种数据类型的数学性质和表示特点。对实数型和整数型的数来说，值0既

数来说，值0既不看作为正的又不看作为负的。

4.2.1 **整数型**。一个整数数据总是整值的确切表示。它可为正的、负的和零值。但它只能为整值。

4.2.2 **实数型**。一个实数数据是加工程序处理的该实数的近似值。它可为正的、负的和零值。

4.2.3 **双倍精确度型**。一个双倍精确度的数据是加工程序处理的一个实数的近似值。它可为正的、负的和零值。虽然近似度未定义，但必须比该实数型要高。

4.2.4 **复数型**。一个复数数据是加工程序处理的复数的近似值。该近似值表示为有序实数对的形式。数对的第一部分表示为实部，第二部分为虚部。因此，每一部分如对一个实数数据一样，有同样的近似度。

4.2.5 **逻辑型**。一个逻辑数据只可为真值或假值。

4.2.6 **Hollerith型**。Hollerith数据传递符号信息（和数值或逻辑值不一样）。符号信息可由加工程序中所能表示的任何符号组成。在Hollerith数据中，空白字符是一个有效的且有意义的字符。

5. 数据和过程的标识

用名字去引用或标识数据和过程。

对于数，采用术语**引用**意味着：在执行含有这种引用的语句时用的是该数的当前值。如果该数被标识，但不须要用，那末，称该数被**命名**。特别重要的一种情况（其中数被命名）是给数进行**赋值**，于是，定义了或重新定义了这个数。

对于过程，采用术语**引用**意味着：由该过程所规定的一些动作将被应用之。

不看作为正的又不看作为负的。

4.2.1 **整数型**。一个整数数据总是整值的确切表示。它可为正的、负的和零值。但它只能为整值。

4.2.2 **实数型**。一个实数数据是加工程序处理的该实数的近似值，它可为正的、负的和零值。

4.2.3

4.2.4

4.2.5

4.2.6

5. 数据和过程的标识

用名字去引用或标志数据和过程。

对于数，采用术语**引用**意味着：在执行含有这种引用的语句时用的是该数的当前值。如果该数被标识，但不须要用，那末，称该数被**命名**。特别重要的一种情况（其中数被命名）是给数进行赋值，于是，定义了或重新定义了这个数。

对于过程，采用术语**引用**意味着：由该过程所规定的一些动作将被应用之。

引用、定义（包括再定义）的完整又严格论述见第10节。

5.1 数据和过程的名字。数据名字标识一个常数、变量、数组或数组元素、或块（7.2.1.3）过程名字标识一个函数或子例程。

5.1.1 **常数**。常数是一个名字，它引用由该名字得来的数值或符号信息。常数不能再被定义。

整数、实数或双倍精确度常数前面缀有加号或减号者，称为带符号的常数。同样，对这些类型的数来说，任何带符号的常数或则是一个常数，或则是带符号的常数。

5.1.1.1 **整常数**。整常数，是由非空数字串形成的。按这种方法形成的数解释为：由该数字串所表示的数值。

5.1.1.2 **实常数**。一个基本实常数依序由整数部分、小数点和小数部分组成。整数部分和小数部分都是由数字串形成，两者之一可为空，但不得全为空。按这种方法形成的数被解释为代表一个值，这个值是由整数部分和小数部分所表示的数的近似值。

十进指数是由字母E后跟带任意符号的整常数形成的。这种指数被解释为其前面常数的乘数，即E后的数作为它的10的方幂。

一个实常数是一基本实常数、基本实常数后跟一个十进指数、或整常数后跟一个十进指数。

5.1.1.3 **双倍精确度常数**。除把字母E改为字母D外，形成和解释双倍精确度的指数与十进指数完全一样。

一个双倍精确度常数是基本实常数后跟一个双倍精确度指数，或整常数后跟一个双倍精确度指数。

引用、定义（包括再定义）的完整又严格论述见第10节。

5.1 数据和过程的名字。数据名字标志一个常数、变量、数组、或数组元素。过程名字标识一个函数或子例程。

5.1.1 **常数**。常数是引用一个值的名字。常数不能再被定义。

整数或实数常数前面缀有加号或减号者，称为带符号的常数。同样，对这些类型的数来说，任何带符号的常数或则是一个常数或则是带符号的常数。

5.1.1.1 **整常数**。整常数是由非空数字串形成的。按这种方法形成的数解释为：由该数字串所表示的数值。

5.1.1.2 **实常数**。一个基本实常数依序由整数部分、小数点和小数部分组成。整数部分和小数部分都是由数字串形成，两者之一可为空，但不得全为空。按这种方法形成的数被解释为代表一个值，这个值是由整数部分和小数部分所表示的数的近似值。

十进指数是由字母E后跟带任意符号的整常数形成的。这种指数被解释为其前面常数的乘数，即E后的数作为它的10的方幂。

一个实常数或为一基本实常数或为基本实常数后跟一个十进指数。

5.1.1.3