SAMS

C Primer Plus（第五版）中文版

〔美〕Stephen Prata 著
云巅工作室 译

人民邮电出版社
POSTS & TELECOM PRESS

C Primer Plus（第五版）中文版

[美] Stephen Prata 著

云巅工作室 译

人民邮电出版社

图书在版编目（CIP）数据

C Primer Plus：第5版 /（美）普拉塔（Prata, S.）著；云巅工作室译.
—北京：人民邮电出版社，2005.2
ISBN 7-115-13022-1

I. C… II. ①普…②云… III. C语言—程序设计 IV. TP312

中国版本图书馆CIP数据核字（2005）第004888号

版 权 声 明

C Primer Plus（第五版）中文版

◆ 著 [美] Stephen Prata
译 云巅工作室
责任编辑 陈冀康

◆ 人民邮电出版社出版发行 北京市崇文区夕照寺街14号
邮编 100061 电子函件 315@ptpress.com.cn
网址 http://www.ptpress.com.cn
读者热线 010-67132705
北京顺义振华印刷厂印刷
新华书店总店北京发行所经销

◆ 开本：787×1092 1/16
印张：40
字数：1 298千字 2005年2月第1版
印数：1-4 000册 2005年2月北京第1次印刷

著作权合同登记 图字：01-2004-4417号
ISBN 7-115-13022-1/TP·4411
定价：60.00元

内 容 提 要

本书全面讲述了C语言编程的相关概念和知识。

全书共17章。第1、2章学习C语言编程所需的预备知识。第3到15章介绍了C语言的相关知识，包括数据类型、格式化输入输出、运算符、表达式、流程控制语句、函数、数组和指针、字符串操作、内存管理、位操作等等，知识内容都针对C99标准；另外，第10章强化了对指针的讨论，第12章引入了动态内存分配的概念，这些内容更加适合读者的需求。第16章和第17章讨论了C预处理器和C库函数、高级数据表示（数据结构）方面的内容。附录给出了各章后面复习题、编程练习的答案和丰富的C编程参考资料。

本书适合希望系统学习C语言的读者，也适用于精通其他编程语言并希望进一步掌握和巩固C编程技术的程序员。

前　言

1984 年，当 *C Primer Plus* 的第一版刚刚完稿的时候，C 还是一种相对鲜为人知的语言。这种语言从那时才开始兴起，很多人都是在该书的帮助下掌握 C 语言的。实际上，已经有超过 50 万的人购买过 *C Primer Plus* 的各个不同版本的书。

随着 C 语言从最初的非正式的 K&R 标准过渡到 1990 ISO/ANSI 标准，进而发展到 1999 ISO/ANSI 标准，*C Primer Plus* 也不断地成熟，并发展到第五版。在所有这些版本中，我的目标都是致力于编写一本富有指导性的、清晰的 C 语言教程。

本书的方法和目标

我编写这本书的目标是让人们能够把它当作一个友好的、易于使用的、便于自学的指南。为了实现这个目标，本书采用了以下的策略：

- 在介绍 C 语言细节的同时，还阐述了编程概念。本书假定读者并非专业的程序员。
- 每次通过很多简短的、易于录入的实例来说明一两个概念，因为边干边学是掌握新的信息的最有效的方式之一。
- 只用语言难以阐述的概念，采用图表来澄清。
- 突出显示的板块总结了 C 语言的主要特征，以便于参考和复习。
- 每章最后的复习题和编程练习帮助你测试和加深对 C 语言的理解。

为了求得最佳学习效果，在学习本书内容的时候，你应该尽可能地扮演一个积极的角色。不仅只是阅读例子，还要把它们输入到你的系统，然后运行。C 是一种可移植性很好的语言，但你还是会发现某个程序在你的系统下运行的结果和在我们的系统下运行的结果会有所不同。不妨做个试验，改变程序的某一部分来看看有什么效果。修改程序来做略微有些不同的事情。不必理会无关的警告，主要是看一下执行了一个错误操作时会发生什么。尝试提出问题和做练习。实践的越多，你所学到和记住的也就越多。

我希望你能够通过本书最新的版本，愉快而又高效地走入 C 语言的学习殿堂。

关于作者

Stephen Prata 在加利福尼亚州的 Kentfield 得 Marin 学院教授天文学、物理学和程序设计课程。他在加州工业学院获得学士学位，从加州大学伯克利分校获得博士学位。他最早接触计算机，始于对星河的计算机建模。Stephen 已经编写或与他人合作编写了十多本书，其中包括 *C++ Primer Plus* 和 *Unix Primer Plus*。

目　录

第1章 概 览

在本章中您将学习下列内容：

- C的历史和特性。
- 编写程序所需的步骤。
- 关于编译器和链接器的一些知识。
- C的标准。

欢迎来到C的世界！C语言是一种强大的专业化编程语言，深受业余和专业编程人员的欢迎。本章为学习和使用这一强大而流行的语言做准备，并介绍了开发C程序时最可能使用的几种环境。

首先，让我们来看一看C的起源及其特性，包括它有哪些优点和缺点。接着我们将了解编程的起源并探讨编程的一些基本原则。最后，我们讨论在一些常见系统上运行C程序的方法。

1.1 C语言的起源

贝尔实验室的Dennis Ritchie在1972年开发了C，当时他正与Ken Thompson一起设计UNIX操作系统。然而，C并不是完全由Ritchie构想出来的。它来自Thompson的B语言，而B语言则来自……噢，这又是另外一个故事了。重要的是，C是作为从事实际编程工作的程序员的一种工具而出现的，所以其主要目标是成为一种有用的语言。

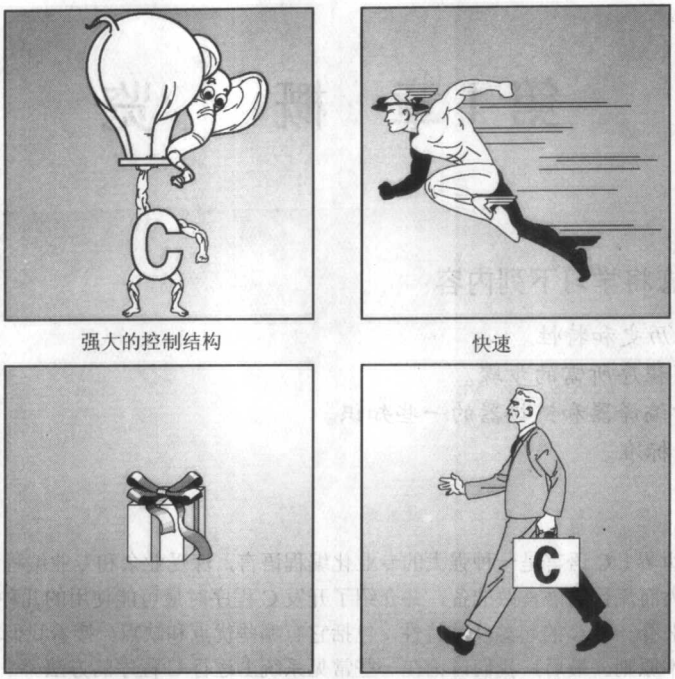
多数语言都以实用为目标，但它们往往也会考虑其他一些方面。例如，Pascal的主要目标是为学习良好的编程原则提供一个扎实的基础，而BASIC则是模仿英语，以便让不熟悉计算机的学生能够轻松地学会这种语言。这些目标很重要，但它们并不总是与实际的使用需要相符。而C则是为编程人员开发的语言，这使得它成为当今人们首选的编程语言之一。

1.2 使用C语言的理由

在过去的30年中，C已经成为最重要和最流行的编程语言之一。它之所以得到发展，是因为人们尝试使用它后都喜欢它。过去10年中，许多人从C转而使用更强大的C++语言，但C有其自身的优势，仍然是一种重要的语言，而且它还是通往C++的必由之路。学习C的过程中，您将认识到它的许多优点（见图1.1）。现在让我们首先来看其中的几个优点。

1.2.1 设计特性

C是一种融合了控制特性的现代语言，而我们已发现在计算机科学的理论和实践中，控制特性是很重要的。其设计使得用户可以自然地采用自顶向下的规划、结构化的编程，以及模块化的设计。这种做法使得编写出的程序更可靠、更易懂。



强大的控制结构　　快速

紧凑的代码——程序更小　　可移植到其他计算机

图 1.1　C 的优点

1.2.2　高效性

C 是一种高效的语言。在设计上它充分利用了当前计算机在能力上的优点。C 程序往往很紧凑且运行速度快。事实上，C 可以表现出通常只有汇编语言才具有的精细控制能力（汇编语言是特定的 CPU 设计所采用的一组内部指令的助记符。不同的 CPU 类型使用不同的汇编语言）。如果愿意，您可以细调程序以获得最大速度或最大内存使用率。

1.2.3　可移植性

C 是一种可移植语言。这意味着，在一个系统上编写的 C 程序经过很少改动或不经修改就可以在其他系统上运行。如果修改是必要的，则通常只须改变伴随主程序的一个头文件中的几项内容即可。多数语言原本都想具有可移植性，但任何曾将 IBM PC BASIC 程序转换为 Apple BASIC 程序（它们还是近亲）的人，或者试图在 UNIX 系统上运行一个 IBM 大型机 FORTRAN 程序的人都知道，移植至少是在制造麻烦。C 在可移植性方面处于领先地位。C 编译器（将 C 代码转换为计算机内部使用的指令的程序）在大约 40 种系统上可用，包括从使用 8 位微处理器的计算机到 Cray 超级计算机。不过要知道，程序中为访问特定硬件设备（例如显示器）或操作系统（如 Windows XP 或 OS X）的特殊功能而专门编写的部分，通常是不能移植的。

由于 C 与 UNIX 的紧密联系，UNIX 系统通常都带有一个 C 编译器作为程序包的一部分。Linux 中同样也包括一个 C 编译器。个人计算机，包括运行不同版本的 Windows 和 Macintosh 的 PC，可使用若干种 C 编译器。所以不论您使用的是家用计算机、专业工作站还是大型机，都很容易得到针对您的特定系统的 C 编译器。

1.2.4　强大的功能和灵活性

C 强大而又灵活（计算机世界中经常使用的两个词）。例如，强大而灵活的 UNIX 操作系统的大部分便

是用 C 编写的。其他语言（如 FORTRAN、Perl、Python、Pascal、LISP、Logo 和 BASIC）的许多编译器和解释器也都是用 C 编写的。结果是，当您在一台 UNIX 机器上使用 FORTRAN 时，最终是由一个 C 程序负责生成最后的可执行程序的。C 程序已经用于解决物理学和工程学问题，甚至用来为《角斗士》这样的电影制造特殊效果。

1.2.5　面向编程人员

C 面向编程人员的需要。它允许您访问硬件，并可以操纵内存中的特定位。它具有丰富的运算符供选择，让您能够简洁地表达自己的意图。在限制您所能做的事情方面，C 不如 Pascal 这样的语言严格。这种灵活性是优点，同时也是一种危险。优点在于：许多任务（如转换数据形式）在 C 中都简单得多。危险在于：使用 C 时，您可能会犯在使用其他一些语言时不可能犯的错误。C 给予您更多的自由，但同时也让您承担更大的风险。

另外，多数 C 实现都有一个大型的库，其中包含有用的 C 函数。这些函数能够处理编程人员通常会面对的许多需求。

1.2.6　缺点

C 确实有一些缺点。和人一样，缺点和优点往往是同一特征相对的两个方面。例如，我们前面曾说过，C 在表达方面的自由会增加风险。尤其是 C 对指针（在本书后面部分将学到）的使用，意味着您可能会犯非常难以追踪的编程错误。正如以前一位计算机专家曾经指出的，自由的代价是永远的警惕。

C 的简洁性与其丰富的运算符相结合，使其可能会编写出极难理解的代码。没有谁强迫您编写含糊难懂的代码，但存在这样的可能性。试问，除 C 之外还有哪种语言存在一年一度的“含糊代码”（Obfuscated Code）竞赛呢？

此外，C 还有许多的优点，但毫无疑问，C 还有一些缺点。我们不想在这一点上多费笔墨，还是换一个新的话题吧。

1.3　C 语言的发展方向

20 世纪 80 年代初，C 在 UNIX 系统的小型机世界中已经是主导语言了。从那时开始，它已经扩展到个人计算机（微型机）和大型机（庞然大物），如图 1.2 所示。许多软件开发商都首选 C 语言来开发其子处理程序、电子表格软件、编译器和其他产品。这些公司知道，C 可以产生紧凑而高效的程序。更重要的是，他们知道这些程序易于修改而且易于适应新的计算机模式。

对于公司和熟悉 C 语言的人有益的东西，对其他用户同样有益。越来越多的计算机用户已转向使用 C 以便利用其优点。不一定非得是计算机专业人员才能使用 C。

在 20 世纪 90 年代，许多软件开发商开始转向使用 C++语言来进行大的编程项目。C++向 C 语言嫁接了面向对象编程工具（面向对象编程是一种哲学思想，它试图让语言来适应问题，而不是让问题来适应语言）。C++差不多是 C 的一个超集，意味着任何 C 程序都同时是，或差不多是一个有效的 C++程序。通过学习 C，您还会学习到 C++的许多知识。

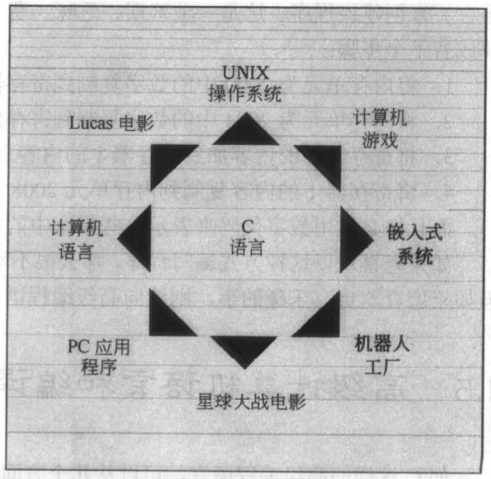


图 1.2　C 的应用领域

不管 C++和 Java 这样较新的语言如何流行，C 在软件产业中仍然是一种重要的技能，在最想获得的技能中，它一般都列在前 10 名。特别是在嵌入式系统的编程中，C 已开始流行。也就是说，它将用来为汽车、照相机、DVD 播放器和其他现代化设备中逐渐普及的微处理器编程。同样，C 已开始进入长期以来一直属于 FORTRAN 的科学编程领域。最后，由于它是一种适合用来开发操作系统的语言，C 在 Linux 的开发中也扮演着重要的角色。因此，在 21 世纪的前 10 年中，C 仍将保持强劲的势头。

简言之，C 是最重要的编程语言之一，并将继续如此。如果您想找一份编写软件的工作，则首先您应该能够回答“是”的一个问题就是：“请问，您会使用 C 吗？”

1.4 计算机工作的基本原理

既然打算学习如何用 C 编程，您就应了解计算机工作原理方面的一些知识。这些知识会帮助您理解用 C 编写程序与运行该程序时最终会发生的事情之间的联系。

现代计算机可分为几个部件。中央处理单元（或称 CPU）担负着绝大部分的计算工作；随机访问存储器（或称 RAM）作为一个工作区来保存程序和文件；永久存储器，一般是硬盘，即使在计算机关机时也能记下程序和文件；还有各种外围设备（如键盘、鼠标和监视器）用来提供人与计算机之间的通信。CPU 负责处理程序，所以我们集中来讨论它的功能。

CPU 的工作非常简单，至少在我们所做的这一简短描述中是这样的。它从内存中获取一个指令并执行该指令，然后从内存中获取下一个指令并执行。一个千兆 CPU 可以在一秒钟内进行大约一亿次这样的操作，所以 CPU 能以惊人的速度来从事其枯燥的工作。CPU 有自己的小工作区，该工作区由若干个寄存器（registers）组成，每个寄存器可以保存一个数。一个寄存器保存下一条指令的内存地址，CPU 使用该信息获取下一条指令。获取一条指令后，CPU 在另一个寄存器中保存该指令并将第一个寄存器的值更新为下一条指令的地址。CPU 只能理解有限的指令（指令集）。还有，这些指令是相当具体的，其中许多指令要求计算机将一个数从一个位置移动到另一个位置，例如，从内存单元移到寄存器。

这段说明中有两个有趣的地方。首先，存储在计算机中的一切内容都是数字。数字是以数字形式存储的，字符（如文本文档中使用的字母字符）也是以数字形式存储的，每个字符有一个数字代码。计算机装载到寄存器中的指令是以数字形式存储的，指令集中的每条指令具有一个数字代码。其次，计算机程序最终必须以这种数字指令代码（或称为机器语言）来表示。

明白了计算机运行方式的一个结果就是：如果您希望计算机做某件事，就必须提供一个特定的指令列表（一套程序）确切地告诉计算机要做的事及如何去做。您必须以一种计算机可以直接理解的语言（机器语言）来创建该程序。这是一项繁琐、乏味、费力的任务。即使将两个数字相加这样简单的事也必须被分解成若干个步骤：

1. 将内存单元为 2000 中的数字复制到寄存器 1。
2. 将内存单元为 2004 中的数字复制到寄存器 2。
3. 将寄存器 2 的内容加到寄存器 1 的内容上，答案保留在寄存器 1 中。
4. 将寄存器 1 的内容复制到内存单元 2008。

而且您必须用数字代码来表示这些指令中的每一个！

如果您喜欢以这种方式编写程序，那么很不幸，您将会发现机器语言编程的黄金时期已经过去很久了。但如果您喜欢更有乐趣的事，则请向高级编程语言敞开您的心扉。

1.5 高级计算机语言和编译器

如 C 这样的高级编程语言，可以从几个方面简化您的编程过程。首先，您不必用数字代码表示指令。其次，您所使用的指令更接近您考虑问题的方式，而非接近计算机使用的详细操作步骤。现在您不用再考

虑特定 CPU 实现特定任务所必须采取的精确步骤，而是可以在更抽象的层次上表达您的意图。例如，要对两个数求和，您可以编写下列内容：

```
total = mine + yours;
```

看到这样的代码，您就会清楚地知道它的作用。但如果看到用数字代码表示的由若干条指令组成的机器语言等价代码，则不会让人这么明白。

不幸的是，对计算机来说正好相反。对计算机来说，高级指令是不能理解的胡言乱语。而这正是出现编译器的原因。编译器是将高级语言程序解释成计算机所需的详细机器语言指令集的程序。您进行高级思考，编译器则负责乏味的琐碎工作。

采用编译器还有另一个好处。一般来说，每种计算机在设计上都有其自身特有的机器语言。所以用机器语言为一个 Intel Pentium CPU 编写的程序对 Motorola PowerPC CPU 来说什么都不是。但您可以将编译器匹配一种特定的机器语言。这样，使用正确的编译器或编译器集，您就可以将同一高级语言程序转换为各种不同的机器语言程序。您解决一个编程问题只须一次，然后可以让编译器将该解决方案解释为各种机器语言。

简言之，高级语言（如 C、Java 和 Pascal）都以更抽象的方式描述动作，并且没有与特定的 CPU 或指令集相关联。同样，高级语言更易于学习，而且用高级语言编写程序比用机器语言容易得多。

1.6　使用 C 语言的 7 个步骤

正如您所看到的，C 是一种编译性语言。如果您习惯于使用编译性语言，例如 Pascal 或 FORTRAN，您会熟悉建立 C 程序的基本步骤。然而，如果您的背景是解释性语言（例如 BASIC），或面向图形界面的语言（例如 Visual Basic），或者您根本没有任何背景，则需要学习如何进行编译。我们很快就会看到这个过程，您会看到该过程直截了当而且容易理解。首先，为了让您对编程有一个概括了解，我们将编写 C 程序的过程分解为 7 个步骤（见图 1.3）。注意这是理想化的。在实践中，尤其是在较大的项目中，您可能需要做一些反复工作，用后一步骤中所了解到的内容来改进前一个步骤。

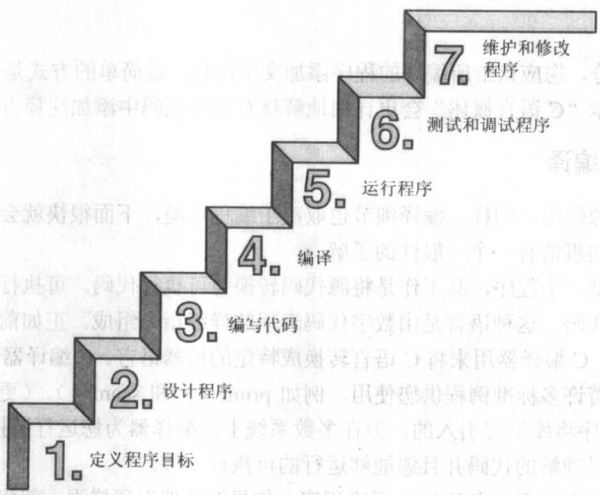


图 1.3　编程的 7 个步骤

1.6.1　第 1 步：定义程序目标

非常自然地，在开始时，您应对希望程序做什么有一个清晰的想法。考虑程序需要的信息、程序需要