
FORTRANZZ
FORTRANZZ
FORTRANZZ
FORTRANZZ
FORTRANZZ
FORTRANZZ
FORTRANZZ
FORTRANZZ
FORTRANZZ
FORTRANZZ

FORTRAN 77

Donald M. Monro 著

翁瑞琪等译

许镇宇 校

天津科学技术出版社

FORTRAN 77

Donald M. Monro 著
翁瑞祺　　等译
许镇宇　　校

天津科学技术出版社出版
天津市赤峰道124号
天津市新华印刷厂印刷
新华书店天津发行所发行

开本 787×1092 毫米 1/16 印张 29.5 字数718,000
一九八五年九月第一版
一九八五年九月第一次印刷
印数：1—12,000
书号：15212·164　定价：5.45 元

内 容 简 介

本书较全面、系统地介绍了FORTRAN77语言及其程序设计方法。全书共分十九章，内容包括FORTRAN77中的各种数据类型、各种语句、程序结构以及程序设计的基本原理和方法等，还给出多个实例研究，各章附有大量练习题。

本书内容深入浅出、简明易懂，是一本较好的FORTRAN77程序设计的教科书。可供高等院校、中等专业学校计算机软件与硬件专业、理工科其他专业师生，以及研究、应用计算机的科研人员，工程技术人员和管理人员学习参考，还可供计算机培训班参阅，或作自学用书。

译者前言

程序设计语言FORTRAN是科学计算中使用极为广泛的一种计算机语言。FORTRAN77是FORTRAN的最新发展。由于在语言结构上的改进，FORTRAN77能较好地体现结构程序设计的原则，加以在一些语法规则上的放宽和用于文本处理方面的一些新功能的引入，它不仅在数值计算方面继续保持其重要地位，而且在非数值计算方面也为人们所注目。

本书较全面、系统地介绍了FORTRAN77语言及其程序设计方法，是一本较好的学习FORTRAN77程序设计的教科书。

本书译自Donald.M.Monro所著《FORTRAN77》（1982年第一版）一书。译文中，由于许多术语尚无统一译法，我们尽量采用较通用的译法。对原书中的一些错误也作了改正。

本书是根据讲授FORTRAN77程序设计课程的教学需要而翻译的，由翁瑞琪译第一、六、七、十六、十七各章，艾德才译第十三、十四、十五、十九各章，任焱译第十、十一、十二各章，汪大菊译第二、三、四各章，散襄军译第八、九各章，王万年译第十八章，李英慧译第五章。全书由翁瑞琪负责初校和统一，最后由许镇宇教授校定。

由于时间仓促和限于水平，肯定有不少缺点和错误，恳切希望读者批评指正。

1984年5月25日

目　录

第一章 绪 论

1.1 程序设计是什么?

人们在日常生活中习惯于对各种事情给出指示，甚至最谦让的人有时也要留下一个通知给送牛奶的工人，要求额外增加一品脱牛奶。这就是以书面命令为牛奶工人编程序的一种方法。同样，为计算机编程序（叫做程序设计）也是把机器要遵循的指示（叫做指令）列出清单。当然，计算机所做的事情总在某些方面与数字有关；人们未必（还未）要求计算机来拉下窗帘，但是很可能要求把两个数加在一起。要把两个数加在一起，程序员可说：

"Take number A and add it to number B and call the result C"（取出A这个数，与B这个数相加，结果称为C。）

这是用自然语言（英语）表达的一个程序，这段英语也可以是计算机语言，虽然这样陈述未免太罗嗦了。不会有多少程序员会长时容忍编写这样冗长的语句（虽然有些COBOL程序员似乎很能容忍）。较切实的是可以说

```
LET  C = A + B        (BASIC)
```

或只说

```
C = A + B             (FORTRAN)
```

或可说

```
C : = a + b           (PASCAL或ALGOL)
```

计算机提供了一系列便于语言实施的设施，而程序设计语言本身也包含了支持其使用的结构。这些设施是完善的，许多程序命令计算机实现十分复杂的任务。一项任务能否成功地完成取决于指令或"处方"的正确性。这样的处方称为一种算法，而程序设计就是用程序设计语言表达一种算法或"可计算的处方"的过程。

因为计算设施功能很强，因而也很昂贵，所以程序员应关心所用算法的效率。然而，由于程序往往是复杂的，使人难于理解，甚至包括原先的编程序者。如果任何事情多少值得用计算机来做，必定会使其他人感到兴趣。所以技巧复杂而又杂乱无章的程序往往是令人讨厌的。程序员还应注意程序的格调与组成；换言之，应注意其结构。一个好程序员应在格调与效率之间进行平衡，所以程序设计是一种技巧。一位不太著名的英国哲学家D.D.Burgess曾阐明了音乐的艺术与技巧之间的区别，这一区别可用在计算机程序设计上：不管程序员怎样有灵感，占第一位的是技巧的训练。

1.2 关于FORTRAN

FORTRAN 这个名字是FORmula TRANslation（公式翻译）这两个英文字的缩写。FORTRAN在1954年作为一种较简单的、范围较窄的语言开始出现，但是在1958年的修订以及1966年的再修订之后，它变成FORTRAN Ⅳ，成为在科学与工程方面的数据处

理与数值计算的最通用的工具。出现了许多种专用语言，这些专用语言各自理想地适用于某一类难于解决的任务，在学习FORTRAN之后再去学习这些专用语言的任一种都不会有任何特殊困难。

使用FORTRAN的经验使人们察觉到了它的缺欠，尤其是关于它在处理文本（书写的字）方面的缺欠，因为它的控制语句不太适宜表现结构化程序这一概念。新版本FORTRAN 77 在改正这些缺欠方面大有进展，它使该语言在其卓越的持久性方面又进入一个新纪元。所以，FORTRAN仍然是在商业范围（在商业范围流行的是COBOL）之外的领域中最有用的一种计算机语言。

所有语言都具有构造规则或“语法”，在计算中语法必须精确，以使语言中的语句不能有一个以上的含义。FORTRAN 77 对其以前几个版本的严格的语法有所放松，这点在第一次学习时是很重要的。由第二章第一个例子可知，FORTRAN 77 可用自然的方式来写出一些事情，这是FORTRAN Ⅳ 所不能容许的。这种新的灵活性可使初学者把更多的注意力集中在他所做之事的意义上，即FORTRAN程序设计的“语义”。从现在起，本书中采用FORTRAN这个字来表示FORTRAN 77 。

1.3 关于计算机

人类的一个特征是创造精神，这种创造精神导致他们开发出一些能减轻其负担或加强其能力的一些工具。每种新的开发不可避免地又需要其它方面的开发。计算机一方面具备了使早先的创新所需要的重复计算能自动进行的能力，而同时却又要求发展程序设计的技巧。

每台计算机是这样构造的一种机器，它具有一个由简单指令组成的指令系统，在人的指导下机器盲目地服从这些指令。把这些指令编制成一个合适算法的工作叫做程序设计。所完成的指令表叫做程序，它是用程序设计语言（经常用FORTRAN）向机器表达的。计算机无法知道给它的指令是否有意义或程序员真正打算的是什么。它完全按字义解释这些指令，很可能反复进行同一个毫无意义的操作而被卡住，一直到人去干预它或用一个定时电路使它停止。试图获得一个正确工作的程序的人可能有许多错误，但他（通常）知道自己的意图因而能推断出哪里出了错。计算机程序设计的大部分工作就用在找出程序中的错误。

机器通常不会出错，但也不进行判断。当初学者在他们自己的程序中不能发现错误时，就急于归咎计算机出了错。但是，应当知道，在你作程序员的最初几小时内，你自己的错误会多得无法计算下去，而在你一生中也未必能遇到真正由于机器本身所造成的错误。

倘若适当地指挥计算机，它就能在几秒钟内或几分钟内完成超过人工计算一生所完成的计算量。这就是计算机对社会产生了如此深远影响的原因。这些影响并不总是有经济效益的，尤其当“计算机化”的决定是在无视涉及到大量开销与高度专门化技巧的情况下作出时，更是如此。但是，计算机可在一秒内加一百万个数，它能以高精度快速相乘。一台计算机能在其存贮器中存贮成千个或成万个结果，并能在一微秒内再调用其中任一个。

程序设计的一个关键部分是作出判断，从而使机器具有智能（响应）的外表，但是这个智能来源于人类程序员，而计算机的明显的错误几乎也都来自人类程序员。值得注意的是，一些预言家们正在谈论用计算机开发它们自己的智能这一天的到来。

为使一个计算机系统能够使用，必须用设备馈入信息，并用“系统程序”进行任务指

导。学习FORTRAN的人们可以通过一个带有键盘与打印机的终端进行通信，如果运气差些，或许必须提交一些卡片给一个“成批服务”，并随后由它取回打印结果。

一台计算机可能连接卡片阅读机、纸带阅读机、卡片穿孔机、纸带穿孔机、磁带机、行式打印机与磁盘存贮器。所有这些装置是为输入信息（到计算机）与(从计算机)输出信息而提供的。每个装置有一个控制它的“驱动”程序，而对驱动程序还要有一个管理程序（可能对管理程序还要有一个监督的程序）。在可引入FORTRAN程序之前，一个计算机系统就是由所有这些装置与程序组成的（图1.1）。

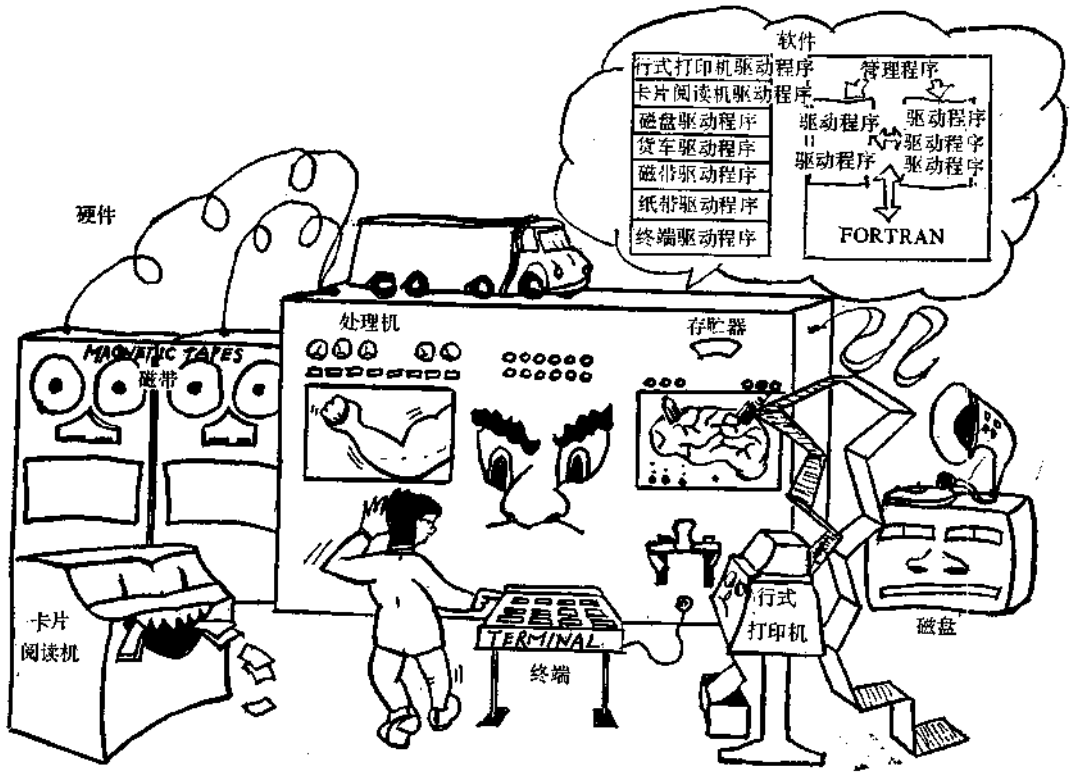

图1.1 计算机机房内的各种装置

1.4 交互计算与分时

一个交互计算机系统使程序员与计算机直接通信，通常是通过一台打字机进行的。所以开发新程序与查找并校正错误的“周转时间”减为几秒钟。程序本身可写成这样，程序员在计算机执行它时给它以信息，所以他能在计算进行之中控制计算的步骤。当用交互系统运行FORTRAN时，可在一个终端上快速开发程序并检查与校正之。学习过程缩短了，而且变得更彻底了，因为计算机的快速响应和语言的直截了当的特点能获得学生的好感，并鼓励他们进行尝试。

从程序员的角度看，一个交互计算机系统可处于两种操作方式中的一种。这两种操作方式称为“程序定义”方式与“程序执行”方式，这由控制事件的究竟是程序员还是计算机来

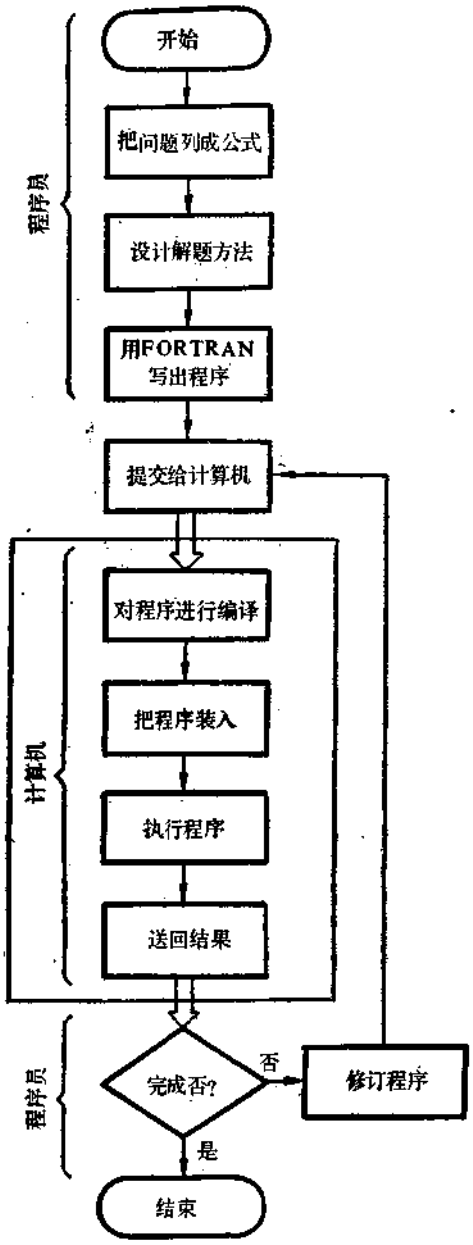

图1.2 开发一个FORTRAN程序的各个步骤

区分，如图1.2所示。在程序定义方式中，程序员将建立、编辑与校正他的程序，是他本身在进行控制。主信息流是从终端到计算机，计算机的任何响应是程序员活动的结果。程序员可输入命令给系统，命令中的一个可能是把程序提交给计算机。如果这一点已完成的话，系统就变为执行方式，于是，只有当程序已作出从终端输入的安排时，才要求用户响应。计算机实现图1.2所示的步骤通常是在一个看不见的操作中发生的。主信息流将是从计算机到终端，通常在程序完成时程序员取回控制，虽然若有需要他可手控地停止执行。

计算机本身并不了解FORTRAN，计算机所服从的机器代码是一系列望之生厌的数字。这需要有一个翻译程序或“编译程序”，它取入FORTRAN程序，而把它变换成机器语言。FORTRAN本身就有许多功能，而又需要检查FORTRAN程序的文法，所以编译程序本身是一个很大的程序。但是，对FORTRAN的多年经验，已使人们能够编制出效率很高的FORTRAN编译程序。

制备一个FORTRAN程序的步骤如图1.2所示，该图是本书中第一个流程图，本书中有许多用来描述主要由计算机程序实现的那些进程的流程图。这一个流程图所描述的是开发一个计算机程序的各个阶段。第一步是对问题的说明，它可能是最关重要的部分，这一步要求程序员必须学会清晰地写成公式。然后用第六章所介绍的“逐步求精法”设计一个过程。程序要用FORTRAN写出，然后把它提交给计算机。机器把程序编译成其自己的语言，然后将它装入机器本身并遵循或“执行”该程序。计算结果要打印出来并交给用户。然后程序员决定结果是否令人满意，判定总是用一个菱形框来表示，如图1.2所示。如果由于某些原因结果是不正确的，程序员就修订程序再提交给计算机。这修订“循环”有可能要重复许多次。成批系统与分时系统之间的差别在于所用的机构与用以实现其机器部分的方法。初学者无需详尽了解所有这一切，但FORTRAN是这样一种语言，它允许内行扩展系统的许多功能。

1.5 成批处理

计算机服务最可能遇到的方式是成批处理，虽然成批处理对学习并不理想，但是对大多数应用中的生产工作，确是一个高效率的系统。在这一类的计算机构中，在一中心场所提交程序，而把这些程序成批地馈送到计算机中。过一些时，当机器已完成这一批程序的处理而转向另一批程序时，打印结果就和程序一起退回，在大多数情况下这程序要加以修正并再次提交。即使是一些不大的作业，成批服务的“周转”在最好的情况下也需要几个小时，有时往往通宵。所以，学习FORTRAN的人们的进展可能很受阻碍。象这样的服务，要求人来居间：组织输入、排序输出并应付成百的抱怨的用户。用户本身也多半要对接待部门发生意见。应当记住，处理的延迟不一定是接待部门的（倒霉的）负责人的过错；多数计算机似乎具有在最坏的时刻令人伤脑筋的怪癖。

某些开明的计算机中心允许用户自己利用足够的设备来运行他自己的“作业”，用这方法来缓和成批处理的延迟，以使顾客满意。允许用户不仅看到并且“接触”设备这一想法所引起最初骚乱平息之后，通常这种组织良好的“动手作坊”是成功的，虽然有时不够整洁。在这种作坊中，人们可读入自己的卡片并撕走自己的输出。

几乎所有的成批处理工作是由穿孔卡片进行的，组成一个“作业”的卡片叠必须包含全部FORTRAN程序和使计算机系统运行所必需的指令，以及要求程序去处理的任何数据。就如同卡片叠的组织有不同方式一样，这些表现为“控制卡片”或“作业控制语言”形式指令也因不同的计算机而大不相同。典型的动作是要用控制卡片去启动把原始的“源代码”（FORTRAN）翻译或“编译”成机器语言这项工作，把已“编译”好的程序或“目标代码”装入计算机中，查找系统程序库看程序位有无丢失然后使之运行或“执行”。其各个步骤仍与图1.2中所示出的一样，只是程序员要卷入实际的步法，所以要花费气力去弄清楚计算机的更详尽步骤。对于经验较丰富的程序员来说，成批处理的好处是有很多其他设施可供利用，这也就是必须更详尽地弄清所要求的各个步骤的原因。

1.6 如何使用本书

本教程要从头学起，每隔几页都有练习，应全部完成。每章末了的问题是按由易到难的顺序排列的，学生应利用这些问题来提高其技巧，而教师则可用它对学生进行评定。

在试图解答问题之前，应通读与该问题有关的各节内容。即使是最初步的解法，如事前写出，也可节省上机时间。年复一年都见学生临到键盘面前才第一次读资料和思索解法，这耗费的时间是可悲的；就是这些人却又常抱怨接触机器的时间不够。

有些节只对经验丰富的程序员才感兴趣，可跳过这些节，不会漏掉FORTRAN的任何基本部分。

应征求内行的意见，首先是对FORTRAN的意见，因为曾作出各种错误的人在认出这些错误的方法（这就是所谓的“经验”）上是很聪明的，还有关于对计算机系统的意见，计算机系统给出的困难在最初可能比FORTRAN还多。

第二章 用实数计算

2.1 引 言

用一系列指令指挥计算机，确切地告诉计算机每一个计算阶段做什么，一组这样的指令称为程序。使用程序设计语言来表示指令，其方法与计算机的操作细节无关。FORTRAN 77是一种高级语言，因为计算是用人所熟悉的名词来表达的，而不是用机器所熟悉的名词来表达的。FORTRAN程序使用普通的英语名词和数学运算。但是，因为是和计算机通信，给出的指令必须是确切的，而不允许有多义性。因此FORTRAN的文法，象任何其它计算语言一样，受一组周密定义的规则所约束，这些规则控制什么样的文法构造或"语法"将被机器所理解。

因为规则明确，用FORTRAN进行程序设计是几乎每个人都能做的事；经验有助于该语言的容易而流畅的使用。这就是为什么FORTRAN是科学计算的通用语言并已经持续许多年的原因之一。在本章只给出最基本的文法和构造，足以进行简单的计算，虽然某些资料必须在以后详细介绍。

现在用一个完整独立的程序的很简单的例子来引出该语言并指出构造的某些特点。以下是一个完整的FORTRAN 77 程序，它计算并打印2.0+2.0之和：

```
PRINT *, 2.0+2.0
END
```

这个程序中有些东西看来面熟，另一些几乎本身就说明了。FORTRAN使用英语中的字和某些可认识的数字符号，但它也有非常严格的标点规则。

这样一个简单程序的结构是足够直截了当的。程序的每一行称为FORTRAN的一个语句，以后我们将会看到有些语句是加标号的。当一个FORTRAN程序被一台计算机"执行"时，语句的顺序指出了这个程序中给出的指令所遵循的顺序。所以这个例子恰好按读它那样是逐行执行的。

PRINT语句指挥计算机把2.0和2.0加在一起并在某处打印结果。给出这种PRINT语句，机器将把答案打印在明确的地方——在分时系统的终端或在成批环境的宽行打印机上。星号后的逗号则是所需的标点符号。

END语句表示程序的结束，当执行到这个语句时，计算机就知道这个作业已经完成。所有程序都用一个END语句结束。

用流程图来思考程序往往方便一些，流程图说明程序的结构。在图2.1中给出上例的流程图，它用图示的方法指出程序各个步骤的导向。

由上例可得知，简单计算是能立即进行的。

练习：

指出以下几个短程序的意思：

(i) PRINT*, 4.0-2.0