湖南省高等院校统编教材　湖南省九五重点建设教材

C语言程序设计与分析教程

主编：郭浩志　副主编：林亚平

湖南科学技术出版社

Yuyan Chengxu Sheji yu Fenxi Jiaocheng

湖南省高等院校统编教材　湖南省九五重点建设教材

C语言程序设计与分析教程

顾　　问：李洪基
主　　编：郭浩志
副 主 编：林亚平
编　　委：（以姓氏笔画为序）
陈洛资　周大庆　林亚平　郝三如
郭浩志　谢深泉　蒋大文　曾三槐
组织策划：蒋易春　易建华　王文斌　刘少波

湖南科学技术出版社

湖南省高等院校统编教材　湖南省九五重点建设教材

C 语言程序设计与分析教程

主　　编：郭浩志
副 主 编：林亚平
责任编辑：胡海清
出版发行：湖南科学技术出版社
社　　址：长沙市展览馆路 66 号
印　　刷：湖南省新华印刷一厂
厂　　址：长沙市芙蓉北路 564 号
邮　　编：410008
（印装质量问题请直接与本厂联系）
出版日期：1998 年 7 月第 1 版第 1 次
开　　本：787mm×1092mm　1/16
印　　张：26.25
字　　数：592000
印　　数：1—10100
书　　号：ISBN 7—5357—2467—1/TP・112
定　　价：20.00 元

内 容 简 介

本书为湖南省普通高校非计算机专业C语言程序设计课程的全省统编教材，湖南省“九五”重点建设教材。它是在湖南省教委高教处的领导与组织下，由国防科技大学等八所大学的教授编写而成的。全书包括三篇。第1篇为基础篇，重点介绍与语言、程序有关的基础概念，以及用C语言编写程序的预备知识；第2篇为程序设计与分析篇，详细阐述了十种程序设计和分析的基本方法与技巧；第3篇是程序开发与调试篇，讨论了开发与调试Turbo C程序的方法、工具与环境。各章均有丰富的练习与实习。全书遵循全国、全省有关大纲，精选了典型数据结构和算法，强化编写程序、阅读程序和调试程序，并安排了课程设计。

本书是大学生的公共基础课，不仅适用于众多非计算机专业，也适用于计算机专业；不仅可供教师参考，也可供企事业管理人员和科技人员使用。

前言
PREFACE

本书为湖南省普通高校非计算机专业C语言程序设计的全省统编教材，“九五”重点建设教材。它是在湖南省教委高教处的领导与组织下，由国防科技大学等八所高等院校的教授依照国家教委面向21世纪教学内容和课程体系改革的精神及1998年《湖南省普通高校非计算机专业学生计算机应用水平等级考试大纲》的要求编写而成的。主编是郭浩志教授，副主编是林亚平教授。

撰写本书的指导思想主要有三点：

1. 加强大学生计算机素质教育。本课程属于计算机基础教学第二层次的范畴，主要训练学生利用计算机处理问题的思维方式与工作方式，这是21世纪人材知识、能力和素质结构中不可缺少的重要组成部分，是当今高等教育的公共基础课。

2. 培养各专业复合型人材的计算机软件技术基础。围绕C语言这一当今使用最为广泛的人机交互工具，使学生掌握计算机软件技术中有关程序的基础知识、程序设计与分析方法，以及用计算机解决主专业问题的能力。

3. 与等级考试接轨。本书内容力求覆盖等级考试中关于程序语言以及程序设计、分析和调试能力的要求。

全书共分三篇。第1篇为基础篇，重点介绍与语言、程序有关的基础概念，C语言的主要特点、简史、用处以及用C语言编写程序的预备知识；第2篇为程序设计与分析篇，详细阐述了十种程序设计和分析的基本方法和技巧；第3篇是程序开发与调试篇，主要以Turbo C为背景，讨论了C语言程序开发、调试的方法、工具与环境。

本书主要特点如下：

1. 精选了常见的数据结构与典型算法；

2. 为强化编写程序和阅读程序的训练，第2篇各章均有代表性示例分析编写程序的思路和方法，解剖程序结构和功能；

3. 为加强动手能力，多数篇章给出了内容和形式都比较丰富的、有针对性的练习题及实习示例题；

4. 突出程序调试训练。第3篇专门讨论了有关的程序排错方法、调试工具和

常见错误分析；

5. 安排了小型课题（约几百行）的课程设计，为进一步学习软件开发打下初步的基础；

6. 与等级考试接轨。凡参加一、二级考试者均可适用。

建议本科学时数为 60～80 学时，大专学时数为 50～70 学时。凡标有‘*’的章节专供本科和参加二级考试使用。

参加本书编写的人员有：国防科技大学郭浩志（前言，第 2 篇第六、七、八章），湖南大学林亚平（第 1 篇第一、二章）和李军义（第 2 篇第三、四、五章），湖南财经学院周大庆（第 2 篇第九、十、十一、十二章）以及长沙铁道学院陈洛资（第 3 篇第十三、十四、十五章 15.1 节）、黄汉永（15.2 节）和杨彦（15.3 节）。全书由郭浩志统编。蒋易春、易建华、王文斌、刘少波和胡春晖等做了大量组织工作。郭旭东为本书很多示例做了调试工作。本书撰写过程中，得到了中南工业大学、长沙铁道学院、湖南大学、湖南师范大学、湖南农业大学、湖南财经学院、湘潭大学、国防科技大学以及其他很多高等院校的关心和支持。湖南科学技术出版社，尤其是胡海清编审为本书的出版付出了辛勤的劳动。在此一一表示衷心的感谢！

虽然本书大多数内容已在若干所大学试用，并反复修改过，但是由于编写时间仓促，有些院校近年教学改革的好经验，可能未来得及融入本书之中。加上编者水平和经验有限，书中错误、缺点在所难免，热忱欢迎读者提出宝贵的意见和建议。

编　者

1998 年 5 月

目录
CONTENTS

第一篇 基础

第一章 高级语言程序设计基础

程序设计语言主要经历了机器语言、汇编语言和高级语言这三代语言的演变。机器语言和汇编语言是面向机器的低级语言，编制程序十分不便。传统的高级语言是面向过程的语言，其优越性是提高了编程人员的工作效率，比较低级语言，用高级语言编制程序更容易编写、调试和阅读理解。由于不依赖机器，在某种机器或某种操作系统下用高级语言编制的程序，容易移植到其他机器或操作系统下执行，也就是说高级语言编制的程序具有很好的可移植性。

现代计算机技术的发展，又导致了使用更为方便的第四代语言的出现，但从严格意义上说，第四代语言与前述的三代语言有着明显的差别，它属于一类自动化程度比较高，且通常是面向某些专门应用的开发工具或开发环境，如数据库查询语言，用于 Web 网页制作的 HTML 语言（超媒体语言），以及诸如报表、屏幕格式等应用生成程序。前三代语言在程序设计上更讲究方法和技巧，强调的是如何做程序；而第四代语言强调的是自动化地做什么应用。

综上所述，为了理解程序设计的概念，设计方法和技巧，乃至理解各种形式计算机软件（包含第四代语言）的使用，学习高级语言程序设计是十分必要的。因此，高级语言程序设计也属于计算机基础教育中的一种基本训练。本章主要讲述高级语言程序设计的基本知识。

1.1 高级语言程序设计概述

程序设计语言是人们为了描述计算过程而定义的一组有一定语法和语义规则的记号。语法表示程序的结构或形式，它描述了实现语言基本记号的组合规则，用户如果违背了这些规则，编译程序时便会出错。语义表示的是程序的含义，它描述了语言记号组合所表示的特定含义，语义的解释与语言本身及编译程序相关，而与用户使用一般无关。

在高级语言发展初期，流行的语言主要有 Fortran、Algo60 和 Cobel 语言。Fortran 语言是第一个问世的高级语言，现在 Fortran 语言仍然流行，目前常用的版本是 Fortran77 和 Fortran90；Algo60 语言现在已不流行，但它对后续的一些语言如 Pascal 等有着重要影响；Cobel 语言在数据库技术出现以前，广泛用于商业事务处理，它对数据库技术的发展有着重要作用。继 Fortran 之后高级语言发展迅速，出现了数百种语言，其中 Basic 语言以其简单易学而拥有最为广泛的用户群。

结构程序在程序设计语言发展过程中具有重要意义，1968 年 E. W. Dijkstra 指出高级语言中使用转向语句带来的问题，导致结构程序设计这一新的程序设计方法问世。

E. W. Dijkstra 证明，任何程序都可以只用顺序、选择和循环这三种结构的语句构造，因此可以避免使用转向语句带来的程序阅读性差和其他问题。在这一时期出现了以 Pascal 为代表的结构程序设计语言。尽管目前大多数高级语言都保留了转向语句，但其功能和形式都变得简单，一般只用于辅助其他的控制结构。不少语言在控制结构上提供了丰富的顺序，选择和循环语句，如 Pascal，C 等语言。新出现的 Basic 语言版本（如 True Basic 等）也在控制结构上进行了扩充，提供了多种循环控制语句。语言的控制结构加上自顶向下，逐步求精的模块化程序设计方法构成结构程序设计的框架。结构程序设计的思想也扩充到现代软件工程。

目前比较流行，有代表性且广泛用于教学的传统高级语言主要有 Basic，C，Fortran 和 Pascal 等。比较而言，Basic 语言简单易学，适合初学者；Fortran 语言注重效率，适合科学计算；Pascal 语言精致简明，可阅读性好，适合用于教学和描述算法；本书介绍的 C 语言则十分灵活，是目前最为流行的系统软件与应用软件开发程序语言之一。

如前所述，传统高级语言是一种面向过程的程序语言，或称强制性语言。在高级语言发展过程中，人们也开始注意发展其他风格，其他范型的语言。这些语言主要可以分为函数型，逻辑型和面向对象型语言。函数型语言因为其形式很类似数学上的函数而得名。典型的例子有 Lisp 语言，它主要用于人工智能处理等方面；逻辑型语言以逻辑程序设计思想为理论基础，利用逻辑规则和推理机制产生结果，最有代表性的逻辑语言是 Prolog 语言，它主要用于人工智能领域如专家系统方面的程序设计；面向对象高级程序语言的出现是程序语言的一次重大变革。它与传统过程性语言的主要区别在于：在传统过程性语言中把数据以及处理它们的子程序当作互不相关的成分分别加以处理，而在面向对象语言中则把这两者统一作为对象封装在一起进行处理。面向对象的思想充分体现了软件工程要求的模块性，信息隐蔽性和抽象性。最早的面向对象程序设计语言是 Smalltalk，该名字的本义就是“少说话”，意指只要很少的语句就可以做许多工作。继 Smalltalk 之后，许多面向对象的程序语言或系统先后出现，如 C++，VB，Java 等有代表性的语言。面向对象技术已成为目前计算机软件的主要开发技术。

用编辑程序编制的程序称之为源程序，源程序必须翻译成机器指令才能在机器上执行。根据翻译的方式不同，可将高级语言分成编译型和解释型。大多数高级程序属于编译型，即源程序通过编译生成目标程序，然后经连接程序（Linker）生成可执行程序；解释型高级语言是对程序语句一边翻译一边执行，Basic 语言支持解释程序。近年来出现的 Java 语言具有另一特点，Java 源程序经过编译后生成的代码是一种与机器无关的虚目标代码，这种代码在不同的机器上再通过与机器相关的解释程序执行。Java 这种编译和解释相结合的翻译方式，不存在程序移植问题，因此特别适合采用流览器/Web 服务器这种跨平台方式的 Web 应用。

1.2 算法与数据结构

程序设计中一个有名的公式是：程序＝算法＋数据结构，其基本意思是指编一个好的程序取决于选用好的算法和设计有效的数据结构。算法决定了程序解题时的运行时间复杂性和存储空间复杂性，而数据结构的设计则是为实现算法的程序存取数据更为方便和有效。因此，了解算法和数据结构的基本概念对程序设计十分有益。

1.2.1 算法

为求解一个问题而采用的方法和解题步骤称为算法。求解同一问题可能存在不同的算法，因此存在算法的选择问题。例如，判定某不小于 3 的自然数 n 是不是素数，可以用 [2，$n-$

1] 间的自然数依次去除 n 而得到答案；但也可以只用 2 到 $\sqrt{n}$ 之间的自然数依此去除 n，就可得到答案。显然，第二种算法比第一种算法解题更快。在第一种算法中，为确定 n 是素数，必须做 $n-2$ 次除法，其时间复杂性与 n 属于同一级别，我们用 $O(n)$ 表示；而用第二种算法确定 n 是素数，只须做 $\sqrt{n}$ 次除法，当 n 较大时，$\sqrt{n}$ 比 n 小得多，两者不是一个数量级别，我们记第二种算法的时间复杂性为 $O(\sqrt{n})$。由此可知，选择好的算法可以事半功倍。

算法可以分为数值计算算法和非数值计算算法，数值计算指的是通常意义下的计算，如求线性方程组的解，求方程的根等计算方法；非数值计算算法包含对表、树、图等非数值性的结构的处理方法，以及对各种数据进行排序，查找等方面的算法。

1.2.2 数据结构

在确定了解题的算法，准备编程实现算法时，首先要考虑的是如何将算法中涉及的数据用合理、有效的结构予以组织，以便存取数据。这便关系到数据结构问题。

数据是描述客观事物的数、字符、符号、表格以及图形、图像、声音等可由计算机程序处理的对象集合。简单而言，数据结构是指带有结构的数据元素的集合。数据结构包含两个方面的意义，一方面指的是数据元素之间的逻辑关系，即说明元素之间的某种顺序关系，称之为逻辑结构；另一方面指的是数据的物理结构，它描述的是数据逻辑结构在计算机存储器中的映像，也称为数据的存储结构。数据的存储结构可以分为顺序存储结构和链式存储结构两大类，所谓顺序结构指的是数据元素按照它的逻辑结构顺序地存放在一组连续的存储单元中；而链式结构中则不要求数据元素连续存储，但需要采用一种指针变量来指示数据元素之间的顺序关系。相同逻辑结构的数据元素采用相同的存储结构，因此，一般统一用数据结构这一名词陈述。

程序语言中提供的数据类型是语言已经实现的一些基本数据结构。高级语言中提供的数据类型主要可分为三大类：

1. 简单类型：包括整型、实型、字符型等数据。

2. 构造类型：构造类型是由已知类型如简单类型按一定规则构造而成的类型，例如数组就是最简单但又最常用的构造类型，数组可以有整型数组、字符型数组等。C 和 Pascal 语言中还提供更为复杂的构造类型，如结构（Pascal 语言中称记录）和联合。

3. 指针类型：它主要用于构造链表、树、图等各种复杂的数据结构。通过指针这种特殊的变量来指示数据元素之间的顺序关系。在 C 语言中，指针实际上就是存储单元的地址，其应用范围尤其广泛，但要小心使用。

程序语言提供的数据类型丰富如否，在很大程度上决定了该语言的强弱，C 和 Pascal 语言的数据类型就比 Basic 和 Fortran 丰富得多，因此用 C 和 Pascal 实现一些复杂的算法（特别是非数值算法）比 Basic 和 Fortran 方便得多。

利用高级语言提供的基本数据类型，可以构造其他有用的数据结构。例如，高级语言都提供有数组，在数组的基础上，我们可以构造队列，堆栈等线性表结构。许多算法的实现都依赖一定的数据结构，我们将结合 C 语言，在以后的章节中适当予以介绍。

1.3 程序设计基础知识

早期的程序设计，在确定了算法和数据结构之后，便交给程序员去编写程序，编写的程序只要在机器上通过，便算完成任务。但如果编写的程序较大或从软件工程的角度去衡量，情

况就不一样了。从软件工程的角度出发，编写的程序不仅要保证正确性，而且要保证程序的可读性，可修改性，和可维护性。只有保证了这些因素，才能说保证了程序的质量。因此，程序员编写的程序不仅要考虑自己读懂，还要考虑别人看懂，而且还要提供可修改的余地。由此可见，编制程序必须注意程序设计基本方法和风格，以保证程序的可读性和可维护性。

1.3.1 算法描述工具

编写程序（特别是大型程序）之前，最好先用一种方法描述算法步骤，以方便编程。常用的方法主要有自然语言、程序流程图、问题分析图（PAD 图）、N—S 图（也称盒状图）等。下面主要介绍程序流程图和 N—S 图两种方法。

一．程序流程图

程序流程图采用一些几何图形表示算法步骤中的各种操作，图 1.3.1 列出了常用的流程图符号。

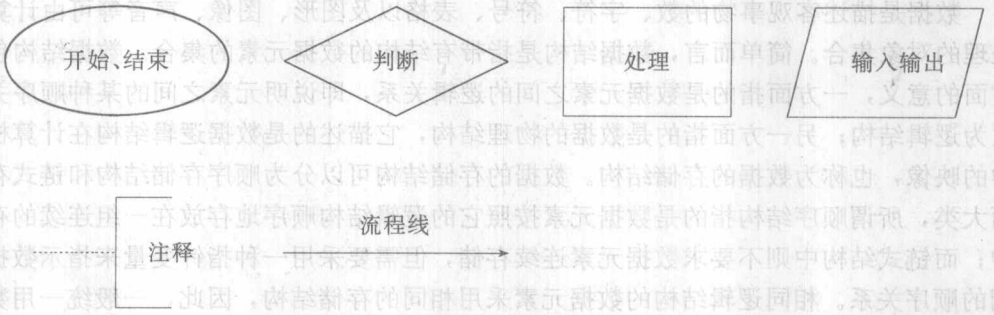

图 1.3.1　常用程序流程图符号

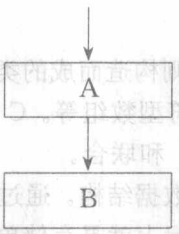

图 1.3.2　顺序结构

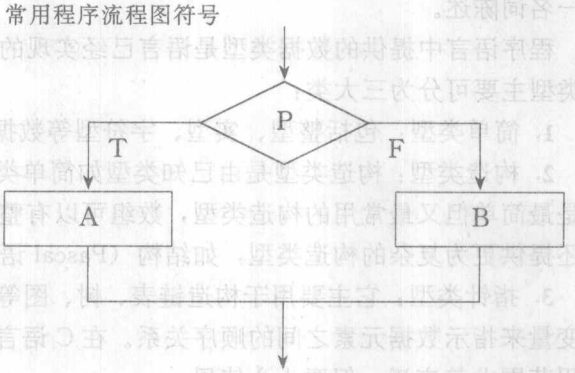

图 1.3.3　选择结构

利用流程图表示算法，直观形象化，容易理解和掌握，但由于它允许流程线指向图中任意一个程序框，导致程序走向任意，因而违反结构程序设计原则，在此基础上编制程序逻辑思路不清楚，可读性差。因此，在使用程序流程图时，应限制流程图只由几种基本控制结构嵌套而成。图 1.3.2 到图 1.3.4 分别给出了顺序、选择和循环三种基本控制结构所对应的程序流程。其中，图 1.3.2 和 1.3.3 分别表示顺序和选择控制，图 1.3.4 分别给出了几种常用的循环控制结构，在图 1.3.2 中，表示 A 操作完成后，执行 B 操作；图 1.3.3 中表示当条件 P 满足时，执行 A 操作，不成立时执行 B 操作；图 1.3.4（a）中，表示当条件 P 成立时，反复执行 A 操作；循环结构的另一种形式见图 1.3.4（b），它表示反复执行 A 操作，直到条件 P 被满足。注意，图 1.3.4（a）中，表示先判定条件，再决定是否执行操作，因此 A 操作可能一次也不执行；而图 1.3.4（b）中，表示先执行操作，再判定条件，决定是否继续执行操