



高职高专计算机精品课程系列规划教材

# C语言程序设计

孙凤美 主 编

刘 莹 李明仑 姜大庆 副主编



中国铁道出版社  
CHINA RAILWAY PUBLISHING HOUSE

高职高专计算机精品课程系列规划教材

# C 语言程序设计

孙凤美 主编

刘莹 李明仑 姜大庆 副主编

**中国铁道出版社**  
CHINA RAILWAY PUBLISHING HOUSE

---

## 内 容 简 介

本书是作者根据多年的软件开发经验和教学经验编写而成的；在内容的编排上本着精选理论、强化实践、突出技能的原则，内容前导后续，衔接、过渡自然，结构清晰，充分体现了易学易用的特点；在文字叙述上条理清晰、简洁，便于阅读理解。

全书共分 12 章，主要内容包括：C 语言程序设计概述，C 语言程序设计基础，顺序结构程序设计，选择结构程序设计，循环结构程序设计，数组，函数，指针，结构体，联合体，枚举，位运算，编译预处理，文件。通过典型例题分析，着重强调应用 C 语言进行程序设计的方法和技巧，深入浅出。每章配有大量的例题、重点内容总结和大量的习题，以有利于读者全面、系统地理解和掌握 C 语言程序设计。

本书覆盖了全国计算机等级考试二级大纲所要求的范围，适合作为高职高专教材，同时也可作为参加计算机等级考试人员的自学或培训教材。

### 图书在版编目（CIP）数据

C 语言程序设计/孙凤美主编. —北京：中国铁道出版社，2008.12

（高职高专计算机精品课程系列规划教材）

ISBN 978-7-113-09473-7

I. C… II. 孙… III. C 语言—程序设计—高等学校：技术学校—教材 IV. TP312

中国版本图书馆 CIP 数据核字（2008）第 198680 号

书 名：C 语言程序设计

作 者：孙凤美 主编

策划编辑：翟玉峰 沈 洁

责任编辑：王雪飞

封面设计：付 巍

责任印制：李 佳

编辑部电话：（010）63583215

封面制作：白 雪

出版发行：中国铁道出版社（北京市宣武区右安门西街 8 号 邮政编码：100054）

印 刷：三河市兴达印务有限公司

版 次：2008 年 12 月第 1 版 2008 年 12 月第 1 次印刷

开 本：787mm×1092mm 1/16 印张：15.25 字数：353 千

印 数：4 000 册

书 号：ISBN 978-7-113-09473-7/TP·3101

定 价：23.00 元

版权所有 侵权必究

本书封面贴有中国铁道出版社激光防伪标签，无标签者不得销售

凡购买铁道版的图书，如有缺页、倒页、脱页者，请与本社计算机图书批销部调换。

# 前言

## FOREWORD

C语言是一种结构化语言，层次清晰，便于按模块化方式组织程序，易于调试和维护。C语言的表现能力和处理能力极强，不仅具有丰富的运算符和数据类型，便于实现各类复杂的数据结构，还可以直接访问内存的物理地址，进行位（bit）一级的操作。由于C语言实现了对硬件的编程操作，因此C语言集高级语言和低级语言的功能于一体，既可用于系统软件的开发，也适用于应用软件的开发。此外，C语言还具有效率高、可移植性强等特点。因此，C语言从诞生之日起就被广泛地移植到了各种类型的计算机上，从而受到了广大编程人员的喜爱。

现在许多大中专院校都开设了C语言课程；越来越多的计算机程序设计人员也把C语言作为入门语言；教育部考试中心在全国计算机等级考试的考试大纲中，也把C语言作为程序设计二级考试的可选语种之一和三级考试的必考语种。

本书概念准确，叙述流畅，重点突出，例题实用性强，通俗易懂。

本书主要有以下特点：

- 结构清晰、紧凑。
- 精选理论、强化实践、突出技能。
- 本着循序渐进的原则，先讲基本概念和语法规则，然后通过实例分析加深理解，前导后续，最后对重点、难点进行总结。
- 例题丰富，趣味性较强，程序全部上机调试通过。
- 例题程序书写规范，读者通过学习和模仿，有利于养成良好的编程习惯。
- 习题量大，针对性强。

不少人在学习C语言时，感到入门难，对很多问题的理解支离破碎，作者建议初学者一定要多读、反复研读、勤于思考，然后试着去写，举一反三。多读、多写、多上机调试，只有这样才能尽快掌握和运用C语言去解决实际问题。

本书由孙凤美任主编，刘莹、李明仑、姜大庆任副主编。其中，孙凤美编写第3~12章，刘莹编写第1章，李明仑编写第2章，姜大庆编写附录。在编写过程中得到了秦永生、刘鹏、亓胜田等的大力支持和帮助。全书由孙凤美统稿定稿。

本书的编写和出版在各方面得到了许多友人的支持和帮助，马宝林、朱云萍、白翔、王海霞、李利萍、姜伟、王传东、王丰、张义明等为本书的编写和出版提供了支持，在此一并表示感谢。

由于作者水平有限，书中难免存在疏漏和不足之处，敬请批评指正。

编 者

2008年10月

# 目 录

## CONTENTS

|                         |    |
|-------------------------|----|
| 第 1 章 C 语言程序设计概述 .....  | 1  |
| 1.1 C 语言的发展历史 .....     | 1  |
| 1.2 C 语言特点 .....        | 2  |
| 1.3 算法 .....            | 3  |
| 1.3.1 算法特性 .....        | 3  |
| 1.3.2 算法描述 .....        | 3  |
| 1.4 结构化程序设计和模块化结构 ..... | 4  |
| 1.4.1 结构化程序设计 .....     | 4  |
| 1.4.2 模块化程序设计 .....     | 6  |
| 1.5 C 语言程序的编辑与运行 .....  | 7  |
| 本章小结 .....              | 9  |
| 习题一 .....               | 9  |
| 第 2 章 C 语言程序设计基础 .....  | 11 |
| 2.1 简单 C 语言程序的构成 .....  | 11 |
| 2.2 标识符 .....           | 12 |
| 2.3 常量 .....            | 13 |
| 2.3.1 直接常量 .....        | 14 |
| 2.3.2 符号常量 .....        | 15 |
| 2.4 变量 .....            | 15 |
| 2.4.1 整型变量 .....        | 16 |
| 2.4.2 实型变量 .....        | 17 |
| 2.4.3 字符型变量 .....       | 17 |
| 2.4.4 强制类型转换 .....      | 18 |
| 2.5 运算符和表达式 .....       | 18 |
| 2.5.1 算术运算符和算术表达式 ..... | 18 |
| 2.5.2 关系运算符和关系表达式 ..... | 19 |
| 2.5.3 逻辑运算符和逻辑表达式 ..... | 20 |
| 2.5.4 赋值运算符和赋值表达式 ..... | 21 |
| 2.5.5 自增、自减运算符 .....    | 22 |
| 2.5.6 逗号运算符和逗号表达式 ..... | 23 |
| 2.5.7 其他运算符 .....       | 23 |
| 2.6 应用举例 .....          | 23 |
| 本章小结 .....              | 24 |
| 习题二 .....               | 25 |
| 第 3 章 顺序结构程序设计 .....    | 28 |
| 3.1 赋值语句 .....          | 28 |

|                                          |           |
|------------------------------------------|-----------|
| 3.2 输入/输出 .....                          | 29        |
| 3.2.1 printf 函数 (格式输出函数) .....           | 29        |
| 3.2.2 scanf() 函数 (格式输入函数) .....          | 32        |
| 3.2.3 字符型输入/输出函数 (getchar/putchar) ..... | 35        |
| 3.2.4 字符串输入/输出函数 (gets/puts) .....       | 35        |
| 3.3 空语句 .....                            | 36        |
| 3.4 复合语句 .....                           | 36        |
| 3.5 顺序结构程序举例 .....                       | 37        |
| 本章小结 .....                               | 39        |
| 习题三 .....                                | 39        |
| <b>第 4 章 选择结构程序设计 .....</b>              | <b>44</b> |
| 4.1 if 语句 .....                          | 44        |
| 4.1.1 简单的 if 语句 .....                    | 44        |
| 4.1.2 if-else 语句 .....                   | 45        |
| 4.1.3 复杂的 if-else 语句 .....               | 46        |
| 4.1.4 if 语句的嵌套 .....                     | 48        |
| 4.1.5 条件表达式 .....                        | 49        |
| 4.2 switch 语句 .....                      | 51        |
| 4.3 选择结构程序举例 .....                       | 53        |
| 本章小结 .....                               | 55        |
| 习题四 .....                                | 55        |
| <b>第 5 章 循环结构程序设计 .....</b>              | <b>58</b> |
| 5.1 while 语句 .....                       | 58        |
| 5.2 do-while 语句 .....                    | 61        |
| 5.3 for 语句 .....                         | 64        |
| 5.4 循环嵌套 .....                           | 66        |
| 5.5 break、continue 语句 .....              | 69        |
| 5.6 goto 语句和标号语句 .....                   | 70        |
| 5.7 几种循环比较 .....                         | 71        |
| 5.8 循环结构程序举例 .....                       | 72        |
| 本章小结 .....                               | 76        |
| 习题五 .....                                | 76        |
| <b>第 6 章 数组 .....</b>                    | <b>80</b> |
| 6.1 一维数组 .....                           | 80        |
| 6.1.1 一维数组的定义 .....                      | 80        |
| 6.1.2 一维数组的初始化 .....                     | 81        |
| 6.1.3 数组元素的引用 .....                      | 82        |
| 6.2 二维数组 .....                           | 84        |
| 6.2.1 二维数组的定义 .....                      | 85        |

|              |                      |            |
|--------------|----------------------|------------|
| 6.2.2        | 二维数组的初始化 .....       | 85         |
| 6.2.3        | 二维数组元素的引用 .....      | 86         |
| 6.3          | 字符数组与字符串 .....       | 88         |
| 6.3.1        | 字符数组的定义、初始化及引用 ..... | 88         |
| 6.3.2        | 字符串和字符串结束标志 .....    | 90         |
| 6.3.3        | 字符数组的输入/输出 .....     | 91         |
| 6.3.4        | 常用字符串处理函数 .....      | 91         |
| 6.4          | 数组应用举例 .....         | 92         |
|              | 本章小结 .....           | 94         |
|              | 习题六 .....            | 95         |
| <b>第 7 章</b> | <b>函数 .....</b>      | <b>98</b>  |
| 7.1          | 函数分类 .....           | 98         |
| 7.2          | 函数定义 .....           | 99         |
| 7.3          | 函数调用 .....           | 101        |
| 7.3.1        | 函数调用的一般形式 .....      | 101        |
| 7.3.2        | 函数调用方式 .....         | 101        |
| 7.3.3        | 函数的参数和函数的值 .....     | 102        |
| 7.3.4        | 数组作为函数参数 .....       | 104        |
| 7.3.5        | 函数的嵌套调用 .....        | 109        |
| 7.3.6        | 函数的递归调用 .....        | 111        |
| 7.4          | 变量的作用域 .....         | 113        |
| 7.4.1        | 局部变量 .....           | 113        |
| 7.4.2        | 全局变量 .....           | 114        |
| 7.5          | 变量的存储类型 .....        | 116        |
| 7.5.1        | 自动变量 .....           | 117        |
| 7.5.2        | 外部变量 .....           | 119        |
| 7.5.3        | 静态变量 .....           | 119        |
| 7.5.4        | 寄存器变量 .....          | 121        |
| 7.6          | 内部函数和外部函数 .....      | 121        |
| 7.6.1        | 内部函数 .....           | 122        |
| 7.6.2        | 外部函数 .....           | 122        |
| 7.7          | 函数应用举例 .....         | 122        |
|              | 本章小结 .....           | 124        |
|              | 习题七 .....            | 124        |
| <b>第 8 章</b> | <b>指针 .....</b>      | <b>128</b> |
| 8.1          | 指针简介 .....           | 128        |
| 8.2          | 指针变量的操作 .....        | 129        |
| 8.2.1        | 指针变量的赋值 .....        | 129        |
| 8.2.2        | 指针变量的运算 .....        | 130        |

|                             |            |
|-----------------------------|------------|
| 8.3 数组与指针 .....             | 133        |
| 8.3.1 一维数组与指针 .....         | 133        |
| 8.3.2 字符串与指针 .....          | 135        |
| 8.3.3 用数组名作函数参数 .....       | 138        |
| 8.4 二维数组与指针 .....           | 139        |
| 8.4.1 二维数组的地址 .....         | 139        |
| 8.4.2 行指针变量 .....           | 140        |
| 8.4.3 指针数组 .....            | 141        |
| 8.4.4 main()函数的参数 .....     | 142        |
| 8.5 指针应用举例 .....            | 143        |
| 本章小结 .....                  | 145        |
| 习题八 .....                   | 146        |
| <b>第9章 结构体、联合体、枚举 .....</b> | <b>149</b> |
| 9.1 结构体 .....               | 149        |
| 9.1.1 结构体类型定义和结构体变量说明 ..... | 149        |
| 9.1.2 结构体成员的引用 .....        | 152        |
| 9.1.3 结构体变量的初始化 .....       | 152        |
| 9.1.4 结构体变量的赋值 .....        | 153        |
| 9.1.5 结构体数组 .....           | 154        |
| 9.1.6 结构体指针变量 .....         | 156        |
| 9.1.7 结构体指针变量作为函数参数 .....   | 159        |
| 9.2 利用结构体变量构成链表 .....       | 160        |
| 9.2.1 动态存储分配 .....          | 160        |
| 9.2.2 链表概述 .....            | 162        |
| 9.2.3 链表的基本操作 .....         | 162        |
| 9.3 联合体 .....               | 167        |
| 9.3.1 联合体的定义 .....          | 167        |
| 9.3.2 联合体变量的说明 .....        | 168        |
| 9.3.3 联合体变量的赋值和使用 .....     | 169        |
| 9.4 枚举 .....                | 170        |
| 9.4.1 枚举类型的定义和枚举变量的说明 ..... | 170        |
| 9.4.2 枚举类型变量的赋值和使用 .....    | 171        |
| 9.5 类型定义符 typedef .....     | 172        |
| 9.6 结构体综合应用举例 .....         | 175        |
| 本章小结 .....                  | 178        |
| 习题九 .....                   | 179        |
| <b>第10章 位运算 .....</b>       | <b>182</b> |
| 10.1 位运算符 .....             | 182        |
| 10.2 位运算符的运算功能 .....        | 182        |

|                               |            |
|-------------------------------|------------|
| 10.3 位运算应用举例 .....            | 185        |
| 本章小结 .....                    | 186        |
| 习题十 .....                     | 187        |
| <b>第 11 章 编译预处理 .....</b>     | <b>188</b> |
| 11.1 宏定义 .....                | 188        |
| 11.1.1 无参宏定义 .....            | 188        |
| 11.1.2 带参宏定义 .....            | 191        |
| 11.2 文件包含 .....               | 194        |
| 11.3 条件编译 .....               | 195        |
| 11.4 编译预处理应用举例 .....          | 197        |
| 本章小结 .....                    | 199        |
| 习题十一 .....                    | 199        |
| <b>第 12 章 文件 .....</b>        | <b>201</b> |
| 12.1 文件概述 .....               | 201        |
| 12.1.1 文件的分类 .....            | 201        |
| 12.1.2 文件的处理 .....            | 202        |
| 12.1.3 文件类型指针 .....           | 203        |
| 12.2 文件的打开与关闭 .....           | 204        |
| 12.2.1 文件打开 (fopen()函数) ..... | 204        |
| 12.2.2 文件关闭 (fclose 函数) ..... | 206        |
| 12.3 文件的读/写操作 .....           | 207        |
| 12.3.1 文件的字符读/写函数 .....       | 207        |
| 12.3.2 文件的字符串读/写函数 .....      | 209        |
| 12.3.3 格式化读/写函数 .....         | 211        |
| 12.3.4 文件的数据块输入/输出函数 .....    | 213        |
| 12.4 文件定位函数 .....             | 215        |
| 12.5 文件应用举例 .....             | 217        |
| 本章小结 .....                    | 219        |
| 习题十二 .....                    | 219        |
| 附录 A 常用 ASCII 码字符对照表 .....    | 221        |
| 附录 B 运算符的优先级和结合性 .....        | 222        |
| 附录 C C 语言中的关键字 .....          | 224        |
| 附录 D C 语言中的头文件 .....          | 225        |
| 附录 E C 语言中的库函数 .....          | 227        |
| 参考文献 .....                    | 231        |

# 第 1 章 C 语言程序设计概述

C 语言是一种通用的程序设计语言，由于其功能非常强大，因此可以用来完成一些非常复杂的工作。很多操作系统都是用 C 语言编写的，例如 MS-DOS、UNIX、Microsoft Windows、Linux 等。它不仅具有丰富的运算符和数据类型，便于实现各类复杂的数据结构；还可以直接访问内存的物理地址，进行位（bit）一级的操作。由于 C 语言实现了对硬件的编程操作，C 语言集高级语言和低级语言的功能于一体，既可用于系统软件的开发，也适用于应用软件的开发。此外，C 语言还具有效率高、可移植性强等特点。因此，C 语言从诞生之日起就被广泛地移植到各种类型的计算机上，也受到广大编程人员的喜爱。

## 1.1 C 语言的发展历史

### 1. 程序设计语言分类

程序设计语言一般分为机器语言、汇编语言和高级语言三大类。

① 机器语言：计算机只能接收和处理由 0 和 1 构成的二进制指令和数据，由于这种形式的指令是面向机器的，因此也称“机器语言”。一组机器指令就是程序，称为机器语言程序。

② 汇编语言：为了便于理解与记忆，人们采用能帮助记忆的英文缩写符号（称为指令助记符）来代替机器语言指令代码中的操作码，用地址符号来代替地址码。

③ 高级语言：机器语言和汇编语言都是面向机器的语言，一般称为“低级语言”。低级语言较抽象且难以理解，现在更多人习惯使用接近日常使用的自然语言和数学语言作为语言的表达式，便于理解和记忆，学习和使用起来十分方便，这种语言称为“高级语言”。

C 语言是国际上流行的、很有发展前途的计算机高级语言，适合作为“系统描述语言”，既可以用来编写系统软件，也可以用来编写应用程序。

### 2. C 语言的发展

C 语言是在 B 语言的基础上发展起来的，图 1-1 给出几种主要语言的派生关系。

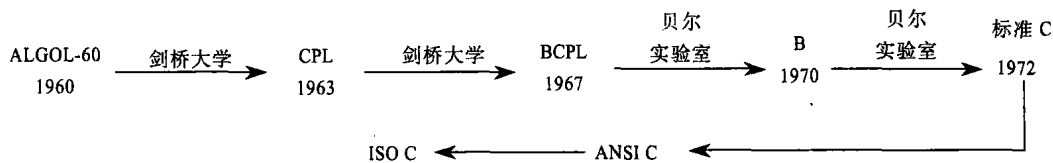


图 1-1 C 语言发展过程

ALGOL-60: 一种面向问题的高级语言。ALGOL-60 离硬件较远, 不适合编写系统程序。

CPL (combined programming language, 组合编程语言): CPL 是一种建立在 ALGOL-60 基础上更接近硬件的语言。CPL 规模大, 实现困难。

BCPL (basic combined programming language, 基本组合编程语言): BCPL 是对 CPL 进行简化后的一种语言。

C 语言: 在 B 语言基础上增加数据类型而设计出的一种语言。C 语言取 BCPL 的第二个字母。C 语言诞生后, UNIX 很快用 C 语言进行了改写, 并被移植到其他计算机系统。

标准 C、ANSI C、ISO C: C 语言的标准化, 本书的叙述基本上以 ANSI C 为基础。

C 语言从一开始就是用于编写大型、复杂系统软件的, 当然 C 语言也可以用来编写一般的应用程序。

## 1.2 C 语言特点

C 语言从“组合编程语言”CPL 发展而来, 既具有一般高级语言特性 (ALGOL-60 带来的高级语言特性), 又具有低级语言特性 (BCPL 带来的接近硬件的低级语言特性)。

C 语言具有以下特点:

### 1. 语言简洁

C 语言语法简单、紧凑, 使用方便、灵活, 书写形式自由, 一共有 32 个关键字, 九种控制语句。

### 2. 数据类型丰富

C 语言具有整型、实型、字符型、数组类型、指针类型、结构体类型、组合体类型、枚举类型等数据类型, 能方便地构造复杂的数据结构 (如使用指针构造链表、树、栈等)。

### 3. 运算符丰富

① C 语言具有算术运算符 (\*、/、%、+、-)、关系运算符 (>、>=、<、<=、==、!=)、逻辑运算符 (!、&&、||)、按位运算符 (&、|、^、~、<<、>>) 等, 复合的赋值运算符 (+[-、\*、/、%]=[加等、减等、乘等、除等、求余等]、>>=[右移等]、<<=[左移等]、“&[^|]”=[按位与等、按位异或等、按位或等])、sizeof 运算符 (求字节数)。

② C 语言有条件运算符 “?:” 可代替简单的 if-else 语句。

### 4. 生成的代码质量高

C 语言编译后生成的目标代码小、质量高, 程序的执行效率高 (有资料显示只比汇编代码效率低 10%~20%)。

### 5. 具有良好的可移植性

C 语言编制的程序基本上不需要修改或只需要少量修改就可以移植到其他的计算机系统或其他的操作系统。

### 6. 具有结构化语言特征

C 语言具有结构化的控制语句 (if-else、switch-case、for、while、do-while), 函数是 C 语言程序的模块单位。

### 7. C 语言可以实现汇编语言的大部分功能

• C 语言可以直接操作计算机硬件如寄存器、各种外设 I/O 端口等。

- C语言的指针可以直接访问内存物理地址。
- C语言类似汇编语言的位操作可以方便地检查系统硬件的状态。
- C语言适合编写系统软件。

## 1.3 算 法

学习计算机程序设计语言的目的，是要用语言作为工具，设计出可供计算机运行的程序。

在拿到一个需要求解的问题之后，怎样才能编写出程序？除了选定合理的数据结构外，一般来说，十分关键的一步是设计算法，有了一个好的算法，就可以用任何一种计算机高级语言将其转换为程序。

### 1.3.1 算法特性

算法是指解决某个特定问题而采取的确定且有效的步骤。一个算法应当具有以下五个特性：

#### 1. 有穷性

一个算法应当包含有限个操作步骤。也就是说，每一步都在合理的时间内完成，在执行若干个操作步骤之后，算法将结束。

#### 2. 确定性

算法的每一条指令必须有确切的含义，不能有二义性，对于相同的输入必能得出相同的执行结果。

#### 3. 可行性

算法中指定的操作，都可以通过已经实现的基本运算执行有限次后实现一定的合理的功能。

#### 4. 有零个或多个输入

在计算机上实现的算法是用来处理数据对象的，在大多数情况下这些数据对象需要通过输入来得到。

#### 5. 有一个或多个输出

算法的目的是为了求“解”，这些“解”只有通过输出才能得到。

### 1.3.2 算法描述

算法可以用各种方法来描述。一般常用以下四种方法进行描述：

#### 1. 自然语言表示法

使用自然语言来描述算法通俗易懂、不受语法约束；缺点是不够直观形象，计算机不能识别执行。

#### 2. 伪代码表示法

伪代码是一种近似于高级语言但又不受语法约束的语言描述方式，在英语国家使用起来更为方便。

#### 3. 流程图示法

##### (1) 传统流程图

传统流程图由图 1-2 所示的几种基本图形组成。



图 1-2 传统流程符号及功能

由这些框和流程线组成的流程图表示的算法形象直观、简单方便，但是这种流程图对于流程线的走向没有任何限制，可以转向，在描述复杂的算法时所占篇幅较多，费时费力且不易阅读。

## (2) N-S 图（无流程线）

N-S 图是一种不允许破坏结构化原则的图形算法描述工具，又称盒图。

N-S 图有以下几个基本特点：

- ① 功能域比较明确，可以从框图中直接反映出来。
- ② 不可能任意转移控制，符合结构化原则。
- ③ 很容易确定局部和全程数据的作用域。
- ④ 很容易表示嵌套关系，也可以表示模块的层次结构。

三种基本程序结构（顺序结构、选择结构、循环结构）流程图分别如 1.4 节中的图 1-3、图 1-4、图 1-5、图 1-6 所示。

## 4. 高级语言表示法

以计算机能识别的接近人们习惯的自然语言和数学语言作为语言的表达式，便于理解和记忆。人们学习和使用起来十分方便的语言，即计算机语言表示法，也就是我们后面要学习的编写程序。

# 1.4 结构化程序设计和模块化结构

程序设计的基本过程可以理解为：程序设计 = 算法 + 数据结构 + 方法 + 工具。首先，问题分析，分析问题的性质、输入/输出数据、数学模型或常用的方法；其次，结构特性的设计，选择控制结构和数据结构；再次，算法的设计和流程的描述；最后，选择实现程序的工具。编写程序的基本要求是要掌握程序设计的方法：结构化程序设计和模块化程序设计。

## 1.4.1 结构化程序设计

结构化程序设计要求把程序的结构限制为顺序、选择和循环三种基本结构，以便提高程序的可读性。这种结构化程序具有以下两个特点：

① 以控制结构为单位，只有一个入口和一个出口，使各单位之间的接口比较简单，每个单位也容易被人们所理解。

② 缩小了程序的静态结构与动态执行之间的差异，使人们能方便、正确地理解程序的功能。

三种基本结构都可以用传统流程图和 N-S 图表示。

### 1. 顺序结构

顺序结构按语句在程序中出现的先后顺序逐条执行，没有分支，没有转移，也没有重复。顺序结构的结构化流程图如图 1-3 所示。

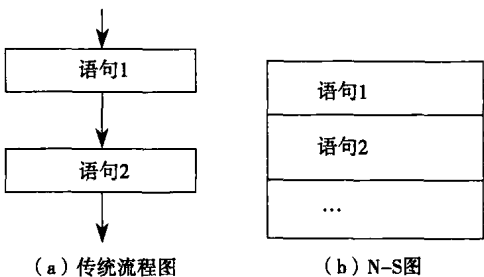


图 1-3 顺序结构的结构化流程图

## 2. 选择结构

选择结构又称分支结构，是判断条件是否成立并根据不同的条件执行不同分支中的语句，选择结构的结构化流程图如图 1-4 所示。

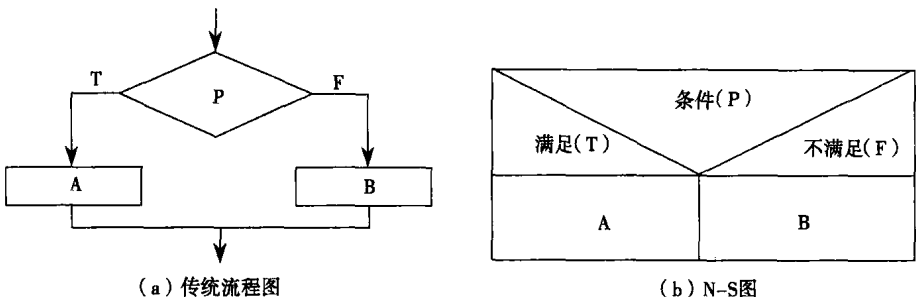


图 1-4 选择结构的结构化流程图

## 3. 循环结构

循环结构又称重复结构，即反复执行某一部分的操作，分为当型循环结构和直到型循环结构。

### (1) 当型循环结构

当型循环结构是当给定的条件 P 成立时反复执行 S，否则结束循环，执行下面的语句。当型循环结构的结构化流程图如图 1-5 所示。

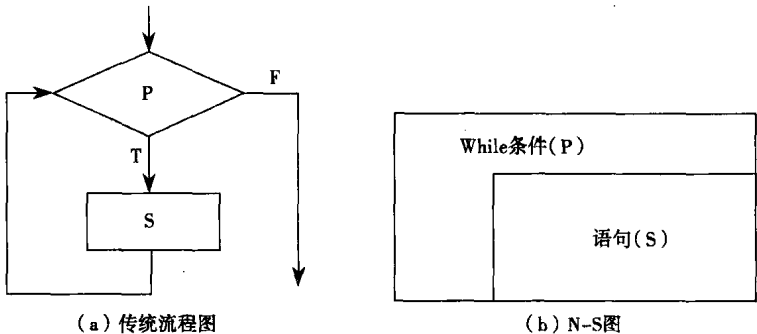


图 1-5 当型循环结构的结构化流程图

### (2) 直到型循环结构

直到型循环结构是先执行循环部分 S，然后判断条件 P，当条件不成立时执行循环体 S，条件成立时结束循环。直到型循环结构的结构化流程图如图 1-6 所示。本书中没有介绍直到型循环语句。

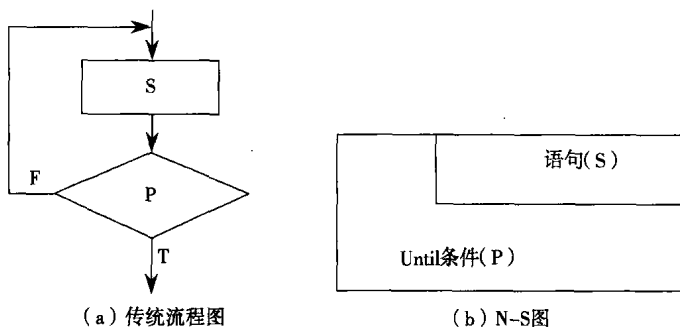


图 1-6 直到型循环结构的结构化流程图

图 1-7 是顺序输出 3~100 之间所有素数的结构化 N-S 流程图，在这个流程图中，表示了三种基本结构互相嵌套的情况。

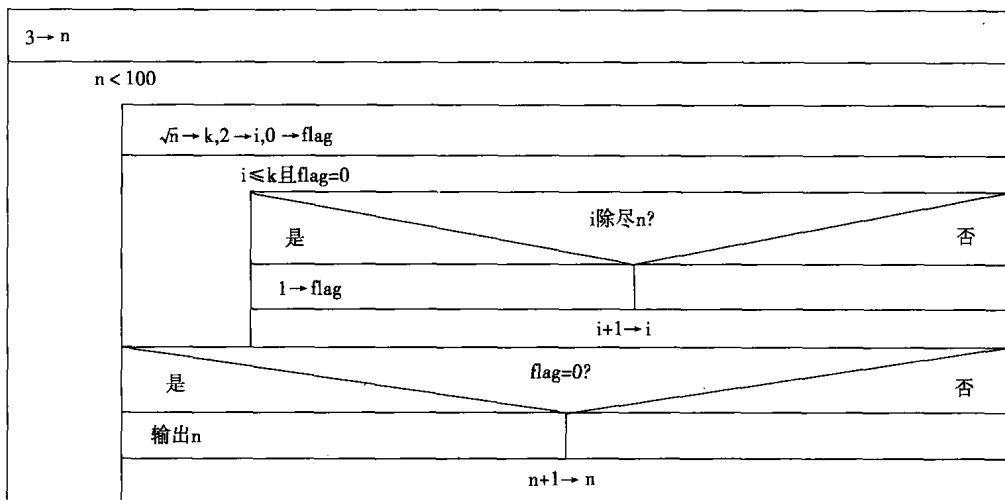


图 1-7 三种基本结构互相嵌套的情况

### 1.4.2 模块化程序设计

模块化设计是指把一个程序按人们能理解的大小规模进行分解。C 语言程序完全是由函数构成的，而且每个程序可由一个或多个函数组成。C 语言程序的函数化结构使得 C 语言程序非常容易实现模块化，便于阅读和维护。

#### 1. 按功能划分模块

- ① 模块间的接口关系比较简单，并且每个模块都是人所能及的。
- ② 各模块的功能比较单一，当需要修改某一功能时，一般只涉及一个模块，不会影响到其他模块。
- ③ 人们脱离程序的上下文也能单独地验证一个模块的正确性。
- ④ 在扩充系统或建立新系统时，可以充分利用已有的一些模块，用积木式的方法进行开发。

#### 2. 按层次组织模块

程序编制人员进行程序设计的时候，首先集中考虑主程序中的算法，写出主程序后再

逐步完成子程序的调用。对于这些“子”程序也可用调试主程序同样的方法逐步完成其下一层子程序的调用。这就是自顶向下、逐步细化、模块化的设计过程，它包括以下两个方面：

① 将一个复杂问题的解法分解和细化成由若干模块组成的层次结构。

② 将一个模块的功能逐步分解细化为一系列的处理步骤，直到某种程序设计语言的语句或某种机器指令。

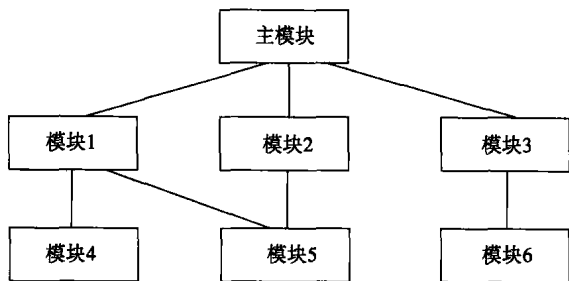


图 1-8 按层次组织模块

## 1.5 C 语言程序的编辑与运行

编写一个 C 语言程序到完成运行，一般要经过以下几个步骤：开机进入 C 语言编辑环境→输入与编辑源程序（.c）→对源程序进行编译产生目标代码（.obj）→连接各个目标代码、库函数生成可执行程序（.exe）→运行程序。

C 语言程序经过编辑、编译、连接到运行的全过程如图 1-9 所示。

早期的 C 语言编译环境是在 DOS 系统下的 Turbo C，随着 Windows 操作系统的普及与发展，人们越来越习惯使用 Windows 操作系统的窗口界面和鼠标进行操作。而 DOS 系统应用不太方便，使用者越来越少。下面简要介绍在 Windows 环境下使用 WIN-TC 编辑与运行 C 程序的步骤。

WIN-TC 是一个集成开发环境，它可以完成一个 C 程序的编辑、编译、连接和运行的全过程。

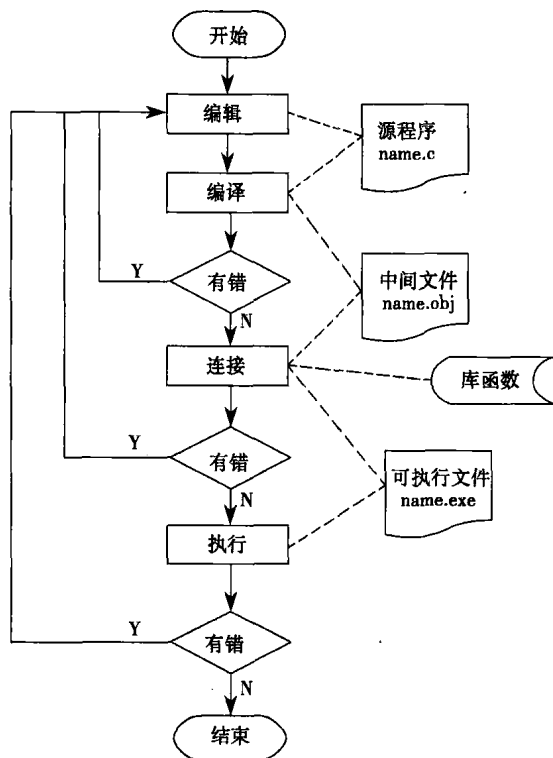


图 1-9 C 语言编辑运行步骤

WIN-TC 是 TC 2.0 在 Windows 平台下的开发工具。该软件以 TC 2.0 为内核, 提供 Windows 平台的开发界面, 因此也就支持 Windows 平台下的功能, 例如使用鼠标剪切、复制、粘贴和查找替换等, 而且在功能上也有它的独特之处, 例如语法加亮、C 内嵌汇编、自定义扩展库的支持等, 并提供一组相关辅助工具, 使用户在编程过程中更加游刃有余。WIN-TC 简、繁双语版可以正常运行于 Windows 98 及其以上的版本操作系统上。

### 1. 启动 WIN-TC

安装 WIN-TC 后, 在开始菜单的程序组里面将生成 WIN-TC 的程序组, 单击 WIN-TC 图标。WIN-TC 开始运行后, 在屏幕上显示如图 1-10 所示的主菜单窗口。

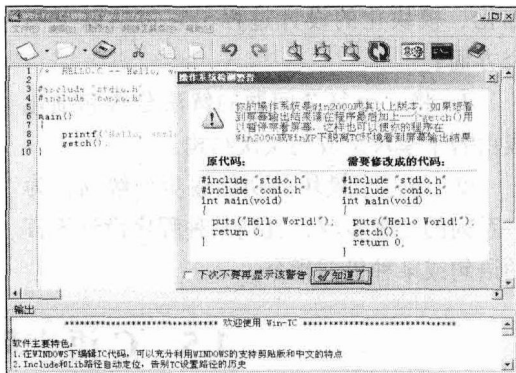


图 1-10 WIN-TC 开发集成环境

主菜单、常用菜单项及其主要功能如表 1-1 所示。

表 1-1 主要菜单及其主要功能

| 菜单选项名称 | 主要功能                   |
|--------|------------------------|
| 文件     | 新建、打开、保存文件, 退出编程环境     |
| 编辑     | 编辑程序文本、剪切、复制、粘贴、查找、替换等 |
| 运行     | 编译、连接、运行当前程序、编译配置等     |
| 超级工具集  | 英文 DOS 或中文 DOS 系统运行程序等 |
| 帮助     | WIN-TC 帮助文件、TC 教程等     |

### 2. 编辑源文件

如果系统是 Windows 2000 或 Windows XP, 将会先看到一个操作系统警告的对话框, 单击“知道了”按钮进入编辑状态, 在编辑窗口中输入程序, 在 main()函数的结尾处加上 getch()来暂停观看一下屏幕输出结果。WIN-TC 默认打开的文件是 WIN-TC 安装目录下的 noname.c, 可以在此基础上修改程序, 也可以新建自己的程序。

### 3. 编译源程序、运行程序

接下来就可以编辑或者编译 C 语言程序, 只有通过编译, 代码才能变成能够高效运行的软件。通过编译生成的文件扩展名是 .obj, 对于 TC 来说, 一般习惯是直接编译连接生成与 .c 文件同名的 .exe 文件。

工具栏里有两个按钮:  : 第一个是编译连接按钮, 第二个是编译连接并运行按钮 (还可以使用“运行”菜单中的命令)。这两个按钮都可以编译用户的代码, 所不同的是, 编译连接并运行按钮, 可以在编译后立即运行程序来检验是否是所期望的结果。

编译成功后出现“恭喜, 编译成功”信息提示, 如图 1-11 所示, 单击“确定”按钮就会看到程序的运行结果, 如图 1-12 按钮。如果程序有错误, 会在下面的提示信息框中提示错误类型信息, 可以编辑修改再编译运行, 直到输出正确结果。