



普通高等教育“十二五”创新型规划教材

C Yuyan  
Chengxu Sheji

# C语言程序设计

■ 主 编 郝桂英 赵敬梅  
■ 副主编 戴青云 苏 雪 刘 凤 赵裕民

普通高等教育“十二五”创新型规划教材

# C 语言程序设计

主 编 郝桂英 赵敬梅

副主编 戴青云 苏 雪 刘 凤 赵裕民

 **北京理工大学出版社**  
BEIJING INSTITUTE OF TECHNOLOGY PRESS

## 内 容 简 介

本书从高校学生的思维方式、理解能力等因素出发,以“实用、够用”为原则,用通俗易懂的语言和与实际密切相关的例题深入浅出地介绍 C 语言程序设计的基本概念和设计方法。全书共 9 章,第 1 章介绍了程序设计的基础知识;第 2 章介绍了 C 语言的数据类型、常量和变量、运算符、表达式及数据的输入与输出;第 3 章介绍了 C 语言程序的三种控制结构;第 4 章介绍了数组在 C 语言中的定义和使用;第 5 章介绍了 C 语言中函数的定义、调用、参数传递以及变量的作用域和存储类型;第 6 章介绍了指针的定义和使用方法;第 7 章介绍了两种构造类型的使用;第 8 章介绍了 C 语言的编译预处理命令;第 9 章介绍了 C 文件的概念及操作。

本书中的全部例题、习题和上机实训均在 Turbo C 及 Visual C++ 6.0 环境下调试、运行通过,便于读者直接上机验证。为了方便老师授课,本书还配有电子教案以及所有例题、习题和上机实训的源程序。本书既可以作为高等院校学生“C 语言程序设计”课程的理想教材,也可作为全国计算机等级考试二级 C 语言的培训或自学教材。

版权专有 侵权必究

---

### 图书在版编目(CIP)数据

C 语言程序设计 / 郝桂英,赵敬梅主编. —北京:北京理工大学出版社, 2010. 12

ISBN 978 - 7 - 5640 - 4214 - 1

I. ①C… II. ①郝… ②赵… III. ①C 语言—程序设计—高等学校—教材 IV. ①TP312

中国版本图书馆 CIP 数据核字(2011)第 011451 号

---

出版发行 / 北京理工大学出版社

社 址 / 北京市海淀区中关村南大街 5 号

邮 编 / 100081

电 话 / (010)68914775(办公室) 68944990(批销中心) 68911084(读者服务部)

网 址 / <http://www.bitpress.com.cn>

经 销 / 全国各地新华书店

印 刷 / 保定市中华美凯印刷有限公司

开 本 / 787 毫米 × 1092 毫米 1/16

印 张 / 16

字 数 / 372 千字

版 次 / 2010 年 12 月第 1 版 2010 年 12 月第 1 次印刷

印 数 / 1 ~ 2000 册

定 价 / 35.00 元

责任校对 / 王 丹

责任印制 / 边心超

---

图书出现印装质量问题,本社负责调换

# 前 言

C 语言是国内外广泛使用的一种计算机程序设计语言，其功能强大，使用方便、灵活，可移植性好，既具有高级语言的优点，又具有低级语言的许多特点，因此成为编制系统软件和应用软件的首选语言。

C 语言课程是我国各高校都开设的一门重要的基础课程，在高等院校计算机专业的课程体系中尤为重要，它是学习其他程序设计语言及专业课程的基础。在本书的编写过程中，我们针对高校学生自身的特点和培养目标，从高校学生的思维方式、理解能力等因素出发，以“实用、够用”为原则，结合多年从事 C 语言教学的经验，对全书的内容做了精心的安排，用通俗易懂的语言和与实际密切相关的例题深入浅出地介绍 C 语言程序设计的基本概念和设计方法。

本书在体系结构上尽可能地将概念、知识点与具体实例结合起来，同时借助于“说明”和“注意”等提示内容，帮助学生准确理解相关教学内容。另外，每章后面都配有一定数量的与所讲述内容以及计算机二级等级考试相匹配的习题和上机实训，借助于每章的习题帮助学生加深对教学内容的理解和掌握，借助于上机实训提高学生 C 语言编程的实际动手能力。

本书全面介绍了 C 语言的概念、特点和结构化程序设计方法。全书共 9 章，第 1 章介绍了程序设计的基础知识，并对 C 语言的基本知识和操作方式进行了简单介绍；第 2 章介绍了 C 语言的数据类型、常量和变量、运算符、表达式及数据的输入与输出；第 3 章介绍了 C 语言程序的 3 种控制结构：顺序结构、分支结构、循环结构；第 4 章介绍了一维、二维数组和字符数组在 C 语言中的定义和使用；第 5 章介绍了 C 语言中函数的定义、调用、参数传递以及变量的作用域和存储类型；第 6 章介绍了指针的定义和使用方法；第 7 章介绍了两种构造类型的使用：结构体与共用体，并简单讲述了枚举类型和用 typedef 进行类型定义的相关知识；第 8 章介绍了 C 语言的编译预处理命令；第 9 章介绍了 C 文件的概念以及文件的打开、读写、定位、关闭等操作。

本书的编写，参考了国家教育委员会考试中心编写的《全国计算机等级考试考试大纲》中的二级考试大纲中“C 语言程序设计考试要求”，以及部分省市计算机应用知识和应用能力水平考试大纲对 C 语言部分的要求。

本书中的全部例题、习题和上机实训均在 Turbo C 及 Visual C++ 6.0 环境下调试、运行通过，便于读者直接上机验证。为了方便老师授课，本书还配有电子教案以及所有例题、习题和上机实训的源程序。本书既可以作为高等院校学生“C 语言程序设计”课程的理想教材，也可作为全国计算机等级考试二级 C 语言的培训或自学教材。

本书由郝桂英、赵敬梅老师担任主编，戴青云、苏雪、刘凤、赵裕民担任副主编。郝桂英编写第1章、第5章、第6章，赵敬梅编写第2章、第3章、第4章，戴青云编写第7章，苏雪编写第8章，刘凤编写第9章，赵裕民编写附录I～附录IV。全书由郝桂英拟定编写大纲并负责统稿。

在本书的编写过程中，得到了许多从事计算机教学工作的同事的帮助和大力支持，他们对本书提出了很多宝贵的建议，在此向他们表示衷心的感谢。

尽管我们做了大量的工作，但由于我们水平有限，书中难免存在不足和疏漏之处，敬请广大读者不吝赐教。

编 者

# 目 录

|                           |     |
|---------------------------|-----|
| 第1章 概述                    | 1   |
| 1.1 程序设计基础知识              | 1   |
| 1.2 C语言概述                 | 8   |
| 1.3 本章小结                  | 14  |
| 习题                        | 15  |
| 上机实训                      | 16  |
| 第2章 数据类型、运算符、表达式及数据的输入/输出 | 20  |
| 2.1 C语言的数据类型              | 20  |
| 2.2 常量与变量                 | 21  |
| 2.3 基本数据类型的常量与变量          | 23  |
| 2.4 变量赋初值                 | 28  |
| 2.5 运算符和表达式               | 28  |
| 2.6 数据类型转换                | 39  |
| 2.7 数据的输入与输出              | 41  |
| 2.8 本章小结                  | 47  |
| 习题                        | 48  |
| 上机实训                      | 51  |
| 第3章 程序的控制结构               | 53  |
| 3.1 C语句概述                 | 53  |
| 3.2 赋值语句                  | 54  |
| 3.3 顺序结构程序设计              | 55  |
| 3.4 分支结构程序设计              | 56  |
| 3.5 循环结构程序设计              | 68  |
| 3.6 综合应用程序举例              | 78  |
| 3.7 本章小结                  | 79  |
| 习题                        | 80  |
| 上机实训                      | 86  |
| 第4章 数组                    | 88  |
| 4.1 数组概述                  | 88  |
| 4.2 一维数组的定义和引用            | 89  |
| 4.3 二维数组的定义和引用            | 93  |
| 4.4 字符数组                  | 96  |
| 4.5 本章小结                  | 104 |

|                      |            |
|----------------------|------------|
| 习题                   | 104        |
| 上机实训                 | 108        |
| <b>第 5 章 函数</b>      | <b>110</b> |
| 5.1 函数概述             | 110        |
| 5.2 函数定义             | 111        |
| 5.3 函数调用             | 113        |
| 5.4 函数之间的数据传递        | 117        |
| 5.5 函数的嵌套调用和递归调用     | 123        |
| 5.6 变量的作用域和存储类型      | 127        |
| 5.7 内部函数和外部函数        | 136        |
| 5.8 本章小结             | 137        |
| 习题                   | 138        |
| 上机实训                 | 143        |
| <b>第 6 章 指针</b>      | <b>148</b> |
| 6.1 指针与指针变量的概念       | 148        |
| 6.2 指针变量的定义和引用       | 150        |
| 6.3 指针与数组            | 157        |
| 6.4 指针与字符串           | 166        |
| 6.5 指针与函数            | 171        |
| 6.6 二级指针             | 174        |
| 6.7 本章小结             | 176        |
| 习题                   | 177        |
| 上机实训                 | 181        |
| <b>第 7 章 结构体与共用体</b> | <b>184</b> |
| 7.1 结构体类型与结构体变量      | 184        |
| 7.2 结构体数组            | 190        |
| 7.3 结构体指针            | 191        |
| 7.4 共用体              | 195        |
| 7.5 枚举类型             | 198        |
| 7.6 用 typedef 定义类型   | 200        |
| 7.7 本章小结             | 201        |
| 习题                   | 201        |
| 上机实训                 | 205        |
| <b>第 8 章 预处理命令</b>   | <b>206</b> |
| 8.1 宏定义              | 206        |
| 8.2 文件包含             | 211        |
| 8.3 条件编译             | 213        |
| 8.4 本章小结             | 214        |
| 习题                   | 215        |

---

|                              |            |
|------------------------------|------------|
| 上机实训                         | 218        |
| <b>第9章 文件</b>                | <b>220</b> |
| 9.1 C语言文件的概念                 | 220        |
| 9.2 文件类型指针                   | 222        |
| 9.3 文件的打开与关闭                 | 223        |
| 9.4 文件的读写                    | 225        |
| 9.5 文件的定位和检测                 | 231        |
| 9.6 本章小结                     | 233        |
| 习题                           | 234        |
| 上机实训                         | 235        |
| <b>附录 I 常用字符与 ASCII 码对照表</b> | <b>237</b> |
| <b>附录 II 关键字及其用途</b>         | <b>239</b> |
| <b>附录 III 运算符的优先级和结合方向</b>   | <b>240</b> |
| <b>附录 IV C语言常用语法提要</b>       | <b>242</b> |
| <b>参考文献</b>                  | <b>246</b> |

# 第1章 概述

C语言是目前流行的一种结构化程序设计语言，既适合于编写系统软件，又适合于编写应用软件。

作为本书的开篇，本章首先介绍程序设计的基本概念和基础知识，带领读者迈入程序设计的大门，在此基础上对C语言这种应用广泛的计算机高级程序设计语言进行进一步探讨。

## 【学习目标】

- (1) 掌握程序、程序设计、算法的基本概念。
- (2) 了解程序语言的发展和程序设计的步骤。
- (3) 掌握算法的基本特征和描述方法。
- (4) 掌握结构化程序设计方法。
- (5) 了解C语言的发展特点和C语言的字符集、标识符。
- (6) 掌握C语言程序的基本构成和开发流程。

## 1.1 程序设计基础知识

### 1.1.1 程序设计基本概念

#### 1. 程序和程序设计语言

计算机是一种能高速运算、具有存储和记忆能力的电子设备。它最本质的使命就是执行指令所规定的操作。如果我们需要计算机完成什么工作，只要将其步骤用一条条指令的形式描述出来，并将其存入计算机的存储器中，需要结果时就向计算机发出一个简单的命令，计算机就会自动按顺序执行这些指令，全部指令执行完后就得到了预期的结果。这种可以被连续执行的一系列指令的集合就称为计算机的程序。

众所周知，计算机中的指令都是二进制编码，用它编写程序既难记忆又难掌握，所以计算机工作者就研制出了各种计算机能够懂得、人们又方便使用的计算机语言，程序就是利用计算机语言来编写的。因此，计算机语言通常被称为“程序设计语言”，一个计算机程序总是用某种程序设计语言书写的。

从计算机问世以来，程序设计语言也伴随着计算机技术的发展而不断变化。大体经历了以下几个过程。

(1) 机器语言：这是最早产生和使用的程序语言，又称为第一代计算机语言。其每条指令都是由二进制代码0和1组成的一个序列。任何一种计算机，都有各自的一套指令系

统，每套指令系统都由一组指令组成。例如某指令系统有如下两条指令。

①10000000 表示两数相加。

②10010000 表示两数相减。

由于计算机的指令系统往往各不相同，所以，在一台计算机上执行的程序要想在另一台计算机上执行，必须另编程序，这就造成重复工作。另外，使用这种语言编写程序往往相当困难，只有少数人能掌握。但是，因为机器语言使用的是针对特定型号计算机的语言，故其运算效率是所有语言中最高的。

(2) 汇编语言：这是第二代计算机语言。为了减轻使用机器语言编程的负担，人们进行了一种有益的改进：用一些简洁的英文字母、符号串等助记符来替代特定指令的二进制串，比如用 ADD 表示加法，用 SUB 表示减法等，这样一来，就使人们更容易理解和编写程序了。

这种用一些简单助记符构成的指令系统就称为汇编语言。但是计算机是不认识这些助记符的，这就需要有一个工具，将汇编语言程序翻译成二进制的机器语言，这种工具就称为汇编程序。

使用汇编语言编程效率仍十分高。针对计算机特定硬件而编制的汇编语言程序，能准确地发挥计算机硬件的功能和特长，所以至今仍是一种常用的软件开发工具。但汇编语言同样依赖于机器硬件，移植性不好，因此它和机器语言都属于面向机器的语言。

(3) 高级语言：随着计算机技术的发展，人们开始意识到应该设计一种这样的语言：这种语言接近于数学语言或人的自然语言，同时又不依赖于计算机硬件，编出的程序能在所有机器上使用。经过努力，1954 年，第一个完全脱离机器硬件的高级语言——FORTRAN 问世了。50 多年以来，高级语言的发展也经历了从面向过程的语言到面向对象的语言的发展。影响较大的面向过程的语言有 FORTRAN、COBOL、BASIC、Pascal、C 等。用面向过程的语言解决问题时，人们首先要理解问题要求我们“做什么”，然后去构造“怎么做”的解题过程。程序设计者要详细规定计算机操作的每个细节。而面向对象的语言只需告诉计算机“做什么”，无需去构造“怎么做”的过程。如 C++、VC、VB、Delphi、Java 等均属于此类语言。这种语言使用起来更为简便，但是它的运行效率和灵活性都不如面向过程的语言。另外，对于程序设计的初学者，面向过程的语言是程序设计的基础，因此本书将以面向过程的 C 语言为背景介绍程序设计的基本概念和方法。

## 2. 程序设计步骤

使用计算机解决问题，必须从问题描述入手，经过解题过程的分析、算法的设计，直到最后程序的编写、调试和运行等一系列过程，最终得到要求解问题的结果，这一过程称为程序设计。一般一个简单的程序设计包含 4 个步骤。

### 1) 分析问题，建立数学模型

使用计算机解决具体问题时，首先要对问题进行充分分析，确定问题是什么，解决问题的步骤又是什么，然后针对所要解决的问题找出已知的数据和条件，确定所需的输入、处理和输出对象，最后将解题过程归纳为一系列的数学表达式，建立起解决问题的数学模型。模型的好坏在很大程度上决定了程序的正确性和复杂度。

### 2) 确定数据结构和算法

根据建立的数学模型，对指定的输入数据和预期的输出结果，确定存放数据的数据结

构,在此基础上选择合适的算法加以实现。

### 3) 编写程序

根据确定的数据结构和算法,用某种程序语言把这个解决方案完整地描述出来,也就是编写出程序代码。

### 4) 调试运行程序

在计算机上用实际的输入数据对编好的程序调试,分析得到的运行结果,进行程序的测试和调整,直到得到预期的结果。

由此可见,一个完整的程序要涉及4方面的问题:数据结构、算法、程序设计语言和程序设计方法。这4个方面的知识都是程序设计人员必备的。关于数据结构的知识有专门的著作,本书重点介绍算法、程序设计语言和程序设计方法的相关知识。

## 1.1.2 算法和算法的描述

### 1. 算法的概念

所谓算法,就是指解决某一问题的方法和步骤。根据制定的算法,编写出计算机可以执行的命令序列,就是编制程序。

一个完整的算法应具有如下5个特征,这5个特征也是算法的判断标准。

#### 1) 有穷性

任何一个算法都必须能在执行有限步之后结束。例如:

$$N! = 1 * 2 * 3 * \dots * (N-1) * N$$

其中N是一个特定数,比如50,则这个描述可称之为一个算法。而下式:

$$\text{sum} = 1+2+3+\dots+N+\dots$$

就不能称之为算法,因为该式在执行有限步后仍不能结束,它只能被称为一个计算方法,因此,计算方法不等同于算法。

#### 2) 确定性

算法中每一步骤的含义必须是确切的,不能出现任意二义性。

#### 3) 有效性

算法中的每一个步骤都应当是可以被执行的,并能得到正确的结果。

#### 4) 有0个或多个输入

所谓输入是指在算法开始之前所需要的初始数据。输入的多少取决于特定的问题,有些特殊的算法可以没有输入。

#### 5) 有1个或多个输出

算法既然是为解决特定问题而设计的,那么它至少包含一个输出结果。无任何输出信息的算法是无意义的。

下面通过3个简单的问题说明算法的特性。

【例 1-1】有红和黑两个墨水瓶,但错把红墨水装在了黑墨水瓶里,而黑墨水错装在了红墨水瓶里,要求其交换。

算法分析:这是一个非数值运算问题。因为两个瓶子的墨水不能直接交换,所以,解决这一问题的关键是需要引入第3个墨水瓶。设第3个墨水瓶是白色的,其交换步骤如下。

- (1) 将黑瓶中的红墨水装入白瓶中。
- (2) 将红瓶中的黑墨水装入黑瓶中。
- (3) 将白瓶中的红墨水装入红瓶中。
- (4) 输出红瓶和黑瓶中的结果，交换结束。

【例 1-2】设计一个算法，让计算机来解方程： $ax+b=0$ ，其中参数  $a$ ， $b$  由键盘任意输入，最后计算机输出结果。

算法分析：这是一个数值运算问题。对于该问题，我们不能简单地让计算机输出“ $x=-b/a$ ”的结果，因为当  $a=0$  时，这种输出是错误的。所以我们需要分情况讨论。其具体步骤如下。

- (1) 输入  $a$ ， $b$ 。
- (2) 若  $a \neq 0$ ，则输出  $x=-b/a$ ，执行第 (4) 步，否则执行第 (3) 步。
- (3) 若  $a=0$ ，则分两种情况：
  - ① 若  $b=0$ ，方程的解是全体实数，执行第 (4) 步；
  - ② 若  $b \neq 0$ ，方程没有实数解，执行第 (4) 步。
- (4) 算法结束。

【例 1-3】设计一个算法，求两个自然数  $M$  和  $N$  的最大公约数。

算法分析：这也是一个数值运算问题。我国古代数学家秦九韶在《算书九章》一书中曾记载了这个算法。 $M$  和  $N$  的最大公约数就是能同时整除  $M$  和  $N$  的最大正整数。一般用辗转相除法求解。其具体步骤如下。

- (1) 输入两个确定的自然数  $M$  和  $N$ 。
- (2) 判断  $M$  是否等于  $N$ ，如相等，则执行第 (5) 步，否则执行第 (3) 步。
- (3) 当  $M$  大于  $N$  时，从  $M$  中减去  $N$ ，即  $M$  减去  $N$  后再将结果赋给  $M$ ；否则，从  $N$  中减去  $M$ ，即  $N$  减去  $M$  后再将结果赋给  $N$ 。
- (4) 回到第 (2) 步。
- (5) 输出结果， $N$  为所求最大公约数，算法结束。

## 2. 算法的描述

描述算法有多种不同的工具，采取不同的描述工具对算法的质量有很大的影响。常见的描述算法的工具具有：自然语言、流程图、N-S 图、PAD 图、伪代码等。这一节主要介绍应用最广泛的流程图和 N-S 图。

### 1) 流程图

流程图是一种用图框来表示各种操作步骤的算法描述工具，这是一种描述算法的传统工具。美国国家标准化协会 ANSI 规定了一些常用的流程图符号，如图 1-1 所示。

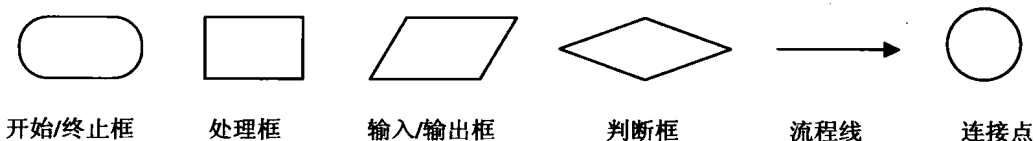


图 1-1 流程图中的各种符号

### 2) N-S 图

随着结构化程序设计方法的出现，两个美国学者 Nassi 和 Shneiderman 在 1973 年提出

了一种新的流程图形式，即著名的盒图，又称为 N-S 图（N-S 取自两学者名字的第一个字母）。它和传统流程图的区别在于去掉了流程线，算法的每一步都用一个矩形盒来描述（盒图因此而得名），把一个个矩形盒按执行的次序连接起来就是一个完整的算法描述。

在本章 1.1.3 节中将会看到这两种算法描述工具的基本形式。

### 1.1.3 结构化程序设计方法

程序设计方法是程序设计人员必须掌握和理解的内容。现在广泛使用的程序设计方法主要有结构化程序设计方法和面向对象的程序设计方法。由于 C 语言是一种面向过程的设计语言，利用这种语言编写程序时采用的是结构化的程序设计方法，所以本书对结构化程序设计方法作一个简单的介绍。

结构化程序设计的本质是对问题进行分解。也就是说，从代表目标系统整体功能的复杂问题着手，自顶向下不断地把复杂的问题分解成简单的子问题，这样一层一层地分解下去，直到只剩下若干个容易实现的子问题为止。当所分解出的子问题已经十分简单，其功能显而易见时，就停止这种分解过程，并写出各个最底层子问题的处理描述。然后再用结构化编程技术编制出各个子问题的程序块，进而构造出整个问题的求解程序。

结构化的程序设计方法一般包括以下几个特征。

（1）整个程序采用模块化结构，用自顶向下、逐步求精的方式进行设计。

（2）设计程序时只采用 3 种基本的程序控制结构来编制程序。这 3 种基本程序控制结构分别是顺序结构、选择结构和循环结构。

下面对结构化程序设计时所需要的 3 种基本程序控制结构进行简单介绍。

#### 1. 顺序结构

这种结构程序的执行是按照程序中语句书写的先后顺序逐条执行的，没有分支，没有转移。顺序结构可用图 1-2 表示，其中左边是传统的流程图，右边是 N-S 流程图。

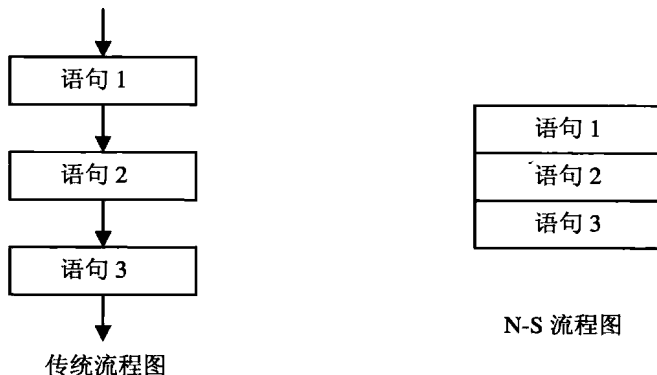


图 1-2 顺序结构

#### 2. 选择结构

选择结构又称为分支结构。此结构中必须包含一个条件判断，程序执行时根据条件判断的结果来决定其走向，它向我们提供了根据条件取值来选择不同分支的方法。选择结构可用图 1-3 表示，其中左边是传统的流程图，右边是 N-S 流程图。

### 3. 循环结构

循环结构是一种根据某种条件对某一语句块反复执行若干次（可以是多次、一次或零次）的结构。

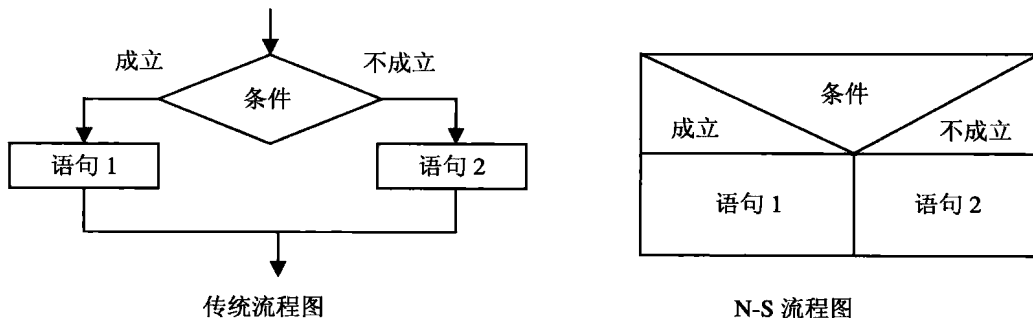


图 1-3 分支结构

循环结构又分为“当型循环”和“直到型循环”两种。“当型循环”的特点是先判断条件，后执行循环体；“直到型循环”是先执行循环体，后判断条件。图 1-4 和图 1-5 分别是“当型循环”和“直到型循环”的示意图。其中左边是传统的流程图，右边是 N-S 流程图。

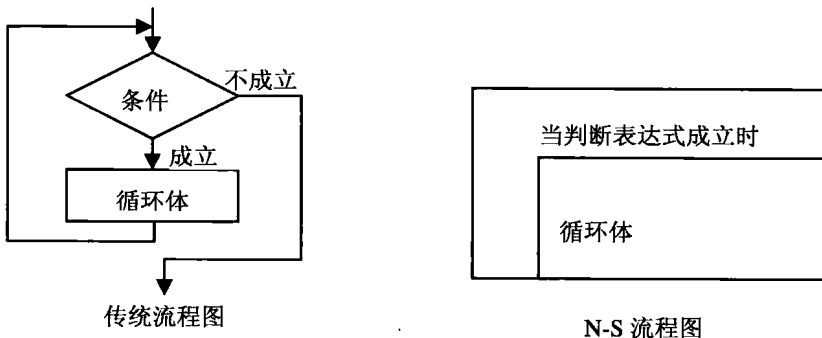


图 1-4 当型循环结构

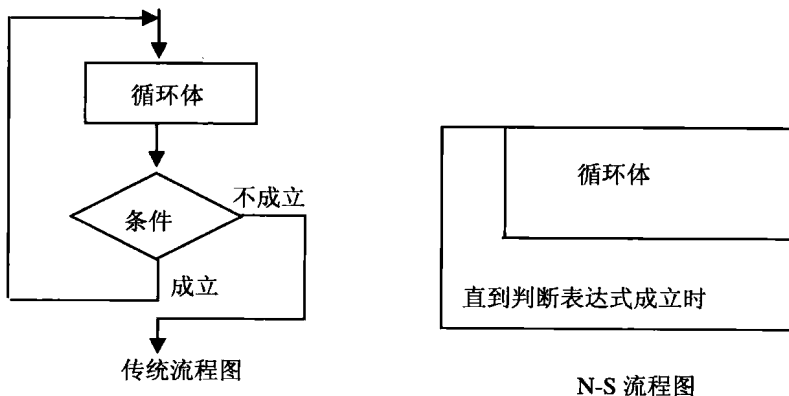


图 1-5 直到型循环结构

以上 3 种基本程序控制结构必须具有以下特点。

- (1) 每种基本结构必须只有一个入口和一个出口。
- (2) 每种基本结构都有一条从入口到出口的路径通过。
- (3) 结构内不允许出现“死循环”。

任何复杂问题都可以通过这3种基本结构的组合构成。由这2种基本结构所组成的算法称为结构化算法，由这3种基本结构所组成的程序称为结构化程序。

比如上述例1-1、例1-2和例1-3的算法，就可以用这3种基本结构的流程图表示（见图1-6、图1-7和图1-8）。

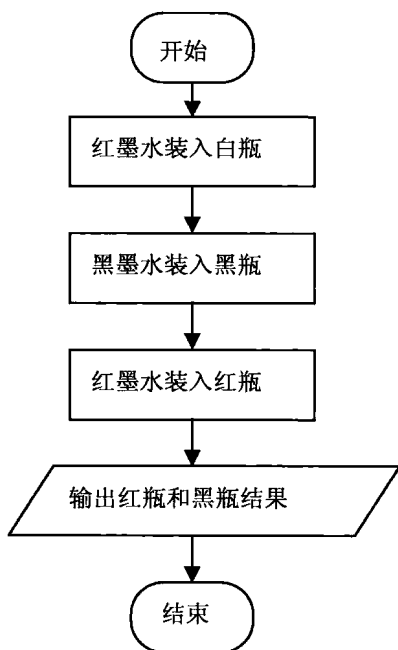


图 1-6 例 1-1 流程图

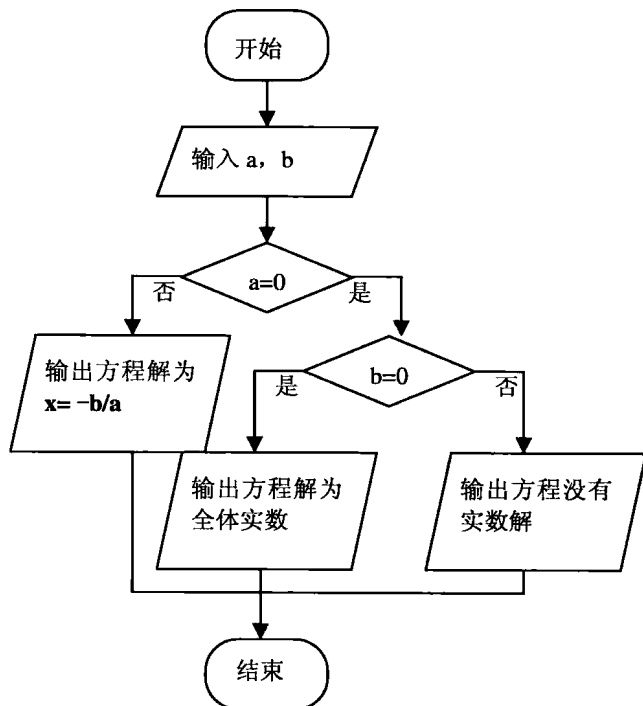


图 1-7 例 1-2 流程图

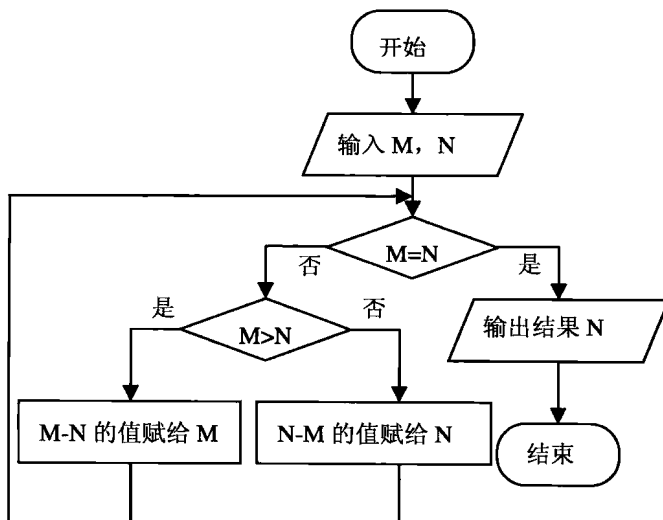


图 1-8 例 1-3 流程图

## 1.2 C 语言概述

### 1.2.1 C 语言的发展和特点

C 语言是一种面向过程的高级程序设计语言，它的根源是 ALGOL 60，1960 年出现的 ALGOL 60 具有可读性和可移植性好等特点，但它不能直接对硬件进行操作；不宜用来编写系统程序。1963 年，英国剑桥大学和伦敦大学将 ALGOL 60 语言发展成为 CPL 语言，1967 年，剑桥大学的 Martin Richards 将 CPL 改制成 BCPL 语言。1970 年，美国贝尔实验室的 Ken Thompson 在 BCPL 语言的基础上，设计出了 B 语言（B 语言取名自 BCPL 语言的第一个字母），并用 B 语言开发出了第一个高级语言的 UNIX 操作系统。之后，美国贝尔实验室的 D. M. Ritchie 为了更好地描述和实现 UNIX 操作系统，在 B 语言的基础上开发出了 C 语言（C 语言取名自 BCPL 语言的第二个字母）。C 语言既保持了 BCPL 和 B 语言的优点（精练，接近于硬件），又克服了它们的缺点（过于简单，数据无类型等）。到了 1973 年，Ken Thompson 和 D. M. Ritchie 两个人合作把 UNIX 90% 以上的内容用 C 语言进行了改写，即 UNIX 第五版。可以说，C 语言是为开发 UNIX 操作系统而研制的，它随着 UNIX 的出名而闻名，因 UNIX 的巨大成功而得到了广泛的使用。另外，C 语言的广泛应用又促使其新版本不断出现，其性能也越来越强。到了 1975 年，随着 UNIX 第六版的推出和面向对象程序设计技术的出现，C 语言的突出优点引起了人们的普遍关注。1978 年，Brian W. Kernighan 和 D. M. Ritchie 合写了 C 语言的经典著作《C 程序设计语言》，奠定了 C 语言完整而坚实的基础。1983 年，美国国家标准化协会制定了 ANSI C 草案，形成了 C 语言的标准。以后 C 语言又几经修改和完善，发展到了目前可在微机上运行的 Microsoft C/C++，Turbo C，Quick C，Borland C，Visual C/C++ 等版本。

目前，不同类型的计算机和不同的操作系统，都可以应用 C 语言进行程序设计。由于 C 语言本身具有许多优点，现在它已经成为在各种计算机上进行各种程序设计，并在各方面都得到广泛使用的程序设计语言。

C 语言之所以能够在与众多的高级语言竞争中脱颖而出，成为高级语言的佼佼者，主要是因为与其他高级语言相比，它具有以下特点。

#### 1. 语言简洁灵活

C 语言只有 32 个标准的关键字，9 种程序控制语句，程序书写形式自由，主要用小写字母表示。压缩了其他高级语言中的冗余部分。

#### 2. 运算符丰富

C 语言提供了 45 种标准的运算符，运算功能十分强大，它把圆括号、逗号、等号都作为运算符来处理，从而使其运算类型极其丰富，可以实现在其他高级语言中难以实现的运算。

#### 3. 数据结构丰富

C 语言除提供整型、实型、字符型等基本数据类型外，还提供了用基本数据类型构造出的各种复杂的数据结构，如数组、结构体、共用体等。另外，C 语言还引入了与地址密

切相关的指针类型，使得 C 语言的计算功能、逻辑判断功能非常强大。

#### 4. 允许直接访问物理地址

C 语言中的位运算符和指针运算符能够直接对内存地址进行访问操作，可以实现汇编语言的大部分功能，即直接对硬件进行操作。

#### 5. 生成的目标代码质量高

C 语言提供了一个相当大的运算符集合，而且其中大多数运算符与一般机器指令相一致，可直接翻译成机器代码，因此，用 C 语言编写的程序生成的代码质量高，从而带来了编译和执行的高效率。C 语言的代码效率只比汇编语言低 10% ~ 20%，而比其他高级语言的代码效率都高。

#### 6. 可移植性好

C 语言所提供的与硬件有关的操作，如数据的输入/输出等，都是通过调用系统提供的库函数来实现的。库函数本身不是 C 语言的组成部分，因此用 C 语言编写的程序能够很容易地从一种计算机环境移植到另一种计算机环境中。

C 语言是一种很灵活的程序设计语言，在很多方面都没有过多的限制。这对程序设计者来说是很方便的，非常适合对 C 语言很熟悉的程序员的胃口。但任何事物都是一分为二的，C 语言的过于灵活和较少的限制，对于 C 语言的初学者来说，却存在着不安全的因素。因此，在使用 C 语言进行程序设计时，一定要注意这一点。

### 1.2.2 C 语言的字符集和标识符

C 语言根据其特点，规定了其所需的字符集和标识符。

#### 1. 字符集

C 程序中允许出现的所有基本字符的组合称为 C 语言的字符集。C 语言的字符集如下所述。

- (1) 字母：小写字母 a~z 有 26 个，大写字母 A~Z 也有 26 个，共 52 个。
- (2) 数字：阿拉伯数字 0~9 共 10 个。
- (3) 特殊符号：共 28 个，如表 1-1 所示。

表 1-1 C 语言编程中可以使用的特殊符号

|   |   |   |   |   |   |      |
|---|---|---|---|---|---|------|
| + | - | * | / | % | ^ | =    |
| ~ | ( | ) | [ | ] | { | }    |
| ; | " | ' | : | . | ! | 空格   |
| < | > | # | & | ? |   | _下划线 |

#### 2. 标识符

C 语言程序中出现的任何对象（比如常量、变量、函数、数组、文件、类型）一般都要有一个“名字”，这些对象的“名字”就是 C 语言的标识符。

C 语言的标识符包括 3 类：保留字、预定义标识符和用户定义标识符。下面就对其分别加以介绍。