

■ 高等院校基础课教材

程序设计基础 (C语言)

马保东 编著

南开大学出版社

TP312/2651

2008

程序设计基础（C 语言）

马保东 编著

南开大学出版社
天 津

内容简介

本书详细介绍了程序设计的基本思维方式和 C 语言的基础知识, 特别重点介绍了选择程序设计, 循环程序设计, 逐步细化, 递归程序设计, 模块化程序设计, 状态转移, 抽象数据类型设计等方面的思维方式, 以及 C 语言的两个高级特性——指针与位运算。除此之外, 本书还简要介绍了文件、输入输出、预处理、算法与数据结构等方面的基础知识。

本书可用作普通高等学校理工科各专业的程序设计基础 (C 语言) 课程的教材, 也可供初学程序设计的读者自学时参考。

图书在版编目(CIP)数据

程序设计基础: C 语言 / 马保东编著. —天津: 南开大学出版社, 2008. 1

ISBN 978-7-310-02824-5

I. 程… II. 马… III. C 语言—程序设计 IV. TP312

中国版本图书馆 CIP 数据核字 (2007) 第 195534 号

版权所有 侵权必究

南开大学出版社出版发行

出版人: 肖占鹏

地址: 天津市南开区卫津路 94 号 邮政编码: 300071

营销部电话: (022) 23508339 23500755

营销部传真: (022) 23508542 邮购部电话: (022) 23502200

*

南开大学印刷厂印刷

全国各地新华书店经销

*

2008 年 1 月第 1 版 2008 年 1 月第 1 次印刷

787×1092 毫米 16 开本 25.75 印张 652 千字

定价: 40.00 元

如遇图书印装质量问题, 请与本社营销部联系调换, 电话: (022) 23507125

前 言

自从 1946 年第一台电子计算机诞生以来,计算机的应用领域越来越广泛。20 世纪 50 年代,计算机只是用于军事和科学计算领域,而 21 世纪的今天,几乎任何领域都不同程度地受到了计算机的影响。计算机应用越来越广泛的原因之一是软硬件技术的飞速发展,硬件技术的发展为我们提供了运算能力更强、存储容量更大的计算机,软件技术的发展则为更快更好地开发软件产品提供了可能。

尽管计算机技术在过去的几十年里发展很快,但仍然不能满足日益增长的需求。即使在科技发达的美国,计算机技术人员也始终是紧缺人才。在科技水平相对落后的我国,更需要优秀青年(包括在校大学生)参与其中,成为计算机技术人才。

程序设计基础(或称为程序设计入门)是计算机教育的基础课程之一,所涉及的是计算机领域中较为“古老”的内容。应该说,无论在什么领域,掌握了最新的思想、方法和知识的人就是该领域的领先者,这是毫无疑问的。不过,“古老”的内容也是经过多年的淘沙积淀下来的精华,一般对本领域有很强的辐射力。因此,对有志于深入了解计算机技术的读者来说,学好程序设计基础尤其必要。

在学习之前,我们首先要了解学习该课程的目的,在学习的时候重点应该掌握什么。简单地说,学习该课程的目的就是学会编写程序,重点是程序设计的基本概念、基本方法和基本思维方式。我们编写程序的目的是为了解决问题,而解决问题首先需要找出合适的解决方案,然后用合适的程序设计语言把方案描述出来。显然,该过程的关键是分析问题并找出解决方案,然后才是用语言描述方案。因此,该课程的首要目标是培养解决问题的基本思维方式,其次才是掌握语言知识。

目前,我国不少高校都开设 C 程序设计课程,以此作为程序设计的入门课程。根据笔者了解,很多初学者都觉得 C 程序设计很难学,有的人甚至下了很大工夫,学了不止一遍,可总觉得不会编写程序。造成这种情况的首要原因,也是根本原因,就是没有养成正确的思维方式。

当然,C 语言本身的复杂性也是其比较难学的原因之一。C 语言的语法和语义非常繁琐,对初学者来说完全掌握语言知识就不是一件容易的事。更有甚者,C 语言是为开发系统程序而设计的,相比其他语言,C 语言又增加了许多特点,这更增加了学习的难度。

另外,由于 C 语言通常用来开发系统软件,学习 C 语言的时候往往要涉及一些计算机原理的相关知识。对初学者来说,理解计算机的原理是很费时间的。

基于上面的考虑,本书有意识地将内容划分为不同的层次。第一层次是程序设计的基本概念和基本思维方式,以及 C 语言的基础知识;第二层次是程序设计规范;第三层次是计算机原理方面的知识。

本书的正文主要讨论的是第一层次的内容,并兼顾第二层次的程序设计规范(主要是通过例子体现)。本书较少涉及 C 语言繁琐的语法和语义细节,目的就是让初学者把学习的重点放到培养思维方式上。只要掌握了程序设计的思维方式,即使自学 C 语言也是完全可能的。

相反, 如果没有建立起正确的思维方式, 即使语言知识掌握得再好, 也不会设计程序。

为了保证程序设计思维这个重点, 避免喧宾夺主, 本书把属于第三层次的计算机原理的相关知识尽可能地放在附录中。

综上所述, 本书的内容强调的是程序设计思维和 C 语言的基础知识, 并兼顾程序设计规范, 并没有花很多笔墨详细介绍 C 语言。换句话说, 本书是培养初学者程序设计思维的教材, 而不是深入透彻地讲授 C 语言的语法和语义的教材。读者如果能够有意识地从不同的层次阅读本书, 可能会有更好的效果。

本书的组织结构和对读者的建议

第 1 章重点介绍程序的概念和程序的基本模式, 包括程序处理简单数据的模式、根据条件做出选择的模式、循环计算的模式、注释的模式等。尽管只是介绍性的内容, 但从未接触过程序设计的初学者必须认真阅读这些内容。除此之外, 第 1 章还简单介绍了程序处理大量数据的模式 (数组) 和复杂程序的构成模式 (函数), 对这部分内容, 刚开始学习的时候只需要简单了解即可。

第 2 章讨论了程序的数据类型、运算符和表达式, 第 3 章讨论了选择和循环程序设计的思维方式, 第 4 章讨论了数组等复合数据类型。这部分讨论的是程序设计的基础知识和基本思维方式, 初学者必须掌握, 不能有丝毫的含糊。除此之外, 如果有兴趣和时间, 最好能进一步阅读附录, 尤其是与第 2 章相关的附录 A、附录 B、附录 C。

第 5 章讨论了函数, 其中函数的设计与调用、函数执行的基本原理、一维数组作参数的原理都是初学者必须掌握的。第 5 章的最后讨论了递归, 并给出了一个递归的实例。读者可首先掌握递归函数的概念及其在计算机内的执行过程, 如果觉得实例比较难理解, 可先略过该部分内容, 等到对函数的理解更深刻的时候再重新学习。

第 6 章首先以简单实例的形式展示了模块化程序设计的方法, 初学者可仔细体会其中蕴含的思维方式。除此之外, 第 6 章还介绍了作用域和存储类别的概念, 这是 C 语言方面的知识。

第 7 章讨论了指针, 这是 C 语言的一个重要特性, 也是将来学习面向对象的程序设计的基础。读者必须掌握与指针相关的概念、计算机的相关执行原理、使用指针设计程序的模式。当然, 动态内存分配实例和链表应用实例相对来说比较繁琐, 初学者可以暂且略过, 待基础牢固之后再重新学习。

第 8 章讨论文件和 C 语言的格式化输入输出函数。初学者可以首先学习两类文件的应用场合和编程模式, 而深入了解两种文件的编程需要融会贯通字符的原理和输入输出的实质。附录 A.3 介绍了字符的原理, 附录 G.3 中关于 `scanf`, `fscanf`, `sscanf`, `printf`, `fprintf`, `sprintf`, `fread`, `fwrite` 等函数的内容蕴含了输入输出的实质, 读者可根据自己的情况酌情阅读。格式化输入输出函数的用法比较繁琐, 初学者只要掌握基本用法即可。

第 9 章讨论了位运算在软件开发中的应用, 电子信息、机电、控制等专业的读者需要仔细学习该章的内容, 其他专业的读者可酌情考虑。

第 10 章讨论了 C 语言的基本预处理命令, 除了条件编译属于了解性内容之外, 初学者应该掌握其他内容 (`#include` 和 `#define`)。

第 11 章介绍接口与库的开发, 第 12 章介绍算法分析、数值方法、状态转移分析方法,

第 13 章则介绍数据结构和抽象数据类型。该部分主要起引导的作用，向在程序设计方面基本入门的读者介绍更高层次的内容，并希望引导他们进入其中。

请读者注意，程序设计是实践性很强的课程。读者除了要学习程序设计的基本概念和基本思维方式之外，还要特别重视上机实践，以培养自己调试程序的能力，丰富自己的编程经验。我们建议，读者理论学习与实践的时间比例至少应该达到 1:1，甚至应该达到 1:2 或 1:3 或更高。

由于作者水平有限，书中肯定有许多不足之处，热切盼望广大读者和专家批评指正。我们的联系方式是：baodongma@163.com。

编 者

2007 年 8 月

目 录

第1章 C语言程序设计概述	1
1.1 程序的概念	2
1.2 C语言的历史与特点	3
1.3 简单的C程序	4
1.4 变量	5
1.5 循环	7
1.6 选择	8
1.7 数组	10
1.8 函数	12
1.9 注释	14
1.10 运行程序	15
1.11 小结	17
习题1	17
第2章 数据类型、运算符和表达式	19
2.1 变量命名	20
2.2 计算机内数据的特点	20
2.3 整数	21
2.4 实数	24
2.5 字符	25
2.6 常用算术运算符	28
2.7 赋值运算符	29
2.8 自增和自减运算符	30
2.9 数据类型转换	32
2.10 小结	34
习题2	34
第3章 控制流	36
3.1 关系运算符	37
3.2 逻辑运算符	39
3.3 if语句	44
3.4 条件运算符	49
3.5 for语句	50
3.6 while语句	55
3.7 for语句与while语句的关系	58
3.8 break语句	59

3.9 嵌套循环	62
3.10 switch 语句	65
3.11 小结	68
习题 3	68
第 4 章 复合数据类型	72
4.1 数组	73
4.2 数组编程	77
4.3 二维数组	82
4.4 字符串	87
4.5 记录	91
4.6 枚举类型	95
4.7 typedef	99
4.8 小结	99
习题 4	100
第 5 章 函数	104
5.1 定义函数	105
5.2 调用有返回值的函数	108
5.3 调用无返回值的函数	111
5.4 函数调用的原理	113
5.5 一维数组作参数	118
5.6 二维数组作参数	121
5.7 递归调用	124
5.8 小结	132
习题 5	132
第 6 章 模块化设计初步	135
6.1 模块化设计实例: 日历程序	136
6.2 作用域	143
6.3 变量的存储类别	146
6.4 外部函数与内部函数	152
6.5 小结	153
习题 6	153
第 7 章 指针	156
7.1 指针的概念	157
7.2 使用指针	158
7.3 指针作函数参数	163
7.4 用指针处理一维数组	165
7.5 用指针处理多维数组	170
7.6 动态内存分配	177
7.7 动态内存分配实例*	181

7.8 动态数据结构——线性单链表*	189
7.9 链表应用的实例*	199
7.10 指向函数的指针	208
7.11 指针和字符串	212
7.12 小结	217
习题 7	218
第 8 章 I/O	222
8.1 ASCII 文件	223
8.2 二进制文件	227
8.3 格式化输出函数 printf	231
8.4 格式化输入函数 scanf	237
8.5 字符和字符串的输入输出	243
8.6 小结	245
习题 8	245
第 9 章 位运算	248
9.1 位运算符与运算规则	249
9.2 位运算实例	252
9.3 小结	259
习题 9	259
第 10 章 预处理	261
10.1 #include	262
10.2 #define	263
10.3 带参数的宏	264
10.4 条件编译	265
10.5 小结	268
习题 10	268
第 11 章 接口与库	270
11.1 接口	271
11.2 解读接口 time.h	272
11.3 接口的设计原则	277
11.4 接口设计的实例 (BigNum.h)	278
11.5 库代码的实现 (BigNum.c)	285
11.6 使用库解决问题	300
11.7 小结	301
习题 11	302
第 12 章 算法初步	303
12.1 排序算法	305
12.2 查找算法	308
12.3 数值算法	311

12.4 状态转移表	316
12.5 小结	322
习题 12	322
第 13 章 数据结构初步	324
13.1 数据结构的概念	325
13.2 抽象数据类型	328
13.3 队列	329
13.4 堆栈	339
13.5 堆栈的应用实例	346
13.6 小结	352
习题 13	352
附录 A 计算机的数据表示原理	354
附录 B 数据类型的转换	363
附录 C C 语言的常量	367
附录 D ASCII 码对照表	369
附录 E C 语言的运算符	371
附录 F C 语言的关键字	373
附录 G 常用的 ANSI C 库函数	377
附录 H Turbo C 2.0 集成环境	395
参考文献	401

第 1 章 C 语言程序设计概述

学习目的

- 理解程序的概念
- 了解 C 语言的历史与特点
- 掌握 C 程序的基本组成部分
- 了解程序处理数据的基本模式
- 了解程序的三种基本结构（顺序、循环、选择）
- 了解数组及其应用场合
- 了解函数的用途
- 掌握注释的作用和注释的风格

本章首先介绍程序的概念和程序设计的实质,然后简单介绍 C 语言的历史与特点,最后介绍 C 程序的基本构成、处理数据的基本模式、重复计算的基本模式、根据条件做出选择的基本模式等。

1.1 程序的概念

当前,广泛应用的计算机系统包括两大部分:硬件系统和软件系统。硬件系统是由各种部件构成的物理上的计算机,比如说,中央处理器、内存储器、显示器等都是硬件设备。软件则主要指运行在计算机硬件系统上的程序,比如说,Windows 操作系统、金山词霸、各种游戏程序等都属于软件产品。

从用户的角度看,程序是运行在计算机上能够完成特定任务的产品。比如说,Microsoft Word 是处理文档的程序,Microsoft Excel 是处理电子表格的程序,Microsoft PowerPoint 是制作演示文稿的程序。但如果要学习程序设计,那就必须从更深入的角度,即从程序员的角度,来理解程序。

为了更好地理解程序的概念,我们首先来看一个形象的例子。现在有一个机器人罗伯特,它可以为我们提供服务。前提是必须预先给罗伯特设计好工作步骤,并用罗伯特可以理解的语言描述工作步骤,然后存储到它的脑子里。比如说,让罗伯特把桌子上的铅笔交给我们,设定的工作步骤可能是:

“向左转”

“向前一步走”

“伸手拿铅笔”

“向右转”

“向前一步走”

“放下铅笔”

在这个例子中,罗伯特的角色相当于计算机硬件,6 条命令相当于罗伯特执行的程序(汉语描述的程序),而使用特定语言设计工作步骤的技术人员就是程序员。

由此可见,从程序员的角度看,程序就是按一定顺序组织的、能完成某项任务的一组计算机指令。程序设计的实质就是按一定的方法组织计算机指令,程序设计工作通常分两个阶段:(1)设计计算机的工作步骤;(2)用特定的语言把工作步骤描述出来。通常,前一阶段称为算法设计,后一阶段称为编码。

当然,真正的计算机并不理解汉语、英语或德语等自然语言。因此,计算机也不能理解用自然语言描述的工作步骤。在设计程序的时候,程序员必须使用特定的程序设计语言编制程序。

目前,计算机领域中可用的程序设计语言很多。比如说,FORTRAN, C/C++, Java 等。在不同的场合,可以选择不同的程序设计语言。通常,FORTRAN 多用于科学计算领域,Java 在互联网中被广泛使用, C/C++ 则特别适合开发系统软件。

1.2 C语言的历史与特点

C语言诞生于1972年,由著名的美国贝尔实验室科学家Dennis M. Ritchie发明。当时Ken Thompson刚刚使用汇编语言和B语言开发出UNIX操作系统,但用汇编语言开发系统非常烦琐,而B语言的功能又不够强大。为了更好地开发新版本的UNIX,Dennis M. Ritchie在B语言的基础上设计了C语言。除了系统的最核心部分,UNIX的后来版本基本都是用C语言开发的。可以说C语言和UNIX有着不可分割的渊源,C语言的不少特点就是由当时的UNIX所赐,有些特点一直保留到今天。

从1972年到现在,C语言已经成为世界上最流行的程序设计语言之一。C语言既可以用来开发系统软件,也可用来开发应用软件,它属于通用(general-purpose)的程序设计语言。C语言之所以深受程序员的欢迎,是因为它有很多独特的优点。

C语言的特点

(1) 简洁、紧凑。与学习自然语言一样,掌握任何程序设计语言都需要掌握一些关键字(key words,也称为保留字),即基本词汇。相对于其他语言,C语言的关键字比较少,便于记忆。另外,完成同样的任务,C程序往往比其他语言的程序短,因此有利于提高程序员的编程效率。

(2) 运算符、数据类型、数据结构丰富。C语言支持多种算术和逻辑运算,不仅可以处理基本类型的数据,如整型、实型、字符型等,还可以处理复杂结构的数据,如数组、记录、链表等。

(3) C语言具有部分低级语言的特点,因此能够用来开发系统程序。C语言允许程序根据地址直接访问内存,也允许程序按位处理数据,对于开发系统软件来说,这些特点都是不可或缺的。

(4) 运行效率高。在相同的计算机上完成相同的任务,C程序往往比其他语言的程序运行时间短,占用内存空间少。这是因为C语言与计算机硬件的关系比较紧密^①,用它编写的程序可以更有效地利用硬件资源,提高运行效率。当然,从另外一个角度看,因为与硬件紧密相关,所以学习C语言的时候往往需要了解一些计算机的基本原理,这也是C语言难学的原因之一。

(5) 可移植性好。在一种计算机上开发的C程序,经过少量的修改,甚至不经修改,就可以在其他类型的计算机上运行。C语言优良的可移植性能够有效地提高程序的利用率,避免不必要的重复开发。

(6) 模块化程度高。C程序由多个互不隶属的函数构成,因此在很大程度上减少了函数之间的相互影响。如同在工厂里组装机器,如果某零件的设计或制造缺陷对其他零件影响不大,那显然可以提高设计和制造的生产率。因为C语言的模块化程度好,也就是说函数(零

^① 程序设计语言与硬件的关系越紧密,其程序效率越高,要求程序员对计算机的理解越深。机器语言和汇编语言与硬件直接相关,属于低级语言。Pascal、FORTRAN、BASIC等语言与硬件的相关程度较低,属于高级语言。C语言介于二者之间,我们习惯上称其为高级语言,有时也被程序员看作中级语言。

件)之间的相互影响不大,因此为提高软件开发的效率提供了便利。

当然,C语言也并非没有缺点。除了与硬件相关程度高、对程序员的要求较高外,其过于灵活的书写风格、复杂的语法等特点通常会严重影响程序的可读性^①。因此,学习C程序设计的时候,要特别注意养成良好的编程习惯,遵守编程规范,提高程序的可读性。

1.3 简单的C程序

学习程序设计的最好方法就是阅读典型的例子,然后模仿例子,自己动手编写程序。根据这一原则,本节将介绍一个简单的例子,以便读者了解C程序的基本构成模式。

例 1-1 编写程序,在屏幕上显示两行文字^②:

This is my first program.
I will be a good programmer.

显然,完成这项任务的工作步骤非常简单。第一步显示“This is my first program.”,第二步在新的一行显示“I will be a good programmer.”。现在,问题的关键是如何用C语言把工作步骤描述出来。

```
#include <stdio.h>
main()
{
    printf("\nThis is my first program. ");
    printf("\nI will be a good programmer. ");
}
```

例 1-1 有一个 main(),它是程序的主函数。所有的C程序都由函数构成,每个程序有且只有一个主函数,它是程序执行的起点。

在C语言里,计算机执行的指令称为语句。语句必须包括在函数的一对大括号{ }之间,每条语句都必须以分号作为结束标志。

printf的作用是:无论小括号内的双引号之间是什么文字,程序执行时它们都被显示在屏幕上。因此如果需要在屏幕上显示文字,只要把文字放在双引号之间就可以了。

读者可能注意到,printf的显示文字中有换行符\n (a new line),它表示后面的文字将在

① 请读者特别注意:如果某人编写的程序,除了他自己,别的程序员很难看懂,则说明其程序的可读性差。我们认为他是蹩脚的程序员,决非高水平的权威,根本不会有公司聘任他。

② 本书以圆角的矩形表示屏幕,矩形内的文字表示显示的结果。

新的一行显示^①。如果删除两个\n，文字将显示在同一行，结果为：

This is my first program. I will be a good programmer.

程序经常需要在屏幕上输出文字或从键盘输入数据，但从计算机的工作原理上说，把文字显示在屏幕上是很烦琐的。如果没有其他帮助，程序员需要书写很多语句才能完成这项工作。通常，C语言的开发工具会提供给程序员一些库函数，每个库函数可完成一项基本功能。在程序设计时，程序员可直接使用库函数，而不必每次都书写很多语句。如同在工厂组装机，工人可以直接使用其他厂家提供的零件，而不必每次都使用原材料制造零件。例1-1中的printf就是一个库函数，它可以完成在屏幕上显示文字的任务。

C语言规定，在使用函数之前，首先要声明函数（即告诉计算机哪个零件将被使用）。stdio.h文件包含了许多标准输入输出函数的声明^②，用编译预处理命令#include <stdio.h>把文件包含到程序中，程序就可以随意使用stdio.h声明的任何函数^③，printf就是其中之一。请读者注意，#include <stdio.h>不是语句，而是编译预处理命令，因此该行的后面没有分号。

例1-1虽然简单，但反映了C程序的基本结构，即：

- (1) 程序有且只有一个main函数，它是程序执行的起点。
- (2) 语句包括在函数的大括号{ }之间。
- (3) 每条语句以分号为结束标志。
- (4) 程序从main的第一条语句开始顺序执行。
- (5) 如果使用库函数，必须首先把其对应的头文件（.h文件）包含到程序中。

通常情况下，按一定顺序组织的C语句称为源程序，也称为源代码。不过从原理上说，计算机并不能理解C语言源程序，其只能执行二进制程序，即机器语言程序。因此必须把C程序翻译成机器语言程序，然后才能被计算机执行。所幸的是，程序员不必关心如何翻译，编译软件可以完成这项工作。

C语言的相关编译软件很多，Turbo C 2.0就是其中之一。虽然实际的软件开发很少使用Turbo C 2.0，但由于其易学易用，所以对初学者来说仍然是个不错的实验工具^④。本书的1.10节将介绍翻译的过程，附录H中介绍了Turbo C 2.0的使用方法，请读者自行参阅。

1.4 变量

例1-1的程序非常简单，它没有进行任何数据计算。其实，C语言的功能很强大，可以

① 更多的程序员喜欢在显示一行文字之后就换行，比如说，

```
printf("This is my first program.\n");
printf("I will be a good programmer.\n");
```

② 通常，标准输入指的是从键盘输入，标准输出指的是在屏幕上显示。

③ 通常，我们称.h文件为头文件(header)。

④ 本书的例子都已在Turbo C 2.0环境下调试通过。

处理多种类型的数据。本节将简单介绍 C 程序处理数据的基本模式。

例 1-2 如果买钢笔花了 12 元钱, 买橡皮花了 6 元钱。编写程序计算总共花了多少钱, 并在屏幕上显示计算结果。

```
#include <stdio.h>
main()
{
    int pen, eraser, sum;
    pen=12;
    eraser=6;
    sum=pen+eraser;
    printf("\nThe sum is %d", sum);
}
```

■ int pen, eraser, sum;

该行声明: 程序将使用 3 个整型数据元素, 名字分别是 pen, eraser, sum, 它们分别表示买钢笔的费用、买橡皮的费用、总费用。^①

其中, int 是 C 语言的关键字, 表示整数类型^②。

pen, eraser, sum 称为变量, 用来存储被处理的数据元素。C 语言规定, 程序必须首先声明变量, 然后才能使用。声明变量的时候, 程序员必须给出变量的名字, 同时指明变量的数据类型。

■ pen=12;

eraser=6;

读者要注意, 这里的等号并不是数学方程中的相等。在 C 语言中, 等号是一种运算符, 表示赋值操作, 即把右边的计算结果保存到左边的变量中^③。上面的 2 条语句执行后, 12 被保存到 pen 中, 6 被保存到 eraser 中。

■ sum=pen+eraser;

显然, 右边的计算结果为 18, 保存到左边的 sum 变量中, sum 的值亦为 18。

■ printf("\nThe sum is %d ", sum);

该语句在屏幕上显示: The sum is 18。其中, %d 是格式说明, 表示数据按十进制格式输出^④。printf 执行的时候, 普通字符按原样显示在屏幕上, %d 被替换为数据的指定格式。在本例中, %d 将被替换为 18, 即 sum 的十进制整数格式。

① 本书对程序的解释采用这种格式。

② int 是英文 integer (整数) 的缩写。

③ 例: x=5;

x=x+2;

在数学方程中, $x=x+2$ 永远不能成立。在 C 程序中, 该语句表示: 把 $5+2$ 的结果 7 保存到 x 中, x 的数值由 5 变成 7。这也是为什么称 x 为变量的原因。

④ d 是 decimal (十进制) 的首字母。

程序的执行结果是：

The sum is 18

例 1-2 展示了 C 程序处理数据的基本模式，即：

- (1) 被处理的数据通常保存在变量中。
- (2) 首先要声明变量，声明就是给变量命名^①，并指明它的数据类型。
- (3) 可以使用赋值运算符（等号）将数值保存到变量中。
- (4) 赋值后的变量可参加各种运算，如加、减、乘、除等。
- (5) 数据可以按一定的格式显示在屏幕上。

当然，C 语言可处理的数据不仅仅整型一种，本书将在第 2 章详细讨论 C 语言的基本数据类型。

1.5 循环

聪明的读者可能会提出疑问，按照 1.4 节的程序模式，如果要计算 $\sum_{i=1}^{50} i$ ，岂不是要声明 50 个变量，写 50 条语句，那不是太烦人了吗^②？不必担心，C 语言可以让程序员轻松解决这一问题。

计算 $\sum_{i=1}^{50} i$ ，需要计算机重复做 50 次加法运算。使用循环语句，可以规定计算机重复执行的次数（或条件）。因此，程序员只需要书写 1 条语句，即只向计算机下 1 条命令，就可以完成类似的工作。

例 1-3 编写程序计算 $\sum_{i=1}^{50} i$ ，并在屏幕上显示结果。

```
#include <stdio.h>
main()
{
    int sum, counter;
    sum=0;
    for(counter=1; counter<=50; counter=counter+1){
        sum=sum+counter;
```

① 变量的命名也要遵守一定的规则，本书的第 2 章将给出变量命名的详细规则。

② 古代有富家子弟学字，先生教之曰：一横为一，两横为二，三横为三。富家子弟大悟，以为得之，遂逐师。恰有人姓万，来访。令是儿书客名，自晨至午，仅画五百余横。