教育部师范教育司组织编写

中学教师进修高等师范本科（专科起点）教材

程序设计方法

薛锦云 主编

薛锦云 李云清 杨庆红 王洪发 编著

高等教育出版社

内容提要

本书是教育部师范教育司组织编写的中学教师进修高等师范本科（专科起点）“程序设计方法”课程教材。本书的主要内容包括程序设计方法概论、C 语言基本要素、C 语言中的控制结构、结构化程序设计方法、C 语言中的组合数据类型、基于功能抽象的程序设计方法、抽象数据类型、基于分划和递推的程序设计方法、面向对象程序设计等。本书主要以方法为主导，结合 C 语言，把程序设计方法学研究中若干成熟的理论和方法用通俗易懂的语言描述出来。本书还选取趣味性强、技巧性高、能够启发学生创造性思维的例题，以适应指导中学生参加各类程序设计比赛的需要。

本书主要面向进修高等师范本科（专科起点）的中学教师，也可作为普通高等师范院校计算机专业和非计算机专业学生的教材或教学参考书。

图书在版编目（CIP）数据

程序设计方法 / 薛锦云主编. —北京：高等教育出版社，2001.12
师范类专升本计算机专业教材
ISBN 7-04-010180-7

Ⅰ.程... Ⅱ.薛... Ⅲ.程序设计-方法-高等教育：师范教育-教材 Ⅳ.TP311.11

中国版本图书馆CIP数据核字（2001）第079242号

程序设计方法
薛锦云 主编

出版发行	高等教育出版社		
社　　址	北京市东城区沙滩后街 55 号	邮政编码	100009
电　　话	010-64054588	传　　真	010-64014048
网　　址	http:// www.hep.edu.cn http:// www.hep.com.cn		
经　　销	新华书店北京发行所		
印　　刷	北京市鑫鑫印刷厂		
开　　本	787×960　1/16	版　　次	2001 年 12 月第 1 版
印　　张	15.75	印　　次	2001 年 12 月第 1 次印刷
字　　数	280 000	定　　价	13.70 元

本书如有缺页、倒页、脱页等质量问题，请到所购图书销售部门联系调换。

前　言

当前常见的高级语言程序设计教程以介绍程序设计语言为中心，很少介绍程序设计方法，致使书中只有程序，没有程序设计过程，教师和学生都难以说清楚程序是如何得来的，不能很好地起到培养学生程序设计能力的作用。事实上，科学的程序设计方法至关重要，是各级各类程序设计人员必须掌握的知识和技能。许多有识之士已注意到这一问题。教育部颁发的《中小学信息技术课程指导纲要》已将“程序设计方法”作为全国高中学生信息技术基础课程的重要内容。但目前国内系统介绍程序设计方法的教材还不多见，能够见到的只是程序设计方法学教程。这是计算机专业高年级学生和硕士研究生专业课的教材。这类教程介绍了多种程序设计方法，但其中有些内容涉及严格的数学推演，难以被非计算机专业的学生和进修高等师范本科（专科起点）的中学教师理解和掌握。为此，急需一本系统地反映国内外成熟的程序设计方法的通俗易懂的新教材。编著出版本书的目的正是为了满足这一需要。我们长期从事程序设计方法的研究和教学，试图将这一领域实用的理论、方法和最新研究成果，经过化繁就简，化难为易，体现在教材中。

本教材根据教育部师范教育司中学教师进修高等师范本科（专科起点）教学计划中“确保高等师范本科的教育水平，科学地处理专科知识与本科知识的衔接关系，避免知识重复”的要求组织内容。教材以 C 语言为程序设计的主要语言，假定读者已初步掌握 C 语言及其程序设计的基本知识，着重介绍 C 程序中的数据类型、数据结构的设计、程序流程和控制等高级语言机制。现代程序设计理论认为，语言是方法的具体实现。本书主要以方法为主导，结合 C 语言，系统地阐述结构化程序设计、模块化程序设计、抽象数据类型、功能抽象和数据抽象等成熟的程序设计概念和方法，并吸收当前国外同类教材的经验，把程序设计方法学研究中若干成熟的理论和方法（如程序规约、循环不变式等）用通俗易懂的语言描述出来，使之成为一般程序员也能理解和掌握的实用程序设计技术。算法设计是程序设计的核心、关键和难点。在介绍程序设计方法时突出介绍算法的地位和算法设计的方法。

本书主要面向进修高等师范本科（专科起点）的中学教师。首先满足他们在中学从事各类程序设计课程教学的需要，并选取趣味性强、技巧性高、能够启发学生创造性思维的例题，以适应指导中学生参加各类程序设计比赛的需

要。考虑到目前普通高等师范院校尚没有合适的程序设计方法教材，本教材也可作为普通高等师范院校计算机专业和非计算机专业学生的教材或教学参考书。我们力争使本书既能反映程序设计方法学科规律，突出基本概念和基本方法，又能体现该领域最新科研成果和现代教学思想，便于培养创新思维能力，便于自学，做到科学性、实用性和易理解性的统一。

全书共分 9 章，建议按 72 学时进行教学。第 1 章是程序设计方法概论，建议用 4 学时；第 2 章介绍 C 语言的基本要素，宜用 6 学时；第 3~7 章，各安排 8 学时为好；第 8 章内容较新较多，需安排 12 学时；第 9 章为选学内容，可安排 10 学时。

本书第 1 章、第 2 章和第 8 章由薛锦云撰写，第 3 章和第 4 章由杨庆红撰写，第 5 章和第 6 章由王洪发撰写，第 7 章和第 9 章由李云清撰写，全书由薛锦云统稿和定稿。华东师范大学黄国兴教授认真审阅了全书，提出了宝贵的修改意见。研究生骆健和赖勇参加了部分编写工作，在此一并表示衷心感谢。

本书在内容和课程体系方面的革新，体现了我们承担的教育部“高等师范教育面向 21 世纪教学内容和课程体系改革计划”课题取得的研究成果。这是我们的初次尝试，再加上时间仓促和水平所限，书中难免存在错误和疏漏，恳请广大专家和读者不吝赐教。

编 者

2001 年 8 月

目　录

第 1 章　程序设计方法概论

【学习目标】

通过本章的学习，学生应该能够：

- 深刻理解程序、程序设计语言和程序设计方法之间的关系，明确程序设计方法在程序设计中的核心作用；
- 了解程序设计方法形成和发展的过程；
- 掌握目前常见程序设计方法的概况。

1.1　程序设计语言和程序设计方法

计算机的出现是 20 世纪最杰出的发明之一。进入 20 世纪以后，科学的迅猛发展，迫切要求有计算快、精确度高、能按规定自动计算和进行自动控制的新型计算工具。因此，计算机应运而生。从 1946 年第一台电子计算机在美国问世以来，计算机使我们的工作、生活发生了革命性的改变。这与计算机程序设计技术的发展密不可分。

计算机及其应用技术发展到今天，程序设计已不再是单纯技巧性的活动，与其他技术科学一样，程序设计也有了自身的科学理论和方法，既程序设计方法学。程序设计方法学的诞生使程序设计从技巧升华为科学。

1.1.1　程序、语言和程序设计方法

纵观人类的发展史，语言的出现使得人类与其他动物区分开来。人们用语言表达思想，进行交流。在计算机的世界里，要使计算机能够按人的意图工作，就要向计算机发出命令，这就需要语言。然而，计算机不懂人类的自然语言，为了能与计算机交流，需要有计算机能够理解和执行的特殊语言，这种语言专门用于程序设计（程序是语言描述的指令或语句的集合，它规定了计算机的每一步操作），称为程序设计语言。

新型程序设计方法的研究必然会导致新的程序设计语言的产生，回顾历史我们可以看到：当软件危机出现后，Wirth 和 Dijkstra 等人提出了结构化程序设计方法的思想，对这种方法的深入研究导致 Pascal 语言的产生；当抽象

数据类型概念被提出的时候，诞生了基于对象的程序设计方法，而这以后出现了具有数据抽象能力的 ADA 语言。可以说，语言是程序设计方法不断向前发展的产物。

从应用的角度来看，语言又是程序设计方法的体现，不同类型的语言体现了不同程序设计方法的设计思想。例如，Pascal 语言所编写的程序，由一系统过程组成，体现了结构化程序设计方法的自顶向下、逐步求精、模块化的思想，而 ADA 所提供的程序包可将数据和对数据的操作封装在一起，这正体现了基于对象程序设计中数据抽象和数据封装的思想。

程序设计语言与程序设计方法之间这种密切的关系，对于程序设计的学习具有指导意义。学习任一种新型程序设计语言的重点是理解该语言所体现的方法，掌握方法所体现的原理和思想精髓，才能增强写出高效程序的能力。语言的语法和语义的学习是第二位的。这样才能培养和提高程序设计的能力，而不至于陷入只懂得机械背诵语言的语法和语义，却不会编制程序的怪圈。

1.1.2　语言作为程序设计工具

产生程序的全部技术活动序列称为程序设计，它主要包括需求分析、算法设计、用某种计算机语言实现算法、程序正确性验证和测试等步骤。用计算机语言实现算法是程序设计过程中不可缺少的环节，语言在此充当着描述工具的角色，它将解决某项任务所用的算法以计算机能够接受的形式表达出来，便于计算机处理。

学习作为工具的语言并不是学习程序设计的主要目的。程序设计的着重点和技术关键是算法设计。这是一项极富挑战的创造性劳动，是程序设计能力的重要标志。用高级语言编码是第二位的，是一项非创造性劳动，可以由计算机机械完成。

1.1.3　程序设计方法的作用

在正式学习本书之前，了解一下程序设计方法的作用是十分必要的。它能让我们明确学习程序设计方法的意义及今后学习所要达到的目标。

首先，从哲学的角度来看，任何实践活动都需要理论的指导，否则将是盲目的实践。程序设计作为人类在计算机领域的实践活动自然也不例外。Brooks 曾生动地描述在程序设计方法学产生前程序员编制和调试程序所处的困境："像巨兽在泥潭中作垂死挣扎，挣扎得愈猛，泥浆就沾得越多，最后没有一个野兽逃脱淹没在泥潭中的命运……程序设计就像是这样一个泥潭……一批批程序员在泥潭中挣扎……"。但是初学者千万不要被刚才的话语所吓倒，不要失去学习程序设计的信心和勇气。因为这样的时代已经过去，程序设计已有了科

学的理论和方法作为指导。只要掌握了科学的利器，学习程序设计将是一件令人身心愉悦的事。

其次，通过程序设计方法的学习，可以极大地提高程序设计能力。程序设计用英语表示为 Programming。Programming 常常可理解成规划，也就是说程序设计的能力实际上一种设计和规划的能力，因此，通过对程序设计方法的学习，不仅可以提高对相应语言的运用能力，而且可以从全局的角度对整个软件系统进行规划。

最后，科学的程序设计方法是程序质量的可靠保证。如果没有程序的正确性，软件的正确与否将无从谈起。程序正确性是程序员首先应该追求的目标，如果脱离程序的正确性，程序的效率和精美将变得没有意义。科学的程序设计方法将指导人们设计出尽可能正确的程序。

1.2 程序设计方法的形成和发展

1.2.1 程序设计技巧阶段

要使计算机按照人的意图工作，则必须使计算机懂得人的意图，接受人发出的命令，因此，与计算机交流就存在着“语言”的问题。计算机语言的发展经历了一个由低级走向高级的过程。

1. 机器语言

计算机并不能识别人们日常生活中所使用的自然语言，它只能识别“0”和“1”两种状态，人们与计算机交流时，只能使用由 0 和 1 组成的代码，这种代码称为机器指令，一条机器指令能够控制计算机执行一种操作，机器指令的集合称为机器语言。

然而，机器语言又称为是计算机专家的“专用语言”，如果不了解计算机内部结构及其指令系统则无法用机器语言编程。使用机器语言编写程序有难学，难记、难读、难修改、通用性差等缺点，严重阻碍了计算机的普及推广。

2. 汇编语言

为了克服机器语言的缺点，方便计算机的使用，20 世纪 50 年代初，计算机工作者开始使用一系列便于理解和记忆的符号来代替 0 和 1 组成的难记、难读的代码，这种符号指令的集合称为汇编语言。

汇编语言的出现，大大减少了程序编写、阅读、修改和维护等方面的工作量，然而汇编语言仍然是一种面向机器、通用性差的低级语言，它与人们习惯使用的自然语言仍有较大的差异，并且程序员仍不能完全摆脱了解计算机内

部细节的困境。

3. 高级语言

20 世纪 50 年代末，高级语言的诞生让程序员摆脱了低级语言所带来的种种不便。用高级语言编写的程序独立于机器，程序员编程时无需关心计算机内部的细节，从而大大提高编程效率。高级语言种类繁多，使用较广泛的几种语言有 FORTRAN、BASIC、COBOL、Pascal、C 等。

20 世纪 60 年代末，高级语言的兴起使计算机应用日益广泛，逐步渗透到各行各业，所开发的程序也越来越庞大，越来越复杂，当时判断好程序的标准是：指令少，运行较快，占用内存少。这使得许多程序员都过分地注意追求语言的表述能力和技巧，力求达到语句少，占用内存少的目标。程序中常常无限制的使用 goto 语句或引入一些不合理、难以理解的成分。在这个时期，软件开发主要是依赖程序员的个人经验和技巧，缺乏科学理论和方法作指导；对于开发出的软件或程序，只能靠测试来查找错误，没有科学的方法来保证程序的正确性。这些种种因素使得许多大型软件崩溃。软件开发周期长、生产效率低、可靠性及可维护性较差，成为 20 世纪 60 年代末开发大型复杂软件所面临的突出问题，并最终引发了“软件危机”。

总之，从计算机诞生到 20 世纪 60 年代末这段时间，程序设计没有科学理论和方法的指导，主要靠灵感和技巧，致使程序质量差，开发效率低。

1.2.2 程序设计从技巧上升为科学

软件危机的爆发使人们不得不去思考和程序设计相关的许多重要问题。例如，程序调试能否保证程序正确？怎样保证程序正确？程序好坏的评价标准是什么？指导程序设计的原理和方法是什么？

1968 年，ACM 通讯发表了 E. W. Dijkstra 的文章“Goto statement considered harmful”被称为程序设计方法革命的第一里程碑。Dijkstra 认为：Goto 语句是有害的，它造成了程序结构的混乱，高级程序设计语言应取消 Goto 语句。由此，引发了关于 Goto 语句的大讨论。而这一场讨论，又引发了关于程序设计首先是讲究好的结构，还是讲究效率的讨论。

Dijkstra 还提出了结构化程序设计思想，给出了好结构程序的概念和标准。好结构程序应具有以下特点：①单入口单出口；②无死语句，即程序的每部分都应至少有一条从入口到出口的路径通过它。Dijkstra 希望通过程序静态结构上的良好性来保证程序正确性，并提出“程序测试只能发现程序中的错误而不能保证程序无错误”的科学断言。

与此同时，瑞士的 Wirth 在文章“自顶向下逐步求精”中提出了自顶向下逐步求精的程序设计方法。自顶向下逐步求精的程序设计方法实质上是“分而

治之”策略在程序设计中的应用，它将一个复杂的问题，分解成若干个相互独立的、相对更为简单的模块，再将模块分解成更简单的子模块，依此类推，一直到所有问题能用程序设计语言简单方便地解决。

采用自顶向下逐步求精和结构化程序设计方法，能有效地控制程序设计的复杂性，有益于提高程序设计的效率。

在这一时期，另一个引人注目的研究方向则是“程序正确性证明”。什么样的程序才是正确的？如何来保证程序正确？这些问题是计算机广泛应用后迫切需要解决的问题。许多著名的计算机科学专家在这个方向上付出了不懈的努力。

1967 年，Floyd 和 Naur 提出了证明框图程序正确的归纳断言法（又称中间断言法），他们在程序框图的每个结点处设置中间点，给出在程序执行时，保持为真的断言，并给出中间点的验证条件，然后用数学方法证明这些验证条件为真。

1969 年，Hoare 在 Floyd 工作的基础上提出了公理方法，他将描述程序的语义性质的语句用逻辑公式表示出来，使程序正确性证明成为可能，但是在实际应用时仍有许多困难，程序证明需要的中间断言及证明循环程序必须的循环不变式都难以构造。

1975 年，Dijkstra 提出了最弱前置谓词的概念和程序推导的方法，解决了中间断言构造难的问题，并可从程序规约推导出正确程序，使程序正确性证明开始变得实用。

20 世纪 80 年代，Gries 综合了以谓词演算为基础的程序设计理论和方法的成果，完成了“The Science of Programming”一书，首次把程序设计从经验、技巧上升为科学，程序设计完成了从技巧上升为科学的过渡。

1.2.3 几种实用的程序设计方法

1. 过程式程序设计方法

从过程式程序设计的角度来看，程序是若干个过程（函数）组成的序列，它采用自顶向下逐步求精的手段，实现模块分解和功能抽象。程序按功能被划分成若干个模块，各模块在功能上相对独立。每个模块可以用过程或函数来实现。因此，过程式程序设计也称作模块化程序设计。过程式程序设计常采用自顶向下和自底向上相结合的分析和设计方法。

在过程式程序设计中，数据和过程是相互独立的两个实体。要保持数据与程序的相互一致，程序员将要增加许多额外的工作量。

主要的过程式程序设计语言有 FORTRAN、BASIC、Pascal、C 等。

2. 基于对象的程序设计方法

该方法一方面采用自顶向下逐步求精的手段，对程序按功能进行模块划分，这就是所谓的功能抽象；而另一方面，把数据和对数据进行的操作密切联系在一起，做成一个实体封装起来。对于外界，这个封装实体的内部细节可以是不可见的，从而实现了数据抽象。主要的基于对象的程序设计语言有 ADA、APLA 等。基于对象的程序设计方法以抽象数据类型的理论和思想为基础。

3. 面向对象的程序设计方法

面向对象的程序设计方法是一种直观方法，它模拟了人类认识问题中的分类过程。现实世界的某一类事物对应于面向对象系统的“类”；现实世界的某一具体物质对应于面向对象系统的“对象”。面向对象程序设计追求的目标就是尽量将现实世界的事物映射到软件系统的解空间。

面向对象程序设计采用数据抽象和信息隐藏技术，将数据及对它的操作封装在一起。类的继承性是实现代码重用的有效途径。有了高质量的可重用代码就能降低软件的复杂度，提高软件开发效率。这也正是面向对象程序设计方法的优越性所在。与基于对象的程序设计方法相比较，面向对象的程序设计方法增加了继承和多态两大功能。面向对象程序设计方法的理论基础也是抽象数据类型。

4. 函数式程序设计方法

在函数式程序设计中，程序被看作是描述输入与输出之间关系的一个数学函数。程序中完全没有变量这一概念，并且具有数学引用的透明性，即不存在赋值语句。主要的函数式程序设计语言有 LISP。

5. 逻辑程序设计方法

在逻辑程序设计中，逻辑被看作知识表示和推理的工具。程序被看作描述输入与输出之间各种关系的一组方程。程序设计可归结为事实列举、定义逻辑关系、逻辑公式演绎，以提问方式求解。Prolog 是一种典型的逻辑程序设计语言。

6. 形式化程序设计方法

形式化程序设计方法以提高软件可靠性和生产效率以及实现程序设计自动化为目标。它用谓词逻辑和数学公式精确地描述算法和程序的功能或需求，构成形式化规约。这种形式的规约精确而没有二义性，便于进行数学的变换，为开发正确的算法程序提供了有力的支持。VDM、Z、RAISE 等就属于这种类型的形式化程序设计方法。更高层次的形式化方法不仅支持形式功能规约的开发，还支持算法程序开发的全过程，也就是能够基于形式功能规约，通过一系列形式化数学变换，推导出正确的程序；或者对已有的程序，用数学的方法进行严格的正确性证明。这样的形式化程序设计方法为实现程序设计自动化提供了有力的支持。由法国学者 Abrial 发明的 B 方法，以及本书要介绍的 PAR

方法都属于这类形式化程序设计方法。

习 题 一

1.1 试述程序、程序设计语言和程序设计方法之间的正确关系，并以实例说明程序设计方法的作用。

1.2 试用本章学习的内容分析你所学过的某些语言程序设计教程的优、缺点。

1.3 试用简洁的语言描述程序设计由技巧上升为科学的过程。

第 2 章　C 语言基本要素

【学习目标】

通过本章的学习，学生应该能够：

- 掌握 C 语言的主要特点以及 C 语言程序的结构；
- 熟悉 C 语言的数据类型和常量、变量的定义方法；
- 掌握 C 语言的各种运算符和它们的优先级，以及表达式的构成规则；
- 熟练掌握输入输出语句的功能和格式。

本章学习的重点和难点是数据类型、运算符、赋值表达式、逗号表达式。

2.1　C 语言概述

C 语言是国际上广泛应用的计算机高级语言。C 语言的产生主要归功于贝尔实验室的 D. M. Ritchie。为了保持 BCPL 和 B 语言精练、接近软件的优点，克服 B 语言过于简单的局限性，他于 1972 年在小型机 PDP-11 上实现了 C 语言。C 语言作为描述和实现 UNIX 操作系统的工作语言，随着 UNIX 的日益广泛使用，也被迅速地推广。

实用的 C 编译系统种类繁多。适用于微型机的 C 编译系统主要有 Microsoft C、Quick C、Turbo C 等，它们都可在 PC-DOS 或 MS-DOS 系统下运行。

2.1.1　C 语言的特点

C 语言被称为高级语言的低级形式以及低级语言的高级形式。这说明 C 语言同时具有高级语言与低级语言的许多优点。C 语言以其功能强大、表达灵活而著称，几乎具有无所不能的本领。因此，它广泛应用于编写各种系统软件和应用软件。与其他计算机语言相比较，C 语言具有以下特点：

1. C 语言的优点

（1）语言表达能力强。C 语言兼有高级语言和低级语言的特点，可以直接处理字符、数字、地址，几乎具有汇编语言的全部功能。

（2）具有丰富灵活的数据结构和结构化的控制语句。

（3）语言简洁，共有 32 个关键字和 9 种控制语句；程序书写格式自由。

（4）可移植性好。C 程序基本上不用或少加修改就可以用于不同的计算机和操作系统上。

（5）生成的代码质量较高，通常其代码效率只比汇编语言描述的代码低10%~20%。

2. C 语言的不足

（1）C 语言是非强类型语言，数据类型的转换较为随便，这就带来了不安全因素。

（2）C 语言不具备自动数据边界检查功能。

2.1.2 C 语言的程序结构

在深入学习 C 语言之前，先例举一个简单而完整的 C 程序实例，以说明 C 程序的结构、特点及设计风格。

例 2.1 简单的 C 程序。

```
main ( )                          /*主函数*/
{                                 /*开始*/
  int a,b;                        /*定义变量*/
  float  sum;
  scanf ("%d",&a);                /*输入语句*/
  b=2; sum=a +b ;                 /*赋值语句*/
  printf(" sum =%f " ,sum );      /*输出语句*/
}                                 /*结束*/
```

（1）C 程序由函数构成。一个可执行的 C 程序必须有且只能有一个主函数 main()，并且总是从主函数开始执行。

（2）一对大括号是用来构成函数体的分隔符，表示函数的开始与结束。

（3）第 3 行、第 4 行是定义变量语句。关键字 int 说明 a 和 b 是整型变量，float 则说明 sum 是浮点型变量。C 语言中所有的完整语句都必须以分号“;”结束。

（4）第 5 行是输入语句，从键盘接受输入并赋给变量 a。

（5）第 6 行为赋值语句。C 程序允许若干条语句串写在同一行，彼此之间用分号分隔。

（6）第 7 行是输出语句，即在屏幕上显示 sum 的值。scanf()和 printf()是 C 的库函数，用于实现标准的输入和输出。

（7）有一对注释符“ / *”和“* / ”括起的内容是 C 语言中的注释内容。注释不属于程序本身的内容，只是提高程序的可读性而加的。“ / *”是注释的开始，“* / ”是注释的结束。