

中小学生的理想计算机课外辅导读物 信息学(计算机)奥林匹克竞赛的最好教材

信息学[计算机]奥林匹克

提高篇下册

吕 品 编著 北京大学出版社

信息学
[计算机]
奥林匹克丛书



信息学（计算机）奥林匹克丛书

信息学（计算机）奥林匹克

（提高篇下册）

 品 编著

北京大学出版社
北 京

内 容 提 要

本书是《信息学(计算机)奥林匹克系列丛书》提高篇的下册。本书由浅入深、系统地介绍了信息学(计算机)科学核心内容之一——算法的一些基本知识和人工智能的一些初步知识。主要内容有结构化程序设计、数据类型、基本数据结构、常用算法策略、递归、无信息搜索、启发式搜索、动态规划等。全书共分十一章,每章配有相应的习题。

本书可作为各地信息学(计算机)奥林匹克学校、计算机课外小组的教材。适用对象是对信息学(计算机)奥林匹克感兴趣的初、高中学生。本书理论联系实际,有一定理论深度但不脱离学生实际,有较多实例帮助理解,分析叙述也较详尽,并有充足的习题加深理解和巩固,因此也适合广大青少年自学使用。

图书在版编目(CIP)数据

信息学(计算机)奥林匹克:提高篇 下册/吕品编著. 北京:北京大学出版社,1997

(信息学(计算机)奥林匹克丛书)

ISBN 7-301-03083-5

I. 信… II. 吕… III. 计算机课-中学-课外读物 IV. G 634.674

中国版本图书馆CIP数据核字(97)第14399号

书 名: 信息学(计算机)奥林匹克(提高篇下册)

著作责任者: 吕 品

责任编辑: 郭佑民

标准书号: ISBN 7-301-03083-5/TP·291

出 版 者: 北京大学出版社

地 址: 北京市海淀区中关村北京大学校内 100871

电 话: 出版部 62752015 发行部 62559712 编辑部 62752032

排 印 者: 中国科学院印刷厂

发 行 者: 北京大学出版社

经 销 者: 新华书店

787×1092 16开本 14.75印张 370千字

1997年8月第一版 1997年8月第一次印刷

定 价: 24.00元

序 言

科学技术普及工作是传播科学知识、科学思想和科学方法，提高全民科学文化素质的关键措施，是实施“科教兴国”战略，加速科学技术进步，促进社会主义物质文明和精神文明建设的重要内容。在科技竞争、人才竞争已成为世界各国竞争焦点的今天，科学技术的普及活动也已成为衡量综合国力的一个重要尺度。从培养 21 世纪人才着眼，抓好青少年科技教育，是一件十分有意义的事情，这也是我们编写这套丛书的初衷。

众所周知，现代信息技术正在对人类社会产生着难以估量的深远影响，并将成为新世纪的一个标志。将有关信息科学知识纳入到未来人才的知识结构中来，是时代的需要。人们越来越认识到：计算，跟语言一样是人类社会每时每刻都不可缺少的事情；计算已经成为与理论研究和实验研究并列的第三种研究方法；计算机成了“人类通用智力工具”；计算机和现代通信网络结合成了教育面向世界、面向未来、面向现代化的支撑环境；信息技术与基础教育相结合已经成为当今世界的大趋势。国家教委副主任柳斌同志在论述“为什么要重视计算机教育”时说：“经验证明，计算机技术越是高度发展，计算机人才就越重要，计算机教育就越重要。只有培养一批又一批掌握现代化已经成熟的电子计算机技术人才，并不断发展和提高我国的计算机技术水平，我们才能加速我国走向现代化，走向世界，走向未来的历史进程。”

信息学奥林匹克竞赛最根本的一条宗旨是推动计算机在青少年中的普及。从 1989 年到 1995 年已经成功地举行过七届世界大赛。在这七届大赛中，我国的青少年选手取得了名列前茅的好成绩。这表明，中国孩子有志向、有能力、有条件掌握先进的信息科学知识，具备成为合格的跨世纪人才的素质条件。学科奥林匹克竞赛对青少年树立自立、自信、自尊、自强精神，起到了激励和鼓舞作用。

信息学奥林匹克竞赛属于课外因材施教活动，重在培养能力，特别是创造能力。按照教育心理学的观点，创造能力的培养必须以实践为基础。在计算机上编程解题是一项极好的实践活动，它可以训练观察能力、逻辑思维能力、抽象化形式化描述问题能力、规划能力、动手动脑分析问题和解决问题的能力。总之，这是一项用电脑帮助开发人脑的有益活动。正因为如此，热心于普及教育的老师们集体编著了这套丛书。这套丛书按不同年龄段设计了不同的内容，分层次由浅入深，介绍计算机的典型算法和基本数据结构知识，重点讲解编程解题的思路与技巧，有丰富的例题和习题，许多都是能够引发兴趣的题目，有的有相当难度。这里需要强调：计算机是实践性很强的学科，不上机是学不会的。阅读丛书应和上机实践紧密配合才能真正学懂，学到手。另外，书上给出的方法也只是为了抛砖引玉。我们鼓励创造，尤其希望读者编出的程序更有效、更高明。“精心育桃李，切望青胜蓝”是我们的座右铭。

中国计算机学会普及委员会主任
国际信息学奥林匹克中国队总教练

吴文虎

1995 年 12 月于清华园

前 言

当今时代是一个迅速向信息化迈进的时代。信息在人们社会生活中越来越重要，已成为继物质和能源之后，第三大社会资源。掌握信息学科学的基本知识也已成为 21 世纪人才的基本素质之一。因此广大青少年学生，怀着渴求知识的强烈愿望，要求学习有关的知识。为满足广大学生的迫切要求，我们曾在 90 年代初，编写和出版了《信息学（计算机）奥林匹克》丛书。该书一出版，就获得了广大师生的欢迎，常供不应求。几年来在普及信息学知识，提高学生水平中作出了一定的贡献。由于当时编写仓促，书中存在不少缺点和错误，并且近年来形势也在发展。为了更好地满足广大师生的需求，适应新的形势，重新编写了这套丛书。

这套书继承了原书的宗旨：面向广大青少年学生，介绍信息学科学的入门知识，力求通俗，便于理解。但又根据国际、国内信息学（计算机）奥林匹克新的发展，适当地扩展了一些新的内容和知识。

本书是这套书的提高篇。对象是具有程序设计初步知识的初、高中学生。本书不是全面论述程序设计及算法的专著，而仅仅是在有关方面引导广大青少年计算机爱好者进入计算机科学殿堂的大门。更重要的是通过学习，提高学生的各种能力，特别是分析问题和解决问题的能力。因此，本书有以下几个特点。

PASCAL 语言的创始者 N·沃思 (Niklaus Wirth) 曾说过一个公式：“算法 + 数据结构 = 程序”。这个公式形象地、精辟地概述了这三者的关系。当然随着程序设计的理论和实践的发展，人们发现要设计出一个好的程序，除了算法和数据结构这两个关键因素外，还要有好的程序设计方法。因此目前很多人认为，沃思的公式应发展为：

$$\text{算法} + \text{数据结构} + \text{程序设计方法} = \text{程序}$$

本书就是以这一思想为依据来编写的，即希望读者通过本书的学习，能在算法、数据结构和程序设计方法上有所收获，有所提高。

语言是程序的载体，相同的程序可以用不同的语言来实现。单就本书的内容而言，我们认为以 PASCAL 语言作为载体较合适。本书中的程序是以目前社会较流行的 Turbo PASCAL 6.0 环境为基础编写和调试的。为照顾多数，本书是从没学过 PASCAL 语言的基础上起步的，因此，只要具有一些简单的程序设计知识（如 LOGO、BASIC 等）的学生，就可以学习本书。

从感性上升到理性，再从理性回到实践，是人认识世界的一般规律。本书的编写遵循这一规律，大部分章节的写法是，从实际问题出发，通过解决实际问题引入一般的方法，提高到理论的高度，然后再以理论为指导，举一反三，解决其他的实际问题。

本书是在作者多年的讲稿基础上编写而成的。多年的经验告诉我们，语言、数据结构和算法的学习三者是密不可分的，不可能截然分开学习。因此本书在基本按语言、数据结构、算法、方法的顺序学习的基础上，采取了交叉学习的方法，在讲授语言的同时还学习数据结构和算法的内容，在讲解算法的同时也学习程序设计方法等。

本书分为十一章。第一章对程序设计作概要介绍，其中有些内容较深，对于初学程序的

同学来说可能难以理解。因此对于初学者这些内容可以先跳过，仅学习第一章的 3, 4 两节。学习到一定深度后，再回头学习这部分内容，就会容易理解，会有更深的体会。第二、三章主要是 PASCAL 语言的入门知识，同时还有简单的数据结构和结构化程序设计的基本知识，如三种程序结构、过程、基本数据类型等。第四章内容主要是数据结构，如线性表（栈、队列、链表）、树、图、记录、指针等知识。但也有一些算法。第五、六章主要介绍常用的算法。第七至十一章的内容为人工智能中的一些基本算法，如无信息搜索、启发式搜索、动态规划等。

本书中很多内容，是作者多次聆听了吴文虎教授的亲自讲授和教诲后写成的。可以说没有吴教授的指导和帮助，就没有这本书。因此在此对吴文虎教授表示衷心的感谢。

本书编写过程中也得到了同仁们的热情帮助，特别是北京十二中庄燕文老师、北京 161 中俞咪华老师和北京西城教研中心滕秋莉老师的大力协助，借此对他们表示深切的谢意。

由于本人水平和能力有限，时间又很仓促，书中定有不当和错误之处，恳请读者批评指正。

作 者

1997.5

目 录

第一章 概述	(1)
1.1 计算机语言及其发展	(1)
1.2 计算机语言的分类	(3)
1.3 程序设计	(5)
1.4 结构化流程图	(7)
1.5 程序设计方法的发展	(8)
习题一	(10)
第二章 结构化程序设计初步	(12)
2.1 程序基本结构.....	(12)
2.2 算术运算.....	(14)
2.3 编程环境.....	(17)
2.4 循环结构.....	(21)
2.5 选择结构.....	(26)
2.6 字符处理.....	(28)
2.7 文件操作.....	(31)
2.8 数组及其应用.....	(33)
习题二	(35)
第三章 子程序	(40)
3.1 过程.....	(40)
3.2 带参过程.....	(43)
3.3 自定义函数.....	(45)
3.4 全程变量和局部变量.....	(47)
3.5 值参数和变量参数.....	(49)
习题三	(51)
第四章 数据结构	(53)
4.1 子界和集合类型.....	(53)
4.2 记录.....	(55)
4.3 指针.....	(62)
4.4 堆栈.....	(65)
4.5 队列.....	(71)
4.6 链表.....	(77)
4.7 树.....	(81)
4.8 图.....	(84)
习题四	(88)

第五章 递归	(89)
5.1 递归概念	(89)
5.2 几个典型例子	(94)
5.3 消除递归	(106)
习题五	(109)
第六章 常用策略	(112)
6.1 筛选策略	(112)
6.2 归纳策略	(118)
6.3 分治策略	(123)
6.4 最优策略	(126)
6.5 穷举策略	(130)
6.6 回溯策略	(134)
6.7 产生式系统	(136)
习题六	(141)
第七章 深度优先搜索	(143)
7.1 引 入	(143)
7.2 深度优先基本算法 (一)	(146)
7.3 深度优先搜索实例	(148)
7.4 深度优先搜索基本算法 (二)	(157)
7.5 小 结	(159)
习题七	(161)
第八章 广度优先搜索法	(165)
8.1 基本算法	(165)
8.2 应 用	(173)
8.3 小 结	(178)
习题八	(179)
第九章 启发式搜索	(181)
9.1 启发函数	(181)
9.2 A 算法	(186)
9.3 A* 算法	(193)
习题九	(194)
第十章 分枝定界	(195)
10.1 旅行问题	(195)
10.2 任务安排问题	(201)
10.3 任务的最佳排序	(206)
习题十	(210)
第十一章 动态规划	(212)
11.1 最佳原理	(212)
11.2 生产规划	(216)

11.3 0/1 背包问题	(219)
11.4 最短航线	(221)
习题十一	(223)

第一章 概 述

1.1 计算机语言及其发展

一、什么是计算机语言

计算机是按照人们的命令工作的。人们把自己的思想、要做的事告诉别人，要用人與人交流信息的工具，这个工具就是语言。同样，人们要把需要做的事告诉计算机，也要用“语言”，但必须是计算机能理解、能执行的语言，我们称之为计算机语言。而人类之间交流思想的语言称之为自然语言。

目前计算机技术还远远达不到人的大脑水平，即能理解一些模糊的、多义的语言程度。例如“请您把头抬高些！”这句话就是一种模糊的说法，因为抬高多高？没有明确的规定。这种情况在自然语言中，特别是在口语中经常出现，但在计算机语言中是不允许的。也就是说，计算机语言必须是无二义性的、无感情色彩的、语法语义完全确定的语言。

为了避免二义性和不确定性，也为了便于计算机处理，目前计算机语言一般都是抽象符号形式的语言，简称形式化语言。例如在数学问题中“方程 $ax=b$ ($b \neq 0$) 当 $a=0$ 时方程无解，当 $a \neq 0$ 时方程的解为 $x=b/a$ 。”这一段话，用计算机语言（以 BASIC 语言为例）写出是：

```
IF a=0 THEN PRINT “无解” ELSE x=b/a
```

计算机语言是一种形式化语言，但并不是所有形式化的语言都可以作为计算机的语言。计算机语言与其他形式化语言（如数学语言、格式说明语言等）最重要的一点区别是，它可以直接地或间接地在现有计算机上有效地执行。

二、计算机语言的发展史

随着计算机技术的发展，计算机语言也经历了不断发展的过程。

1. 机器语言

最初，计算机操作人员是直接用二进制代码表示的机器指令来编写程序的。例如，在某一台计算机上，做加法 $32+72$ 的指令是：

```
10110000      00100000
00000100      01001000
```

这种二进制代码形式的指令是计算机能直接执行的。

这种计算机唯一能直接执行的二进制代码就叫做机器语言。机器语言是面向具体机器的，不同的计算机就有不同的机器指令，也就是有不同的机器语言，相互之间一般没有通用性。而且机器语言很不直观，难懂、难记也难读，容易出错。

2. 汇编语言

为了克服机器语言的缺点，人们用英文缩写的助记符来代替机器指令。例如上面的二进

制代码用助记符可以写成：

MOV AL, 20 (取十六进制数 20 存入累加器 AL)

ADD AL, 48 (把累加器 AL 中的数和十六进制数 48 相加)

其中 MOV 是英语 MOVment 的缩写, ADD 就是英语加法。这种用符号来表示机器指令的语言就叫做汇编语言。汇编语言比机器语言容易记, 也容易读, 容易编程和修改。

但是计算机是不能直接执行汇编语言的, 必须翻译成机器语言后, 才能由计算机执行。这个翻译工作是由计算机的程序自动完成的。把汇编语言写成的程序(叫做源程序)翻译成机器语言程序(称为目标程序)的过程叫做“汇编”, 完成这项工作的计算机程序就叫做“汇编程序”。

使用汇编语言虽然可以比机器语言提高工作效率, 但是它与机器语言是一一对应的, 一条汇编语言对应于一条机器语言(目前宏汇编语言可以一条对应几条机器语言), 仍是面向具体机器的, 不同的机器有不同的汇编语言, 没有通用性。另外它与我们常用的自然语言和数学语言相距很大, 用它来编制程序仍是一项工作量大, 比较困难的工作。

机器语言和汇编语言都是面向具体机器的语言, 因此称它们为低级语言。

3. 高级语言

高级语言是完全独立于具体机器的语言, 它定义了一些语义与英语接近的词和语法, 可以和自然语言一样对处理过程进行描述, 和数学语言一样来表示计算公式。例如前面举例的加法, 用高级语言表示(BASIC)是:

PRINT 32+72

其中 PRINT 就是英语“打印输出”, 算式和数学中的完全相同。

但高级语言和汇编语言一样, 计算机是不能直接执行的, 必须先用高级语言写成“源程序”, 然后翻译成机器语言程序(目标程序)才能由计算机执行。把高级语言翻译成机器语言, 一般有编译和解释两种方式, 完成这两种工作的计算机程序分别称为编译程序和解释程序。

编译方式是将用高级语言编制的源程序输入计算机后, 编译程序一次全部将源程序翻译成机器语言的目标程序, 然后再逐条执行目标程序。编译程序在进行翻译时, 一般要经过词法分析、语法分析、代码生成和优化、生成目标程序等几个过程, 因此编译程序一般比较大, 占内存较多。如 C 语言、PASCAL 语言等都属于编译方式。

解释方式是将高级语言编写的程序输入计算机后, 由解释程序对逐条语句加工解释, 并立即执行这条语句。如果某一语句发现错误, 解释程序就给出错误信息, 并暂停执行, 改正错误后再执行。一般微机上的 BASIC 语言、LOGO 语言是解释方式。

解释方式并不把源程序翻译成可执行的程序, 而是依照程序中的语句和控制直接进行对数据的处理和加工。因此解释方式有较大的灵活性, 占内存也较小。但由于一些重复执行的语句(如循环体中的语句)每次都要重复进行解释, 一般执行效率比较低, 运行速度较慢。

两种翻译方式的区别可以用下面的示意图(图 1.1)来表示。

由于高级语言容易学, 编程效率也比较高, 因此获得了较大的发展。目前高级语言有几百种之多, 世界上大部分计算机程序是用高级语言编写的。

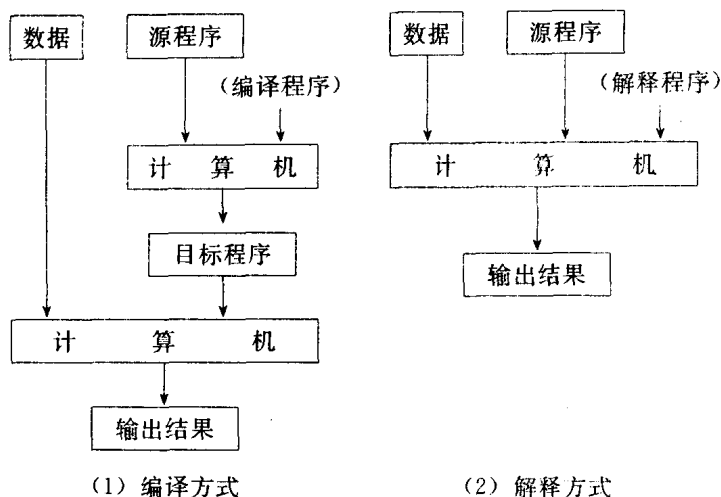


图 1.1

1.2 计算机语言的分类

计算机语言分为命令语言、查询语言、程序设计语言和定义语言等几种。

命令语言定义了一些基本的操作命令，人们可以输入这些命令，让计算机执行某些操作，但它一般不提供将这些命令组合起来的手段。例如 DOS 操作系统的命令就是一种命令语言。

查询语言是用来存取数据库中信息的一类语言。例如数据库管理系统 FoxBASE 就是一种查询语言。

程序设计语言是用于书写计算机可以执行的程序的语言。程序设计语言是计算机语言的主要组成部分。计算机上运行的各种操作系统、应用软件等都是用程序设计语言编写成的。

自计算机诞生以来的 50 多年中，程序设计语言得到了很大的发展，到目前已有上千种之多，常用的也有一二十种，形成一个很大的家族。

对于程序设计语言可以从许多不同的角度加以分类。主要有：

(1) 依照语言的应用范围，可分为通用语言（支持各种领域中的程序开发）和专用语言（仅用于某一专门领域）。

(2) 依照与计算机的相关程度，可以分为低级语言（与机器密切相关的，如机器语言和汇编语言）、高级语言（独立于具体计算机的，如 BASIC，PASCAL）。

(3) 依照描述的程序中同时存在的控制流（进程）数目，可分为串行语言、并行语言和并发语言等。

(4) 依照描述计算的基本方式，可分为过程式语言、函数式语言、逻辑式语言、面向对象式语言、数据流语言等等。

下面介绍一些常用的高级程序设计语言。

一、LOGO 语言

由美国麻省理工学院的教授西蒙·佩伯特（Seymour Papert）60 年代根据儿童的心理特点研究成功的一种程序设计语言。这种语言的特点是采用“海龟”形式进行绘图，既直观有

趣又容易学，特别适合少年儿童学习。

二、BASIC 语言

BASIC 是英语 Beginner's All-purpose Symbolic Instruction Code 的缩写，中文含义是“初学者通用符号指令代码”。是微型计算机常用语言之一。设计它的目的正如它的名称所表示的，是面向初学者的，语法比较简单，有较多的人机对话，因而简单易学，便于修改和调试，特别适合非计算机专业的工作者编写自己的应用程序。

随着计算机技术的发展，BASIC 语言也在不断地发展、改进。目前的 Visual BASIC 其功能之强大，已可以和其他高级语言媲美。而且它的大部分的屏幕界面可以直接进行“可视性”的设计，即用系统提供的绘画工具，直接在屏幕上作图，系统自动地转换为程序，节省了编程人员大量的时间和精力。并且还提供了编译的方式。它是编制 Windows 环境的应用程序良好的工具。

三、PASCAL 语言

它是由瑞士沃思教授 (Niklaus Wirth) 在 70 年代初研制出来的，目的是为教学和结构化程序而设计的。由于它符合结构化程序设计的原则，有丰富的数据类型和各种控制结构，适用于计算机专业的语言教学，也适用于编写系统软件。

四、FORTRAN 语言

它是研制较早 (1957 年) 的一种高级语言。它比较适用于数值计算和工程计算，因此在一些科学研究机构和工程设计单位比较流行。

五、C 语言

它是由美国贝尔实验室开发的高级编程语言。它是高级语言，但也能在低的级别和汇编语言级别上操纵计算机，因此更适用于系统软件的开发。

六、FORTH 语言

它是为小型计算机设计的高级语言。它是一种多级语言体系，可在机器语言、汇编语言和高级语言下编制程序，因此有人称它为中级语言。它提供了便于用户扩充功能的手段，具有运行速度快、所需内存少的特点，在军事和实时控制中应用较多。

七、COBOL 语言

它是一种高级商业用编程语言，在国外商业界应用较广。但由于用它书写的程序较长，并使用了很多的英语单词，所以在中国应用不多。

八、LISP 语言

它是 60 年代发明的一种人工智能语言。它是第一个突破冯·诺依曼机制，以符号为处理对象，以函数递归为基本工具的语言，因此也称为表处理语言。目前已成为主要的人工智能通用程序设计语言。

九、PROLOG 语言

它是 1972 年发明的一种逻辑程序设计语言。它突破了传统程序设计概念，形成了一种新的逻辑程序设计的风格。它是以符号为处理对象的描述性语言。目前在人工智能界，PROLOG 语言已成为与 LISP 语言齐名的通用人工智能程序设计语言，几乎所有计算机都有不同版本的 PROLOG 语言在运行。

十、Smalltalk 语言

它是用面向对象的思想 and 概念设计的计算机语言，是第一个典型的纯面向对象的语言（环境）。1971 年开始研制，1980 年完成最后版本。它的问世开创了一种新的程序设计风格——面向对象的程序设计。它也影响到其他语言的发展，例如 TURBO PASCAL 6.0 和 C++ 等语言版本都增加了面向对象的程序设计的功能和环境。

十一、C++ 语言

它是 1983 年由贝尔实验室开发，用于管理、设计和维护大型软件系列的高级语言。它在 C 语言的基础上增加了多种功能，特别是面向对象的程序设计功能。它是一种目前最常用的面向对象的编程语言。

十二、DELPHI 语言

它是在 IBM 兼容个人计算机上运行的面向对象的程序设计语言。它在 Microsoft Windows 下运行并具有类似于 PASCAL 的语法，使程序员可以容易地从传统语言转到面向对象的语言。它是“可视性”语言的一种，一些界面的设计、图形等用户可以不编程，只需直接在窗口中进行直观的设计，由系统自动地翻译成相应程序。它特别适宜于编制 Windows 环境下的应用程序。

1.3 程序设计

一、什么是程序设计

程序从形式上看，是一系列指令，或叫做命令、语句。但它又不是任意组合的命令，一个特定的程序能完成一定的任务或解决某个问题。所以程序是：

能完成某项任务的一系列指令就是程序。

什么是程序设计呢？我们要编制一个程序，就要首先分析问题，找出解题方法，然后写出程序，最后上机调试通过。因此，程序设计就是设计、书写和调试程序的过程。

二、为什么要进行程序设计

我们日常工作学习和生活中，要做某件事，如果事先毫无计划，只是想一步做一步，做完一步再想下一步该做什么，是绝对做不好工作的。要比较好地、高效率地完成某件事，必须事先有一个计划，考虑好第一步做什么，下一步做什么，最后一步做什么。即先考虑好做这件事的所有步骤，然后按部就班地去完成它。在计算机上解决某个问题也是一样，也要事

先制定一个解题计划，这就是程序设计。

另外，目前计算机还不会像人那样地进行思考，没有人那样的理解力。因此不能像告诉人那样，简单地“告诉”计算机要完成任务的内容和原则，而是我们要为计算机设计好解决问题的具体方法，然后把这个方法分解成计算机能执行的具体步骤和指令，再交给计算机去完成。这个过程也就是程序设计。

三、怎样进行程序设计

一个完整的程序设计过程一般要经过以下 5 个步骤。

1. 提出问题

首先要明确问题的已知条件、原始数据以及问题要达到的要求，即程序要完成的功能，最后结果的内容形式和精度等。也就是说要明确程序的输入、输出和要处理的内容。

2. 建立数学模型

有的问题的原始形式不适合计算机直接处理，就要对问题加以简化、转换，把问题用计算机便于处理的数据及其相应的结构来表示；有的问题还要选择合适的运算公式。这些工作我们叫做建立数学模型。

3. 设计算法

算法通俗地说就是解决问题的方法、步骤，确定算法就是找到一个合适的、能在计算机上实现的解题方法与步骤。

算法确定后，就要用一种工具把它精确地、清楚地表示出来。常用的表示算法的工具具有：自然语言、流程图、程序设计语言等。本书将用 N-S 图来表示算法。另外，算法中有的步骤还比较粗糙，还要对它们进行细化，使这些步骤能用程序设计语言编制出来。

4. 编写程序

根据问题的性质，选择一种合适的计算机语言，把算法转换成计算机能执行的程序。

5. 上机调试

把编好的程序输入计算机，试运行，改正其中的错误，最后得到正确的结果。一般的程序要多次使用，所以还要存储起来，以便今后应用。

四、结构化程序设计原则

在计算机发展的初期，由于计算机的运算速度不快，内存容量也不大，因此评判一个好程序的标准是长度短（占内存少），执行效率高（速度快）。但 70 年代以来计算机已发展到大规模集成电路时代，速度和内存量已不是主要问题，而由于程序越来越大，程序的可靠性、可读性和可维护性成了主要矛盾。在这种情形下，以德克斯特拉（Dijkstra）和 C. A. R. 霍尔（Hoare）为首的一批科学家提出了新的评价程序好坏的标准，即结构化程序设计的思想，很快得到了大家的公认。这个思想或原则，简单地说有以下几点：

1. 自顶向下，逐步求精

进行程序设计时，先把对问题的处理（即算法）粗分成几大步骤，叫做模块。然后对每个模块再细化为几个小模块，每个小模块再细分为若干个更小的子模块，……，这样细化下去，直到每个小模块能直接用语言来实现为止。这种逐步细化的过程是模块化的，使程序结构清晰，易于保证程序的正确性，也便于修改和维护。

2. 模块化

整个程序和大模块，都由几个功能相对独立的模块组成，每个模块只有一个入口和一个出口与外界联系。这可以使程序结构清楚，便于调试和修改。

3. 三种基本结构

整个程序中的大小模块仅由顺序、分支、循环三种基本控制结构组成，这三种基本结构也只有一个入口和一个出口（见图 1.2）。

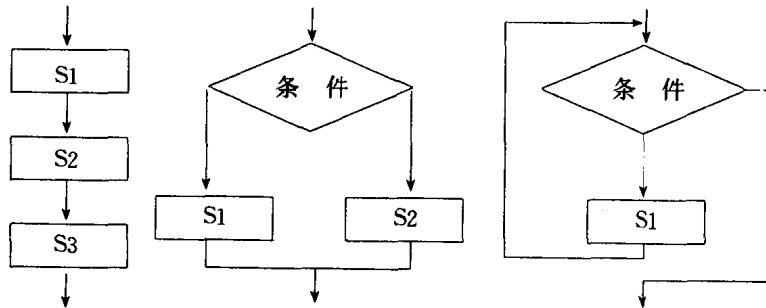


图 1.2

1.4 结构化流程图

算法设计是程序设计的核心，而流程图是一种很好的描述算法的工具，较早使用的流程图一般是用一些几何图形表示不同的操作，用流线表示程序流向。但这种流程图由于流线可以任意使用，程序的结构很容易被破坏。美国学者耐西（I. Nassi）和史内特曼（B. Shneiderman）1973 年提出了一种新的结构化流程图，它完全去掉了流线，根据这样的流程图表示的算法编出的程序能保证结构化，有利于程序的易读、易调试和正确性。因此受到计算机界的欢迎。这种流程图一般称为 N-S 图，也称为盒图。

一、顺序结构

N-S 图的顺序结构是一个个相邻的长方形，每个长方形内简明地写出操作内容，见图 1.3。

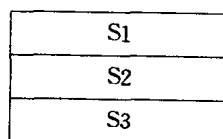


图 1.3 顺序结构

二、选择结构

选择结构的 N-S 图如图 1.4 所示。它表示当条件成立时（即真，用 T 表示），执行操作 S1，如条件不成立（即假，用 F 表示）则执行操作 S2。

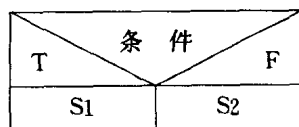


图 1.4 分支结构

三、循环结构

循环结构的 FOR 循环 N-S 图，见图 1.5。用“厂”形框表示循环的条件和范围，“厂”形框下面的矩形框表示循环体，即重复执行的操作。

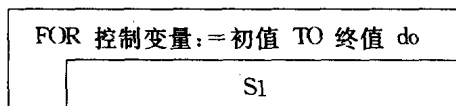


图 1.5 循环结构

【例 1.1】 输入 10 个数，输出它们之中的最大数。

分析：用一个变量 max 存最大数，开始存入计算机可表示的最小数。然后用一个循环读入 10 个数，每读入一个数，就和变量 max 中的值比较，如果更大，则用这个数代替变量中原来的数。最后输出变量 max 中的值，就是 10 个数中最大的数。

流程图如下（图 1.6）：

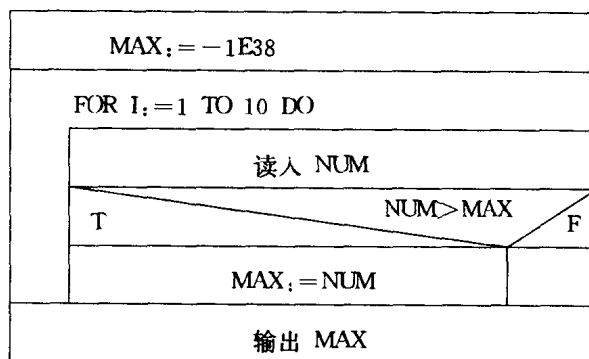


图 1.6

1.5 程序设计方法的发展

创立程序设计语言的目的是为了更好地进行计算机程序的设计。语言是工具，是载体，程序设计是核心。不同的语言可以用同一种程序设计方法和思路进行设计，解决同一个问题。而对于同一个问题，虽然用同一种语言，由于采用的思考方法不同，可以编制出风格完全不同的程序来。因此语言和程序设计是两种不同的概念。但是，一些语言是在某种程序设计的方法指导下设计出来的，显然这种语言用相应的程序设计思路来编程序会更适合，编程效率会更高。

编程时所采用的思考方法和设计思想，我们称之为程序设计范式。程序设计范式随着计算机诞生和发展，应用领域的日益扩大，也在不断地变化和发展。其发展大约经历了四个阶