

GAODENGXUEXIAO JISUANJISHU JICHU JIAOYU CONGSHU

高等学校计算机基础教育丛书

谭浩强 主编

软件技术基础 实用教程

RUANJIANJISHUJICHUSHIYONGJIAOCHENG



史济民 史令 李建平 编著

人民邮电出版社

图书在版编目(CIP)数据

软件技术基础实用教程/史济民等编著. —北京:人民邮电出版社, 1998. 3

(高等学校计算机基础教育丛书/谭浩强主编)

ISBN 7-115-06780-5

I. 软… II. 史… III. 软件工程-教材 IV. TP311.5

中国版本图书馆 CIP 数据核字(98)第 00458 号

内 容 简 介

在微机环境下开发应用软件,一要有科学的方法,二要有适当的平台。本书集“软件开发技术”与“微机操作环境”于一书,由算法与数据结构,软件开发技术的演变,结构化分析与设计,软件测试与质量保证,用户界面,环境、平台与操作系统, DOS 平台, Windows 平台等八章组成。全书复盖了算法、数据结构、软件工程、操作系统以及 PC 机上最流行的 DOS、UCDOS 与 Windows 平台等基础知识,是一本兼有工具书作用的教材和自学书。由于采取了先综述、后展开的结构安排,层次清楚,叙述简明,凡学过一门高级语言的读者均可看懂。

本书适用于普通高校理工科非计算机专业和师范、教育院校计算机专业的学生用作《软件开发技术》、《软件技术基础》、《微机环境与工具》等课程的教材,也可供从事大、中学校计算机基础教育的教师,初、中级计算机应用人员在教学与工作中作参考。

高等学校计算机基础教育丛书

软件技术基础实用教程

Ruanjian Jishu Jichu Shiyong Jiaocheng

◆主 编 谭浩强

◆编 著 史济民 史 令 李建平

◆责任编辑 赵桂珍

◆ 人民邮电出版社出版发行 北京崇文区夕照寺街 14 号
北京顺义振华印刷厂印刷
新华书店总店北京发行所经销

◆ 开本: 787×1092 1/16

印张: 22.5

字数: 560 千字

1998 年 5 月第 1 版

印数: 1—3 000 册

1998 年 5 月北京第 1 次印刷

ISBN7-115-06780-5/TN · 560

定价: 29.00 元

《高等学校计算机基础教育丛书》序

当前我们面临着计算机普及的又一次高潮。这一次普及的特点是向一切领域、各个层次的人群普及,也就是全方位、多层次的普及。现在,计算机是作为文化来普及,作为工具来使用的。这一高潮的到来不是偶然的,它是我国计算机应用发展的必然产物。实践证明,只有大力发展计算机应用,才能促进计算机产业的发展。只有把计算机技术与各行各业的各个专业领域结合起来,才能打开计算机应用的局面,才能从根本上推动计算机产业的发展。

计算机的应用是分层次的,计算机的教育也是分层次的。人们可以按以下几个层次循序渐进地学习计算机知识。第一个层次是:计算机的初步知识和计算机的操作使用(包括 DOS 操作命令、汉字输入、字表处理、数据库的简单操作等)。这是最初级的层次。第二个层次是:高级语言程序设计。达到这一层次要求的人员应该能用一种高级语言(如 BASIC、FORTRAN、PASCAL、C)或一种数据库语言(如 dBASE 或 FoxBASE)熟练地编写程序,而程序设计是计算机应用人员的基本功,是计算机基础教育的基础和重点。第三个层次是:进一步学习计算机软、硬件知识。达到这一层次要求的人员应具有开发应用软件或硬件系统的初步开发能力。第四个层次是结合本专业领域对计算机应用的需要进行学习。

国家教委工科计算机基础课程教学指导委员会据此制定了五门计算机课程的基本要求。这五门课是:《微机系统应用基础》(属第一层次)、《高级语言程序设计》(属第二层次,可选 BASIC、FORTRAN、PASCAL 和 C 语言之一)、《微机原理与应用》(属第三层次的偏硬的方向,使学生具有开发微机硬件系统的初步能力)、《软件开发技术》和《计算机软件基础》(这两门课属第三层次的偏软的方向,使学生具有开发应用软件的初步能力)。以上课程的设置不仅适用于工科,对其它专业也具有参考意义。目前,多数高校的非计算机专业已按以上方案开设了计算机课程。

为了进一步普及计算机教育,国家教委考试中心也已推出面向社会的“全国计算机等级考试”,设一、二、三个级别的考试,相当于上面的前三个层次的内容。

以上的安排为广大计算机的初学者和初、中级计算机应用人员提供了很好的教学体系。

根据广大读者要求,我们参照国家教委工科计算机基础课程教学指导委员会制定的计算机课程基本要求和国家教委考试中心公布的《全国计算机等级考试大纲》,编写了这套《高等学校计算机基础教育丛书》。本丛书的读者对象是高等学校非计算机专业的学生,也可供其它计算机爱好者以及计算机应用人员学习参考。本丛书包括三个层次的各门课程所用的教材,将陆续配套出齐。

本丛书的特点是:实用性强、内容新颖、通俗易懂、概念清晰。我们邀请高等学校中有丰富

教学经验的老师作为丛书的作者。相信这套丛书的出版会受到高校和社会的欢迎。

由于计算机科学技术发展很快。需要学习的内容在不断变化,因此上面介绍的几个层次的内容以及课程的设置也会随着形势的发展而变化。我们将根据计算机科学技术的发展和社会的需要,及时修改补充丛书的内容。

我们希望得到各位专家以及广大读者的支持与帮助,请及时给我们提出宝贵的批评意见,以使丛书的内容不断完善。

谭浩强

前 言

本书是为高等学校非计算机专业学生编写的一本计算机基础教材,可供学习过一门高级语言的读者进修软件后续课程使用。全书由两篇八章组成,即:

上篇 软件开发技术

- | | |
|-----------------|---------------|
| 第一章 算法与数据结构 | 第二章 软件开发技术的演变 |
| 第三章 结构化分析、设计与编码 | 第四章 软件测试与质量保证 |

下篇 微机操作环境

- | | |
|------------|----------------|
| 第五章 用户界面 | 第六章 环境、平台与操作系统 |
| 第七章 DOS 平台 | 第八章 Windows 平台 |

80 年代中、后期,国内就有少数高校开设了《软件技术基础》课,出版了包含数据结构、操作系统、软件工程和数据库等基本内容的若干教材。他们开创了用一门课概括计算机软件知识的先河,缓解了计算机基础课时不足的矛盾。但由于内容中心过多,加上受篇幅的限制,使早期教材多数偏重于知识介绍,很难学了就用。为了弥补这一不足,编者为本书确定了以下的编写原则:

一、基础知识与应用能力并重。在拓宽学生知识面的同时,尽量结合实际应用,克服部分同类教材早先重理论轻实践、现在又只讲应用不讲原理的偏向。

二、突出“软件开发”这个主题,围绕“开发技术”与“操作环境”两个方面讲述软件的基础知识,使全书各章在同一主题下逐步展开,以“众星拱月”代替多个中心。

为了适应非计算机专业学生起点低、又缺乏充分时间阅读参考书等特点,本书采用了“综述”和“细化”相结合的结构。第二章中的“软件开发技术的演变”,综述了各种设计和开发方法的来龙去脉,在上篇 4 章中起着承上启下的作用。而下篇则先用第五、六两章阐明现代操作系统的基本概念,然后用第七、八两章介绍了用户最多的两类 PC 机软件平台——DOS 平台和 Windows 平台(含 Windows 3. X 与 Windows 95)。这一“由顶向下”、“逐步细化”的描述,既改善了全书的可读性,也有利于提高教学的效果。

本书内容参考了高等学校工科计算机基础课程教学指导委员会在 1994 年和 1996 年先后提出的《软件开发技术》和《软件技术基础》等课程的“教学基本要求”,可供普通高校理工科非计算机专业和师范院校数学、物理和计算机等专业本科生、研究生用作 3 学分课程的教材。通过适当地裁剪,也可供 2 学分的《软件技术基础》(选择第一、二及六至八章)、《软件开发技术》(选择第一至五章)、《微机环境与工具》(选择第二章及五至八章)等选修课作为教材。由于在一本书中集成了 DOS、Windows 以及算法、数据结构、软件工程等多方面的软件知识,内容丰富,

叙述简明,对于高等学校计算机基础课教师、中学计算机课教师、教育学院学员、以及其他从事计算机教学和应用的在职人员,均不失为一本有益的参考书。

本书由史济民主编。四川联合大学李建平和华东理工大学史令分别参加了第二章和第八章的编写。原课委会主任委员胡正家教授和副主任委员谭浩强教授对本书的出版均甚关心,使我们深受鼓舞。鉴于计算学科高速发展,软件技术日新月异,编者水平有限,不妥之处在所难免,诚恳希望读者不吝指正。

编 者

目 录

上篇 软件开发技术

第一章 算法与数据结构	3
1.1 计算机算法	3
1.1.1 算法的性质与描述	3
1.1.2 算法分析的基本概念	9
1.1.3 算法设计的基本方法	13
1.2 数据结构	22
1.2.1 数据结构的地位与作用	23
1.2.2 数据结构的种类与基本操作	25
1.2.3 数据结构应用举例	29
习题	44
第二章 软件开发技术的演变	45
2.1 从结构程序设计到软件工程	45
2.1.1 关于 GOTO 语句的争论	45
2.1.2 程序设计风格的转变	46
2.1.3 结构程序设计	47
2.1.4 软件工程	48
2.2 软件工程范型	50
2.2.1 软件工程范型的共性	50
2.2.2 传统的软件工程范型——瀑布模型	51
2.2.3 快速原型范型	53
2.2.4 4GT 范型	54
2.3 软件设计基础	55
2.3.1 软件设计的目的与任务	55
2.3.2 软件设计的基本原则	56
2.3.3 模块化设计和由顶向下设计	59
2.3.4 三类不同面向的设计方法	64

习题	65
第三章 结构化分析、设计与编码	67
3.1 应用软件的分类	67
3.1.1 应用软件的的服务领域	67
3.1.2 应用软件的数据管理方式	68
3.2 结构化分析	69
3.2.1 分析的目的与任务	69
3.2.2 结构化分析的工具	72
3.2.3 结构化分析的步骤	78
3.2.4 结构化分析举例	80
3.2.5 结构化分析小结	83
3.3 结构化设计	84
3.3.1 结构化设计的目的与任务	85
3.3.2 结构化设计的工具	86
3.3.3 变换分析	87
3.3.4 事务分析	90
3.3.5 结构化设计的指导规则	92
3.3.6 结构化设计举例	94
3.4 详细设计与编码	98
3.4.1 逐步细化的设计方法	98
3.4.2 结构化编码	100
3.4.3 编码语言	107
3.4.4 模块设计举例	112
3.5 数据库应用系统的开发特点	115
3.5.1 两类应用系统的比较	115
3.5.2 开发活动的差别	115
习题	116
第四章 软件测试与质量保证	118
4.1 测试的基本概念	118
4.1.1 目的与任务	118
4.1.2 测试的特性	119
4.1.3 测试的种类	120
4.1.4 测试的文档	120
4.2 测试的策略与技术	121
4.2.1 概述	121
4.2.2 黑盒测试	122
4.2.3 白盒测试	125
4.2.4 测试终止标准	133
4.2.5 测试用例设计举例	134
4.3 纠错的策略与技术	139

4.3.1	纠错的策略	139
4.3.2	纠错的技术	141
4.3.3	两个例子	143
4.4	多模块程序的测试	149
4.4.1	测试的层次性	149
4.4.2	程序错误的类型	150
4.4.3	单元测试	152
4.4.4	综合测试	154
4.4.5	高级测试	156
4.5	软件的质量保证	157
4.5.1	评审与测试	157
4.5.2	程序正确性证明	158
4.5.3	软件配置控制	159
4.5.4	软件开发规范	159
	习题	160

下篇 微机操作环境

第五章	用户界面	165
5.1	用户界面的作用与发展	165
5.1.1	用户界面的作用	165
5.1.2	界面的分类	166
5.1.3	界面的分代	166
5.2	用户界面的友好技术	168
5.2.1	多窗口技术	168
5.2.2	菜单技术	169
5.2.3	联机帮助技术	171
5.3	用户界面的操作	176
5.3.1	键盘操作	176
5.3.2	鼠标器操作	177
5.3.3	键盘操作和鼠标器操作的比较	179
5.4	界面技术的实现	179
5.4.1	硬件支持	179
5.4.2	软件支持	180
	习题	186
第六章	环境、平台与操作系统	188
6.1	软件支持环境	188
6.1.1	开发环境与操作环境	188
6.1.2	从 SE 到 CASE	189
6.1.3	CASE 环境的组成与模型	190

6.2	CASE 环境的类型	192
6.2.1	按支持对象分类	192
6.2.2	按宿主机构成分类	194
6.2.3	应用生成系统	194
6.3	硬件平台与软件平台	196
6.4	操作系统的基本概念	197
6.4.1	操作系统的地位	197
6.4.2	操作系统的分类	198
6.5	操作系统的硬件资源管理	200
6.5.1	进程与中断	200
6.5.2	处理机管理	203
6.5.3	存储管理	205
6.5.4	设备管理	210
6.6	操作系统的文件管理	214
6.6.1	基本功能	214
6.6.2	文件的结构与存取方法	214
6.6.3	文件的目录管理	217
6.6.4	外存空间的管理	218
6.6.5	文件的使用与保护	220
	习题	221
第七章	DOS 平台	223
7.1	MS-DOS	224
7.1.1	DOS 的组成与启动	224
7.1.2	DOS 的资源管理	228
7.1.3	DOS 的用户界面	237
7.1.4	DOS 的优化	250
7.2	CCDOS 与 UCDOS	253
7.2.1	汉字信息处理原理	253
7.2.2	CC-DOS	257
7.2.3	UCDOS	260
7.3	DOS 的批处理与系统配置	263
7.3.1	DOS 的批处理	263
7.3.2	DOS 的系统配置	269
7.4	DOS 的工具软件 Dosshell	278
7.4.1	Dosshell 概述	278
7.4.2	Dosshell 的管理功能	281
7.4.3	DOS 平台的新面貌	287
7.5	Turbo 集成开发环境	287
7.5.1	集成化的工具环境	287
7.5.2	易操作的用户界面	288

习题	289
第八章 Windows 平台	290
8.1 Windows 3.x 的特点、组成与安装启动	290
8.1.1 Windows 的特点	290
8.1.2 Windows 的基本组成	291
8.1.3 Windows 的安装与启动	294
8.2 Windows 的界面及其基本操作	296
8.2.1 Windows 的窗口	296
8.2.2 Windows 的图标	299
8.2.3 Windows 的菜单	300
8.3 Windows 的程序管理	302
8.3.1 程序的编组	303
8.3.2 程序的运行	306
8.3.3 多任务切换	308
8.4 Windows 的文件管理	309
8.4.1 文件管理器的菜单	309
8.4.2 多目录窗显示	310
8.4.3 用鼠标拖放文件	310
8.4.4 多个文件的选定与撤选	312
8.5 信息共享与应用程序集成	313
8.5.1 信息共享的常用术语	313
8.5.2 信息共享的三种方式	313
8.5.3 信息共享的常用命令	315
8.5.4 应用系统集成举例	316
8.6 Windows 的其它管理功能	318
8.6.1 打印管理	318
8.6.2 修改系统设置	319
8.6.3 运行非 Windows 应用程序	323
8.7 Windows 的联机帮助	325
8.8 Windows 95 简介	326
8.8.1 Windows 95 的新功能	326
8.8.2 易学易用的操作平台	329
8.8.3 简短的结论	337
习题	337
附 录 UNIX 简介	339

上 篇

软件开发技术

第一章 算法与数据结构

算法与数据结构是计算机程序的两大基础。对计算机算法的研究由来已久。而对数据结构最早进行系统地讨论,并阐明算法与数据结构的相互依赖关系的当推 Hoare 在 1972 年发表的“数据结构札记”一文^[1]。N·Wirth 在谈到算法与数据结构二者的联系时,明确地指出“程序就是在数据的某些特定的表示方法和结构的基础上对抽象算法的具体表述”^[2],并于 1976 年公开发表了他的名著《算法+数据结构=程序》,使“程序设计=算法设计+数据结构设计”的概念从此深入人心。

在高级语言程序设计一类入门性课程中,读者已接触过算法与数据结构的初步概念。本章将着重从实用的角度出发,集中介绍一些有关的基本知识。

1.1 计算机算法

计算机解题一般可分解成若干操作步骤。通常把完成某一任务的操作步骤称为求解该问题的算法。在说明算法与程序的关系时,美国《计算科学基础》一书简明地指出,“算法代表了对问题的解”,而“程序则是算法在计算机上的特定的实现”^[3]。由此可见,一个有效的程序首先要求有一个有效的算法。评价程序质量的标准,诸如清晰、高效、可读性、可修改性和可维护性等,无一不受到算法的影响。所以算法设计实际上可说是程序设计的核心,必须给予足够的重视。本节将首先概述算法的性质和描述,然后简单介绍算法的分析和设计方法。

1.1.1 算法的性质与描述

一、算法的性质

著名计算机科学家、《计算机程序设计技术》系列丛书的作者 knuth,曾把算法的性质归纳为以下 5 点^[4]:

1. 有穷性(finiteness)

一个算法必须在执行有穷步之后结束。换言之,任何算法必须在有限时间内完成。有穷性也隐含执行时间应该合理的涵义。如果一个算法在计算机上要运行上千年,就失去了实用的价值。

2. 确定性(definiteness)

算法中的每一步都必须有清楚的定义,不允许存在多义性和模棱两可的解释。

3. 输入

算法开始前,允许有若干个输入量,也可以没有输入量。

4. 输出

算法执行完毕,必须有一个或数个输出。

5. 能行性(effectiveness)

算法中的每一步操作都应是是可以执行的。当 $B=0$ 时 A/B 就无法执行,不符合能行性的要求。“给 X 增加一个值”也是不可行的,因为它没有说明增加多少。

二、算法的种类

求解不同的问题需要有不同的算法。按照不同的应用领域,计算机算法可以区分为数值计算算法与非数值计算算法两大类。

1. 数值计算算法

主要用于科学计算。随着“数值分析”的发展,对各类数学模型都设计了许多行之有效的算法。例如计算数值积分有梯形法、辛普生法与柯特斯法;对方程式求根有二分法、迭代法或牛顿法;解线性方程组可使用消去法和迭代法等等。在这类算法中,算术运算居于主要的地位。许多复杂的计算,常常被转换为重复进行的简单算术运算。一般地说,它们使用的数据结构比较简单,“简单变量”加“数组”,大致已能满足需要。

数值计算是计算机最早的应用领域,所以对这类算法的研究开展较早。在高等学校中,以数值计算算法为对象的《计算方法》课也已有较长的历史。因此,大多数数值计算问题现均有现成的算法可供选用。

2. 非数值计算算法

常用于数据管理、实时控制以及人工智能等应用领域。与数值计算的算法不同,逻辑判断通常在这类算法中处于主导地位,而算术运算则居于相对次要的地位。算法处理的内容,也从单纯的数值运算,扩展到对数据、图形和字符信息的综合处理。这类算法起步略晚,但发展也很迅速。“排序”和“查找”,就是在这类算法中发展比较成熟、且为大家所熟悉的两个例子。早在1973年,Knuth就已发表了有关这两类算法的专著^[5]。

由于这类算法使用的数据结构一般比较复杂,对算法的设计与选择,往往在很大程度上依赖于处理对象所用的数据结构。所以这类算法的设计通常都与数据结构的选择联系在一起,使问题变得较为复杂。70年代初,Wirth就著文指出“数据结构要与逻辑过程一起细化”^[6],随后又提出了“算法+数据结构=程序”的著名公式,正是反映了非数值计算算法的发展需要。

总的来说,非数值计算算法发展较晚,涉及面也广,远不及数值计算算法成熟。如果说求解数值计算类的问题主要包括选择适用的算法并组织好各种算法的衔接,则求解非数值计算类的问题往往任务更重,有时需要程序员根据具体的问题,自行设计出适用的算法与数据结构。

三、算法的描述

把算法用文字或图形表达出来,就是算法的描述。描述算法可使用文字或图形工具。前者如PDL语言,后者有流程图、N-S图、PAD图等。现分别简介如下:

1. PDL 语言

1975 年,Caine 与 Gordon 在“PDL—— 一种软件设计工具”一文中^[7],发表了他们所设计的 PDL 语言。表 1-1 列出了用 PDL 的 IF 语句和 DO 语句所描述的选择结构和循环结构的格式。

表 1-1 由 PDL 描述的控制结构

选 择 结 构	循 环 结 构
IF <条件> S 或 S* ELSE IF <条件> S 或 S* : ELSE S 或 S* ENDIF	DO WHILE <条件> (或 DO UNTIL <条件> 或 DO FOR <条件>) S 或 S* ENDDO

注:S 代表语句,S* 代表语句群。

表中的<条件>子句可使用简单明了的自然语言来描述。例 1.1.1 列举了怎样在 3 种不同的 DO 结构中用自然语言来描述结束循环的条件。

[例 1.1.1] 用自然语言描述的条件子句举例

- (1) DO WHILE THERE ARE INPUT RECORDS
S
ENDDO
- (2) DO UNTIL “END” STATEMENT HAS BEEN PROCESSED
S
ENDDO
- (3) DO FOR EACH ITEM IN THE LIST EXCEPT THE LAST ONE
S
ENDDO

有些文献将 PDL 一类描述语言称作“伪代码”(pseudo code),因为它们表面上很像高级语言的代码,但又不像高级语言那样要接受严格的语法约束。事实上,不同单位或程序员使用的伪代码也经常略有差异。例如 DO UNTIL 有时也写成 REPEAT UNTIL,IF<条件>有时也写成 IF<条件>THEN,等等。图 1.1.1 是用伪代码描述算法的例子。其中有不少地方使用了仅由动词和宾语组成的祈使句,简单明了,如 execute process X,select process X 等。

```

      :
execute process a
REPEAT UNTIL condition X3
  execute process b
  IF condition X1
  THEN BEGIN
    execute process f
    IF condition X5
    THEN
```

```

        REPEAT UNTIL condition X7
            execute process i
        ENDREP
    ELSE BEGIN
        execute process g
        execute process h
    END
ENDIF
END
ELSE CASE OF X1
    WHEN condition X2 SELECT
        DO WHILE condition X5
            execute process c
        ENDDO
    WHEN condition X3 SELECT process d
    WHEN condition X4 SELECT process e
ENDCASE
ENDIF
ENDREP
execute process j
END

```

图 1.1.1 用伪代码书写的程序逻辑一例

2. 流程图(flowchart)

流程图是使用最早的算法和程序描述工具。它符号简单,表现灵活(例如可用箭头线来表示 GOTO 操作),在很长时期中曾广泛流传。自结构程序设计问世以来,流程图中能随意表达任何控制结构的特点反而成为它的缺点,它在算法描述中的统治地位也逐渐被 N—S 图等其它图形描述工具所取代。这里就不细讲了。

3. N—S 图(N—S diagram)

其名称来源于它的两位作者:Nassi 和 Shneiderman^[8]。图 1.1.2 显示了它的 5 种符号。由于各种控制结构均用方框表示,所以也称为方框图(block diagram)。

N—S 图的优点是只能描述结构化程序所允许的控制结构,从而迫使程序员遵守结构程序设计的原则。方框可以嵌套,也可以并列,能清楚地显示出程序的结构。它的缺点是如果嵌套层数太多,内层的方框将越画越小,影响图形的清晰度。

4. PAD 图(Problem Analysis Diagram)

全名为“问题分析图”,是由日本人二村良彦设计的一种描述工具。与 N—S 图相似,这种图也只能描述结构化程序所允许的控制结构。图 1.1.3 显示了它使用的各种符号。

用 PAD 图描述的算法通常呈树形结构。它既克服了传统流程图不能清晰地显示出程序控制结构的缺点,又不像 N—S 图那样必须将全部算法(或程序)约束到一个方框内,有时会显得十分拥挤。所以从 80 年代初发表之后,很快就从日本走向世界,成为受到广泛欢迎的一种算法描述工具。

以上 4 种描述工具,也称为程序设计工具。一般地说,图形描述工具比较直观,尤其是 N—