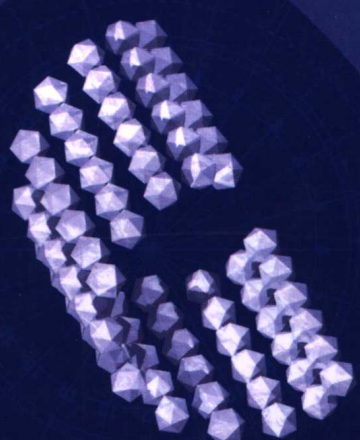
高等学校计算机精品课程系列教材

C语言程序设计教程

张　磊　冯伟昌　主编



中国铁道出版社
CHINA RAILWAY PUBLISHING HOUSE

高等学校计算机精品课程系列教材

C 语言程序设计教程

张　磊　冯伟昌　主　编

黄忠义　李竹健　王桂东　副主编

中国铁道出版社
CHINA RAILWAY PUBLISHING HOUSE

内 容 简 介

本书的内容组织遵循“概念—方法—应用”的基本思路，突出C语言程序设计的主干知识，由浅入深，循序渐进，实例丰富，便于教、便于学。本书既注重理论知识介绍，又突出体现了高级语言程序设计的应用性、实践性特点，强化实践应用，贯穿能力培养主线，力求做到知识学习和能力培养并重。

全书共分12章。第1章～第3章介绍了程序设计的基本概念和C语言的入门知识，第4章和第5章介绍了C语言程序设计的逻辑控制内容，第6章～第10章分别介绍了数组、函数、指针、结构体和文件的有关内容，第11章简单介绍了共用体、枚举和位运算的有关知识，第12章简单介绍了面向对象技术的基本概念和C++语言非面向对象方面对C语言的扩展知识。

本书配有电子教案和《C语言程序设计题解与实验指导》教材。

本书可作为高等院校C语言程序设计课程的教学用书，也可作为工程技术人员的参考书。

图书在版编目（CIP）数据

C语言程序设计教程/张磊，冯伟昌主编. －北京：中国铁道出版社，2007.7
（高等学校计算机精品课程系列教材）
ISBN 978-7-113-08090-7

Ⅰ.C... Ⅱ.①张…②冯… Ⅲ.C语言－程序设计－高等学校－教材 Ⅳ.TP312

中国版本图书馆CIP数据核字（2007）第112588号

书　　名：C语言程序设计教程
作　　者：张　磊　冯伟昌　等
出版发行：中国铁道出版社（100054，北京市宣武区右安门西街8号）
策划编辑：严晓舟　秦绪好
责任编辑：翟玉峰　包　宁
封面设计：付　巍
封面制作：白　雪
印　　刷：三河市华晨印务有限公司
开　　本：787×1092　1/16　**印张**：20.5　**字数**：477千
版　　本：2007年8月第1版　2007年8月第1次印刷
印　　数：1～5 000册
书　　号：ISBN 978-7-113-08090-7/TP·2448
定　　价：28.00元

前言

FOREWORD

本书是C语言程序设计精品课程建设的结晶。本书的内容组织遵循“概念—方法—应用”的基本思路，突出C语言程序设计的主干知识，由浅入深，循序渐进，便于教、便于学。本书既注重理论知识介绍，又突出体现了高级语言程序设计的应用性、实践性特点，强化实践应用，贯穿能力培养主线，力求做到知识学习和能力培养并重。

本书作者是长期从事高级语言程序设计课程教学的一线教师，对计算机语言课程的教学特点、教学规律、C语言教材教法等有较为系统的认识和研究，在精品课程建设和教材建设方面做了大量的工作，积累了丰富的经验。本书是作者在吸收借鉴已有教材长处的基础上，融入多年的教学研究成果而编著完成的，编写时力求做到注重概念、强调方法、突出应用、语言流畅、通俗易懂。

本书在知识选取上采取“瘦身”措施，消除了很多C语言教材中存在的“知识臃肿”现象。教材内容突出C语言程序设计的主干知识，淡化分支知识，摒弃末叶知识。在对C语言程序设计知识点进行系统论证分析的基础上，合理取舍各章的教学内容，将主干知识放在首位，将分支知识作次要介绍，对不利于课程主体内容教学、容易使初学者陷入迷魂阵的“末叶”知识坚决舍弃。这种突出主干知识的选材原则，使教材知识体系脉络清晰，提高了教材的可读性和易学性。

本书突出C语言程序设计课程的“应用性、实践性”特点，贯穿能力培养主线，通过强化应用性教学内容，力求达到在应用中学习知识、培养能力的目的。一是针对基本概念、基本原理设置应用性、实践性教学内容，强化对基本概念、基本原理的理解应用；二是对重点、难点知识设置专门的应用性、实践性教学内容，强化对重点、难点年知识的理解应用；三是灵活运用案例教学、项目驱动等多种方法，将知识点融入到各题例中；四是题例由浅入深、循序渐进，实行“六位一体”标准，即：选题符合教学目标，内容具有应用性、趣味性，概念、原理运用恰当，算法分析具体，程序实现经典，能力培养点明确。

本书教学内容符合教指委关于计算机基础知识教学的有关精神，同时兼顾全国计算机等级考试，内容涵盖了计算机等级考试的主要知识点。本书习题紧密结合教学内容选题，而且类型丰富，具有趣味性、启迪性、应用性。

全书共分12章。第1章至第3章介绍了程序设计的基本概念和C语言的入门知识，第4章、第5章介绍了C语言程序设计的逻辑控制内容，第6章至第10章分别介绍了数组、函数、指针、结构体和文件的有关内容，第11章简单介绍了共用体、枚举和位运算的有关知识，第12章简单介绍了面向对象技术的基本概念和C++非面向对象方面对C的扩展知识。

本书前11章的例题程序均在VC++6.0和Turbo C 2.0环境下编译通过，第12章的例题程序全部在Visual C++ 6.0环境下编译通过。

本书配有电子教案和《C语言程序设计题解与实验指导》教材。

本书适合作为高等院校C语言程序设计课程教学用书，也可用作工程技术人员的参考书。

本书由张磊、冯伟昌主编，黄忠义、李竹健、王桂东副主编，主要内容由张磊编写。参加本书编写工作的还有魏建国、高永存、张元国、薛莹、徐英娟、马明祥、徐思杰、周立友、腾秀荣、潘振昌、王金才、王涛。全书由张磊统稿定稿。

本书的编写和出版得到了许多友人的支持和帮助，在此，表示衷心的感谢。

由于作者水平有限，书中缺点和错误在所难免，恳请读者批评指正。

编 者

2007 年 5 月

目录

CONTENTS

第 1 章 程序设计概述

内容概述

本章简要介绍了程序设计的基本概念和 C 语言程序设计的入门知识，主要包括计算机程序的概念、算法的概念及其描述方法、程序设计中的一般问题、C 语言程序的基本结构及特点、C 语言程序的上机实现等内容。在本章最后，通过一个实例对 C 语言程序设计的过程进行了概括介绍。

主要知识点

（1）程序、算法、程序设计、程序的调试和测试等基本概念。

（2）算法设计和算法描述的基本方法。

（3）C 语言程序结构的基本特点。

（4）使用 Turbo C 2.0 或 Visual C++ 6.0 集成环境编辑、运行 C 语言程序的方法。

（5）问题、算法和程序之间的区别与联系，由具体问题到程序实现的一般过程和步骤。

1.1 程序设计的基本概念

1.1.1 计算机语言和程序

计算机语言是计算机能够理解和识别的软件系统，它通过一定的方式向计算机传送操作指令，从而使计算机能够按照人们的意愿进行各种操作处理。计算机能够识别并执行这些指令的前提是，在设计和组织这些指令时必须符合计算机语言的规则。任何一种计算机语言都有一定的使用规则，通常称为语法规则。只有按照特定的语法规则设计的指令，计算机才能够执行它。

计算机语言分为 3 种类型，即机器语言、汇编语言和高级语言。机器语言是一种二进制语言，它直接使用二进制代码描述机器指令，是唯一能够被计算机硬件直接识别、直接执行的计算机语言。用这种语言编写的程序不直观、难懂、难写、难记、难以修改和维护。汇编语言用符号代替了机器指令代码，而且助记符与指令代码一一对应。与机器语言相比，汇编语言比较直观、容易记忆，但它的通用性与机器语言一样，都很差。机器语言和汇编语言都属于低级语言。高级语言是接近于自然语言的一种计算机语言，高级语言有很强的描述能力，能够直观地按照人们的思维方式处理问题。高级语言进一步分为面向过程的程序设计语言和面向对象的程序设计语言。典型的面向过程的程序设计语言有 BASIC、FoxPro、C 等，典型的面向对象的程序设计语言有 Visual Basic、Visual C++、Java、Delphi 等。

学习计算机语言，必须注意学习它的语法规则，就像学汉语要学汉语语法，学英语要学英语语法一样。学习文字语言的目的是为了实现人们之间的交流，而学习计算机语言的目的是为了设计计算机程序，使计算机按照人们的意愿去自动处理问题。

所谓计算机程序就是按照计算机语言规则组织起来的一组命令，或者说计算机程序是计算机能够自动执行的一组指令的集合。如下是一个用C语言编写的计算机程序，它能够计算100以内的所有偶数和。

```
/* 计算100以内的所有偶数和的程序e1-0.c */
main()
{
    int i,s;
    i=0;
    s=0;
    while(i<100)                /* 循环控制 */
    {   s=s+i;                  /* 数据累加 */
        i=i+2;                  /* 生成下一个要累加的数 */
    }
    printf("sum=%d\n",s);       /* 输出结果 */
}
```

用计算机高级语言编写的程序要在计算机上运行，必须依靠语言处理程序的支持。计算机语言处理程序将用户利用高级语言编写的源程序转换为机器语言代码序列，然后由计算机加以执行。计算机语言处理方式有解释方式和编译方式两种。解释方式是对源程序的每条指令边解释边执行，这种语言处理程序称为解释程序；编译方式是将用户源程序全部翻译成机器语言的指令序列，即目标程序，执行时，计算机直接执行目标程序，这种处理程序称为编译程序。

1.1.2 算法

算法是求解问题的方法，是在有限步骤内求解某一问题所使用的一组定义明确的规则，是计算机处理问题所需要的过程。无论是形成解题思路还是编写程序，都是在实施某种算法。前者是推理实现的算法，后者是操作实现的算法。一般而言，若某一过程由一明确规则的有限集合组成，这些规则规定了解决某个问题或某类特定问题集的操作的有限序列，该过程即称为算法。算法的最终实现是计算机程序。

算法的建立通常需要一个逐步求精的过程，先找出解决问题的基本思路，把解决问题的基本过程表达出来，确立粗略的算法框架，然后对框架中的内容进行逐步细化，添加必要的细节，使其成为详细的算法描述。

算法具有如下特性：

（1）有穷性。一个算法必须经过有限步骤之后解决某个问题，即计算机能够在执行有限的步骤后给出问题的正确结果。事实上，仅一般的有穷性限制在实际中是不够的，因为尽管解决某个特定问题的执行步骤量是有限的，但可能对实际计算来说仍太大。一个有用的算法不仅要求步骤有限，同时也要求步骤量合理。

（2）确定性。一个算法通常由一系列求解步骤来完成，各操作步骤之间有严格的顺序关系，每一个步骤所规定的操作必须是无歧义的和精确定义的，在各种情况下动作的执行必须严密地确定。

（3）多样性。在进行算法设计时，还应该注意到，对于同一个问题，可以有不同的解决方法，即一个问题有多种算法。因此，即便使用同一种计算机语言，对一个问题的解决可能编写出多个不同的计算机程序，认识这一点对学习程序设计是非常重要的。

（4）输入和输出特性。每个有意义的算法有零个或多个输入，并且提供一个或多个输出。所谓 0 个输入是指算法本身定义了初始条件，能够提供处理数据；而多个输入是指算法能够从外部获得处理数据。输出是指算法执行后得到的输出信息，以反映对输入数据加工后的结果，没有输出的算法是毫无意义的。

（5）通用性。一个算法最好是适用于某类问题而不只是适用于某一个问题。这种通用性尽管不是必要的，但通常对算法具有通用性要求。

现在，从算法的角度对“计算 100 以内的所有偶数和”问题作进一步讨论，并给出它的算法描述。

最直观的理解，计算 100 以内的所有偶数和，就是求以下代数式的值：

0+2+4+6+…+98

可以有多种方法对这个代数式求值，如可以使用等差数列求和的方法，可以使用逐个数累加的方法等。下面，以累加求和的方法为例进行算法讨论。

假设用 i 表示当前要加的数，i 的开始取值为 0，每加一次，i 的值增加 2；用 s 表示已经累加取得的结果，开始取值为 0。那么，对问题求解的过程就是不断地将 i 加到 s 中，直到 i 的值超过 98 时，将累加的结果显示在计算机屏幕上。

上面的一段叙述，对问题的求解方法进行了基本描述，但作为算法还不够完整，还需要明确地表达出求解问题的步骤。下面是包含了执行步骤的算法描述，是用自然语言对算法进行描述的常见形式。

“计算 100 以内的所有偶数和”问题的算法：

（1）为 i 和 s 赋初值，使 i=0，s=0；继续下一步骤。

（2）判断 i 的值，若 i<100 则继续执行下一步骤；否则，跳转到（6）。

（3）s 加上 i，继续执行下一步骤。

（4）i 加上 2，继续执行下一步骤。

（5）跳转到（2）。

（6）显示 s 的值，继续执行下一步骤。

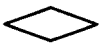
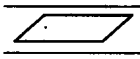
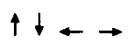
（7）结束。

按照上述算法给定的 7 个步骤，就能求解“偶数和”问题。若选用一种计算机语言正确描述这个算法，就会得到求解“偶数和”问题的计算机程序，运行该程序，将得到“偶数和”问题的计算结果。

为了使算法描述表达得更清晰，更容易实现程序编写，在进行程序设计时通常使用专门的算法表达工具对算法加以描述，如流程图、N-S 图、PAD 图、伪码等。本节只对流程图知识作简要介绍。

流程图是最早使用的一种算法描述工具，它采用不同的几何图形来表示算法的各个步骤，每个几何图形表示不同性质的操作。流程图的特点是绘制简单，结构清晰，逻辑性强，便于描述，容易理解。如表 1-1 所示列出了常用的流程图符号及其功能。

表 1-1 常用的流程图符号及其功能

流程图符号	符 号 功 能
	开始、结束
	处理
	判断
	输入、输出
	流程方向

利用流程图符号可将“计算 100 以内的所有偶数和”算法表达成如图 1-1 所示的算法流程图。

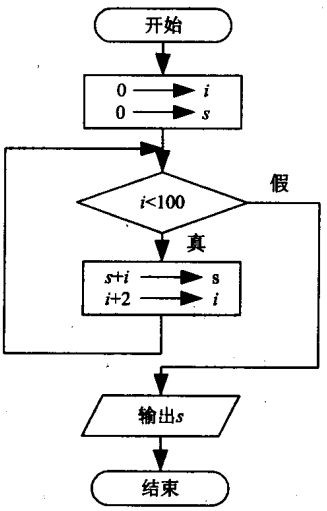


图 1-1 求 100 以内偶数和的流程图

需要指出的是，由于同样的问题可以有不同的算法，自然就产生了算法优劣的评价标准。评价标准涉及很多方面，如算法本身的复杂程度、算法实现后的执行速度、算法对系统资源的需求程度、算法的通用性等，但不管采用什么标准进行评价，正确性和清晰易懂性永远是一个好算法的基本条件。

1.1.3 程序设计

程序设计就是编写程序，它是在对算法进行正确描述的基础上进行的，是用计算机语言（程序设计语言）实现算法的过程。计算机程序是一段文本代码，编写计算机程序需要在文本编辑环境下进行，高级语言系统一般都提供相应的编程环境，当然也可以使用像 Windows 记事本程序那样的文本编辑软件。程序员只有对所用程序设计语言深刻理解、熟练掌握、正确运用，才能编写出高质量的程序代码。

编写程序的基本要求是保证语法上的正确性，只有语法正确的程序才能通过编译系统的语法检查。然后是保证逻辑正确性，这样程序执行后才能得到正确结果。逻辑正确性对一个复杂的程序并不容易做到，通常需要进行反复测试和编辑修改。

高质量的程序还应体现在以下 4 个方面：可靠性高、运行速度快、占用存储空间小和易懂性。通常这 4 个方面不能同时满足，要根据具体情况权衡利弊，兼顾某些方面。在计算机速度越来越快、内存越来越大的今天，程序的易懂性显得更为重要。这是因为一个程序除了能在计算机上运行外，还要求能够让人看懂。只有看懂程序，才能对程序中出现的问题快速地修改，才能根据需要容易地扩充其功能和改善其性能。

程序设计有两种主要方法，即面向过程程序设计和面向对象程序设计，相对于面向过程程序设计而言，面向对象程序设计是一种更新的程序设计方法，但在面向对象程序设计中仍然要用到面向过程的知识，面向过程程序设计是程序设计的基础。

要编写容易读懂的程序代码，从一开始就应养成良好的编程习惯，主要体现在以下几个方面。

（1）合理使用注释。

注释语句是每个程序设计语言都要提供的语句。注释语句对程序的执行结果没有影响，是帮助阅读程序的人理解程序的，它为程序员与程序读者之间建立了重要的信息沟通渠道。一些正规的程序文本中，注释信息会占用大量的篇幅。

编写程序时，一般要在程序的开头编写对程序整体说明的注释，在程序模块（如子程序、函数、过程等）前编写解释该模块作用的注释，在较难理解的语句前后编写对该语句说明的注释。

（2）要使用含义鲜明的符号名。

符号名包括函数名、变量名、常量名等。这些名称应能反映它所代表的实际东西，有实际意义，使其能见名知义。例如，使用 total 表示总量，使用 average 表示平均值，使用 sum 表示累加和等。

（3）程序格式化。

尽量使程序布局合理、层次清晰。一个程序如果写得密密麻麻，没有空白，往往是很难看懂的。恰当地利用空格、空行和缩进可使程序清晰明了。

1.1.4　程序的错误和测试

程序的错误通常有两种，即语法错误和逻辑错误。程序的语法错误，是指程序编写时因不符合程序语言的语法规则而造成的错误。程序存在语法错误时，程序不能运行。程序的逻辑错误，是指程序能够运行，但得不到要求的正确结果。程序的语法错误在编译运行阶段语言系统就会指出来，查错纠错比较容易。对于复杂一些的程序，存在逻辑错误时查找起来则比较困难。

要保证程序的正确性或验证程序的正确性是一个关键的、极为困难的问题。目前，比较实用的验证程序的方法是采用测试法。但是，测试只能证明程序有错，而无法确保程序完全正确，也许通过一系列的测试，程序中还包含未发现的错误。

测试是假设程序中存在错误，通过运行程序来尽可能发现错误。在测试时，输入一组预先设计好的数据，检查运行后是否得到正确的结果。这组输入的数据称为测试用例，如何设计测试用例是测试的关键。目前常用的测试法有黑盒法和白盒法。

黑盒法把程序看成一个黑盒子，只测试程序是否满足它的功能，不考虑程序的内部逻辑和特性。因此，要发现程序中的错误，需要应用穷举法输入每一种数据，进行测试。但事实上要测试每一种数据往往是不可能做到的，因此在程序测试领域出现了众多测试技术，如等价分类法、边值分析法、错误推测法和因果分析法等。

白盒法又称为逻辑覆盖法。使用白盒法需要了解程序内部的详细情况，并在此基础上设计测试用例。测试时，程序中的每一条语句至少要执行一次，最彻底的覆盖程序中的每一条路径。常用的覆盖标准有语句覆盖、判定覆盖、条件覆盖、判定/条件覆盖和条件组合覆盖等。

通过测试发现错误后，要对源程序进行检查，找出错误的位置，并予以改正。这种去除程序错误的过程称为调试。