21世纪高等学校计算机基础教育系列教材
21 SHIJI GAODENG XUEXIAO JISUANJI JICHU JIAOYU XILIE JIAOCAI

计算机软件技术基础——高级程序设计

■ 刘彦明 荣政 编

人民邮电出版社
POSTS & TELECOM PRESS

21世纪高等学校计算机基础教育系列教材

计算机软件技术基础
——高级程序设计

刘彦明　荣　政　编

人民邮电出版社

图书在版编目（CIP）数据

计算机软件技术基础——高级程序设计 / 刘彦明，容政编. —北京：人民邮电出版社，2005.2
（21 世纪高等学校计算机基础教育系列教材）

ISBN 7-115-13028-0

Ⅰ. 计… Ⅱ.①刘…②容… Ⅲ.程序设计—高等学校—教材 Ⅳ.TP311.1

中国版本图书馆 CIP 数据核字（2005）第 005805 号

内 容 提 要

本书是在多年计算机软件技术基础课程教学实践的基础上，根据新的教学计划和学生对原有教材的修改建议编写而成的。它以程序设计能力的培养为目标，系统地介绍高级程序设计方法以及程序实现的两个关键：数据结构和算法设计。其主要内容包括：高级程序设计方法、编写好程序、排错与测试、程序性能、线性表、数组与串、栈与队列、树、图、索引结构与散列技术、缩小规模算法、搜索算法和“难”问题求解算法等。书中既有基础知识的介绍，也有相关知识的应用实例，具有较高的使用价值。

本书可作为高等院校非计算机专业的电子类本、专科学生学习计算机软件基础的教材，也可供自学计算机软件基础知识的读者作为参考。

21 世纪高等学校计算机基础教育系列教材

计算机软件技术基础——高级程序设计

◆ 编　　刘彦明　荣　政

责任编辑　滑　玉

◆ 人民邮电出版社出版发行　　北京市崇文区夕照寺街 14 号

邮编　100061　　电子函件　315@ptpress.com.cn

网址　http://www.ptpress.com.cn

读者热线　010-67129259

北京隆昌伟业印刷有限公司印刷

新华书店总店北京发行所经销

◆ 开本：787×1092　1/16

印张：21.5

字数：519 千字　　2005 年 2 月第 1 版

印数：1－4 000 册　　2005 年 2 月北京第 1 次印刷

ISBN 7-115-13028-0/TP・4414

定价：32.00 元

编 者 的 话

计算机的普及极大地改变了人们的生活。目前，各行业、各领域都广泛采用了计算机信息技术，并由此产生了开发各种应用软件的需求。为了以最少的成本、最快的速度、最好的质量开发出适合各种应用需求的软件，必须遵循软件开发的原则，掌握软件开发的基础知识和基本技能。

为了培养非计算机专业学生具有软件开发的基础知识和基本技能，国家教育部给出了指导性建议：以计算机文化基础、程序设计语言和软件技术基础为主干课程培养学生的计算机基础知识和基本技能。对于计算机软件技术基础课程而言，各学校、各专业的课程内容设置和教学目标都不尽相同，因而配套的教材种类繁多。归纳起来有两种教材：一是以介绍软件技术为主，其主要内容包括软件工程、数据结构、操作系统、数据库技术，主要介绍软件技术的相关概念；二是以介绍数据结构及其典型应用为主，其主要内容包括线性表、数组和串、栈和队列、树、图、排序以及查找等。

对于非计算机专业的计算机基础知识和基本技能培养而言，当前的培养目标是了解计算机软件开发技术的相关知识，但是这些知识的掌握深度未能明确提出，这就造成了学生在学习软件技术基础课程时，其学习方法欠佳，死背概念，考试之后这些概念又都还给了老师。

另一方面，随着我国经济的快速发展，企业需要的人才类型也为我们制定培养目标提出了明确的要求。那么企业需要何种学生呢？企业需要的是进入企业就能为企业创造效益的学生，现在相当多的企业不再愿意培养员工。这就要求我们在制定培养目标时，首先考虑学生基本能力的培养。

对于非计算机专业的计算机软件技术基础课程而言，学习计算机软件技术的目的是开发软件、编写程序。因此，将计算机软件技术基础的培养目标定位于熟练的程序设计能力最为合适。

程序设计能力的培养除了必要的计算机语言功底外，还必须学习数据结构和算法设计。因此，本书作为计算机软件技术基础课程的教材将主要介绍如下内容。

概述（第 1 章）。对本书内容和必要的基础知识作简要介绍。

程序设计技术（第 2 章至第 5 章）。安排本部分内容的目的是对程序设计语言的进一步提高，主要介绍高级程序设计方法、编写好程序、程序的排错与测试和程序性能。其目标是使学生掌握编写好程序的基本能力。

数据结构（第 6 章至第 11 章）。主要介绍线性表、数组和串、栈和队列、树、图和索引结构与散列技术等。

算法设计（第 12 章至第 15 章）。主要介绍缩小规模算法、搜索算法和“难”问题求解算法等。

总之，针对非计算机专业计算机软件技术基础课程，以培养学生程序设计能力为目标，在总结多年教学实践的基础上，编写了本书。期望本书的出版为广大从事非计算机专业计算

机软件技术基础课程教学的教师提供一种更好的选择，同时也希望广大读者通过本书的学习提高自己的程序设计能力。

由于作者的知识水平和写作水平有限，书稿虽然几经修改，仍难免存在缺点和错误。热忱欢迎同行专家和读者批评指正，并提出宝贵建议，使本书在使用中不断得到改进，日臻完善。

在本书的编写过程中，得到了西安电子科技大学教务处、通信工程学院、电子工程学院和机电工程学院等领导和同行的关心与支持，在此，谨向每一位曾经关心和支持本书编写工作的各方人士表示衷心的谢意。

编者

2004 年 9 月

目　　录

第1章 概述

随着计算机科学技术的发展，软件已经成为一种驱动力：它是商业决策的基本引擎；它是解决现代科学研究和工程问题的基础；它也是区分现代产品和服务的关键因素。它可以应用于各种类型的应用系统中，如办公、交通、教育、医药、通信……数不胜数。

软件成为从基础教育到基因工程的所有领域的推进器。软件无所不在，它已经成为人们现实生活中的实用技术，渗透在人们的工作、安全、娱乐、决策、甚至整个社会生活中。

1.1 软件的基本概念

软件的概念越来越模糊，但也越来越简单。从一般意义上讲，能够对人们提供帮助的、能够在计算机上运行的程序都可以称为软件。更准确的定义是：利用计算机本身提供的逻辑功能，合理地组织计算机的工作流程，简化或替代人们使用计算机过程中的各个环节，提供给使用者一个便于操作的工作环境的“程序集”；它包括计算机程序和与之相关的文档资料的总和（文档是指编制程序所使用的技术资料和使用该程序的说明性资料，即开发、使用和维护程序所需的一切资料）。

软件是逻辑的而不是物理的产品，与硬件相比具有完全不同的特征。

（1）软件是由开发或工程化而形成的，而不是传统意义上的制造产生的。硬件在制造过程中可能引入质量问题，软件则几乎不可能发生类似问题；软件成本集中于开发上，而硬件成本除开发成本外还有生产成本。

（2）软件不会磨损。软件的故障率随时间的推移而降低，而硬件的故障率随时间的推移而增加。硬件模块磨损后可以用新的备用模块替换，而软件故障则是其中的错误，没有可替换的备件。

（3）大多数软件是自定义的，而不是通过已有的组件组装而来的。当然，当前基于组件的软件开发已经成为未来的主要发展技术。

1.1.1 软件应用

在某种程度上讲，难以对软件的应用给出一个确切的分类，下面给出

一些软件应用领域，它们可以算作是一种软件的应用分类。

系统软件：一组为其他程序服务的程序。如操作系统、编译器等。

实时软件：管理、分析和控制现实世界中发生的事件的程序称为实时软件。实时软件的组成包括：数据接收部件，负责从外部环境获取和格式化信息；分析部件，负责将信息转换为具体应用所需要的形式；控制/输出部件，负责对外部事件做出响应；管理部件，负责协调其他各部件，使得系统能够保持一个可接受的实时响应时间。

商业软件：商业信息处理是最大的软件应用领域，具体的“信息系统”均可归入管理信息系统（MIS）软件，它们可以访问一个或多个包含商业信息的大型数据库。该领域的应用将已有的数据重新构造，变换成一种能够辅助商业操作或管理决策的形式。除了传统的数据处理应用外，商业应用软件还包括交互式的和客户/服务器模式的计算（如 POS 事务处理）。

工程和科学计算软件：工程和科学计算软件的特征是“数值分析”算法。此类应用涵盖面很广，从天文学到火山学，从汽车压力分析到航天飞机的轨道动力学，从分子生物学到自动化制造。

嵌入式软件：智能产品在工业和消费电子市场上都是必不可少的，而嵌入式软件正是驻留在这些智能产品中的只读内存中，实现产品的智能应用。嵌入式软件能够执行很有限但专职的功能。

个人软件：仅供个人使用的程序称为个人软件，如文字处理、电子表格、多媒体娱乐、数据库管理、个人金融应用及访问外部网络应用等。

人工智能软件：利用非数值算法解决复杂问题，这些问题不能通过计算或直接分析得到答案。如专家系统（基于知识的软件）、模式识别、定理证明及游戏等。

1.1.2 软件危机与神话

软件危机是指在计算机软件开发过程中遇到的一系列问题，如：开发周期延长，成本增加，可靠性降低等。软件危机包含与下列问题相关的问题：

① 如何开发软件?

② 怎样做才能满足对软件不断增长的需求?

③ 如何维护现有的、容量又在不断增加的软件?

软件危机以许多问题为表征，例如：

① 对软件成本、开发成本和开发进度的估计常常不很准确；

② 用户对“已完成的”软件系统不满意的现象经常发生；

③ 软件产品的质量往往靠不住；

④ 软件常常是不可维护的；

⑤ 软件通常没有适当的文档资料；

⑥ 软件成本在计算机系统总成本中所占的比例逐年上升；

⑦ 软件开发生产率的提高速度远远跟不上计算机应用的普及和发展的趋势。

在软件开发领域，目前还没有一种“灵丹妙药”能够完全“治愈”软件开发所面临的危机。

软件危机的真正含义是软件开发过程中遇到的一系列问题。这些问题不仅仅局限于“不能正确完成功能”的软件，还包括那些与如何开发软件、如何维护大量已有的软件以及软件的开发速度如何跟上目前对软件越来越大的需求等相关的问题。

由于软件危机的诸多因素可以追溯到软件开发的早期阶段所产生的神话，这些神话具有极高的欺骗性，如表面上看很有道理，符合人们的直觉，软件是有经验的开发者开发出来的等等。

软件神话已经给管理人员和技术人员带来了下列严重的问题：

1．管理者的神话

神话：我们已经有了关于建造软件的标准和规范，难道它们不能给人们提供所有其需要知道的信息吗？

事实：真正将这些标准和规范用于软件开发了吗？软件开发者知道它们的存在吗？它们完整吗？很多情况下对于这些问题的回答是“不”。

神话：我们已经有了很多好的软件开发工具，而且我们也买了最新的计算机。

事实：为了使用最新型号的计算机而去开发高质量的软件，我们投入了太多的费用。实际上，计算机辅助软件工程工具比起硬件而言对于获得高质量和高生产效率更为重要，但大多数软件开发者并未使用它们。

神话：如果我们落后于计划，可以增加更多的开发人员以便赶上进度。

事实：软件开发并非是个机械制造过程。实际上，给一个已经延迟的软件项目增加新的开发人员只会使其更加延迟。

2．用户的神话

在许多情况下，用户相信关于软件的神话，因为负责软件的管理者和开发者很少纠正用户的错误理解。神话导致了用户过高的期望值，并引起对开发者的极度不满。

神话：有了对目标的一般描述就足以开始编写程序了——我们可以以后再补充细节。

事实：不完善的系统定义是软件项目失败的主要原因。关于待开发项目的应用领域、功能、接口、设计约束及详细的描述是必需的。而这些内容只有用户和开发者之间的交流才能确定。

神话：项目需求总是在不断变化，这些变化能够很容易满足，因为软件是灵活的。

事实：软件需求确实是经常变化的，但这些变化所产生的影响会随着引入的时间的不同而不同。如果在早期系统定义阶段，这时的需求变化相对而言容易满足，这时对成本的影响也相对较小。当在软件设计过程中要求修改，则对成本的影响就大得多。实现阶段的功能、性能、接口及其他方面的修改对成本会产生更大的影响。当软件投入使用后再要求修改，这时所花的代价是非常巨大的。

3．开发者的神话

程序设计被看成一种艺术，这种旧的观念和方式已经成为软件开发人员的最大隐患。

神话：一旦我们编写出了程序并正常运行，我们的工作就结束了。

事实：越早开始写程序，就会花越长的时间才能完成。50%～70%的时间花费在第一次将程序移交给用户之后。

神话：在程序真正运行之前，没有办法评估其质量。

事实：从项目一开始就可以使用最有效的软件质量保证机制——技术复审，这比在软件

测试时发现错误更有效。

神话：一个成功的软件项目惟一应该提交的就是应用程序。

事实：应用程序仅仅是软件的一部分，软件包括：程序、数据和文档等。

1.2 软件技术

软件是由程序、数据和文档组成。为了开发高质量的软件，软件工程师必须遵循一个开发策略，即软件过程模型。软件过程模型的选择取决于项目和应用特点、采用的方法和工具以及需要的控制和缴付的产品。

所有软件开发都可以看成是一个问题的循环解决过程，其中包括四个截然不同的阶段：状态描述、问题定义、技术开发和方案综述。状态描述“表示了事物当前的状态”；问题定义标识了当前要解决的问题；技术开发通过采用某种技术解决问题；方案综述将结果提交给需要方案的人。

目前存在许多软件过程模型，无序模型[RAC95]、线性顺序模型（比较传统的软件过程模型，包括分析、设计、测试和维护）[ROY70]、原型模型（包括听取用户意见、建造/修改原型、用户测试运行原型、听取用户意见等）[BRO75]、RAD 原型（快速应用开发模型，包括业务建模、数据建模、处理建模、应用生成和测试及反复）[KER94]、演化软件过程模型（包括增量模型[MCDE93]、螺旋模型[BOE88]、构建组装模型[NIE92]、并发开发模型[DAV94]）和形式化方法模型[MIL87,DYE92]。

上述模型各有特点，但无论是用何种模型开发软件，都必须采用合适的软件过程管理技术，即软件项目管理技术。

软件的最终目的是形成产品，但形成产品需要开发过程，如何将产品和过程完美结合是当前软件产品化的一个重大课题。而软件技术正是为了解决该课题而形成的一门学科。

软件技术是指开发软件所需的所有技术的总称。按照软件分支学科的内容划分，软件技术有如下几个领域。

（1）软件工程技术

软件工程技术是软件技术的一个分支学科，它主要研究软件开发全过程中的各种技术，包括：软件开发的原则与策略、软件开发方法与软件过程模型、软件标准与软件质量的衡量、软件开发的组织与项目管理以及软件版权等。

（2）程序设计技术

程序设计技术主要包括：程序的结构与算法设计、程序设计的风格、程序设计语言、程序设计方法和程序设计自动化、程序的正确性证明、程序的变换（高效和可读性不可兼得，可读性是低效的，低效的程序变为高效的程序称为程序的变换）等。

（3）软件工具环境技术

软件工具环境技术主要包括：人机接口技术、软件自动生成、软件工具的集成和软件开发环境、软件的复用以及逆向工程等。

（4）系统软件技术

系统软件技术主要包括：操作系统、编译方法、分布式系统的分布处理与并行计算、并

行处理技术以及多媒体软件处理技术等。

（5）数据库技术

数据库技术主要包括：数据模型、数据库与数据库管理系统、分布式数据库、面向对象的数据库技术、工程数据库及多媒体数据库等。

（6）实时软件技术

实时软件技术主要包括：实时监控软件技术和嵌入式实时软件技术。

（7）网络软件技术

网络软件技术主要包括：协议工程、网络管理、局域网技术、网络互连技术及智能网络等。

（8）与实际工作相关的软件技术

与实际工作相关的软件技术主要包括：如何延长软件的使用时间和不断增强软件的性能、如何控制软件的质量、如何改变管理和配置记录、如何设计用户的在线帮助文档和图标、软件规模控制、软件评估和软件开发计划的制订以及软件需求的表示和软件规格说明书的确定。

从软件技术的范畴看，软件开发需要许多相关的知识，但是作为非计算机专业的学生应掌握哪些计算机软件技术呢？笔者认为，作为非计算机专业的开发人员应掌握的最适用的软件技术是程序设计技术和实时软件技术。

本书重点介绍程序设计技术。而程序设计技术的关键是数据结构和算法设计。本章将从程序设计的基本思想入手，介绍程序设计的基本技术，后面的章节将重点讨论数据结构和算法设计。

1.3 程序设计技术

程序设计技术的目标是提高程序的质量与生产率，最终实现程序的工业化生产。影响程序质量的因素很多，如正确性、性能、可靠性、容错性、易用性、灵活性、可扩充性、可理解性、可维护性和复用性等等。有些因素相互重叠，有些则相抵触，因此提高程序质量并不容易。本书所介绍的程序设计技术主要针对规模小且一个人能够独立完成的程序的设计技术。程序设计的主要环节有：明确需求、设计、实现（编码）和测试等，如图1-1所示。

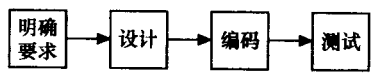

图1-1 程序设计的主要环节

1.3.1 明确需求

程序设计的第一步工作是明确需求，即明确待解决的问题是什么？也称为问题分析阶段。

该阶段是面向“问题”的，它主要是对问题的业务活动（如飞机订票）进行分析，明确在问题的业务环境中，程序应该“做什么”， 后面的设计、编程阶段则是面向“解答”的，这时考虑的是如何构造一个满足问题要求的程序。所以，在问题分析阶段，应集中考虑程序“做什么”，而尽可能少考虑系统将怎样具体实现的问题，这些问题都应尽量推迟到以后的阶段去解决。

为了更好地理解一个问题，必须回答下列问题：

① 谁使用该程序？

② 使用该程序的具体目的是什么？

③ 程序需要做哪些工作？具有的功能是什么？

1.3.2 设计

设计是把需求转化为程序的最重要的环节。设计的优劣在根本上决定了程序的质量。

设计包括程序结构设计、模块设计、数据结构与算法设计和用户界面设计。如果将程序比喻为人体，那么可作如下的描述。

（1）结构就如同人的骨架。如果某个家伙的骨架是猴子，那么无论怎样喂养和美容，这家伙始终都是猴子，不会成为人。

（2）模块就如同人的器官，具有特定的功能。人体中最出色的模块设计之一是手，手只有几种动作，却能做无限多的事情。人体中最糟糕的模块设计之一是嘴巴，嘴巴将最有价值但毫无相干的几种功能如吃饭、说话、亲吻混为一体，使之无法并行处理，真乃人类之不幸。

（3）数据结构与算法就如同人的神经和血液，它让器官具有生命并能发挥功能。数据结构分布在结构上，它将协调系统的各个功能。人的耳朵和嘴巴虽然是相对独立的器官，但如果耳朵失聪了，嘴巴就只能发出“啊”“呜”的声音，等于丧失了说话的功能，可人们却又能用手势代替说话。

（4）用户界面就如同人的外表，最容易让人一见钟情或一见恶心。像人类追求心灵美和外表美那样，程序也追求（内在的）功能强大和（外表的）界面友好。但随着生活节奏的加快，人们已少有兴趣去品味深藏不露的内在美。

1．程序结构设计

结构是程序中最本质的东西。

（1）结构是对复杂事物的一种抽象。良好的结构是普遍适用的，它可以高效地处理多种多样的个体需求。一提起“房子”，人们的脑中马上就会出现房子的印象（而不是洞的印象）。“房子”是人们对住宿或办公环境的一种抽象。不论是办公楼还是民房，同一类建筑物（甚至不同类的建筑物）之间都具有非常相似的结构和构造方式。

（2）结构在一定的时间内保持稳定。程序设计最怕的就是需求变化，但“需求会发生变化”是个无法逃避的现实（在实际的开发过程中）。人们希望在需求发生变化时，最好只对程序做些皮毛的修改，可千万别改动程序结构。就如人们对住宿的需求也会变动，人们可以经常改变房间的装潢和摆设，但不会在每次变动时都要去拆墙、拆柱、挖地基。如果当需求发生变化时，开发人员不得不去修改程序结构，那么这个程序设计是失败的。

良好的结构意味着适应性强、高效和稳定。下面介绍非常通用的程序结构：层次结构。

层次结构表达了这么一种常识：有些事情比较复杂，没法一口气干完，就把事情分为好几层，一层一层地去做，高层的工作总是建立在低层的工作之上。层次关系主要有两种：上下级关系和顺序相邻关系。

上下级关系的层次结构，如同在军队中，上级可以命令下级，而下级不能命令上级，程序的上下层关系也符合上述常识，即上层子程序（模块）可以使用下层子程序（模块）的功能，而下层子程序（模块）不能够使用上层子程序（模块）的功能。

顺序相邻关系的层次结构表明通信只能在相邻两层之间发生，信息只能被一层一层地顺序传递。这种层次结构的经典之作是计算机网络的OSI参考模型。为了减少设计的复杂性，