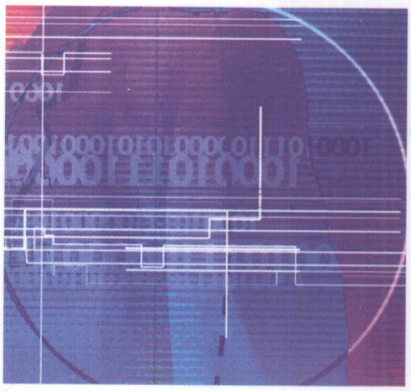
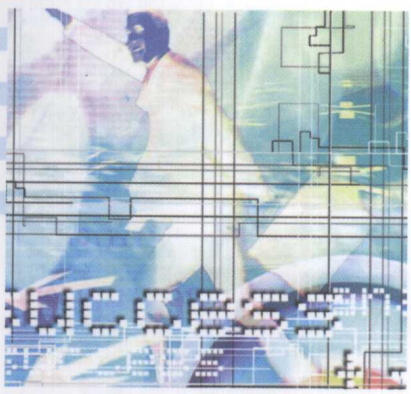
21世纪高等学校应用型教材

C语言程序设计

□ 张 磊 主 编
□ 张元国 李 健 副主编





高等教育出版社
Higher Education Press

21世纪高等学校应用型教材

C语言程序设计

张　磊　主　编
张元国　李　健　副主编

高等教育出版社

内 容 提 要

本书打破了以篇章为结构的传统组织方式，采用教学单元式结构作为教材的基本框架，更贴近于课堂教学。全书通过18个教学单元，从程序设计的基本概念和基本原理出发，立足“理论必须够用为度，强化实践应用，贯穿能力培养主线”的原则，对C语言程序设计知识进行了系统介绍。主要内容包括：程序设计概述、数据及其运算、程序设计基础、分支结构程序设计、循环结构程序设计基础、循环结构程序设计应用、数组基础、数组应用、函数基础、函数应用、指针基础、指针和函数、结构体、链表、文件、共用体/枚举和位运算、编译预处理和数据类型再命名、从C到C++等。

本书针对非计算机专业初学者特点编写，主干知识突出，知识脉络清晰，语言表达流畅，具有良好的可读性和易学性。除主教材外，本书还配有辅导书《C语言程序设计实验与实训指导及题解》。

本书适合作为高等院校各专业“C语言程序设计”公共课教材，也可供编程人员和参加全国计算机等级考试（二级C）的读者学习参考。本书配套电子教案及书中相关源程序均可从高等教育出版社的计算机教学资源网下载，网址为 http://cs.hep.com.cn 或 http://www.hep-st.com.cn。

图书在版编目（CIP）数据

C语言程序设计/张磊主编. —北京：高等教育出版社，2005.1

ISBN 7-04-016437-X

Ⅰ.C… Ⅱ.张… Ⅲ.C语言-程序设计 Ⅳ.TP312

中国版本图书馆CIP数据核字（2005）第007372号

策划编辑 雷顺加　**责任编辑** 萧　潇　**封面设计** 王凌波　**责任印制** 杨　明

出版发行	高等教育出版社	**购书热线**	010-58581118
社　　址	北京市西城区德外大街4号	**免费咨询**	800-810-0598
邮政编码	100011	**网　　址**	http://www.hep.edu.cn
总　　机	010-58581000		http://www.hep.com.cn
		网上订购	http://www.landraco.com
			http://www.landraco.com.cn
经　　销	北京蓝色畅想图书发行有限公司		
印　　刷	中国农业出版社印刷厂		
开　　本	787×1092　1/16	**版　　次**	2005年1月第1版
印　　张	18.5	**印　　次**	2005年1月第1次印刷
字　　数	450 000	**定　　价**	24.00元

物料号 16437-00

前　言

C 语言程序设计是高等学校普遍开设的一门计算机基础课程，在大学生现代化思维训练、创新能力培养、计算机素质教育等方面发挥着重要作用。要学好 C 语言，离不开好的教材。

本书作者是长期从事计算机基础课教学的一线教师，对基础课教学规律、非计算机专业的计算机语言课教学特点、C 语言教材教法等有较深的认识和系统的研究，曾选用过多种不同版本的 C 语言教材实施教学，在精品课程教材建设方面也进行了许多有益的探索。本书是作者在吸收借鉴已有教材长处的基础上，融入多年的教学实践经验和教学研究成果而编著完成的一种新型的 C 语言教材。

本书采用单元式结构作为教材框架，力求使教材在结构形式上更贴近于课堂教学实践，实现教材与课堂教学的进一步融合。本书把 C 语言程序设计知识归结为 18 个教学单元，教学单元之间保持知识的先后连续性，教学单元内保持知识的相对完整性。这种组织形式不但会减小教学实施中对教材内容的重组程度，而且有利于强化“教”和“学”的目标性，“教”者清楚，“学”者明白。

本书针对非计算机专业初学者的特点编写，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，合理取舍每个教学单元的知识内容，将主干知识列入教学目标，放在首位；将分支知识做次要介绍；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。学生在学习时既容易进入主题，又不迷失方向。

本书突出 C 语言课程的“应用性、实践性”特点，贯穿能力培养主线。一是在每个教学单元设置应用性、实践性教学内容，实现基本概念、基本原理的即时应用；二是设置专门的应用性教学单元，使重点、难点知识的应用性教学保持系统性和连续性；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：选题符合单元教学目标，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。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。

本书突出 C 语言程序设计基本概念、基本原理和基本应用的教学，语言流畅，概念准确，通俗易懂。

本书共由 18 个教学单元组成。第 1 单元到第 17 单元对 C 语言程序设计知识进行了系统介绍；第 18 单元介绍了 C++ 的非面向对象知识，这些内容仅仅是对 C 语言的扩展，目的是使读者对 C++ 语言有一个初步认识，也能应用 C++ 的环境编写运行 C 语言程序。

本书还配有辅导书，该辅导书内容包括实验、课程设计、习题解答和典型题例分析四部分。

实验部分有18个实验，对应本书的18个教学单元；课程设计部分通过具体实例，介绍用C语言进行课程设计的方法步骤，并提供用于课程设计的若干题目；习题解答部分对本书的部分习题进行了解答；典型题例分析部分紧密结合本书的教学内容，精选一定数量的C语言等级考试题目，并进行分析解答。

本书由张磊主编，张元国、李健副主编。张磊编写了第1单元～第18单元的主要内容。高永存参加了第1单元～第3单元的编写，王桂东参加了第4单元～第6单元的编写，宋均孝参加了第7单元～第10单元的编写，张元国、王金才参加了第11单元～第14单元的编写，魏建国参加了第15单元～第17单元的编写，李健参加了第18单元的编写。冯伟昌、黄忠义、李竹健参加了本书编写大纲的讨论制定，并编写了部分内容，薛莹、李伟、董春萍、蒋向东、徐英娟参加了本书的程序调试，王治国、梁坤、付延宝对本书的程序进行了归档整理。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。作者联系信箱为：wfuzl@163.com。

编 者

2004年12月

目　录

第1单元

程序设计概述

内容概述

本单元简要介绍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、程序设计的基本问题、程序错误和测试、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。在本单元最后，通过一个实例对C语言程序设计的过程进行概括的介绍。

教学目标

- 掌握程序设计的基本概念，包括程序、算法、程序设计以及程序的调试和测试等。
- 掌握算法设计和描述的基本方法，能对简单的问题设计算法，并用流程图表达出来。
- 了解C语言程序结构的基本特点，能够使用VC++6.0或TC 2.0集成环境运行简单的C语言程序。
- 了解问题、算法和程序之间的关系，熟悉由具体问题到程序实现的有关过程和步骤，初步建立程序设计的基本思路。

特别提示

本单元虽然给出了一些程序示例，但并不要求读者就此能够编写程序，应立足“理解概念、认识程序和掌握方法”的思路来学习本单元知识。

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语言，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，通常称之为语法规则。只有按照特定的语法规则设计的指令，才是有效的指令，计算机

才能够执行它。

计算机语言的种类有很多,大体上经过了由低级语言到高级语言的发展过程,目前广泛使用的是计算机高级语言,如 Basic、FoxPro、C++、Java、Delphi 以及本教程介绍的 C 语言等。

要学习计算机语言,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1 000以内的所有奇数和。

```
/* 计算 1000 以内的所有奇数和的程序 */
main()
{
 int i,s;
 i=1;
 s=0;
 while(i<1000)                    /* 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 */
}
```

1.1.2 算法

算法就是求解问题的方法,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。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,使之成为较为详细的算法描述。

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有明确的含义,不能存在二义性。计算机能够在执行有限的步骤后给出正确的结果。

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,一个问题也可能有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。

现在,从算法的角度对“计算 1 000 以内的所有奇数和”的问题做进一步讨论,并给出它的算法描述。

最直观的理解,计算 1 000 以内的所有奇数和,就是求以下代数式的值:

$$1+3+5+7+\cdots+993+995+997+999$$

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。

假若用 i 表示当前要加的数，i 的开始取值为 1，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，对问题的求解的过程就是不断地将 i 加到 s 中，直到 i 的值超过 999 时，便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，但作为算法这是不够的，通常还需要明确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 1 000 以内的所有奇数和”问题的算法：

步骤① 为 i 和 s 赋初值，使 i = 1，s = 0；继续下一步骤；

步骤② 判断 i 的值，若 i < 1000 则继续执行下一步骤；否则，转步骤⑥；

步骤③ s 加上 i，继续执行下一步骤；

步骤④ i 加上 2，继续执行下一步骤；

步骤⑤ 转步骤②；

步骤⑥ 显示 s 的值，继续执行下一步骤；

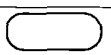
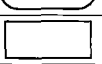
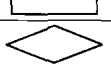
步骤⑦ 结束。

按照上述算法给定的七个步骤，就能求解“奇数和”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“奇数和”问题的计算机程序，运行该程序，将得到“奇数和”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做简要介绍。

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。表 1 - 1 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

现在，利用流程图符号，将“计算 1 000 以内的所有奇数和”的算法表达成如图 1 - 1 所示的程序流程图。

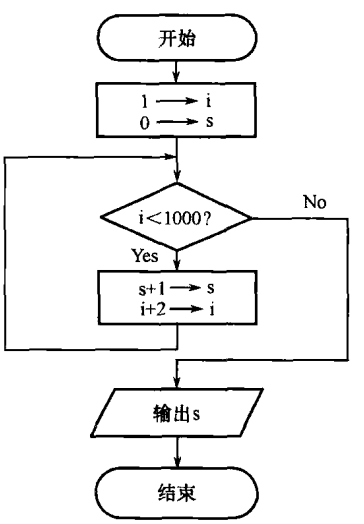


图 1 - 1 “计算 1 000 以内的所有奇数和”的算法流程图

还需要指出的是，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 记事本那样的文本编辑软件。编写程序人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。然后是保证逻辑的正确性,这样程序执行后才能得到正确结果。逻辑的正确性对一个复杂的程序并不容易做到,通常需要一个反复测试和编辑修改的过程。

高质量的程序还应体现在以下四个方面:可靠性高、运行速度快、占用存储空间小和易懂性。通常这四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在计算机上运行外,还要求人能够看懂。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。

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。这些名字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,没有空隙,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。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。存在语法错误时,程序不能正常运行。程序的逻辑错误,是指程序能够正常运行,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的方法。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。目前常用的测试方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,最彻底的是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。

目前,大多数开发工具都提供了跟踪调试工具,一般都提供了设置断点、单步执行、查看变量或表达式值等功能,熟练使用这些调试工具可以提高调试效率。

需要指出的是,给出这一小节的目的,并非要求读者现在就去理解和掌握程序测试的技术,而主要是说明即便是一个熟练的程序设计员,设计的程序仍然会存在一些问题,需要使用一定的技术把问题找出来。那么,作为一个初学者,对所编写的程序即使经过反复调试后,有可能还得不到一个完全正确的程序,这也是很正常的情况。

1.1.5 输入/输出

输入就是通过输入设备为正在运行的计算机程序提供原始数据,输出就是正在运行的计算机程序将处理的结果通过输出设备呈现出来。简单地说,为程序提供数据称为输入,从程序获得数据称为输出。

输入/输出的方式多种多样,程序可以从它支持的任何一种输入设备获得输入数据,同样也可以从它支持的任何一种输出设备输出数据。输入/输出的数据可以是数值、文本、图形、图像等。从键盘输入数据是最基本的数据输入方式,从显示器输出结果是最基本的输出方式。基本的输入/输出一般用于小批量数据的处理,大批量数据的输入/输出通常借助文件来实现,利用事先建立的数据文件为程序提供数据,程序把处理结果保存在磁盘文件上,这称为文件数据的输入/输出。

输入/输出通常是程序的基本功能,在输入/输出时往往有一定的格式要求。

1.2 C语言程序

1.2.1 C语言概述

C语言是1972年由美国的Dennis Ritchie设计发明的,并首次在UNIX操作系统的DEC PDP

-11 计算机上使用。它由早期的编程语言 BCPL(Basic Combind Programming Language)发展演变而来。在 1970 年,AT&T 贝尔实验室的 Ken Thompson 根据 BCPL 语言设计出较先进的并取名为 B 的语言,最后导致了 C 语言的问世。1983 年,美国国家标准化协会(ANSI)根据 C 语言问世以来的各种版本对 C 进行了完善和扩充,制定了 C 的标准,称为 ANSI C。1987 年 ANSI 又公布了新的标准——87 ANSI C。目前流行的 C 编译系统都是以它为基础的。

在 C 的基础上,1983 年贝尔实验室的 Bjarne Stroustrup 又推出了 C++。C++进一步扩充和完善了 C 语言,成为一种面向对象的程序设计语言。C++提出了一些更为深入的概念,它所支持的这些面向对象的概念容易将问题空间直接地映射到程序空间,为程序员提供了一种与传统的结构化程序设计不同的思维方式和编程方法,因而也增加了整个语言的复杂性,掌握起来有一定难度。

一般认为,C 语言具有如下特点:

(1) C 语言是一种结构化语言,它层次清晰,便于按模块化方式组织程序,易于调试和维护。

(2) C 语言的表现能力和处理能力极强,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,它还可以直接访问内存的物理地址。

(3) 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,也适合于应用软件的开发。

(4) C 语言还具有效率高、可移植性强等特点,因此广泛地移植到了各种类型的计算机上,从而形成了多种版本的 C 语言。

1.2.2 简单的 C 程序及其包含的概念

本节通过几个小程序引出 C 语言程序设计的基本概念,使读者对 C 语言程序和程序设计有一个初步认识。

例 1-1 是一个十分简单的 C 语言程序,其功能是对 a 和 b 作加法,然后将结果显示在屏幕上。

例 1-1 一个加法程序。

```
/* 程序 e101.c */
main()
{
 int a,b;
 a=8;
 b=2000;
 printf("%d\n",a+b);
}
```

这个 C 程序(C 语言源程序,简称 C 程序)包含如下概念:程序行、主函数、数据类型、变量、赋值、表达式、系统函数、输出、输出格式、函数体和注释。

这个程序由八行组成,每一行称为一个程序行;第一行是程序的注释,是一些关于程序的说明性信息;第二行的 main()称为主函数;第四、五、六、七行的 a、b 是程序中的变量;第四行的 int 限定变量 a、b 只能代表整数,即 a、b 的数据类型为整型;第五、六行实现了赋值功能,分别使 a、b 具有 8 和 2000 的值;第七行的功能是将表达式 a+b 的值输出在屏幕上,其中 printf()是实现输

出的一个系统函数,“%d\n”是输出数据的格式控制信息;花括号对{}之间的所有信息构成了main()函数的函数体。

从上面的叙述可以知道,这个加法程序的功能是非常单一的,它并不能对任意的两个数进行加法运算,即它不是一个通用的加法程序。

例1-2的程序e102.c是e101.c的改进程序,它实现了任意两个整数(在允许的范围)的加法运算,结果的输出形式也更为明确。它体现的“程序设计”思想正是解决这一问题的思路:

(1) 从键盘任意输入两个整数,最好在需要输入数据时有一个提示信息;

(2) 做加法运算;

(3) 输出清楚明了的运算结果。

例1-2 一个改进的加法程序。

```
/* 程序e102.c */
main()
{
 int a,b;
 printf("Input a,b:");
 scanf("%d,%d",&a,&b);
 printf("a+b=%d\n",a+b);
}
```

程序的第五行是一个输出语句,功能是在屏幕显示“Input a,b:”提示信息;第六行的scanf()是实现数据输入的系统函数,它借助键盘为a,b提供数据;第七行语句实现a+b的运算,并以“a+b=”的形式输出结果。具体执行情况如下:

```
Input a,b: 8,2000 ↵
a+b=2008
```

结果中第一行的“Input a,b:”是执行程序时显示的提示信息,其后的8和2000是输入的两个数据,它们之间以“,”分隔,符号“↵”表示结束输入的回车符。结果中的第二行是输入数据之后的输出结果。

例1-3 最简单的C程序。

```
/* 程序e103.c */
main()
{
  printf("Hello,world! \n");
}
```

这是一个最简单的C语言程序,函数体只有一个输出函数语句,用于输出一个特定的文本信息。该程序执行结果是在屏幕显示如下字符串:

```
Hello,world!
```

总结上述三个程序,不难发现它们具有一个共同的特点,即每个程序都是由main()函数构成的,由于main()函数的函数体内容不同,程序也就各自具有了不同的功能。事实上,任何一个C语言程序,main()函数都是不可缺少的。main()函数的一般结构如下:

```
main()
```

```
{
  函数体语句
}
```

函数体通常分为两部分,前半部分一般是一些说明语句,用于对变量等进行必要的定义说明(例1-3没有需要说明的内容,所以没有说明部分),也称为定义数据结构;后半部分是一些执行语句,完成具体的操作。从操作的角度来认识,程序的功能是由函数体的可执行语句完成的,也就是说,函数的可执行部分实现了问题的"算法"。由此,对"程序"的概念可以理解为

程序=数据结构+算法。

下面是一个稍微复杂一些的例子,它告诉读者,一个C语言程序,在结构上不仅包括main()函数,还可以包括其他独立的函数,一个函数可以在另一个函数中被使用。

例1-4 求最大数程序。

如下程序实现了求两个数中最大数的功能,当程序执行时,用户从键盘输入任意两个整数,计算机就会将其中较大的数显示在屏幕上。

```
/* 程序 e104.c */
main()                                          /* 主函数 */
{
  int x,y,large;                                /* 定义变量 */
  scanf("%d,%d",&x,&y);                         /* 从键盘输入两个数 */
  large=max(x,y);        /* 利用max()函数找出x、y的最大数并赋给large */
  printf("The Max number is %d\n",large);       /* 输出 */
}
/* 以下是一个自定义函数 */
int max(int x,int y)     /* 求x、y中最大值的函数,它是独立于main()函数的 */
{
  int z;
  if (x>y)
    z=x;                 /* 如果x大于y就将x的值赋给z */
  else
    z=y;                 /* 否则就将y的值赋给z */
  return(z);
}
```

这个程序包括了两个函数,一个是主函数main(),另一个是max()函数,其功能是求出两个整数x、y中最大的一个数,这个函数是独立定义的,它在main()函数中被使用。在执行时,程序从main()函数开始,先由scanf()函数从键盘读取两个数值,这时需要由用户从键盘上输入两个数(如3,5)。此时x被赋值3,y被赋值5。然后执行第六行,将x和y的值传入max()函数中。在max()函数中经过判断后,z中的值就是两个数的最大值。用return语句将z的值作为函数的返回值,此时程序又回到第六行,将max()函数的返回值赋值给变量large。第七行将变量large的值输出到屏幕上。

例1-4程序中出现了很多由/*…*/构成的信息,这是程序的注释内容。注释部分不参与也不影响程序的运行,这些注释只是用来帮助人们阅读程序的,注释既可以放在一个语句之后,也可以单独成行。