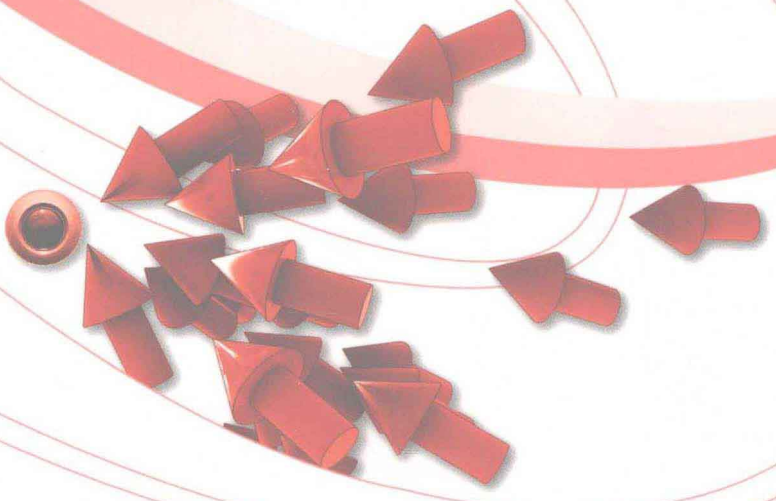

高等学校计算机规划教材

C语言程序设计教程（第2版）

孟宪福　王　旭　编著

電子工業出版社
PUBLISHING HOUSE OF ELECTRONICS INDUSTRY
http://www.phei.com.cn

高等学校计算机规划教材

C语言程序设计教程

(第2版)

孟宪福　王　旭　编著

電子工業出版社
Publishing House of Electronics Industry
北京 · BEIJING

内 容 简 介

本书是大连理工大学精品课程教学成果。本书共 12 章，按照循序渐进的原则，详细地介绍了 C 语言的基本概念和语法规则：数据、运算符、表达式、数据输入输出、基本语句、数组、函数、编辑预处理、结构和联合、位运算、指针、文件操作。最后一章详细分析了一个图形编辑程序的设计与实现过程，并给出完整的源程序。本书在难点的讲解上力求准确、完整。通过精选的典型例题分析，使得读者能够尽快掌握利用 C 语言进行程序设计的技巧和方法。本书配有 PPT、源代码、习题答案等教学资源。

本书可作为高等院校计算机及相关专业 C 语言程序设计的教材或教学参考书，也可作为自学用书。

未经许可，不得以任何方式复制或抄袭本书之部分或全部内容。
版权所有，侵权必究。

图书在版编目(CIP)数据

C 语言程序设计教程 /孟宪福，王旭编著—2 版. —北京：电子工业出版社，2010.9
高等学校计算机规划教材
ISBN 978-7-121-11611-7

Ⅰ. ①C… Ⅱ. ①孟…②王… 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(2010)第 161095 号

策划编辑：史鹏举 杨 寰
责任编辑：史鹏举
印 刷：
装 订：北京市李史山胶印厂
出版发行：电子工业出版社
北京市海淀区万寿路 173 信箱 邮编：100036
开 本：787×1092 1/16 印张：14.75 字数：378 千字
印 次：2010 年 9 月第 1 次印刷
定 价：25.00 元

凡所购买电子工业出版社图书有缺损问题，请向购买书店调换。若书店售缺，请与本社发行部联系，联系及邮购电话：(010) 88254888。

质量投诉请发邮件至 zlts@phei.com.cn，盗版侵权举报请发邮件至 dbqq@phei.com.cn。

服务热线：(010) 88258888。

前　言

C 语言是目前应用最为广泛的计算机高级程序设计语言之一，它短小精悍、功能齐全，是一种结构化程序设计语言，能够运行于多种操作系统环境下，既适合于编写应用软件，又适合于编写系统软件。

作者多年来一直从事 C 语言的教学工作，同时也利用 C 语言开发大型的实际课题。本书的内容吸收了作者多年的教学经验和应用 C 语言的体会，既注重于 C 语言的理论体系，又特别强调 C 语言的应用。

本书是在前一版的基础上，广泛听取读者和同行的建议，并参考最新材料经系统整理而成的。本书的主要特点可归纳如下：

1．在内容编排上，按照循序渐进的原则，逐步介绍 C 语言中的基本概念和理论，在章节内容的安排上，尽可能考虑初学者的接受能力，使整个学习过程按照从简单到复杂的顺序进行。

2．指针是 C 语言中的关键内容，也是初学者难以理解的部分。本书用了大量篇幅由浅入深地介绍了指针的基本概念和应用。除了常用的一级指针之外，还详细地介绍了二级指针、二维数组与指针、二维数组与函数，以及对复杂定义形式的分析等比较深奥的内容，并给出了每种复杂定义形式的具体应用过程。通过对这些内容的学习，能够使读者完整、准确地掌握 C 语言中的精髓内容，从而能够有效地提高读者的语言运用能力和程序设计技巧。

3．在讲授 C 语言基本理论的基础上，着重强调了 C 语言的应用，书中没有深奥的理论和算法，例题中出现的每一个算法，都给出了比较详细的解释，每一章都包含“应用举例”一节，其中的例题涉及本章讲解的主要内容，有些例题还具有一定的难度，通过阅读和分析这些例题，能使读者对本章讲授的内容及应用有一个全面的了解。

4．为了使读者能够尽快掌握利用 C 语言编写大型实用程序的方法，第 12 章详细介绍了利用 C 语言编写的规模较大的图形编辑程序 Panda，在该程序中几乎包含了 C 语言的各个方面，包括菜单设计、图形处理(图形输入和图形编辑)及文件管理等，仔细阅读和分析此程序，无疑会在短期内提高利用 C 语言编写大型程序的能力。

5．每章都附有大量的习题，包括程序分析题和编程题，这些习题对于读者巩固已学习的内容大有益处。

6．在语言的描述上，尽量使用规范化的术语，同时在文字叙述上力求条理清晰、简单明了，以利于读者阅读。

作者认为，要学好 C 语言，除了掌握 C 语言的基本理论外，还必须加强实践环节。本书的所有例题都在微机上调试通过，读者可以边学习边上机，刚开始时可以调试书中的例题，待学习一段时间后，就可以调试自己编写的程序了，只有这样才能加快学习进度，提高学习效率。

本书配有电子课件、程序源代码、习题参考答案，需要者可从华信教育资源网 http://www.hxedu.com.cn 免费注册下载。

由于作者水平有限，经验不足，书中一定会有不少缺点和错误，敬请有关老师和广大读者批评指正。

咨询、意见和建议可反馈至本书责任编辑邮箱：shipj@phei.com.cn

编著者

于大连理工大学

试读结束，需要全本PDF请购买 www.ertongbook.com

目　录

第1章 绪　言

随着计算机硬件技术和软件技术的不断提高，作为人机交流主要工具的计算机程序设计语言也经历了从简单到复杂、从低级到高级的发展过程。在诸多的计算机高级语言当中，C语言是目前国内外最为流行的计算机高级程序设计语言之一，它设计精巧、功能齐全、使用灵活，既适合于编写应用软件，又适合于编写系统软件。

C语言最初是为描述和实现UNIX操作系统而设计和实现的。在此以前像UNIX操作系统这样的系统软件，一般都是利用汇编语言这种低级语言来编写的，自C语言出现以来，使得利用高级语言编写系统软件成为可能。UNIX操作系统90%以上的源代码是由C语言编写的，UNIX操作系统的一些主要特点，如便于理解、易于修改及具有良好的可移植性等，在一定程度上受益于C语言，所以，UNIX操作系统的成功与C语言是密不可分的。最初的C语言是依赖于UNIX操作系统环境的，随着C语言的不断发展及应用的普及，目前，C语言已经能够在多种操作系统如UNIX、Windows和DOS等环境下运行，而且，实用的C语言编译系统种类繁多，适用于PC运行的有Microsoft C(MSC)、Turbo C(TC)、Borland C(BC)和Lattice C(LC)等。

1.1 C语言简介

1.1.1 C语言发展简史

20世纪60年代，随着计算机科学体系的形成与完善，高级程序设计语言的研究得到了长足的发展，但是，在当时出现的高级语言中，缺乏用于编写像操作系统、编译程序等系统软件的工具，系统程序的设计主要还是依赖于汇编语言。为了改变这种状况，1967年Martin Richards设计并实现了BCPL(Basic Combined Programming Language)语言，后来，这一语言被移植到多种计算机上，并得到广泛的应用。此后不久，Ken Thompson在BCPL语言的基础上设计并实现了B语言，并用B语言在PDP-7机上实现了第一个UNIX操作系统。接着，1972年至1973年，D. M. Ritchie在B语言的基础上，又重新设计了一种语言，并在PDP-11机上实现，同时用这种语言重写了UNIX操作系统。由于这一语言是在BCPL语言和B语言的基础上开发出来的，因此称为C语言。由于BCPL语言和B语言是无类型的语言，而C语言却能支持多种数据类型，因此C语言与BCPL语言和B语言是不同的，它更能反映当代计算机的体系结构，因而得到广泛的应用。

1.1.2 C语言的特点

C语言能够成为目前应用最为广泛的高级程序设计语言之一，完全是由其语言特点决定的。C语言的特点可大致归纳如下：

(1) C语言短小精悍，基本组成部分紧凑、简洁

C语言一共只有32个标准的关键字、45个标准的运算符及9种控制语句，不但语言的组成精练、简洁，而且使用方便、灵活。

试读结束，需要全本PDF请购买 www.ertongbook.com

(2) C 语言运算符丰富，表达能力强

C 语言能够处理多种运算符，其运算符包含的内容广泛，所生成的表达式简练、灵活，有利于提高编译效率和目标代码的质量。

(3) C 语言数据结构丰富，结构化好

C 语言提供了编写结构化程序所需要的各种数据结构和控制结构，这些丰富的数据结构和控制结构，以及以函数调用为主的程序设计风格，保证了利用 C 语言编写的程序能够具有良好的结构化。同时，在 C 语言程序设计中，允许将一个复杂的程序分割为多个模块，并可由多人同时编写，当分别调试完成后，再通过连接程序连接到一起，形成一个完整的程序。

(4) C 语言提供了某些接近汇编语言的功能，有利于编写系统软件

C 语言具有“高级语言”和“低级语言”的双重特点，它提供的一些运算和操作，能够实现汇编语言的一些功能，如 C 语言可以直接访问物理地址，并能进行二进制位运算等，这为编写系统软件提供了方便条件。

(5) 由 C 语言程序生成的目标代码质量高

C 语言程序所生成的目标代码的效率仅比用汇编语言描述同一个问题低 20%左右，因此，用 C 语言编写的程序执行效率高。

(6) C 语言程序可移植性好

在 C 语言所提供的语句中，没有直接依赖于硬件的语句，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，而这些库函数本身并不是 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。

当然，C 语言本身也有其弱点：运算符的优先级和结合性比较复杂，不容易记忆；由于 C 语言的语法限制不太严格，这在增强程序设计的灵活性的同时，在一定程度上也降低了某些安全性，对程序设计人员提出了更高的要求。

总之，由于 C 语言的上述特点，使得 C 语言越来越受到程序设计人员的重视，并且已经在广泛的领域里得到应用。

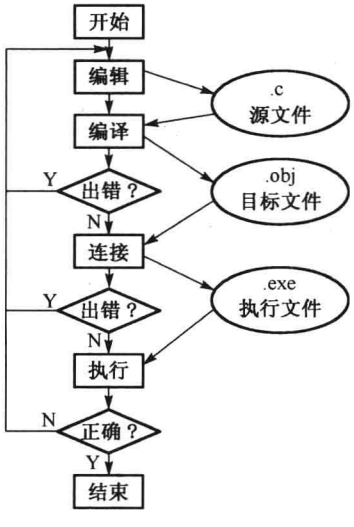

图 1.1 开发 C 语言程序的基本步骤

1.2 C 语言程序的开发步骤

开发 C 语言程序的基本步骤如图 1.1 所示。

1. 编辑

选择适当的编辑程序，将 C 语言源程序通过键盘输入计算机中，并以文件的形式存入到磁盘。在操作系统下，经过编辑后得到的源程序文件以.c 为文件扩展名。

2. 编译

通过编辑程序将源程序输入计算机后，需要经过 C 语言编译器将其编译成目标程序。在对源程序的编译过程中，可能会发现程序中的一些语法错误，这时就需要重新利用编辑程序修改源程序，然后再重新编译。在操作系统下，经过编译后得到的目标文件以.obj 为文件扩展名。

3. 连接

经过编译后生成的目标文件是不能直接执行的，它需要经过连接之后才能生成可执行的代码。在操作系统下，连接后所得到的可执行文件以.exe 为文件扩展名。

4. 执行

经过编译、连接之后，从源程序文件就可以生成可执行的文件，这时就可以执行了。在操作系统下，只要输入可执行的文件名，并按回车键，就可执行文件了。

现在有许多厂家推出了集成环境来处理 C 语言程序，如 Turbo C，在这种集成环境下，对程序的编辑、编译和连接等操作，都可以在一个窗口下进行，使用起来非常方便。

在本书附录 C 简单介绍了 Turbo C 集成开发环境的使用方法，感兴趣的读者可以参考使用。

1.3 C 语言的程序结构

C 语言的程序结构比较简单，很容易掌握，它主要通过函数之间的调用来实现指定的功能。在本节将通过编写几个简单的 C 语言程序，来阐述 C 语言的程序结构，同时对 C 语言的基本语法成分进行相应的说明，以便使读者对 C 语言程序有一个概括的了解，为以后的学习打下基础。

【例 1.1】 编写一个 C 语言程序，用于显示字符串“Hello,World!”。

```
#include "stdio.h"
void main()
{
    printf("Hello,World!\n");
}
```

这是一个简单而完整的 C 语言程序，经过编辑、编译和连接后，其执行结果在屏幕的当前光标位置处显示如下字符串：

Hello,World!

下面对上述程序进行说明。

(1) 一个 C 语言程序可以由多个函数组成，但任何一个完整的 C 语言程序，都必须包含一个且只能包含一个名为 main 的函数，程序总是从 main 函数开始执行的。

(2) 由左右花括号括起来的部分是函数体，函数体中的语句将实现程序的预定功能。在本例中，main 函数的函数体中只有一个 printf 语句，它的功能是进行格式化输出(显示)，即将字符串“Hello,World!\n”显示在终端屏幕上。其中，字符串中的字符“\”和“n”，表示一个“换行”字符，在“换行”字符后面输出的任何字符，将被显示在屏幕的下一行上。

(3) C 语言中的每个基本语句，都是以“;”结束的，分号是 C 语言语句的终结符。

(4) C 语言程序的书写格式比较自由，没有固定的格式要求，在一行内，既可以写一个语句，也可以写多个语句。为了提高程序的可读性，往往根据语句的从属关系，以缩进书写的形式来体现出语句的层次性。

(5) #include 是编译预处理指令，其作用是将由双引号或尖括号括起来的文件中的内容，读入到该语句的位置处。在使用 C 语言输入、输出库函数时，一般需要使用#include 指令将

stdio.h 文件包含到源文件中。有关#include 指令的作用及使用方法，将在第 7 章“编译预处理”做详细介绍。

(6) 函数名前面的 void 用于说明所定义的函数没有返回值。

【例 1.2】 从键盘输入两个整数，并将这两个整数之和显示出来。

```
#include "stdio.h"
int ADDxy(int a,int b)                    /*计算两个整数之和*/
{
    int c;
    c=a+b;
    return(c);
}
void main()
{
    int x,y,z;
    scanf("%d%d",&x,&y);                  /*读入两个整数，存入变量 x 和 y 中*/
    z=ADDxy(x,y);
    printf("The sum of %d and %d is %d",x,y,z);
}
```

上述程序经过编辑、编译和连接后，执行情况如下：

程序从 main 函数开始执行，执行到 scanf 语句时，将等待用户从键盘输入两个整型数据后再继续执行，假如用户输入 10 和 20(此时，x 的值为 10，y 的值为 20)，则屏幕将显示如下信息：

The sum of 10 and 20 is 30

下面，对上述程序进行说明。

(1) 程序中由/*和*/括起来的内容是程序的注释部分，它是为了增加程序的可读性而设置的。注释部分对程序的编译过程和执行结果没有任何影响。

(2) C 语言中的所有变量都必须定义为某种数据类型，同时必须遵循“先定义、后使用”的原则，如语句：

```
int x,y,z;
```

定义了 x、y、z 是三个整型变量，以后就可以使用这三个变量来存放整型数据。

(3) 一个 C 语言程序可以由多个函数组成，通过函数之间的调用来实现相应的功能。程序中所使用的函数，既可以是系统提供的库函数，也可以是用户根据需要自己定义的函数。如上述 main 函数中调用的 scanf 函数和 printf 函数，就是系统提供的库函数，这些函数不需要用户自己定义，在需要时，只要按照指定的格式进行调用即可。C 语言编译系统提供的库函数非常丰富，特别是与硬件打交道的部分，很多工作只要调用库函数就可以实现。而函数 ADDxy 是自定义的函数，它是用户为了实现加法功能而自己编写的函数。每个函数都用于完成一特定的功能，ADDxy 函数的功能是将入口参数 a、b 之和通过 return 语句返回给 main 函数中的变量 z。正确地使用函数，将有助于编写易于理解的、结构化好的程序。有关函数的详细说明请参阅第 6 章“函数”。

(4) 程序中调用的 scanf 函数的作用是进行格式化输入，其中由圆括号括起来的部分是函数的参数部分，不同的函数需要不同的参数。scanf 函数中的参数主要包括两部分内容：“格式控制”

部分，用于对输入数据的格式进行说明；“地址表”部分(本书中出现的表的概念，如地址表、输出表等，是指用逗号分隔的有限个元素序列)，它使用的是存放输入数据的变量的地址。

程序中调用的 printf 函数的作用是进行格式化输出，其参数也包括两部分内容：“格式控制”部分，用于对输出数据的格式进行说明；“输出表”部分，它使用的是存放输出数据的变量本身。有关数据的输入、输出及函数的调用形式，将在后面详细地介绍。

1.4　算法的表示

任何一个程序都是由数据结构和算法等要素组成的，其中数据结构用于管理程序中所使用的数据，如链表和堆栈等就是常用的数据结构，而算法则是为解决某个问题所采取的方法和步骤。

表示算法最直接的方法就是程序设计语言，如利用 C 语言来表示算法。利用程序设计语言表示的算法经过编译和连接之后是能够直接执行的。然而，当一个算法很复杂时，直接利用程序设计语言表示是很困难的，此时就需要用其他的方法来表示算法，自然语言、流程图、N-S 流程图及伪代码等是常用的表示算法的方法。下面简单介绍利用流程图、N-S 流程图及伪代码表示算法的方法。

1.4.1　算法的流程图表示法

利用流程图表示算法，主要是利用一些常用的流程图符号来表示各种操作，这些常用的流程图符号如图 1.2 所示。

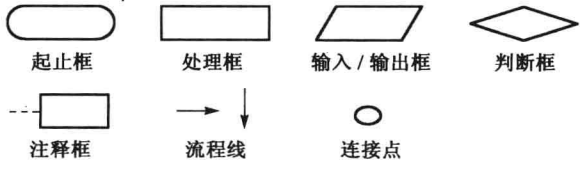

图 1.2　常用的流程图符号

如图 1.2 所示，起止框用于表示流程图的开始和结束；处理框用于表示处理过程；输入/输出框用于表示数据的输入和输出；判断框用于对给定的条件进行判断，以决定下一步的执行流程；注释框用于对流程图中某些框中的操作进行注释和补充说明；流程线用于连接流程图中的各要素并确定算法的执行流程；连接点用于将画在不同位置上的流程线连接起来，一般用在流程图较大的情况。

【例 1.3】　利用流程图来表示计算 $s = 1 + 2 + 3 + \cdots + 100$ 值的算法。

该算法的流程图如图 1.3 所示。

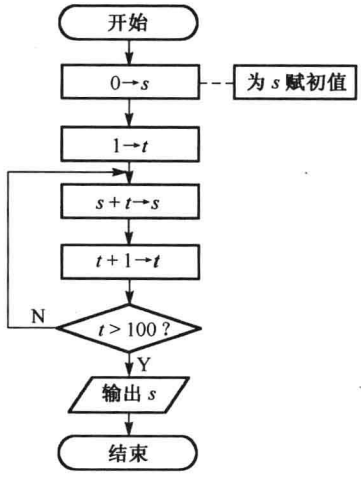

图 1.3　算法的流程图

1.4.2　算法的 N-S 流程图表示法

N-S 流程图是一种去掉所有流程线的流程图形式，全部算法的描述都位于一个矩形框内。N-S 流程图所使用的

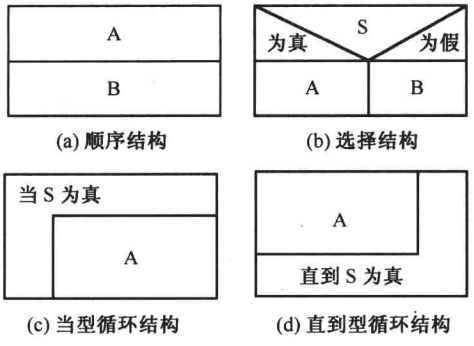

图 1.4 N-S 流程图常用符号

流程图符号如图1.4所示。从图1.4可以看出，N-S流程图中使用的流程图符号，是根据结构化程序的三种基本结构来划分的，分别为顺序结构、选择结构和循环结构，其中循环结构又包括当型循环结构和直到型循环结构。

图1.4(a)为顺序结构流程图符号，它表示按顺序执行操作 A 和操作 B；图1.4(b)为选择结构的流程图符号，表示当条件 S 为真时执行操作 A，当条件 S 为假时执行操作 B；图1.4(c)为当型循环结构的流程图符号，表示当条件 S 为真时重复执行操作 A，直到条件 S 为假时结束循环；图1.4(d)为直到型循环结构的流程图符号，表示重复执行操作 A，直到条件 S 为真时结束循环。与当型循环结构不同，直到型循环结构中的操作 A 至少被执行一次。需要注意的是，图1.4中的操作 A 和操作 B 本身还可以是这 4 种基本结构，也就是说流程图符号是可以嵌套使用的。

【例 1.4】 利用 N-S 流程图来表示计算 $s = 1 + 2 + 3 + \cdots + 100$ 值的算法。

该算法的 N-S 流程图如图 1.5 所示。

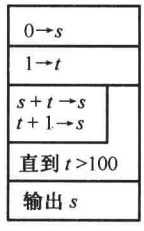

图 1.5 算法的 N-S 流程图

1.4.3 算法的伪代码表示法

前面介绍了算法的流程图表示法和 N-S 流程图表示法，利用这两种方法能够简洁明了地表示一个算法，其缺点是对流程图的编辑(绘制)和修改比较麻烦。为此，可以利用算法的伪代码表示法。伪代码是介于自然语言和计算机语言之间的文字和符号，对这些文字和符号没有严格的规定，只要易于理解、能够将算法描述清楚即可。

【例 1.5】 利用伪代码来表示计算 s=1+2+3+…+100 值的算法。

该算法的一种伪代码表示如下：

```
begin
    0→s
    1→t
    while t<=100
    {
        s+t→s
        t+1→t
    }
    print s
end
```

大型程序设计是一个非常复杂的过程，应养成在利用计算机语言实现算法之前合理地利用流程图或伪代码来描述相关算法的习惯，这对于程序的调试及后续的维护都是非常重要的。

习题1

1.1 简述C语言的主要特点及其用途。

1.2 请参照本章例题，编写一个C语言程序，用于显示以下信息：

We are ready!

1.3 请参照本章例题，分析下面的C语言程序，并给出运行结果，同时画出流程图和N-S流程图。

```
#include "stdio.h"
main()
{
    int a,b,c;
    a=100;
    b=20;
    c=a-b;
    printf("sum=%d",c);
}
```

1.4 请参照本章例题，编写一个C语言程序，用于显示以下信息：

```
##########################
    Happy New Year!
##########################
```

1.5 请参照本章例题，分析下面的C语言程序，并给出模拟运行结果。

```
#include "stdio.h"
main()
{
    int x,y,z;
    scanf("%d%d",&x,&y);
    z=x/y;
    printf("Result=%d",z);
}
```

试读结束，需要全本PDF请购买 www.ertongbook.com