

21

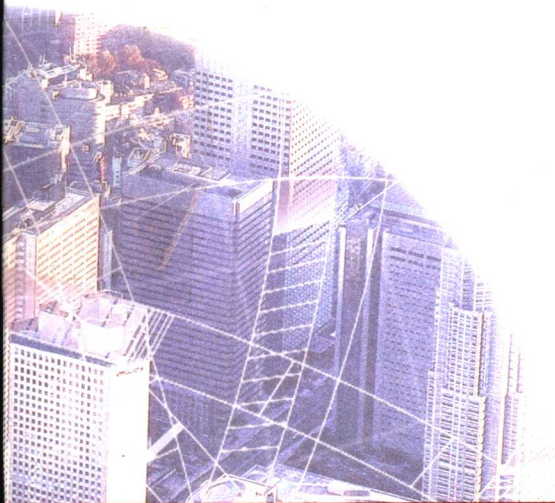
世纪高等学校计算机科学与技术规划教材

FORTRAN 90 程序设计教程

Fortran 90 Chengxu Sheji Jiaocheng

主编 / 刘卫国 蔡旭晖

(第2版)



北京邮电大学出版社
www.buptpress.com



FORTRAN 90 程序设计教程

Fortran 90 (Theory and Study) Manual

1 清华大学出版社

001 001

清华大学出版社

TP312

1070=2

2003

21 世纪高等学校计算机科学与技术规划教材

FORTRAN 90 程序设计教程

(第 2 版)

主 编 刘卫国 蔡旭晖



北京邮电大学出版社
www.buptpress.com

内 容 简 介

FORTRAN 语言是世界上最早出现的高级语言,在科学计算、数值分析和工程设计等领域有着十分广泛的应用。从 1954 年第一个 FORTRAN 语言版本问世至今, FORTRAN 语言有了很大发展。FORTRAN 语言的发展使这门古老的语言焕发出新的活力。

本书介绍 FORTRAN 90 程序设计的基本方法,努力培养读者的程序设计能力。主要内容有: FORTRAN 程序设计基础、顺序结构程序设计、选择结构程序设计、循环结构程序设计、FORTRAN 90 数据类型、格式输入输出、数组、子程序、结构体、指针、文件、FORTRAN 90 图形操作。作者针对初学者的特点,力求做到内容丰富、取材新颖、循序渐进、注重实用、便于教学。

本书可作为高校理工科专业学生学习程序设计的教材,也可供软件开发人员参考。

为配合本书的学习,作者还编写了与本书配套的实验教材《FORTRAN 90 程序设计上机指导与习题选解》(第 2 版),还有与本书配套的教学课件,供教师教学参考。

图书在版编目(CIP)数据

FORTRAN 90 程序设计教程/刘卫国,蔡旭晖主编. —2 版, —北京:北京邮电大学出版社,2007
ISBN 978-7-5635-0659-0

I. F… II. ①刘…②蔡… 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(2007)第 011890 号

书 名 FORTRAN 90 程序设计教程(第 2 版)
主 编 刘卫国 蔡旭晖
责任编辑 陈露晓
出版发行 北京邮电大学出版社
社 址 北京市海淀区西土城路 10 号(100876)
电话传真 010-62282185(发行部) 010-62283578(传真)
电子信箱 ctrd@buptpress.com
经 销 各地新华书店
印 刷 北京忠信诚胶印厂
开 本 787mm×1 092mm 1/16
印 张 18.25
字 数 409 千字
版 次 2007 年 4 月第 2 版 2007 年 4 月第 1 次印刷

ISBN 978-7-5635-0659-0/TP·89

定 价:30.00 元

如有质量问题请与发行部联系
版权所有 侵权必究

21 世纪高等学校计算机科学与技术规划教材

编委会

主任 陈火旺 中国工程院院士,国防科技大学教授

委员 (以姓氏笔画为序)

文双春 湖南大学计算机与通信学院教授

王命延 南昌大学信息工程学院教授

卢正鼎 华中科技大学计算机科学与技术学院教授

宁 洪 国防科技大学计算机学院教授

刘爱民 北京大学信息科学技术学院教授

齐 勇 西安交通大学电子与信息工程学院教授

何炎祥 武汉大学计算机学院教授

李仁发 湖南大学计算机与通信学院教授

李志蜀 四川大学计算机学院教授

杨路明 中南大学信息科学与工程学院教授

杨学军 国防科技大学计算机学院教授

杨放春 北京邮电大学计算机科学与技术学院教授

陈志刚 中南大学信息科学与工程学院教授

周立柱 清华大学计算机科学与技术系教授

周兴社 西北工业大学计算机学院教授

周昌乐 厦门大学信息科学与技术学院教授

孟祥旭 山东大学计算机科学与技术学院教授

姜云飞 中山大学信息科学与技术学院教授

赵书城 兰州大学信息科学与工程学院教授

徐晓飞 哈尔滨工业大学计算机科学与技术学院教授

钱列加 复旦大学信息科学与工程学院教授

蒋昌俊 同济大学电子与信息工程学院教授

戴居丰 天津大学信息学院、软件学院教授

序

自 20 世纪 80 年代以来,高等学校计算机教育发展迅速,计算机教育的内容不断扩展、程度不断加深。特别是近十年来,计算机向高度集成化、网络化和多媒体化发展的速度一日千里;社会信息化不断向纵深发展,各行各业的信息化进程不断加速;计算机应用技术与其他专业的教学、科研工作的结合更加紧密;各学科与以计算机技术为核心的信息技术的融合,促进了计算机学科的发展,各专业对学生的计算机应用能力也有更高和更加具体的要求。

基于近年来计算机学科的发展,以及国家教育部关于计算机基础教学改革的思想,我们确立了这套“21 世纪高等学校计算机科学与技术规划教材”的编写思想与编写计划。教材是教学过程中的“一剧之本”,是高校计算机教学的首要问题。该套系列教材编写计划的制定凝聚了编委会和作者的心血,是大家多年来计算机学科教学和研究成果的体现,并得到了陈火旺院士的亲自指导与充分肯定。

这套系列教材经过了精心的策划和组织,同时在编写过程中,充分考虑了计算机学科的发展与《计算机学科教学计划》中内容和模块的调整,使得整套教材更具科学性和实用性。整套系列教材体系结构按课程设置进行划分。每册教材均涵盖了相应课程教学大纲所要求的内容,既具备学科设置的合理性,又符合计算机学科发展的需要。从结构上遵循教学认知规律,基本上能够满足不同层次院校、不同教学计划的要求。

各册教材的作者均为多年来从事教学、研究的专家和学者。他们有丰富的教学实践经验,所编写的教材结构严谨、内容充实、层次清晰、概念准确、论理充分、理论联系实际、深入浅出、通俗易懂。

教材建设是一项长期艰巨的系统工程,尤其是计算机科学技术发展迅速、内容更新快,为使教材更新能跟上科学技术的发展,我们将密切关注计算机科学技术的发展新动向,以使我们的教材编写在内容上不断推陈出新、体系上不断发展完善,以适应高校计算机教学的需要。

前 言

程序设计是计算机应用人员的一项基本功,也是大学生的一项基本思维方式训练。通过学习程序设计,可以编写出适合自己需要的程序,让计算机完成自己指定的任务,具体了解计算机是怎样进行工作的。只有掌握程序设计的知识,才能具有较强的计算机应用开发能力。

学习高级语言程序设计涉及两方面的问题,一是高级语言的语法规则,这是编程的基础,二是程序设计的基本方法,这是学语言的目的,也是学习高级语言程序设计的重点和难点。现在流行的高级语言很多,语种的选择要从具体情况出发。选择何种高级语言并不是大的原则问题,因为语言只是一种载体,一种学习程序设计的工具。高级语言的选择,应取决于语言的功能、专业的应用背景与工作岗位的需要等。不能简单地说哪一种语言“高级”,哪一种语言“低级”,只能说哪一种语言更适合于哪一个应用领域和哪一类学习对象。无论选用哪种语言,都应掌握程序设计的基础知识与基本编程技术,学习的重点应放在程序设计的基本方法与思路上。

FORTRAN 语言是世界上最早出现的高级语言,在科学计算、数值分析和工程设计等领域有着十分广泛的应用。从 1954 年第一个 FORTRAN 语言版本问世至今,已有半个多世纪。在这期间,FORTRAN 语言有了很大发展,形成了许多版本,如 FORTRAN 66、FORTRAN 77 等。1991 年经 ISO 和 ANSI 双重批准公布了新的 FORTRAN 国际标准 FORTRAN 90,它对 FORTRAN 77 主要扩充了自由的书写格式、模块化机制、派生类型、类型参数化、指针和递归等。而后,出现了基于 FORTRAN 90 标准的集成开发环境,如 Microsoft 公司的 Fortran PowerStation、Compaq 公司的 Visual FORTRAN 等。FORTRAN 语言的发展使这门古老的语言焕发出新的活力。

FORTRAN 90 是 FORTRAN 语言的较新版本,它是一个功能十分强大且体现现代程序设计语言特征的程序设计语言,在社会上有广泛影响。FORTRAN 90 程序设计课程也是高等院校学生学习程序设计的常用课程之一。通过该课程的学习,使学生了解和掌握计算机程序设计的基本知识、基本概念和基本方法,了解和掌握计算机求解问题的机制和方法,熟练掌握 FORTRAN 90 的基本内容、编程技能和开发环境,能熟练运用 FORTRAN 90 语言设计和编写比较复杂的 FORTRAN 90 程序,有效地解决本专业中遇到的实际问题,使学生计算机应用能力得到显著提高,为专业应用和发展奠定基础。

本书介绍 FORTRAN 90 程序设计的基本方法,努力培养读者的程序设计能力。作者针对初学者的特点,力求做到内容丰富、取材新颖、循序渐进、注重实用、便于教学。为了方便教学和读者上机操作练习,作者还编写了《FORTRAN 90 程序设计上机指导与习题选解》(第 2 版)一书,作为与本书配套的实验教材。另外,还有与本书配套的教学课件,

供教师教学参考。

本书可作为高校理工科专业学生学习程序设计的教材,也可供软件开发人员参考。

本书由刘卫国、蔡旭晖主编。第一、第四、第七章由刘卫国编写,第二、第三、第九、第十章由刘胤宏编写,第五章由范臻辉编写,第六章由刘建成编写,第八章由陈昭平编写,第十一、第十二章由蔡旭晖编写。参与部分编写与讨论的还有童键、戴忠、莫照、舒卫真、杨振宇等。全书由刘卫国教授审阅、定稿。

由于编者水平所限,书中不当或错误之处在所难免,敬请广大读者批评指正。

编 者
2007年3月

目 录

第一章 FORTRAN 程序设计基础	(1)
1.1 程序与程序设计	(1)
1.2 算法及其描述	(2)
1.2.1 算法的概念	(2)
1.2.2 算法的描述	(3)
1.3 程序设计方法	(7)
1.3.1 结构化程序设计	(7)
1.3.2 面向对象程序设计	(9)
1.4 FORTRAN 语言的发展	(10)
1.5 FORTRAN 程序的基本结构	(11)
1.6 FORTRAN 90 程序的书写格式	(13)
1.6.1 固定格式	(13)
1.6.2 自由格式	(13)
1.7 FORTRAN 90 开发环境	(14)
1.7.1 系统安装与启动	(14)
1.7.2 上机步骤	(14)
习题	(15)
第二章 顺序结构程序设计	(16)
2.1 FORTRAN 常量	(16)
2.1.1 直接常量	(16)
2.1.2 符号常量	(17)
2.2 FORTRAN 变量	(17)
2.2.1 变量的概念	(17)
2.2.2 变量名	(17)
2.2.3 变量的说明	(18)
2.3 FORTRAN 内部函数	(19)
2.4 FORTRAN 算术表达式	(20)
2.4.1 运算符与其优先级别	(21)
2.4.2 算术表达式中类型转化	(21)
2.5 赋值语句	(22)

2.5.1	赋值语句的格式	(22)
2.5.2	执行赋值语句时的类型转换问题	(22)
2.6	简单输入输出语句	(23)
2.6.1	表控输入语句	(23)
2.6.2	表控输出语句	(24)
2.7	程序执行控制语句	(26)
2.7.1	STOP 语句	(26)
2.7.2	PAUSE 语句	(26)
2.7.3	END 语句	(26)
2.8	程序举例	(27)
	习题	(29)
第三章	选择结构程序设计	(30)
3.1	关系运算和逻辑运算	(30)
3.1.1	关系运算	(30)
3.1.2	逻辑运算	(31)
3.2	块 IF 选择结构	(32)
3.3	块 CASE 选择结构	(36)
3.4	逻辑 IF 语句	(38)
3.5	算术 IF 语句	(39)
3.6	程序举例	(40)
	习题	(42)
第四章	循环结构程序设计	(45)
4.1	用 DO 语句实现循环	(45)
4.1.1	DO 循环一般格式	(45)
4.1.2	DO 循环执行过程	(46)
4.1.3	与循环有关的控制语句	(50)
4.2	用 DO WHILE 语句实现循环	(52)
4.2.1	DO WHILE 循环的一般格式	(52)
4.2.2	DO WHILE 循环的执行过程	(53)
4.3	几种循环组织方式的比较	(55)
4.4	循环的嵌套	(56)
4.5	程序举例	(59)
	习题	(64)
第五章	FORTRAN 90 数据类型	(67)
5.1	数据类型的概念	(67)
5.2	整型数据	(67)
5.3	实型数据	(69)

5.4 复型数据	(71)
5.4.1 复型常量	(71)
5.4.2 复型变量	(72)
5.4.3 复型数据应用举例	(72)
5.5 字符型数据	(76)
5.5.1 字符型常量	(76)
5.5.2 字符型变量	(77)
5.5.3 子字符串	(78)
5.5.4 字符表达式与字符赋值语句	(79)
5.5.5 字符关系表达式	(80)
5.5.6 用于字符处理的内部函数	(80)
5.5.7 字符处理应用举例	(82)
5.6 逻辑型数据	(85)
习题	(87)
第六章 格式输入输出	(89)
6.1 输入输出概述	(89)
6.2 格式输入输出语句	(90)
6.2.1 格式输出	(90)
6.2.2 格式输入	(91)
6.3 格式说明语句	(91)
6.4 常用的编辑描述符	(92)
6.4.1 可重复编辑描述符	(92)
6.4.2 非重复编辑描述符	(97)
6.4.3 纵向走纸控制符	(98)
6.5 输入输出项与编辑符的相互作用	(98)
习题	(99)
第七章 数组	(102)
7.1 数组的概念	(102)
7.2 数组的说明与引用	(103)
7.2.1 数组说明的内容	(103)
7.2.2 数组说明的方法	(104)
7.2.3 数组元素的引用	(105)
7.3 数组元素的存储结构	(106)
7.4 数组的输入输出	(107)
7.4.1 使用 DO 循环输入输出数组元素	(107)
7.4.2 以数组名作为输入输出项	(108)
7.4.3 使用隐 DO 循环进行数组元素的输入输出	(109)

7.5	给数组赋初值	(110)
7.5.1	使用 DATA 语句赋初值	(110)
7.5.2	使用数组赋值符赋初值	(110)
7.6	动态数组	(111)
7.7	数组的应用	(113)
7.7.1	一维数组的应用	(113)
7.7.2	二维数组的应用	(122)
	习题	(124)
第八章	子程序	(127)
8.1	语句函数	(127)
8.1.1	语句函数的定义	(128)
8.1.2	语句函数的调用	(129)
8.2	函数子程序	(130)
8.2.1	函数子程序的定义	(130)
8.2.2	函数子程序的调用	(131)
8.3	子例程序	(133)
8.3.1	子例程序的定义	(134)
8.3.2	子例程序的调用	(134)
8.4	程序单元之间的数据传递	(137)
8.4.1	简单变量作为虚参	(137)
8.4.2	数组名作为虚参	(137)
8.4.3	子程序名作为虚参	(139)
8.4.4	星号(*)作为虚参	(140)
8.4.5	变量的作用域	(141)
8.5	递归调用	(144)
8.5.1	递归的概念	(144)
8.5.2	递归函数	(144)
8.5.3	递归子例程序	(147)
8.6	数据共用存储单元与数据块子程序	(148)
8.6.1	等价语句	(148)
8.6.2	公用数据块	(150)
8.6.3	数据块子程序	(154)
8.7	内部子程序	(155)
8.8	模块	(157)
8.8.1	模块的建立	(158)
8.8.2	模块的调用	(159)
8.9	程序举例	(160)
	习题	(173)

第九章 派生类数据类型与结构体	(178)
9.1 派生类数据类型	(178)
9.2 结构体的定义与引用	(179)
9.2.1 结构体类型定义	(179)
9.2.2 结构体成员的引用	(180)
9.3 结构体的初始化	(180)
9.3.1 利用赋值语句给结构体成员赋值	(180)
9.3.2 定义的同时给结构体成员赋值	(182)
9.4 结构体数组	(184)
9.5 综合实例	(186)
9.5.1 排序	(187)
9.5.2 查找	(187)
9.5.3 插入	(189)
9.5.4 删除	(190)
9.5.5 主程序	(191)
习题	(192)
第十章 指针	(194)
10.1 指针的定义与使用	(194)
10.1.1 指针变量的定义格式	(194)
10.1.2 指针变量的使用	(194)
10.2 动态变量	(196)
10.3 指针与数组	(198)
10.4 指针与链表	(199)
10.4.1 结点的定义	(200)
10.4.2 链表的基本操作	(200)
10.5 综合实例	(204)
习题	(207)
第十一章 文件	(209)
11.1 文件的概念	(209)
11.1.1 文件与记录	(209)
11.1.2 文件的存取方式	(210)
11.1.3 文件的结构	(210)
11.1.4 文件的定位	(212)
11.2 文件的打开与关闭	(212)
11.2.1 OPEN 语句	(212)
11.2.2 CLOSE 语句	(216)
1.3 文件的读写	(217)

11.3.1	有格式顺序存取文件	(217)
11.3.2	有格式直接存取文件	(221)
11.3.3	无格式顺序存取文件	(223)
11.3.4	无格式直接存取文件	(224)
11.3.5	二进制顺序存取文件	(224)
11.3.6	二进制直接存取文件	(225)
11.3.7	内部文件	(225)
11.4	其他文件操作	(227)
11.4.1	文件的定位	(227)
11.4.2	PRINT 语句和 ENDFILE 语句	(229)
11.4.3	INQUIRE 语句和 EOF 语句	(229)
11.5	文件应用举例	(232)
	习题	(236)
第十二章	FORTRAN 90 图形操作	(237)
12.1	QuickWin 绘图基础	(237)
12.2	QuickWin 窗口的创建	(240)
12.2.1	访问窗口属性	(240)
12.2.2	窗口操作	(241)
12.3	图形坐标系	(244)
12.4	图形属性	(247)
12.4.1	图形颜色	(248)
12.4.2	线型	(249)
12.4.3	填充样式	(250)
12.5	基本图形函数	(251)
12.5.1	画像素点	(251)
12.5.2	设置绘图起点	(252)
12.5.3	绘制形状	(252)
12.6	图形字符操作	(257)
12.7	综合实例	(260)
	习题	(267)
附录 1	ASCII 字符编码一览表	(268)
附录 2	FORTRAN 90 内部函数一览表	(272)
	参考文献	(276)

第一章 FORTRAN 程序设计基础

计算机是在“程序”的控制下进行自动工作的,它解决任何实际问题都依赖于解决问题的程序。大多数人已取得这样一个共识,程序设计应是计算机应用人员的一项基本功,也是大学生的一项基本思维方式训练。只有掌握程序设计的知识,才能具有一定的应用开发能力。学习 FORTRAN 程序设计的目的,就是要学会利用 FORTRAN 语言编写出适合自己实际需要的程序,让计算机完成自己指定的任务。

本章将介绍程序设计的基本概念, FORTRAN 语言的发展, FORTRAN 程序的基本结构与书写格式, FORTRAN 90 开发环境等内容。通过这些内容的学习,将使读者对程序设计和 FORTRAN 语言有一个初步认识,从而为以后各章的学习打下必要的基础。

1.1 程序与程序设计

计算机解题的“程序”是用计算机能识别的语言所描述的解决实际问题的方法和步骤。计算机能直接识别的语言是机器语言,但机器语言用二进制代码表示机器指令,且机器指令跟具体的计算机结构有关,程序直观性差、通用性不强。所以初学者一般都学习利用一种高级语言来编写程序。FORTRAN 语言便是在科学计算领域应用十分广泛的一种高级语言。

什么叫程序设计?对于初学者来说,往往把程序设计简单地理解为只是编写一个程序。这是不全面的。程序设计反映了利用计算机解决问题的全过程,包含多方面的内容,而编写程序只是其中的一个方面。使用计算机解决实际问题,通常是先要对问题进行分析并建立数学模型,然后考虑数据的组织方式和算法,并用某一种程序设计语言编写程序,最后调试程序,使之运行后能产生预期的结果,这个过程称为程序设计。具体要经过以下 4 个基本步骤:

- (1) 分析问题,确定数学模型或方法。
- (2) 设计算法,画出流程图。
- (3) 选择编程工具,编写程序。
- (4) 调试程序,分析输出结果。

在拿到一个实际问题之后,应对问题的性质与要求进行深入分析,从而确定求解问题的数学模型或方法,接下来进行算法设计,并画出流程图。有了算法流程图,再来编写程序是很容易的事情。有些初学者,在没有把所要解决的问题分析清楚之前就急于编写程序,结果编程思路紊乱,很难得到预想的结果。

1.2 算法及其描述

1.2.1 算法的概念

在日常生活中,做任何一件事情,都是按照一定规则,一步一步地进行,这些解决问题的方法和步骤称为算法。比如工厂生产一部机器,先把零件按一道道工序进行加工,然后,把各种零件按一定法则组装起来,生产机器的工艺流程就是算法。

同样,要编写解决问题的程序,首先应设计算法,任何一个程序都依赖于特定的算法,有了算法,再来编写程序是很容易的事情。

下面举几个例子,说明计算机解题的算法。

【例 1.1】 求 $u = \frac{x-y}{x+y}$ 。

$$\text{其中 } x = \begin{cases} a^2 + b^2 & a < b \\ a^2 - b^2 & a \geq b \end{cases}, \quad y = \begin{cases} \frac{a+b}{a-b} & a < b \\ \frac{4}{a+b} & a \geq b \end{cases}。$$

这一题的算法并不难,可写成:

(1) 从键盘输入 a, b 的值。

(2) 如果 $a < b$, 则 $x = a^2 + b^2, y = \frac{a+b}{a-b}$, 否则 $x = a^2 - b^2, y = \frac{4}{a+b}$ 。

(3) 计算 u 的值。

(4) 输出 u 的值。

【例 1.2】 输入 10 个数,要求找出其中最大的数。

设 \max 单元用于存放最大数,先将输入的第一个数放在 \max 中,再将输入的第二个数与 \max 相比较,大者送到 \max ,然后将第三个数与 \max 相比,大者放在 \max 中……一直到比完 9 次为止。

算法要在计算机上实现,还需要把它描述为更适合程序设计的形式,对算法中的量要抽象化、符号化,对算法的实施过程要条理化。上述算法可写成如下形式:

(1) 输入一个数,放在存储单元 \max 中。

(2) 设置用来统计比较次数的单元 i ,初值置 1。

(3) 若 $i \leq 9$,做第(4)步,否则做第(8)步。

(4) 输入一个数,放在存储单元 x 中。

(5) 比较 \max 和 x 中的数,若 $x > \max$,则将 x 的值送给 \max ,否则, \max 值不变。

(6) i 增加 1。

(7) 返回到第(3)步。

(8) 输出 \max 中的数,此时 \max 中的数就是 10 个数中最大的数。

【例 1.3】 求两个正整数 m 和 n 的最大公约数。

m, n 的最大公约数即是所有能同时除尽 m, n 的数中的最大数。求两个正整数的最大公约数常用辗转相除法。由于 m 和 n 是两个正整数, 因此存在 q, r , 使 $m = n \times q + r (q \geq 0, r \geq 0, r < n)$ 。如果 $r = 0$, 则 n 即是 m 和 n 的最大公约数。否则, 可以证明 n 和 r 的最大公约数就是 m 和 n 的最大公约数。用同样的方法求 n 和 r 的最大公约数。如此继续下去, 直到余数为 0。

可将上述过程写成如下算法:

- (1) 输入 m 和 n 的值。
- (2) 求 m 除以 n 的余数 r 。
- (3) 若 $r = 0$, 则转至第(6)步, 否则执行第(4)步。
- (4) $n \rightarrow m, r \rightarrow n$ 。
- (5) 转第(2)步。
- (6) 输出 n 。

从上述算法示例可以看出, 算法是解决问题方法的精确描述。算法并不给出问题的精确解, 只是说明怎样才能得到解。每一个算法都是由一系列基本的操作组成的, 所以研究算法的目的就是要研究怎样把问题的求解过程分解成一些基本的操作。

从上面的例子中, 还可以概括出算法的 5 个特征:

- (1) 有穷性。算法中执行的步骤总是有限次数的, 不能无止境地执行下去。
- (2) 确定性。算法中的每一步操作必须具有确切的含义, 不能有二义性。
- (3) 有效性。算法中的每一步操作必须是可执行的。
- (4) 要有数据输入。算法中操作的对象是数据, 因此应提供有关数据。但如果算法本身给出了运算对象的初值, 也可以没有数据输入。
- (5) 要有结果输出。算法的目的是用来解决一个给定的问题, 因此应提供输出结果, 否则算法就没有实际意义。

1.2.2 算法的描述

算法的描述有许多方法, 常用的有自然语言、一般流程图、N-S 图等。前面例 1.1~例 1.3 的算法是用自然语言——汉语描述的, 其优点是通俗易懂, 但它不太直观, 描述不够简洁, 且容易产生二义性。在实际应用中常用流程图表示算法。

1. 一般流程图

一般流程图是一种传统的算法描述方法, 它用不同的几何图形来代表不同性质的操作。例如, 用矩形框表示要进行的操作, 用菱形框表示判断, 用流程线将各步操作连接起来并指示算法的执行方向。

【例 1.4】 用一般流程图来描述例 1.1~例 1.3 的算法。

流程图分别如图 1.1~图 1.3 所示。

一般流程图的主要优点是直观性强, 初学者容易掌握。缺点是对流程线的使用没有严格限制, 如毫无限制地使流程任意转来转去, 将使流程图变得毫无规律, 难以阅读。为了提高算法可读性和可维护性, 必须限制无规则的转移, 使算法结构规范化。

2. 程序的三种基本结构

1966 年 Bohm 和 Jacopini 提出了组成结构化算法的三种基本结构, 即顺序结构、选择结构和循环结构。顺序结构是最简单的一种基本结构, 依次顺序执行不同的程序块, 如图