

潘在元 张素素 编著

FORTRAN 90 教程

浙江大学出版社

FORTRAN 90 教程

潘在元 张素素 编著

浙江大學出版社

(浙)新登字 10 号

内容提要

FORTRAN 90 是国际标准化组织(ISO)颁布的最新版本,比之 FORTRAN 77,在功能与先进性方面都有突破性发展,是最富现代化特性的语言之一。

本书章节命名仍按 FORTRAN 77 习惯,为了能全面、系统地介绍 FORTRAN 90 现代特性的语句与项目,在同名章节下介绍的则是 FORTRAN 90 的新内容。

在本书中介绍的 FORTRAN 90 新概念、新项目,有数据类别、数据属性、派生类型、指针、动态数据、递归过程、内部过程、过程的现代调用方式、超载、接口块、模块、高级传输与低级传输等。

本书以编程的可读性、可维护性为编写指导思想,具体书写时仍循序渐进,深入浅出,引进新概念时阐述特别详尽,适宜大专院校作教科书,也适宜工程人员、科研人员自学进修。对于学过 FORTRAN 的人,更能收到事半功倍之效。

FORTRAN 90 教程

潘在元 张素素 编著

责任编辑 俞妙送 洪保平

浙江大学出版社出版

浙江省煤田地质局制图印刷厂印刷

浙江省新华书店发行

787×1092 16 开 16 印张 字数:487 千字

1993 年 9 月第 1 版

1993 年 9 月第 1 次印刷

印数:0001—5000

ISBN 7-308-01163-1/TP·081

定价:9.80 元

前 言

人们期待已久的 FORTRAN 8X 终于以 FORTRAN 90 的形式问世,它的美国国家标准文件号是 ANSI X3.198-199X., 国际标准文件号是 ISO/IEC 1539:1991, 通俗的称呼是 FORTRAN 90。

我们知道,世界上程序设计的语言种类数不胜数,随生随灭,甚至一度呼声很高的语言几年后也无人问津,而 FORTRAN 却几十年来长盛不衰,原因何在?

FORTRAN 的强大生命力在于能紧跟时代的发展,它不断更新 FORTRAN 的标准文本,为用户提供更方便、更强大、更高效的程序设计语言。例如在 1966 年后推出 FORTRAN IV, 1977 年后推出 FORTRAN 77。每次新的国际文本推出, FORTRAN 的功能都有一次突破性发展,在国外、国内发挥了很大作用。

FORTRAN 90 是国际许多专家自 80 年代起就开始讨论酝酿的文本,他们组织起来,根据当前计算机功能的发展、算法功能的发展、计算机科学理论的发展多次写出讨论稿,发交各国讨论,根据各国提出的意见,再修改草稿,直到 1990 年才正式定稿。其先进性不言而喻。

FORTRAN 90 的先进性首先表现在它的时代特性,它根据计算机软件科学理论,新增了许多有现代性的语句与项目,保证了程序结构化,具有可读性与可维护性。

FORTRAN 90 的先进性还表现了空前强大的算法功能、算法手段。它大大发展了 FORTRAN 原有的数值计算优势,作出了令人惊喜的进步。以数组为例,引进了数组直接运算、数组取函数、数组直接赋值、数组与标量的运算等概念,使数组的运算从繁琐的对数组元素的运算中解脱出来,大大简化了程序的编写,也为以后平行算法准备了条件。

在调用过程时,设计了许多先进的调用手段,如关键字变元调用、可选择变元调用、类属过程调用,大大加强了过程的功能。此外还增加了操作符超载、赋值号超载功能,大大扩展了 FORTRAN 的操作功能。

FORTRAN 90 还允许过程有递归功能。

信息管理功能本是 FORTRAN 语言的弱项,在 FORTRAN 90 里则也变成强项。它引进字符种别概念,使 FORTRAN 能给各国文字(包括汉字)、各种学术符号(如化学符号、数学符号)赋以不同的种别参数,从而可以对例如汉字进行操作,另外,还按数据结构需要,设置派生类型、指针以及模块、模块过程等,很方便地操作各种数据结构。

FORTRAN 90 的另一特色是语法结构严谨规范。从高层语法到底层语法,都有统一的结构,格式一致,形成了可用 BNF 范式推导出的严密体系,把 FORTRAN 77 中许多零乱的说明语句归纳成只有两句说明语句(内部类型说明语句和派生类型说明语句),只是使用不同的属性关键字。这样使 FORTRAN 90 易学易记,编写时也不易出错。只要了解新引进的概念,学习 FORTRAN 90 远比学习 FORTRAN 77 容易。

为了使过去编写的 FORTRAN 77 程序仍能在新编译系统下运行。FORTRAN 90 文本中仍保留了 FORTRAN 77 语句与项目,但实际上 FORTRAN 77 中不少语句与项目已过时(如 GOTO 语句及语句标号),那些应予淘汰的功能完全可由 FORTRAN 90 中现代化语句与项目来替代,其内容在第十章中陈述。

本教材目的是全面系统地阐述 FORTRAN 90 中现代内容,它们无一遗漏地写在本书前九章中。希望读者只用本书前九章的内容编写 FORTRAN 程序,并用它来修改 FORTRAN 77 中的过时语句与项目。

希望读者注意贯串在书中的编程要有可读性、可维护性的指导思想,用于编程实践,不要满足于学习 FORTRAN 90 语句的具体用法。

潘在元

1993 年 7 月 7 日

目 录

第一章 编程基础知识

1.1 FORTRAN 90 的现代特性	(1)
1.2 FORTRAN 90 的程序形式	(1)
1.2.1 程序总体构造形式	(1)
1.2.2 程序单元的编写	(2)
1.2.3 编程实例	(4)
1.3 结构图与流程图	(5)
1.4 程序的运行	(5)
1.5 程序的可读性、可维护性与可使用性	(7)
习题一	(7)

第二章 FORTRAN 的基本语句

2.1 类型说明语句	(9)
2.1.1 变量命令	(9)
2.1.2 类型说明	(9)
2.1.3 种别说明	(11)
2.1.4 属性说明	(12)
2.1.5 使用举例	(13)
2.2 赋值语句	(15)
2.2.1 赋值语句的一般形式	(15)
2.2.2 赋值语句的意义	(15)
2.2.3 赋值语句的类型、种别	(15)
2.2.4 表达式的组成	(16)
2.2.5 编程举例	(21)
2.3 读语句	(22)
2.3.1 读写的必要条件	(22)
2.3.2 缺省部件号的读语句	(23)
2.3.3 表控格式	(23)
2.3.4 带控制信息表的读语句	(24)
2.4 写语句	(25)
2.5 格式编写	(27)
2.5.1 表控格式	(27)
2.5.2 自定格式	(28)
习题二	(31)

第三章 选择结构与程序结构化

3.1 程序结构化	(31)
3.1.1 结构化分析	(34)
3.1.2 结构化设计	(35)
3.2 IF 语句与关系表达式	(36)
3.2.1 IF 语句的一般形式	(36)
3.2.2 关系表达式	(37)
3.3 IF 结构	(38)
3.3.1 IF 结构的一般形式	(38)
3.3.2 IF 结构的控制机制	(39)
3.3.3 IF 结构的取名	(41)
3.3.4 IF 结构块的缺省与嵌套	(41)
3.4 CASE 结构	(44)
3.4.1 CASE 结构的一般形式	(44)
3.4.2 CASE 结构的控制机制	(45)
3.4.3 CASE 结构的取名	(45)
3.4.4 非数值型的情况表达式	(46)
习题三	(47)

第四章 重复结构

4.1 不带循环变量的 DO 结构	(48)
4.1.1 不带循环控制变量 DO 结构的一般形式	(48)
4.1.2 不带循环控制变量的 DO 结构控制机制	(49)
4.1.3 EXIT 语句与 CYCLE 语句	(49)
4.1.4 编程及方法	(51)
4.2 带循环控制变量的 DO 结构	(53)
4.2.1 带循环控制变量 DO 结构的一般形式	(54)
4.2.2 带循环控制变量 DO 结构的控制机制	(54)
4.2.3 使用注意事项	(54)
4.2.4 带循环控制变量 DO 结构的嵌套	(55)
4.2.5 编程及方法	(56)
4.3 隐 DO 循环	(58)
习题四	(59)

第五章 非数值型数据、数组

5.1 逻辑型数据	(61)
5.1.1 逻辑型变量的说明	(61)
5.1.2 逻辑表达式的组成	(62)
5.1.3 逻辑赋值	(65)
5.1.4 逻辑型数据的输入输出	(65)

5.1.5 编程及方法	(66)
5.2 字符型数据	(67)
5.2.1 字符型常数与字符型变量说明	(67)
5.2.2 字符子串	(69)
5.2.3 字符表达式与字符赋值语句	(69)
5.2.4 字符型的输入输出	(70)
5.2.5 字符比较	(72)
5.2.6 编程及方法	(74)
5.2.7 字符函数	(75)
5.3 数组	(78)
5.3.1 有关数组的名词与定义	(78)
5.3.2 数组的存贮与输入输出	(87)
5.3.3 编程与方法	(89)
5.3.4 数组运算与赋值	(93)
5.3.5 WHERE 语句和 WHERE 结构	(96)
5.3.6 数组形式	(97)
5.3.7 ALLOCATE(分配)语句与 DEALLOCATE(释放)语句	(99)
5.3.8 编程与方法	(99)
习题五	(106)

第六章 过程

6.1 过程概念	(109)
6.2 函数过程	(110)
6.2.1 函数程序单元的编写	(110)
6.2.2 哑元的确定及哑元的 INTENT 属性	(112)
6.2.3 函数调用	(113)
6.2.4 接口块	(114)
6.2.5 各种哑数组形式	(116)
6.3 子例子程序过程	(117)
6.4 关键字变元、可选择变元、哑元改名	(119)
6.4.1 关键字变元	(119)
6.4.2 可选择变元	(120)
6.4.3 通过接口块更改哑元名称	(121)
6.5 类属过程、重载操作符、自定义操作符、重载赋值号	(122)
6.5.1 类属过程	(122)
6.5.2 重载操作符、自定义操作符	(123)
6.5.3 重载赋值号	(125)
6.6 外部过程与内部过程	(126)
6.6.1 内部过程的位置	(126)
6.6.2 内部过程的使用	(127)

6.7 多层调用、结构图、哑过程	(128)
6.7.1 多层调用	(128)
6.7.2 分解模块与结构图	(130)
6.7.3 哑过程	(132)
6.8 递归过程	(133)
6.8.1 递归函数	(134)
6.8.2 递归子例子程序	(135)
6.8.3 间接递归	(136)
习题六	(137)

第七章 数据结构、派生类型和模块

7.1 数据结构	(139)
7.2 派生类型	(140)
7.2.1 按结构要求定义派生类型	(140)
7.2.2 派生类型的应用	(142)
7.2.3 结构构造函数	(143)
7.3 模块	(144)
7.3.1 共享	(144)
7.3.2 模块编写	(145)
7.3.3 模块的引用	(146)
习题七	(155)

第八章 指针变量、链表、树

8.1 指针	(156)
8.1.1 指针变量的定义	(156)
8.1.2 指针赋值语句	(157)
8.1.3 包含指针的一般赋值	(158)
8.1.4 指针的使用	(158)
8.1.5 建立新节点、指针置空和状态查询	(162)
8.2 链表	(164)
8.3 树	(167)
8.3.1 二叉树	(168)
8.3.2 二叉树程序	(169)
8.4 指针与数组	(172)
习题八	(173)

第九章 数据文件的输入输出

9.1 数据文件	(175)
9.1.1 数据文件的重要性	(175)
9.1.2 数据文件与部件号的连接	(175)

9.1.3	部件号与数据文件的连接	(176)
9.1.4	数据文件的使用	(177)
9.1.5	读写的格式控制	(178)
9.2	文件与记录	(180)
9.2.1	文件的定位	(180)
9.2.2	文件的状态	(181)
9.2.3	文件的多种属性	(183)
9.2.4	顺序文件与直接文件	(183)
9.2.5	有格式文件与无格式文件	(187)
9.2.6	外部文件与内部文件	(188)
9.2.7	连接表其它说明符	(190)
9.2.8	OPEN 语句的连接表	(191)
9.3	数据传输与传输辅助语句	(191)
9.3.1	读写语句	(191)
9.3.2	高级传输与低级传输	(193)
9.3.3	OPEN 语句与 CLOSE 语句	(194)
9.3.4	BACKSPACE 语句、REWIND 语句及 END FILE 语句	(196)
9.3.5	INQUIRE 语句	(198)
9.4	格式说明中使用的编辑描述符	(201)
9.4.1	数据编辑符	(201)
9.4.2	字符、逻辑值编辑符	(205)
9.4.3	控制编辑符	(205)
习题九		(209)
 第十章 FORTRAN 77 中的过时特性		
10.1	过时的书写格式	(211)
10.2	程序中的过时功能	(212)
10.3	数据说明的过时形式	(215)
10.4	过时的控制转移方式	(217)
10.5	其它过时语句与特性	(220)
 附录 A FORTRAN 90 内部函数与内部子程序		
		(227)
附录 B FORTRAN 90 的程序单元与语句		
		(235)

第一章 编程基础知识

FORTRAN 90 是具有强烈现代特色的语言,它淘汰了 FORTRAN 77 中过时的内容,吸收了 PASCAL、C 语言的长处,总结了现代软件的要求与算法应用的发展,增加了许多现代特征的新概念、新功能、新结构、新形式,把算法语言推进了一大步。

1.1 FORTRAN 90 的现代特性

FORTRAN 90 的现代特性表现在以下四方面:

1. 加强了程序的可读性、可维护性

它淘汰了所有转移语句,用新的控制结构实现选择分叉与重复操作,使程序结构化。

它增加了结构块、模块及过程的调用灵活形式,使程序易读易维护。

2. 发展了现代算法功能

它加强了数组的算法功能,使数组像一个变量一样自由操作。

它提出超载概念与实现方法,允许运算符与赋值号超出原定功能使用。

它提高了文字处理功能,胜任信息管理系统、办公自动化的任务。

3. 扩大与编程者的友好界面

它的编程新形式减少烦琐记忆与人为的格式束缚,接近自然语言与公式演算,有些地方还允许疏忽脱漏、次序颠倒。

它允许在字符数据中选取不同种别,使程序员在字符串中可使用各国文字(例如汉字),还可任意使用化学、物理、数学的各种专业字符,使程序员编程时顺利流畅。

4. 淘汰过时的特性

FORTRAN 语言是最早问世的高级编程语言,国际标准化组织 ISO 每隔 10 年左右就为它制定一次新的国际标准文本供各国遵循:1966 年的 FORTRAN IV,1977 年的 FORTRAN 77 等。FORTRAN 90 是 1990 年制定的国际标准,为了保持向上兼容性,它虽然允许使用 FORTRAN 77 的全部语句,但又明确地把 FORTRAN 77 中 10 多个语句与一些特性定为应予淘汰的过时项目。新编程序时不应再予使用。原有老程序中遇到这些过时部分则用 FORTRAN 90 中现代特性修改替代。本书最后一章“过时的特性”将把过时特性逐一列出。

1.2 FORTRAN 90 的程序形式

1.2.1 程序总体构造形式

FORTRAN 90 程序是一种分块形式的程序,整个程序由若干个程序单元块组成,各单元都有相

似的语句组织形式,其中主程序单元起整体控制作用,各子程序单元各自完成问题中的一个算法。编程时,先把求解的问题分解为若干相对独立的子算法,每个子算法编写成独立的子程序单元。主程序依次调用各子程序单元,控制各子算法的实施,通过主程序对子程序的调用,形成程序的整体运行,完成问题的解。这种控制与调用的整体关系,可以用程序的总体结构图来描绘,如图 1.1 所示。

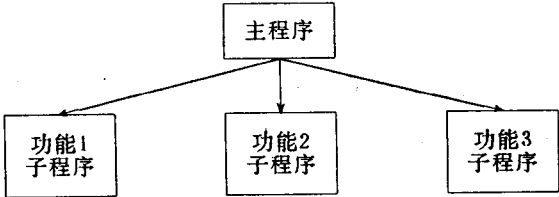


图1.1 程序结构图

若功能 1 子程序算法仍复杂,可再把功能 1 分解为若干更小的算法,分别编写为更低一层次子程序,由功能 1 子程序分别去调用,结构图如图 1.2 所示。这种情况可以类推到其它子程序或更低一层次。

所以:

$\text{FORTRAN 90 程序} = n \text{ 个程序单元} = 1 \text{ 主程序单元} + (0 \sim n-1) \text{ 个子程序单元}$

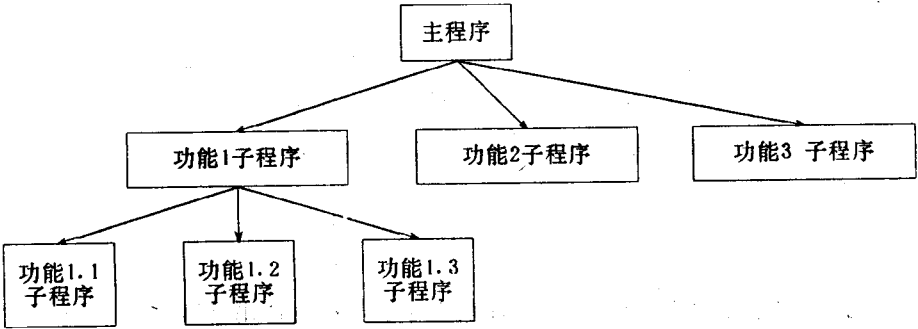


图1.2

在最少情况下,FORTRAN 90 程序只由一个主程序构成,而没有子程序,所有算法都由主程序自身完成。本书前半部分,因还没有介绍子程序编写,程序编写成只有一个主程序形式。这不是好的编程方法,按现代要求,即使功能比较简单的问题,也最好写成主程序调用子程序的形式,以便于维护。

1.2.2 程序单元的编写

1. 程序单元的基本框架

不论是主程序或子程序,每个程序单元都独立编写,形式相似,它的基本框架是标志单元性质的开始语句,然后是程序体,最后是该程序单元的结束语句。编写时,为了阅读醒目,最好把程序体的语句放到比程序开始语句与结束语句后退几格的位置(例如后退 3 格)。如主程序 ADD 单元与函数子程序 A1 形式如下:

主程序单元	函数子程序单元
PROGRAM ADD	FUNCTION A1(X,Y) RESULT(A1_RESULT)
...	...

程序体

...

END PROGRAM ADD

程序体

...

END FUNCTION A1

2. 程序体

各程序单元(除模块程序单元外)的程序体形式相同,共分两部分,前面是说明部分,后面是执行部分,两部分之间没有确切的分界,紧密衔接,但不准彼此穿插,也即:

程序单元=单元起始语句+程序体+单元结束语句

程序体(可执行的)=说明部分+执行部分

说明部分只允许写说明语句,说明语句最基本的一般形式是:

类型关键字::变量名 1,变量名 2,...,变量名 n

类型关键字指明变量的数据类型,如 INTEGER(整型)、REAL(实型),等等,必须严格按语法书写,不可有一点误差,双分隔符::后面的变量名是被说明的对象,它们的类型即关键字指定的类型(所有语句结尾不可有标点符号),例如:

REAL::A,B,C

则说明变量 A、B、C 是实型。其作用一是通知机器系统在内存中分配相应的存储单元,按各自数据类型存储;二是帮助读者理解后面执行部分的算法。

执行部分书写可执行(动作)语句,每一语句控制机器作一具体动作(操作),执行语句最基本的一般形式是:

关键字 变量名 1,变量名 2,...,变量名 m

关键字指定具体操作,如读、写、分配等,变量名是被执行操作的对象。如:

PRINT *,A,B,C

就指定机器打印输出 A、B、C 变量的值。

上面说明语句、执行语句的一般形式只是最简略的共同形式,实际书写时,由于要求指明各种附属性质,往往需增加新的信息。

3. 语句行与注释

程序体的每一部分由语句行及注释组成。语句行书写语句,一般一行只写一个语句,没有标号区、续行区、正文区等区别,可以从行内任何一列写起,一行最多为 132 列,允许在任何列处中断语句,中断处写一个 '&' 号,随后在下一列任何列处继续。一个语句最长不能超过 2640 个字符,语句结尾不可有标号、符号。

在不妨碍程序可读性时,也允许一行中写几个语句,此时各语句间用分号 ';' 分隔,最后一句仍不许有标点符号,例如:

X=1.1;Y=2.2;Z=3.3

与

X=1.1

Y=2.2

Z=3.3

等价。

注释用感叹号 '!' 开头,可以整行作注释行,也可以写在语句行的右边作为对该语句的注释。为了阅读醒目,通常语句用大写英文字母,注释用小写英文字母。注释作用是向程序读者说明下面一段语句或本行语句的算法性质、条件、方法、用途等,以帮助读者理解程序,虽然它不参加编译,在

程序执行中不起任何作用,但却是不可省略的程序重要组成成份,它加强了程序的可读性与可维护性,对以后程序测试、修改、扩充起重要作用。

4. FORTRAN 字符集

FORTRAN 语句中允许使用的字符是有限的,它们的全体称 FORTRAN 字符集,共有 84 个,计有:

26 个大写英文字母: A~Z

26 个小写英文字母: a~z

10 个十进制数码: 0~9

1 个作连接用的下划线: _

21 个特殊字符:

空格 ; = + - * / () , . ' : ! " % & < > ? \$

上面是国际标准规定的字符集。此外,标准还允许 FORTRAN 90 扩充进各国文字、各种专业用符号,这要看厂商装入的编译系统是否支持这些字符的使用。

1.2.3 编程实例

已知直线方程 $y=1.5x$ 。对任意给定点 x 的坐标,计算其对应的 y 值。我们把它编成只有一个主程序单元形式。为了说明方便,在每个语句前加上编号,它不是程序的组成部分。

```
① PROGRAM Y_ACCORDING_TO_X
②      ! for each point on line  $y=1.5x$ , given  $x$ , calculate the  $y$ 
③      REAL::X,Y
④      PRINT *, 'INPUT X'
⑤      READ *, X           ! input the value  $x$ 
⑥      PRINT *, 'X=', X
⑦      Y=1.5 * X          ! calculate the  $y$ 
⑧      PRINT *, 'Y=', Y
⑨      END PROGRAM Y_ACCORDING_TO_X
```

本程序共 9 行。

①是主程序语句,标志主程序单元开始,PROGRAM 是关键字,意为‘程序’,关键字后空一格写程序名,程序名由程序员任取,但不要怕麻烦而使用无意义的短名字,如 X、Y 等,应采用能反映整个程序功能的名称,如果这一功能必须由若干词组成,可以在词之间用下横线‘_’连接成一个名,如这里主程序取名为 Y_ACCORDING_TO_X,意即‘X 对应的 Y’。

②由‘!’开始,表示该行内容是注释,向程序读者进一步解释本程序算法功能,即‘对于直线 $y=1.5x$ 上每一点,给出 x , 计算 y ’。

③REAL::X,Y

进入程序说明部分,在本程序中,说明部分只有这一个说明语句,REAL 是关键字,意为‘实型’,::称双隔号,由两个‘:’连成,::后的 x 、 y 被说明为实型变量,各变量间用逗号‘,’隔开。

④PRINT *, ‘INPUT X’

进入执行部分,PRINT 是关键字,表示‘打印’,* 规定打印的格式,表示程序是不规定打印格式,由机内表控格式规定。用一对撇号定界的‘INPUT X’称字符串,整个语句表示由机器按表控格式打印出‘INPUT X’字符,这是为了人机会话,由机器提示上机者应输入 x 的值。

⑤READ *, X

这是读语句, READ 是关键字, * 是表控格式, 整个语句表示要机器读入一个数给 X, 执行到此语句, 机器会显示一个等待一个输入的信息(各机不一样, 要查上机手册), 等待上机者按键输入 X 的值例如 2.0。本行后半部是注释, 表示‘此处要输入 X 值’。

⑥PRINT *, 'X=', X

机器打印字符‘X=’, 而后再打印 X 的值, 设 X 值是 2.0, 执行到本语句后, 屏幕显示‘X=2.0’字样。本语句目的是把上面输入的 X 值再次显现, 让上机者检验机器收到值与输入值是否符合, 防止程序因输入失误而出错。

⑦Y=1.5 * X

称为赋值语句, 这是 FORTRAN 90 中唯一没有关键字的语句, 写法与数学形式一致, 只是乘号用‘*’表示, 功能是计算 $1.5 * x$, 把值赋给 y。后面是对本语句功能的注释。

⑧PRINT *, 'Y=', Y

按表控格式打印字符‘Y=’, 再输出 Y 的值, 执行此语句, 机器显示: Y=3.0。

⑨是主程序单元结束语句, 标志主程序单元结束, END PROGRAM 是主程序单元结束语句的关键字, 空一格后写本程序名, 此名应与开始语句中程序名一致, 如果漏写程序名, 也可正确执行, 但写上程序名可使程序首尾清晰, 可读性好。

1.3 结构图与流程图

一个大的程序, 应先画出结构图与流程图, 而后才动手编程。结构图是对整个程序结构的设计, 包括划分哪些程序单元, 它们之间的调用隶属关系, 如图 1.1 及 1.2 所示。对于简单的问题, 划分的程序单元不多, 结构图可以省略。

对每一程序单元, 还应画出它们的流程图, 流程图是对该程序单元控制流程的设计, 如果流程图设计得好, 程序编写变得非常容易, 只要按流程中每一框形写入相应语句就可。下面画出 1.2.3 中例题的流程图(图 1.3)。旁边是它的对应 FORTRAN 语句。只要稍加注意, 就可看出流程每一框旁边的语句基本上是对号入座的。

流程图有各种表示方式, 本流程图的表示方式比较直观易学, 它是由表示控制流向的箭头与表示不同功能的六种框形组成。

六种框形的名称与功能见图 1.4。

1.4 程序的运行

编好的程序称为源程序, 还不能直接运行, 上机时还要机器对源程序作编译与连接加工, 得到可执行程序, 才能在机上运行得出结果, 如图 1.5 所示。

上机时, 先将各有关程序单元合放在一个文件内(次序任意, 最好主程序单元放在最前面), 输入机内, 发布编译命令, 将各程序单元分别编译。编译完成两个功能: 一、查出源程序的语法错误, 如显示有错则回上去修改源程序; 二、若语法无误则把源程序翻译成二进制的机器指令代码, 并为每个变量分配相应的存贮单元。

第二步发布连接命令, 把机内一个个编译好的程序单元以及其它需要的机内现成的功能块, 一

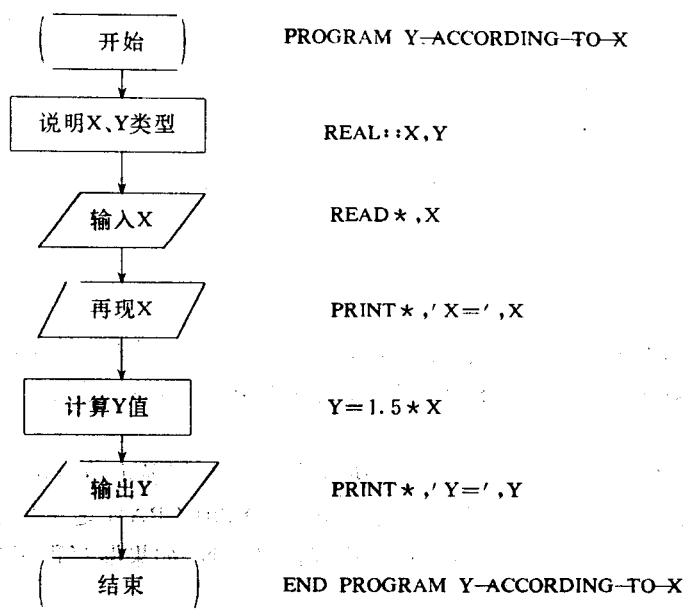


图1.3 流程图

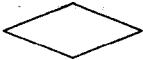
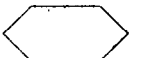
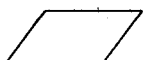
形状	名称	功能
	矩形框	用作一般性处理,如说明类型、计算、赋值等
	菱形框	根据条件判断,选择流向
	六角框	检查循环次数是否满足,而后确定转向
	平行四边形框	表示输入或输出
	圆弧边框	表示流程开始或结束
	圆圈框	当流程图太大,一页内画不下时,断点与续点都用圆圈,圈内写同样字符,表示彼此衔接

图1.4 各框形的名称与功能

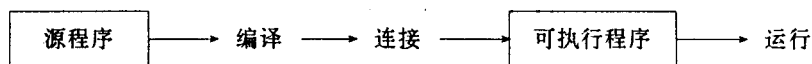


图1.5

起连接成一个完整的 FORTRAN 程序形式,称为可执行程序,如发现缺少某些需要的单元则回上去修改源程序。

第三步发布运行命令,运行可执行程序,输入数据,输出结果,运行上发现数学上的错误,如分母为零、数值溢出、负数开方等还要回上去修改源程序。

上机者在机上操作流程如图 1.6 所示。

因此上机者除了要学会 FORTRAN 语句编程外,还必须学会若干条操作命令、连接命令、执行

(运行)命令以及读懂系统反馈的出错与否信息。

各种操作命令在不同机型上写法不同,要查看具体的上机手册。在某些机上,只需打入一条操作命令就可使编译、连接、执行一起完成。但它在机内加工步骤仍如图 1.5 所示,不同阶段出错时仍反馈不同的出错信息。

程序通过后,得出结果仍可能有错,因此还要进行测试,不要输入一套简单的数据得出满意结果就认为程序的算法是正确的,因为这只是一种特殊情况下的正确,并不能保证数据在另外几种情况下特别是边缘状态情况下也能得出正确结果。因此要作严格的多种角度测试后才能初步认为正确,这过程中又要修改程序。

1.5 程序的可读性、可维护性与可使用性

上节表明,一个程序从编好到能得出正确结果,至少有四次改错阶段,每个阶段内又有许多次具体修改,每次修改都要阅读源程序,读懂源程序各变量意义及语句的算法逻辑。所以,所写程序必须保证有可读性,让调试者、测试者、改正者能顺利清晰地读懂程序,才能查出错误。查出错误后,还要着手修改,又要求所写程序必须易于修改,每次修改只影响到局部范围,不要牵涉很广,牵一发而动全身,以致改正一处,却带进几处新错误,这就要程序具有可维护性。

现代软件要求程序必须有可读性与可维护性。

要使程序有可读性、可维护性,方法有多种。FORTRAN 90 已提供相当多的手段保证程序的可读性、可维护性供我们运用,关键是编写程序者要有一个明确的指导思想:编写程序不只是供机器运行,同时也要供人阅读、供人修改。编程时不仅面向机器,同时要面向人。

程序还要有可使用性,即与用户界面友好,使用方便,使用户只要最少的记忆、最少的预备知识、最少的输入就可取得所需的各种数据。编程时要处处为用户着想,如通过人机对话提示输入数据的类型、属性、物理意义,输入数据作再显示供检验,为不同需求用户提供菜单,运行中各种意外情况的提示与处理对策,为输出提供标题、表格、图形等。

要具有可读性、可维护性、可使用性就必须建立文档,现代软件不只是一整套程序,文档也是软件的重要组成部分,文档是指从课题确立时起,一直到交付使用整个时期内的各种技术档案,包括软件需求分析报告、软件整体设计、结构图、流程图、规范书、程序内部用注释写出的内部文档、使用说明书,等等。这样才能使后来的测试、维护、使用有所遵循。没有文档的程序是不完整的软件。

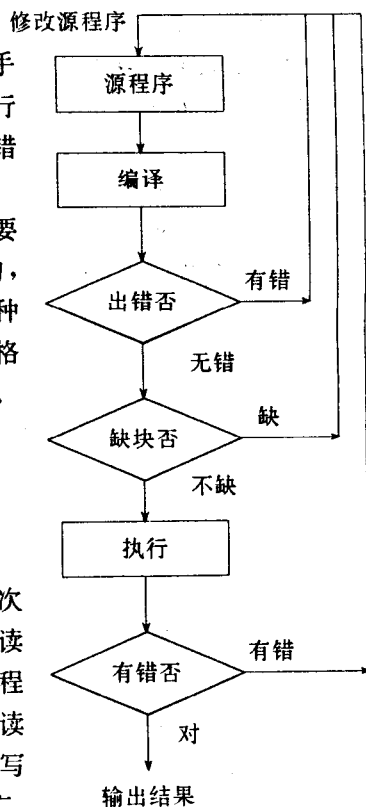


图 1.6

习题一

1. FORTRAN 程序与程序单元是什么关系?