第一章　　引论

FORTRAN 是一种计算机程序设计语言，其功能是进行数字计算。 SVS FORTRAN-77 是由 ANSI 标准化协会提供的，迄今为止仍然是最新的语言版本。这一本FORTRAN参考手册所叙述的语言称为 FORTRAN-77，它是由美国硅谷软件公司（SVS）提供的实现语言，此后，名词 FORTRAN 就代表这一 FORTRAN-77 的实现语

1.1 FORTRAN 语言的概述

一个 FORTRAN 程序最终是由字符组成的。因为由字符组成行，行组成程序单元，程序单元再组成程序。

“行”可以是“注解行”，也可以是语句的“初始行”，或者语句的“继续行”。 “行”所含的字符可以从第 1 列到第 120 列。与较老的 FORTRAN 实现语言相比较，SVS FORTRAN—77 编译程序在用户选择 $COL72 编译指令时将略去各行的 72 列以后的字符（参阅第十三章——“FORTRAN 编译程序时间选择”）。

注解行是一些“空行”，这些空行是用字母 C(大写或小写字母)或者星号‘ * ’写在第 1 列上。注解行可以放置在程序中的任何地方，包括放在语句的初始行和继续行之间。

初始行必须从第 7 列开始，要保持第 6 列是空格或者是一个零字符。一个语句的继续行需要在第 6 列放置一个除了零字符以外的任何的其他字符，或在第一列放置一个“和号

字符”（&）。

一个语句最多可以有 19 个继续行。在语句的初始行的第 1 列到第 5 列之间可以放置一个“语句标号”，每一个语句标号由 1 到 5位数字组成，其中必须有一个数字是非零数字。语句标号的作用是作为某一语句的一个标志，以便其他语句引用。

语句大体上可以分成两类，即可执行语句和非执行语句。可执行语句在程序中所完成的作用是：给变量赋值、计算表达式、改变程序的执行流程和完成数据传送。非执行语句一般是说明程序内容的类型和属性。以下各段将要对语句进行更详细的讨论。

程序的量包括常量和变量。常量是一个数字串或者是一些其他符号，它的数值是不变的。变量占有存储单元，它的数值在程序执行过程中是可变的。变量和常量都可以被赋予一个“名字”和一个“数据类型”。名字的作用是为了标志程序中的一个目标。数据目标类型定义了一些其他内容，例如数据占有的存储量、数据的精度范围以及在某些情况下，这些数据所能完成的运算。在 FORTRAN 中，名字可以是“缺席数据型”的，这种缺席型的名字要按照一种命名规则去确定，或者用一种显式说明方式来确定，而不考虑缺席命名规则。

一个变量可以是单一的，也可以是一个“聚集”。聚集数据变量的形式有两种，即“数组变量”和“字符变量”。一个数组变量是一个数据集合体，它占有连续存储单元。数组的维数最多只能是 7。一个字符变量代表一个字符串数据，并且是一个字符序列，它可以被单独赋值，也可以在一个“子串”的形式下被集体地赋值。

一个完整的 FORTRAN 程序由一个“主程序”和任

何数目的“子程序”组成。子程序的类别有： SUBROUTINE 子程序，它可以通过调用（CALL）语句的作用去执行外部的语句群；FUNCTION （函数）子程序，它在表达式的上下文中可以计算和返回一个数值；以及 BLOCK DATA （块式数据）子程序，它的作用是使数据初始化，这在 COMMON blocks （公用块）中做了说明。主程序和子程序一起便是所谓的“程序单元”。在一般情况下，名词“子程序”和“程序单元”可以被交换使用。由用户定义的“子例行程序”和“函数”均可被称之为“过程”。

对于一个变量可以用“相联处理”的办法给予多个名字。相联数据可以有几种途径。用“公用语句”(COMMON ）可以共享分隔的程序单元之间的数据。用“等价语句”（EQUIVALENCE）可以在同一程序单元中相联变量。当引用子程序或者函数时，变量也可以通过变元扫视机构而被相联。

变量的名字有一个“作用域”，它取决于定义变量的方法。除了那些程序单元名字，公共区域名字，以及某些其他的名字之外，在一般情况下，大多数变量名字的作用域是局部的，它们仅局限于在所定义的程序单元之内。如果一个名字被定义与一个被调用的外部函数一样，则在缺省时，它就具有全局作用域。在公用区域内所定义的名字仅局限于在定义它们的程序单元之内。公用区域本身的名字是全局的。形式参数对于语句函数具有一个作用域，它只局限于这个语句函数的语句本身。

“说明语句”是语句的两类主要群之一。说明语句的作用是解释变量和符号常数。说明语句包括：“类型”语句，它的作用是定义一个变量的数据类型；“维数”语句(DIMENSION)，它的作用是定义数组变量的长度；“公

用”语句（COMMON）和“等价”语句（EQUIVALENCE）可提供变量的相联；“参数”语句（PARAMETER）是给予某一常数一个符号名字；“外部”语句（EXTERNAL）和“内部”语句（INTRINSIC）则具有定义其他程序单元的属性。

“数据”语句(DATA) 是提供数据初始化的一个机构。数据语句中可以有一个“隐循环”语句，它能以紧凑的形式促成数组变量的初始化。

“表达式”将数据和运算符组合在一起，从而产生新的数值。FORTRAN 表达式包括算术表达式、字符表达式、逻辑表达式和关系表达式，只要操作数和运算结果之间有明确定义，那么采用混合型表达式是允许的。

赋值语句（ASSIGN）是将一个表达式的数值赋给一个变量。赋值的形式有三种：即算术的、字符的、和逻辑的。赋值语句(ASSIGN 的作用是将一个语句标号的值，赋值给一个整数变量。

控制语句是在程序中控制执行流程的语句。 IF 语句的作用是为了控制程序的执行，为此，它根据所计算的是逻辑表达式的或是算术表达式的结果而选择各种不同类型的语句。在DO 语句中，将数值序列赋值给一个控制变量，就可使语句块重复执行。所提供的 CALL 语句以及 RETURN 语句是为了执行子例行程序和函数的。所提供的 GO TO 语句的各种变化形式可以在一个程序单元中执行控制的转移。

“语句-函数”语句的特征是在程序单元中定义一个用哑变元操作的单一语句“模板”。语句函数在程序单元中被引用时就好象一个具有实际变元的函数。变元要依据语句-函数的定义而组合以便产生一个可被应用于表达式的结果。

FORTRAN 提供了强有力的输入和输出功能。一个文

件可能是“外部的”（与外部设备相连结），也可能是“内部的”（对应于一个字符变量）；文件可以是“格式化的”，也可是“非格式化的”；文件可以被“顺序地”存取，也可被“随机地”存取。格式化文件可以是数据转换操作的主题，其转换是从内部存储表示到外部字符串表示，或者反之。

格式化转换是通过“读”(READ)，“写”(WRITE)，或者“打印”（PRINT）语句来完成的。“格式说明”有一组功能强的语句去控制转换数据的形式和格式。还有一种表式输入-输出功能，在那里缺席格式化规则可被用于转换进程。

“子例行程序”（SUBROUTINE）和“函数”（FUNCTION）程序单元可以具有变元，在处理进程中用调用程序传送它们。在说明子例行程序或者函数时，它的形式变元也同时被说明了；当子例行程序和函数被引用时，形式变元便被实际变元所代替。一个子例行程序或者函数可以有多重入口点(ENTRY)。除了第一句可执行语句之外，一个入口（ENTRY）语句，可以在一个语句的开始处执行一个子例行程序或函数。在遇到 END 语句，或者执行一个 RETURN语句时，控制可从一个子例行程序或者函数程序单元返回。对于子例行程序，FORTRAN 还提供了另外一种返回说明，它使子例行程序可以返回到与调用（CALL）语句所说明的不同的地方。

FORTRAN 有一组内容众多的内部函数，它们可以完成数据类型的转换，可以提供一组各种类型的算术函数和超越函数的集合体。

1.2 本书所用的记号和专用名词

本节将要叙述应用于 FORTRAN 语言结构的符号。

大写字母和特殊字母将在程序中示出。小写字母和文字所表示的将是在程序的上下文中所描述的实际语句中的内容。一个小写的目标一旦已被定义，它就能够在已被定义的语言结构的其余部分中保持它的意义。

大写和小写的应用实例

Iw 表示描述整数编辑的格式说明，在这里 w 是一个非零的，无正负号的整常数。在一个实际的“格式”（FORMAT）语句中，编辑说明必须写成 I5 或者 I21。对于实数的编辑说明是 Fw.d，此处 d 是一个无正负号的整常数。一个实际的 FORMAT 语句应该包含一个象 F8.4 或 F14.0这样的编辑说明。注意，根据以上定义，句号字符是一个特殊字符，它被看作是一个文字字符。

方括号“〔”和“〕”起闭合任选项的作用。例如，A〔w〕表示或者是 A 或者是 A12 这两种形式都是正确的（这里作为一个字符格式的说明方法）。

略字记号“…”表示写在略字记号前的任选项目可以出现一次或者重复多次。例如，计算的 GOTO 语句可以具有这样形式：

GOTO (s, 〔, s〕…) 〔, 〕i

此处表示语法项', s'可以重复出现任何次数。

空格（空白），在 FORTRAN 语句中通常是无效的（无意义的）。由于在所有的书籍中都提供了空格的解释，所以在本章的最后将介绍空格的一般规则。在本书中，“空格”和“空白”将被看作是同义词。一般情况下使用“空

格”一词。

1.3 FORTRAN 的基本单元

本节要叙述构成一个 FORTRAN 程序所需要的基本的文字单元和语法单元。

1.3.1 FORTRAN 字　　符组

FORTRAN 字符组包括大写字母、小写字母、数字以及特殊字符。

字符表：

字　符	字　符　名　字
	空白或空格
=	等号
+	加号
−	减号（负号）
*	星号
/	斜杠
\	反斜杠
(	左括号
)	右括号
,	逗号
.	小数点
$	货币符号
'	上撇符号（上撇）
:	冒号
&	和符号（和号）

一个字母是下列 52 个字符之一：

ABCDEFGHIJKLMNOPQRSTUVWXYZ

abcdefghijklmnopqrstuvwxyz

一个数字是下列 10 个字符之一：

0 1 2 3 4 5 6 7 8 9

一个字母数字字符是一个字母或是一个数字。

特殊字符见上列字符表。

1.3.2 整理顺序和图形

SVS FORTRAN 应用 ASCII 字符组。在 ASCII 中的整理顺序是：

- 空格（空白）整理为最低级，后随：
- 数字“ 0 ”到“ 9”，后随：
- 大写字母“ A ”到“ Z ”，后随：
- 小写字母“a”到“z”

特殊字符是出现在数字和大写字母之间的，同时也可出现在小写字母的前后。在附录 D 中有一张 ASCII 字符组的图表。

在每一个有序的字符组、数字、大写字母、小写字母之间，字符在这些序列中的排列是紧凑的，即在这些序列之中不存在“ 空缺 ”。

1.3.3 空格或者空白或者横表的应用

SVS FORTRAN 对于横表字符（tab）在所有情况下都解释为一个空白序列（它表示一个或者多个空白），应用它

们可以填满用 8 可除尽的各列的位置。在本书中所有引用的空白皆对应于实际空白或者是由横表字符所说明的空白，而且所有引用的列都是对应于在横表解释后的列，每一行和每一语句的字符总数的限制都是在考虑横表说明之后而定的。

在程序单元中，空格（也称为空白）字符是无意义的，但有例外情况。另一方面，空格可被自由地用来改进程序的格式和提高程序的可读性。在以下各情况中空格是有效的：

·在字符串常数中，在何勒内斯（Hollerich）字段中。

·在编译程序指令行上，将在“应用 FORTRAN 编译程序”一章中讨论。

·在第 6 列上，在该处用一个空格来区别这是一个初始行而不是一个继续行。

·在计算每一行和每一语句的字符数目时要将空格计算在内。

第二章行、语句以及控制流程

本章共包括三节。第一节叙述 FORTRAN 程序中的行的概念，第二节介绍 FORTRAN 语句的规则，最后一节阐明执行顺序，或称为控制流程的概念，即执行 FORTRAN 语句时所依照的次序。

2.1 行

在程序单元中的一个“行”是从第 1 列到第 120 列中的一个字符序列（如果编译任选被选定为 $COL72，则“行”是从第 1 列列第 72 列，这将在第 13 章中叙述），行中的所有字符都必须在第一章中所述的字符组中选定。在注解行（以下叙述）、字符常数以及何勒内斯字段中可以包含任何可印刷的 [redacted] 字符。

某一行上的字符所在的位置称之为列，它们被自左到右从 到 （或者从 到 72）计数。

FORTRAN 编译要略去出现在每一行的第 120 列右面的所有字符（如果编译任选被选定为 $COL72，则第 72 列右面的字符都被略去），这样用户可以将这些略去的列派其他的用途（例如用作为序列信息）。

2.1.1 注 解 行

注解行可以出现在一个程序单元的任何处，包括出现在一个程序单元的第一句语句之前或者在最后一句语句之后。

注解行也可以出现在一个初始行和他的继续行之间，也可以出现在两个继续行之间。

注解行的例子

```
C   This is a comnent line
*   This is also a comment line
C   The line following this one is all blank……

C   ……and is therefore considered to be a
   &comment
*
*   Comment lines are for documentary purpos-
   &es and
*   have no effect on compilation or execution.
```

2.1.2 初　始　行

一个初始行一般地是表示一个语句行的开始。一个初始行是一个任意的非注解行，它在第 6 列应是一个空格字符或者零（0）数字。一个语句的初始行中第 1 列到第 5 列可以有一个语句标号。

因为一个用横表字符开始的行可以具有 8 个初始空格，所以这样的行可以认为是一个初始行。如果一个横表字符后随一个语句标号，它占有了最初 5 列的一部分或者全部，这一个行将仍然被看作为它在第 6 列有一个空格。

初始行的例子

```
C   Here are initial lines without statement labels
C
     GO TO 999
    0GO TO 999
```

```
C
C   Here are initial lines with statement labels.
C
 379 GOTO 999
4850 GOTO 999
```

2.1.3 继　续　行

除了在某一行的第 6 列出现一个空格或一个零数字符之外，一个继续行是在第 6 列中有 FORTRAN 字符组中的一个任何字符，或者是在第 1 列有一个和字符 & 的一个非注解行。继续行必须没有语句标号。在一个语句中最多可以有 19 个继续行。

继续行的例子

```
C
C   These contrived statements each span two
     & lines.
C
      GO TO
     $999
*
843   GO TO
     +999
*
      GO TO
     &999
```

2.1.4 编译指令行

编译指令是 SVS FORTRAN-77 对于 FORTRAN-77 的一个扩充。它在编译功能上提供了一个附加的控制。编译指令行在第 1 列有一个货币符号“ $ ”。虽然某些编译指令必须出现在程序中某些限定的地方，但是一个编译指令行却如同一个注解行一样可以出现在程序中的任何地方。编译指令行中的空格是有效的，它们的作用是确定关键字和文件名的界限。编译指令将在“ FORTRAN 编译程序运行”一章中列出。

编译指令的例子

```
*
*    The following directive instructs the
&    compiler to
*    include the body of the 'rasp•text' into the
*    Program source code
*
$ INCLUDE rasp • text
```

2.2 语　　句

FORTRAN 语言的语句在第六章到第十二章中叙述。语句被用来形成程序单元。

语句可以写在从第 7 列到 72 列的初始行中，也可以写在多达 19 个的继续行中。

END 语句不遵循上述规则。一个 END 语句必须出现在其本身所在的初始行上。在一个程序单元中没有任何其他

的语句能够象 END 语句那样拥有初始行。

一般说来，一个语句必须在新的一行上开始；这就是说，一个语句不可以在其他语句的同一行上开始。逻辑 IF 语句不遵循这一规则。

在语句的前后或之间，所有的空格都是不起作用的，但在字符常数和何勒内斯常数之中的空格例外，在下例中，它们表示空白字符。

语句的例子

```
C     An assignment statement
C
      A=5.0
C
C     A subroutine call statement
C
      CALL COLECT (PAY, PHONE)
C
C     A logical IF statement
C
      IF (DAY. EQ. 'FRIDAY') RETURN
```

2.2.1 语句标号

“语句标号”如同给语句贴“标签”，这样其他的语句就可以引用它。

任何语句都可以有语句标号，但是只有与执行语句以及 FORMAT 语句（格式语句）有关的语句标号才能够被其他语句所引用。

一个语句标号是由 1 到 5 位数字组成，它们被放置在一

个语句的初始行的第 1 列到第 5 列的任何地方。在一个语句标号中至少有一位数字为非零数字。

在任何给定的程序单元中，语句标号必须是唯一的，重复的语句标号是一个错误。

语句标号的例子

```
  123 FORMAT ('The result is', I5)
C
C     An example of a DO block
C
      DO 110 ICON=1,100
      DESK (ICON)=0.0
  110 CONTINUE
```

2.2.2 语句和行的次序

在任何给定的程序单元中，语句的次序必须服从某些规则。这些规则详述如下。

“程序”语句（PROGRAM 语句）只能作为第一句语句出现在主程序中。一个子程序的第一句语句必须是 FUNCTION、SUBROUTINE、BLOCK DATA(块式数据）三个语句之一。

在一个程序单元中，语句必须服从下述次序规则：

1．FORMAT 语句（格式语句）可以在程序中的任何地方出现。

2．所有的说明语句必须是在所有的 DATA 语句，“语句一函数”语句，以及可执行语句之前。

3．所有的“语句-函数”语句必须在所有的可执行语句之前。

4．DATA 语句可以在说明语句之后的任何地方出现。

5．ENTRY 语句，不能出现在一个块式 IF 语句和它的相应的 END IF 语句之间，也不能出现在一个 DO 语句（循环语句）和它的执行循环（DO-LOOP）的终止语句之间，除了这两种情况之外，它可以出现在程序单元的任何地方。换句话说，一个子程序一定不要通过一个 ENTRY 语句进入到一个 IF 块或者 DO 块的语句之中。

在一个程序单元的说明语句中，IMPLICIT 语句（隐式语句）必须放在除 PARAMETER 语句（参数语句）之外的所有的说明语句之前。任何定义符号名字类型的说明语句必须放在一个 PARAMETER 语句之前，参数语句定义某一常数的符号名。

定义常数符号名的 PARAMETER 语句必须放在所有运用这些名字之前。

程序单元的最后一行（非注解行）必须是 END 语句（结束语句）。

以下用图形示出本段所述的语句和行的次序规则，其中语句和注解行可以分散放置。

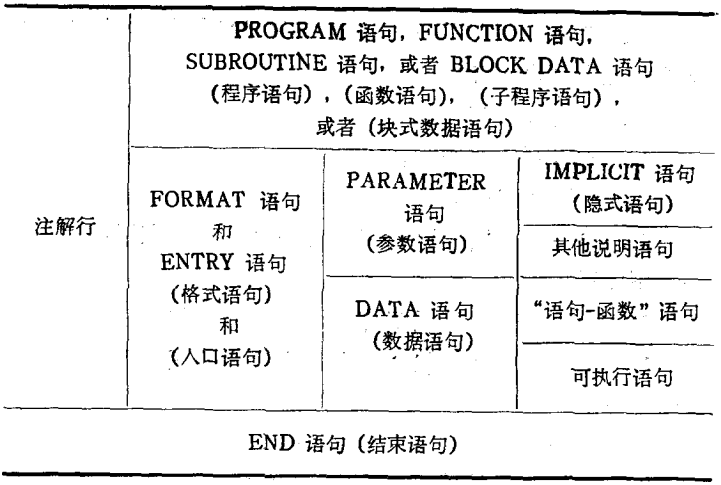

图 2—1 在一个 FORTRAN 程序中语句和行的次序

在以上图形中，用垂直线分隔开的语句群是可以混合的。例如，FORMAT 语句可以和“语句-函数”语句以及可执行语句相混合。

用水平线分隔开的语句群一定不能混合在一起。例如，“语句-函数”语句不能和可执行语句相混合。注意，END语句也是一个可执行语句，但它必须出现在程序单元的最后。

2.3 执行顺序和控制转移

正常执行顺序是指可执行语句在程序单元中所出现的次序。程序的执行是从主程序中第一句可执行语句开始。当引用一个外部过程时，开始执行的第一句语句是在子程序中跟随在 FUNCTION，SUBROUTINE，或者 ENTRY 语句之后的第一句可执行语句。

一个“控制转移”是指“正常执行顺序”已被改变。造成控制转移的语句为：