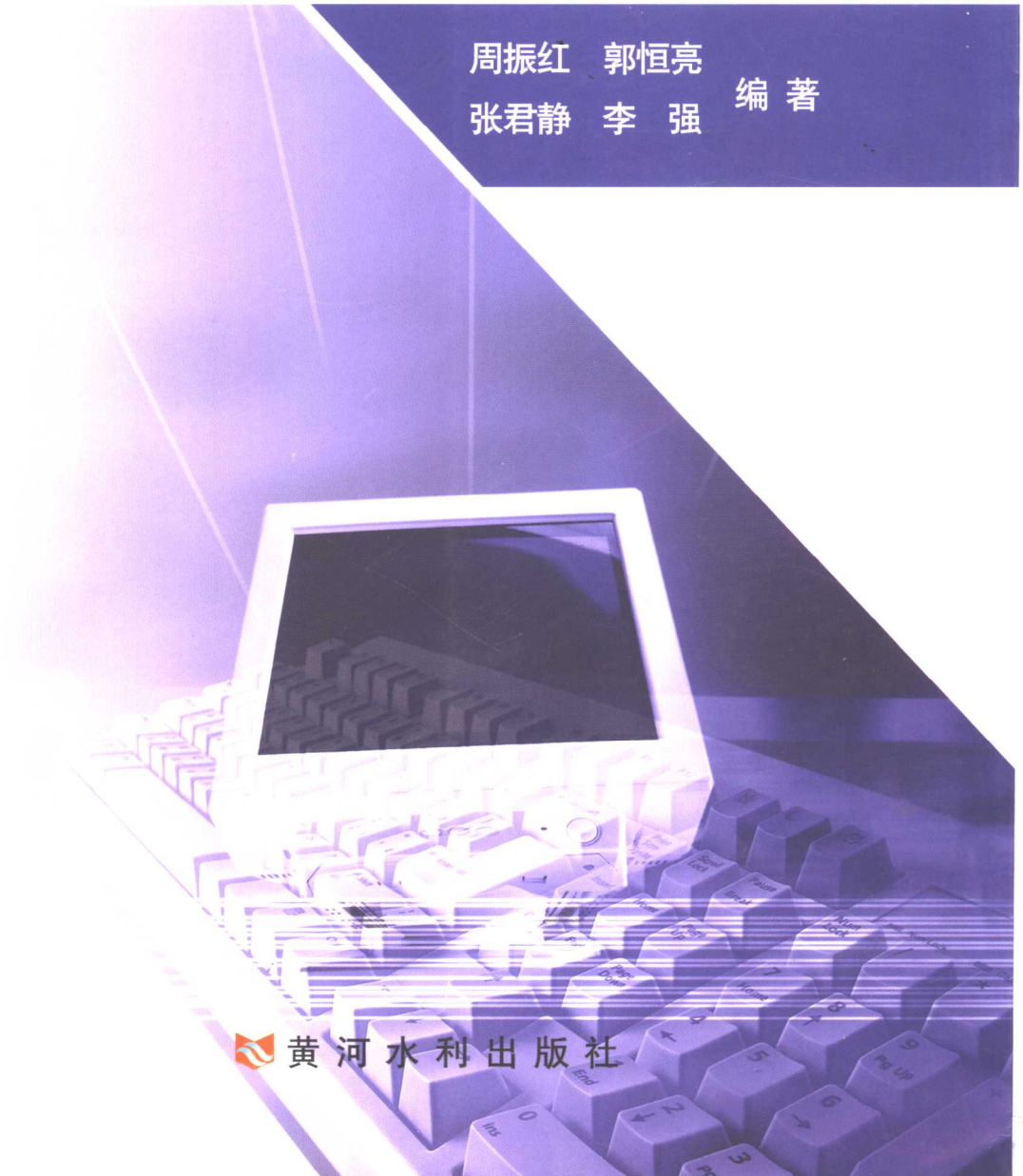
Fortran 90/95 高级程序设计

周振红　郭恒亮
张君静　李　强　编著



黄河水利出版社

Fortran 90/95
高级程序设计

周振红　郭恒亮
张君静　李　强　编著

黄河水利出版社

内容提要

本书系统介绍了当前在科学与工程计算领域广为使用的 Fortran 90/95 高级算法语言，重点讲述 Fortran 90 语法。全书由语言标准和语言扩展两部分组成。语言标准包括程序设计基础、例程和模块、控制结构、数组、派生类型、指针、格式化输入/输出及文件操作；语言扩展包括用模块、接口块、派生类型和指针来模拟 C++面向对象程序设计，Visual Studio 6.0 环境下的 Fortran 与 C/C++混合编译、混成 DLL 并集成到 Win32 应用程序，Fortran COM 组件的创建及其在客户程序中的调用。

本书重点突出，注重实用，文字通俗易懂，面向中、高级读者，适合作为理工科高年级本科生及研究生的教学参考书，也可作为科学研究、应用开发人员的参考书。

图书在版编目(CIP)数据

Fortran 90/95 高级程序设计 / 周振红等编著. —郑州：黄河水利出版社，2005.11
ISBN 7-80621-994-3

Ⅰ.F… Ⅱ.周… Ⅲ.FORTRAN 语言-程序设计-高等学校-教材 Ⅳ.TP312

中国版本图书馆 CIP 数据核字(2005)第 123199 号

出 版 社：黄河水利出版社
地址：河南省郑州市金水路 11 号　　邮政编码：450003
发行单位：黄河水利出版社
发行部电话：0371-66026940　　传真：0371-66022620
E-mail:yrcp@public.zz.ha.cn
承印单位：黄河水利委员会印刷厂
开本：787 mm × 1 092 mm　1 / 16
印张：12.75
字数：292 千字　　印数：1—2 000
版次：2005 年 11 月第 1 版　　印次：2005 年 11 月第 1 次印刷

书号：ISBN 7-80621-994-3 / TP · 21　　定价：26.00 元

前　言

在科学与工程计算领域，有相当多的科技人员对 Fortran 90/95 标准缺乏足够的了解，担心采用新标准会废弃已有的宝贵程序资源，所以仍在使用自己所熟悉的 Fortran 77 进行计算编程。事实上，Fortran 90/95 标准融入了许多现代语言特征，例如模块、接口块、指针、派生类型等，并进一步加强了数组功能，使得程序开发效率和可维护性都得到了极大的提高。

随着 Fortran 标准的不断更新，会逐步淘汰一些过时的语言特征。比如，Fortran 90 标准标明过时的一些语言特征，就从 Fortran 95 标准中去掉了。而且，语言实现机制也在发生变化：Fortran 90/95 标准已带有面向对象的特性，Fortran 2003 则直接支持面向对象的程序设计。所以，眼下从 Fortran 77 转到 Fortran 90/95 已势在必行。

目前，介绍 Fortran 90/95 的书籍还比较少，有的主要讲解的还是 Fortran 77 语法，有的侧重编译器、集成开发环境的介绍，对 Fortran 90/95 新语言特征讲解得不够系统、深入。

针对上述情况，作者在数年《Fortran 90/95 程序设计》教学、科研实践的基础上，对 Fortran 90/95 进行了系统的整理、编撰，并从当前软件开发的实际情况出发，增加了独具特色的语言扩展：模拟 C++面向对象程序设计，Visual Studio 6.0 环境下 Fortran 与 C/C++的混合编译，动态链接库 DLL 和基于组件对象模型 COM 的组件开发，及其在 Visual C++ 6.0、Visual Basic 6.0 和 Delphi 7.0 中的集成。

下面是作者的联系方式，随时欢迎您的指教。通信地址：郑州大学环境与水利学院，450002。电子信箱：zzh6374@163.com

作者

2005 年 6 月

导 读

第一章 Fortran 背景知识。介绍 Fortran 语言发展简史，Fortran 90/95 新的语言特征，以及 Visual Fortran 编译器的演变。

第二章 Fortran 程序设计基础。介绍程序书写、字符集和标识符、数据类型、声明的有关事项、算术表达式及表控输入/输出语句。

第三章 模块化程序设计——例程和模块。从模块化程序设计出发，重点讲述例程和模块，其中例程包括内部例程、外部例程、例程重载、递归例程、例程参数和接口块，另外也介绍了主程序单元的构造。

第四章 结构化程序设计——控制结构。从结构化程序设计出发，重点讲述选择结构(包括 CASE 结构)和循环结构。

第五章 数组。重点讲述数组加强功能，包括数组整体及数组段操作，WHERE 和 FORALL 构造，动态数组、不同类型数组参数及数组型函数。

第六章 派生类型。讲述派生类型的构造、操作符重载及数据库管理应用。

第七章 指针。介绍指针的基本概念，讲述指针数组、指针型函数和指针参数的使用及单链表应用。

第八章 模拟 C++面向对象程序设计。综合运用模块、接口块、派生类型和指针，模拟 C++主要的面向对象特性：封装、继承和运行时多态。

第九章 格式化输入/输出及文件操作。介绍格式编辑符，讲述格式化输入/输出语句的使用，以及内部文件和外部文件的操作。

第十章 Fortran 与 C/C++的混合编译。系统讲述 Fortran 与 C/C++之间调用约定的协调、程序的混合编译，以及 Fortran 模块数据和例程的传递。

第十一章 Fortran 与 C/C++混成 DLL 并集成到 Win32 应用程序。在混合编译的基础上，将 Fortran 与 C/C++建造成动态链接库 DLL，并集成到 Visual C++ 6.0、Visual Basic 6.0 及 Delphi 7.0 中。

第十二章 Fortran COM 组件的创建及其在客户程序中的调用。介绍 COM 对象及组件的基本概念，讲述在 Fortran COM 服务器向导支持下 COM 组件的创建，以及 Fortran COM 组件在 Visual C++ 6.0、Visual Basic 6.0 及 Delphi 7.0 客户程序中的调用。

目 录

第一章 Fortran 背景知识

第一节 Fortran 语言简史

要了解 Fortran 90/95，我们有必要先简单回顾一下 Fortran 语言的发展历史。

1954～1957 年，世界上第一种高级程序设计语言——Fortran 诞生于 IBM 公司。Fortran 是 IBM Mathematical Formula Translation 的缩写，其设计目的在于为科研人员提供一种符合数学思维习惯的高级语言，以满足科学计算的需要。20 世纪 60 年代初，在国防、教育和科技领域对高性能计算工具的迫切需求下，Fortran 语言蓬勃发展，成为当时统治计算机世界的高级语言之王。

1962 年，为了统一不同公司、不同硬件平台上的 Fortran 语言，人们开始了 Fortran 语言标准化的尝试，这也是程序设计语言发展史上的第一次标准化历程。1972 年，Fortran 66 标准(标准编号来自标准草案的制定时间)正式发布。但因为标准文档过于简单，约束力不强，Fortran 66 标准发布后，Fortran 语言的统一问题并没有得到彻底解决。

1978 年，Fortran 语言标准的第一个修订版本正式发布，这就是所谓的 Fortran 77。Fortran 77 细致地描述了 Fortran 语言的各种特征，让 Fortran 成了一种真正规范、高效和强大的结构化程序设计语言。此后，无数性能优异的 Fortran 77 编译器和开发工具的问世更是让 Fortran 77 成为了几乎所有理工科学生的必修课。

尽管 Fortran 77 的影响力一直延续到了今天，但 Fortran 语言不断变革的历程却从未停止过。为了改变 Fortran 77 那种老式的、从穿孔卡片遗留下来的语言风格，并给 Fortran 注入更多的现代语言特征，人们于 1991 年发布了崭新的 Fortran 90 标准。除了自由的代码风格外，Fortran 90 还为 Fortran 语言引入了模块、接口块、自定义(派生)数据类型和运算符、可动态分配和参与复杂运算的数组、例程重载、指针、递归等重要的语法特征。这不但使结构化的 Fortran 语言更趋完善，也使其具备了少量的面向对象语言特征。

1997 年发布的 Fortran 95 标准在 Fortran 90 的基础上，吸收了 HPF(High Performance Fortran，Fortran 语言在并行环境下的一个变种)的优点，提高了 Fortran 语言在并行任务中的表达和计算能力，并进一步完善了派生类型、指针、数组等要素的相关语法。

2004 年 5 月，在 ISO、IEC 的联合工作组 JTC1/SC22/WG5 以及美国 Fortran 委员会 NCITS/J3 的共同努力下，终于推出了 Fortran 2003 标准。Fortran 2003 近乎彻底地解决了 Fortran 语言现代化的问题：完整的面向对象机制、灵活的语法特征、统一的接口标准等。不过，支持 Fortran 2003 标准的编译器要在一两年后才能开发出来。

以 Fortran 66 为基准，我们可以把后续的 Fortran 77/90/95 以及 Fortran 2003 均视为对 Fortran 语言标准的修订。在历次修订中，Fortran 77 和 Fortran 95 是修订幅度相对较小的版本，而 Fortran 90 和 Fortran 2003 则是锐意变革的“大修”版本。

由于支持 Fortran 2003 标准的编译器还不可利用，本书从实际情况出发，重点讲解 Fortran 90 的语法及应用，必要时也介绍 Fortran 95 语法。

第二节 Fortran 90/95 新的语言特征

下面分别介绍 Fortran 90/95 新的主要语言特征。

一、Fortran 90

(一)自由书写格式

Fortran 90 提供了一种新的自由书写格式，行中的位置没有特殊意义，没有保留列，尾部可以出现注释，空格在某些情况下是有意义的。例如：P R O G R A M 并不等于 PROGRAM。

(二)模块

Fortran 90 提供了一种新的程序单元——模块，其功能远比 Fortran 77 数据块程序单元来得强大。模块包含了数据、例程、例程接口等要素的声明，别的程序单元引用后，就可访问其中的数据和例程了。模块要素的可见性可以限制在模块内，以提供数据抽象，编写安全、可移植的程序代码。

(三)自定义(派生)数据类型和操作符

Fortran 90 允许从固有数据类型和派生类型中定义新的数据类型，派生类型可以整体访问，也可直接访问当中的元素；可以重载固有操作符(如+、*)，也可定义新的操作符，以适应派生类型数据操作要求。

(四)数组功能加强

在 Fortran 90 中，固有操作符和相关的固有函数可以直接操作整个数组或数组段。可以对数组整体、部分及隐式赋值(包括可选择性赋值的 WHERE 语句)，声明数组常量，定义派生类型数组值函数，用数组构造子规定一维数组的值，利用 ALLOCATABLE 或 POINTER 属性为数组动态分配内存。新的固有例程能够创建和使用多维数组，支持数组计算(如 SUM 函数累加数组元素的和)。

(五)例程重载

同 C++相似，Fortran 90 也支持例程重载。它通过接口块规定统一的名子,将不同参数表的例程置于接口块内，引用时按参数匹配的原则调用具体的例程。如同固有函数 ABS，其参数既可以是整数，也可以是实数或复数。

(六)指针

Fortran 90 指针允许动态访问和处理数据,可以用来创建动态数组和派生类型的动态数据结构(如链表)。指针可以指向固有数据类型或派生类型，一旦和同类型的目标变量相关联，指针可以代替目标出现在表达式和赋值语句中。

(七)递归

假如 RECURSIVE 关键字添加在例程(FUNCTION 或 SUBROUTINE)原型中，Fortran 90 例程也可递归实现。

(八)接口块

Fortran 90 程序单元(主程序、外部例程和模块)可以包含接口块，接口块用来：①描述外部例程或虚参例程的接口；②为重载的例程规定统一的名称；③定义或扩展操作符等。

(九)封装机制

类似于 C++中的类，Fortran 90 可以将派生类型数据连同其操作例程封装在模块内，通过其公有接口，供别的程序单元使用，以扩展 Fortran 功能，使之适合特殊的应用需求，例如模拟面向对象的程序设计、数据库管理、建立动态数据结构等。

二、Fortran 95

(一)FORALL 语句和构造

在 Fortran 90 中,可以通过数组构造子、RESHAPE 和 SPREAD 固有例程逐元素地构造数组，Fortran 95 的 FORALL 语句和构造则提供了另一种数组操作方式。FORALL 允许通过元素下标对数组元素、数组段、字符子串或指针目标进行操作；类似于隐式 DO 循环，FORALL 构造可使几个数组赋值语句共享相同的下标循环控制表达式。

FORALL 是 WHERE 的一般形式，两者都通过隐式循环对数组进行操作，只不过 FORALL 是使用元素下标，WHERE 则针对整个数组。

(二)PURE 用户定义例程

在用户定义的例程(子程序或函数)原型前添加 PURE 关键字，向系统表明该用户定义例程没有负作用。例如：在例程内改变值传递参数的值。

(三)ELEMENTAL 用户定义例程

在用户定义例程原型前添加 ELEMENTAL 关键字,它是 PURE 例程的特殊形式。

当传递数组参数时，每次对一个数组元素进行操作。要使用 ELEMENTAL 例程，须在调用程序中建立其接口块。

(四)CPU_TIME 子程序

该子程序通过参数返回特定 CPU 处理器的时间，单位为秒，参数须是单一的实数。

(五)NULL 函数

在 Fortran 90 中不能直接初始化指针为空指针。Fortran 90 指针必须先与目标变量相关联，或动态分配内存，然后才能使用 NULLIFY 函数置空指针。Fortran 95 则可利用 NULL 函数直接初始化指针为空指针。

值得注意的是：Fortran 95 已删除了 Fortran 90 标明为过时的某些语言特征，并新标出了一些过时的语言特征。

第三节 Visual Fortran 编译器的演变

当前，国内使用较多的 Fortran 编译器或可视化集成开发环境为 Visual Fortran，它起源于 Microsoft 的 Fortran PowerStation 4.0。这套工具后来卖给了 Digital 公司继续开发，第二个版本称为 Digital Visual Fortran 5.0。Digital 被 Compaq 并购后，接下来的 6.0、6.1、6.5 和 6.6 版本称为 Compaq Visual Fortran。本书使用目前最新的 (Compaq)Visual Fortran 6.6 专业版开发实例。

Visual Fortran 6.6 被组合在 Microsoft Visual Studio 6.0 集成开发环境中。Visual Studio 提供了统一的操作界面，这个界面包括文字编辑器、Project 的管理、调试工具等。而编译器则是使用类似 Plug In 的方法组合到 Visual Studio 中，用户在使用 Visual Fortran 6.6 和 Visual C++ 6.0 时，看到的都是相同的操作界面。

Visual Fortran 6.6 除了完全支持 Fortran 90/95 语法外，扩展部分还提供有完整的 Windows 应用程序开发工具，可以直接建造 Win32 DLL(动态链接库)、基于组件对象模型 COM 的组件等，专业版还内含了 IMSL 数值链接库。另外，它还可以和 Visual C++ 6.0 互相链接，将 Fortran 和 C/C++语言的程序代码混合编译成同一个执行文件(EXE 或 DLL)。

第二章　Fortran 程序设计基础

在学习 Fortran 90/95 高级的语言功能之前，先来了解进行数值计算编程所必备的基础知识：程序书写、字符集及标识符、数据类型、声明的有关事项、算术表达式以及表控输入/输出语句。

第一节　程序书写

一、程序构造形式

我们先来看一个银行存款程序实例。

[例 2-1]　简单 Fortran 程序的构造形式。

```
PROGRAM  MONEY
! Calculates  balance  after  interest  compounded
    REAL   BALANCE, INTEREST, RATE

    BALANCE = 1000
    RATE = 0.09
    INTEREST = RATE * BALANCE
    BALANCE = BALANCE + INTEREST
    PRINT*, 'New   balance:', BALANCE
END   PROGRAM   MONEY
```

第一行的 PROGRAM 关键字标识 Fortran 主程序，后接程序名，这一行是可选的；以感叹号开始的第二行是注释，不参加编译；第三行中的 REAL 将其后面的变量声明为实型数。这三行为非执行部分，之后的部分(END 语句之前)为执行部分。

由此给出简单 Fortran 90 程序的构造形式：

[PROGRAM 程序名]

　　[声明语句]

　　[执行语句]

END [PROGRAM [程序名]]

方括号内的部分是可选的，END 语句是惟一必须的，它通知编译器：程序编译到此结束。END 语句中的程序名可以省略，但若出现程序名，必须同时出现 PROGRAM 关键字。

二、语句

语句是 Fortran 程序的基本单位，一条语句可包含 0～132 个字符。Fortran 77 规定，一条语句的不同部分应从特定的列开始，这样的书写格式称为固定格式，相应的程序文件扩展名为.f 或.for；自由格式的 Fortran 90(文件扩展名为.f90)则无此限制。

除赋值语句外，所有的语句都从一个关键字开始。比如，例 2-1 中出现的关键字：PROGRAM、REAL、PRINT 和 END。

一般情况下，每行一条语句。如一行有多条语句，它们之间以分号间隔。为清楚起见，通常将几条简单的赋值语句写在一行上。例如：

```
A = 1; B = 1; C = 1
```

假如一条语句一行写不完，允许出现续行，但要求被续行最后的非空白字符为&。例如：

```
A = 174.6 * &
    (T-1981.2)**3
```

续行从下一行(非注释行)的第一个非空白字符开始；若下一行的非空白字符为&，则续行从该字符后的第一个字符开始。 Fortran 90 允许出现多达 39 个续行。

三、空白的作用

通常，空白没有意义，它不参加编译。适当地运用空白空间，可以增加程序的可读性，如程序块中的代码缩进。但在代表有意义字符序列的记号(token)内，比如：标号、关键字、变量名、操作符等，不允许出现空白。例如，INTE GER、BAL ANCE 和< = 是非法的(<=为操作符)。

一般情况下，记号之间需留有空白，例如：30CONTINUE 是非法的，因为标号 30 和关键字 CONTINUE 是两个独立的记号。而有的记号间的空白是可选的，例如：END PROGRAM 和 ENDPROGRAM，A*B 和 A*B 均是合法的。

四、注释

Fortran 90 只提供了一种注释方式：以感叹号开始的语句作为注释，字符串内的感叹号除外。注释可以是一整行，也可以是空白行。注释在编译时被忽略。

五、固定格式

早期的计算机，还没有使用键盘/显示器作为输入/输出设备，那时的程序是利用穿孔卡片一张张地记录下来，再让计算机来执行。固定格式正是为了配合早期使用穿孔卡片输入程序所发明的格式。见例 2-2 所示。

[例 2-2]　固定格式编写的程序。

```
C       FIXED FORMAT DEMO
        PROGRAM Fixed
        PRINT*, 'Hello
       $World!'
        PRINT 10
10      FORMAT(1x,'This program is written in fixed format.')
        END
```

在固定格式中，每行有 80 列，这 80 列被分为 4 个区，分别书写不同的内容：

(1)第 1 ~ 5 列为标号区。可以写 1 ~ 5 位整数作为语句标号，也可以没有标号，标号区中的空格不起作用。标号区中某一行的第 1 列若出现“C”或“*”字符，则该行被认为是注释行。该程序中的第一行即为注释行，10 为格式标号。

(2)第 6 列为续行标志区。如果在一行的第 6 列上写上一个非空格或非 0 的字符，则该行作为上一行的续行。程序中的“$”即为续行标志。

(3)第 7 ~ 72 列为语句区。语句可以从第 7 列以后任何位置开始书写，但一行只能写一个语句。语句区内的空格(不包括引号内字符串中的空格)在编译时被忽略。

(4)第 73 ~ 80 列为注释区。注释区用于程序员书写提示信息，在编译时不予处理。

这里介绍固定格式，只是让大家对 Fortran 77 程序有所了解，建议大家在编写新的程序时一律采取前述的自由格式。

第二节　字符集和标识符

一、字符集

Fortran 90 字符集由下列字符组成：

- 26 个英文字母(A ~ Z 和 a ~ z)；

- 数字 0 ~ 9；
- 下划线(_)；
- 特殊字符，如表 2-1 所示。

表 2-1　　字符集中的特殊字符

字符	名称	字符	名称
	空格	:	冒号
=	等号	!	感叹号
+	加号	"	引号
-	减号	%	百分号
*	星号	&	和号
/	撇号	;	分号
(	左括号	<	小于号
)	右括号	>	大于号
,	逗号	?	问号
.	小数点	$	美元符号
'	省略号		

注：(1)表中的标点符号占一个字节，即为半角；
(2)字符串中的字符可以是 Fortran 90 字符集之外的字符。

二、标识符

在给变量、常量、例程等标识符命名时，须以字母(A ~ Z, a ~ z)开头，后可接多达 30 个字母(A ~ Z 或 a ~ z)、数字(0 ~ 9)或下划线(_)。例如：MASS, rate, Npts, I9J7, Time_Rate, Speed_of_Light。针对标识符命名，有下列几点值得注意：

(1)只能以字母开头(3M, _Right 为无效标识符)；

(2)不能含有空格字符(Time Rate 为无效标识符)；

(3)不区分字母大、小写(Vel, VEL, vel 为同一标识符)；

(4)长度限定为 31 个字符(Fortran 77 为 6 个字符)；

(5)避免与关键字、标准例程重名。

第三节　数据类型

程序要和数据打交道，数据都有一个特定的类型。数据类型含有两层意思：一是数据可以取哪些值；二是对数据可以进行哪些运算。例如：整数取 0、±1、±2、±3 等，可以对它们进行算术运算。

Fortran 90 提供了 5 个固有(内建)数据类型，这些数据类型被分成两大类：一类是数值型，包括整型、实型和复数型；另一类是非数值型，包括字符型和逻辑型(或

布尔型)。

在固有数据类型之外，还允许用户定义自己的数据类型——自定义数据类型或派生类型。

本节主要讲解 5 个固有数据类型的取值范围(和精度)，派生数据类型将在第六章中讲解。

一、整数类型

(一)整型变量

声明整型变量的一般形式为：

```
INTEGER I
INTEGER([KIND=]n) I
```

n 是种类参数，取值为 1、2、4、8。

种类参数(KIND)是 Fortran 90 新添加的特性，它通过规定存储数据所用的内存字节数来控制数据的取值范围，1、2、4、8 为整数在内存中的存储字节数。如果种类参数没有特别规定，则取缺省值；而缺省值受编译器选项影响，若没有编译器选项规定，32 位系统下缺省值为 4。不同种类参数的整数取值范围见表 2-2。

表 2-2　　不同种类参数的整数取值范围

参数	整数取值范围
INTEGER(1)	-128 ~ 127
INTEGER(2)	-32768 ~ 32767
INTEGER(4)	-2147483648 ~ 2147483647
INTEGER(8)	-9223372036854775808 ~ 9223372036854775807

Fortran 90 提供的 KIND 函数，用来获取缺省种类参数的值；HUGE 函数则用来获得取值范围的上限；上限加 1 即为取值范围的下限。如下列代码段所示：

```
INTEGER(8) I,Big,Small
Big = HUGE(I)
Small = Big + 1
PRINT*, 'Largest:      ',Big
PRINT*, 'Smallest:     ', Small
```

值得注意的是：在不同的平台(处理器和编译器)下，相同的种类参数可能有不同的取值范围，这极大地影响了程序代码的可移植性。对此，Fortran 90 提供了