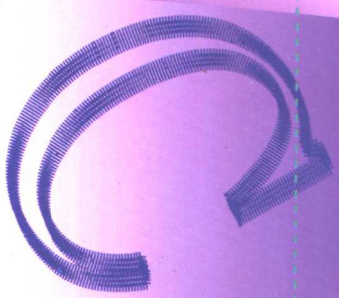
UNIX C程序设计
入门与应用

洪锦魁 编著



- C语言程序设计基础知识
- C语言数据结构
- UNIX系统程序设计

清华大学出版社

UNIX C 程序设计入门与应用

洪锦魁　编著

清华大学出版社

(京)新登字 158 号

内 容 简 介

本书主要介绍 C 语言入门与应用的基础知识，适用于 UNIX 系统。本书共分 3 篇，分别是 C 语言程序设计基础知识、C 语言数据结构和 UNIX 系统程序设计。第 I 篇介绍 C 语言的基础知识，包括基本概念、基本的输入与输出、程序的流程控制、数组和指针、函数变量等；第 II 篇介绍 C 语言数据结构，包括基本表结构、堆栈与队列、二叉树等；第III篇介绍 UNIX 系统的文件管理、程序管理。

本书内容编排合理、由浅入深，程序示例丰富，非常适合对 C 语言感兴趣的初、中级读者使用。

书　　名： UNIX C 程序设计入门与应用
作　　者： 洪锦魁　编著
责任编辑： 杨志娟
出 版 者： 清华大学出版社(北京清华大学学研大厦，邮编 100084)
http://www.tup.tsinghua.edu.cn
印 刷 者： 北京市清华园胶印厂
发 行 者： 新华书店总店北京发行所
开　　本： 787×1092 1/16　**印张：** 20.25　**字数：** 462 千字
版　　次： 2002 年 8 月第 1 版　2002 年 8 月第 1 次印刷
书　　号： ISBN 7-302-05713-3/TP・3370
印　　数： 0001～4000
定　　价： 28.00 元

前　言

随着计算机科学的发展，C 语言的普及和应用面越来越广，要求学习和掌握 C 语言的人越来越多。因为 C 语言同时具备低级汇编语言的功能和高级语言的亲和性，自然也就成了计算机语言的主流。

本书主要介绍如何在 UNIX 系统上使用 C 语言。为了方便读者学习，本书用了将近 220 个程序范例，一步一步对读者进行引导。相信读者只要花时间去学习这些程序，必定可在短时间内精通 C 语言编程。

全书分为 3 篇，共 21 章。第 I 篇为 C 语言程序设计基础，主要介绍 C 语言和一般 C 语言函数的用法；第 II 篇为 C 语言数据结构，通过程序范例说明基本数据结构的应用；第 III篇为 UNIX 系统程序设计，列出了基本 UNIX 系统调用，同时用程序范例说明如何使用这些系统调用来设计有用的系统程序。

第 I 篇共 13 章，具体内容如下：

第 1 章简要介绍 C 语言的发展和 C 语言程序的开发过程。

第 2 章介绍使用 C 语言处理数据的基本概念，包括变量、基本的数据类型、常量的表达方式等。

第 3 章讲述基本的输入与输出，包括 printf()、scanf()、putchar()和 getchar()等函数。

第 4 章介绍 C 语言程序的流程控制，包括程序流程使用的关系运算符和流程控制指令。

第 5 章介绍数组的指针，包括一维和二维数组的使用方式、指针的基本概念，以及数组与指针配合使用的基本常识。

第 6 章介绍函数的应用，包括函数的声明、调用、使用以及程序间参数的传递方式。

第 7 章介绍变量的等级，包括变量的类别和应用。

第 8 章介绍 C 语言的预处理程序，包括预处理程序的定义和用法。

第 9 章介绍自定义数据类型，帮助读者建立自定义数据类型的概念，并通过实例讲解它的使用方法。

第 10 章介绍字符串，包括字符数组和字符串的差异、基本字符串的输入与输出、字符串函数的用法。

第 11 章介绍基本位运算，包括 C 语言的位运算指令及其用法。

第 12 章介绍测试符号与符号转换函数，包括这些函数的用法与应用。

第 13 章介绍文件的输入与输出，包括文件指针的基本知识，以及如何使用文件指针进行一般的文件输入与输出。

第 II 篇共 3 章，具体内容如下：

第 14 章介绍基本的表结构，包括表结构的基本知识，如何配置和释放存储器空间，以及如何将指针的结构数据合成表结构。

第15章介绍堆栈与队列，包括堆栈与队列的基本概念及其应用。

第16章介绍二叉树，包括树状数据结构的基本定义、二叉树的创建方法，以及打印二叉树的基本函数。

第III篇共5章，具体内容如下：

第17章介绍UNIX系统的文件管理，包括文件管理的基本知识、文件的基本操作。

第18章介绍程序的管理，包括进程的定义、建立及应用。

第19章介绍进程间的信息传递，包括进程通信的基本定义，以及进程间通信程序的设计。

第20章介绍屏幕控制，包括屏幕方位的定义，以及如何利用curses函数实现动态画面。

第21章是其他相关函数的总结，介绍了屏幕与文件之间的关系、时间函数的使用原则。

由于我们水平有限，错误之处在所难免，恳请广大读者不吝指正。

目　录

第 I 篇 C 语言程序设计基础

学习 C 语言的首要任务是了解 C 语言数据处理的基本概念。因此本书特将这一部分安排在第 1 章和第 2 章中来说明。

至于 C 语言的其他概念，则分门别类在本篇的其他章节内进行叙述。对于 C 语言而言，另一个很重要的概念是函数的应用，本篇也将一一介绍，以便读者在阅读完本书之后，能对 C 语言有一个综合性的了解。

学习 C 语言最终的目标是能以有效的方式充分地利用系统资源，以实现软件取代硬件的目标。所以，读完本篇后，应该继续阅读第 II 篇 C 语言的数据结构，以了解应该如何有效地设计 C 语言程序。然后学习第III篇 UNIX 的系统资源，以了解应该如何利用 UNIX 的系统资源。

本篇内容：

第 1 章　UNIX 系统 C 语言简介

由于 C 语言同时具备低级汇编语言的功能和高级语言的亲和性，自然就成为未来计算机语言的主流。

本章将对 C 语言的发展做一个简单的介绍。同时，用程序示例说明 C 语言的基本结构、编译以及执行方式。至于 C 语言的其他细节，在以后的章节中将分别介绍。

本章学习目标

- 了解 C 语言的发展
- 熟悉 C 语言的基本构架、编译以及程序执行方法

1.1　C 语言简介

C 语言是 1972 年 Rennis Ritchie 和 Ken Thompson 两人在一起设计 UNIX 操作系统时开发出来的，由于 C 的起源就是为了要设计 UNIX 操作系统，如果只叙述 C 语言本身，而不涉及系统设计，那么读者将无法深入了解 C 语言的实际好处。因此，在本书的第III篇，会详细地说明 UNIX 系统程序设计。

由于 UNIX 已经成为微机或者中型机的主要操作系统，C 语言也就成为这些计算机的主要程序语言。另外，IBM 公司的大型机原本没有 C 编译程序，由于大势所趋也不得不安装这个功能强大的编译程序。在个人计算机中，Turbo C 和 Microsoft C 也成为了学生以及小软件公司的主要程序语言。

1.2　C 语言程序的开发过程

用 C 语言来编写程序，从设计到最后的执行，一般按照下列步骤进行：

(1) 程序规划。
(2) 利用编辑程序编写源程序。
(3) 编译和连接程序，此时系统将产生可执行模块。
(4) 执行此程序。

图 1.1 是上述步骤的流程说明。

1. 规划程序

初学计算机的人常会对程序操作感到头疼。一般而言，要克服困难只有一个方法，那就是多做练习。

惟有多练习才可精益求精。当然，在规划程序过程中，也有简单的基本原则可循。那就是按照程序操作，首先检查有哪些输入的数据，输出的要求是什么。可以将程序操作分成几个区段，然后再一个一个组合起来。图 1.2 是程序操作的流程图。

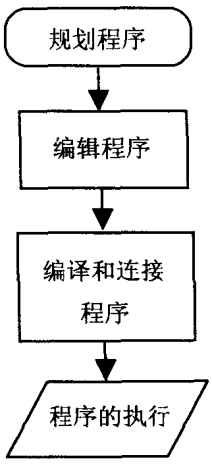


图1.1 C语言程序的流程图

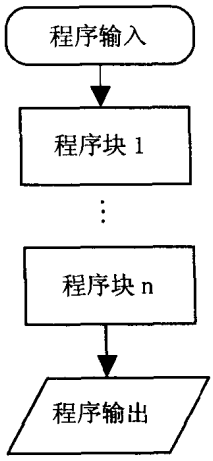


图1.2 规划过程流程图

2. 程序的编辑

在 UNIX 系统中，不像用于个人计算机的 Turbo C 那样有属于自己的全屏幕编辑程序。不过 UNIX 系统上可用的编辑程序有很多，如 ed、ex、edit 和 vi。本书中的程序大多是利用 vi 全屏幕编辑程序完成的。

C 语言程序的文件名称，一般有两部分组成：一个是主文件名，另一个是扩展文件名。其中主文件名可自行命名，扩展文件名是 .c。例如，有一个程序主文件名是 ch1，则这个程序的全名为 ch1.c。

注意： 主文件名和扩展文件名使用点分隔。

一般程序设计，大多是选用能够表示计算机功能的名字来做文件名称。例如，如果设计一个时钟程序，可用 clock.c。

如果设计一个排序程序，则用 sort.c。

ch 是 chapter(章节)的缩写，为了使本书所有程序保持一致性，本书中所有程序都以 ch 开头。

3. 编译和连接程序

UNIX 系统内，C 语言的编译和连接程序称为 cc，要编译时只要输入下列格式即可：

```
cc  文件名
```

假如现在要编译的程序是 ch1.c，则需要输入下列指令：

```
cc ch1.c
```

注意： 在 UNIX 系统中，cc 不仅具有编译的功能，同时也有连接的功能。因此，执

行完 cc 之后，只要程序没有错，即可执行。

4. 程序的执行

在上一节中，程序被编译和连接完之后，会产生一个可执行文件，称为 a.out。要执行此文件，只要输入下列指令即可：

```
a.out
```

如果不特别设置，所有的 C 语言程序，编译完后的可执行文件都为 a.out。当然也可以更改这个可执行文件，下一节将用程序示例来说明这个概念。

1.3　简单的 C 语言程序范例

本节将介绍基本 C 程序的结构，让读者了解 C 语言结构后，再继续说明各种 C 程序的语法。这样，就可以循序渐进地进入 C 语言世界了。

程序范例ch1.c

一个简单的输出程序，其代码如下。

```
01 /*======== program description================ */
02 /* program name:ch1.c                          */
03 /* A first program in c                        */
04 /*============================================= */
05
06 main ()
07 {
08   int i;
09
10    i=1;
11    printf("An Introduction in UNEX C\n");
12    printf("program exercise %\n",i);
13    printf("C is fantastic \n");
14 }
```

要执行上面的程序，可以先输入下列指令，进行编译和连接。

```
cc ch1.c
```

当完成编译和连接之后，系统会自动产生一个文件，文件名为 a.out。要执行此程序，只要输入下列指令就可以了。

```
a.out
```

下面是本程序的执行结果：

```
An Introduction in UNIX C
Program exercise 1
C is fantastic
```

1. 程序的行号

在程序范例 ch1.c 中，在程序的左边有程序的行号。其实 C 语言程序是没有行号的，在此之所以加行号，主要是为了阅读方便。所以在输入程序的时候，可自行将行号删除。

2. 程序的注释

程序范例中，第 1～4 行是注释，如下所示：

```
01 /*========= program description ========*/
02 /*   program name:ch1.c                  */
03 /*A first program in c                    */
04 /* ======================================*/
```

凡是介于 /* 和 */ 之间的文字，C 的编译程序均会忽略，不予编译。设计程序时，最好养成写注释的习惯，便于以后阅读。

3. 主程序main()声明

所有的 C 语言程序均由 main()开头。C 语言程序会执行 { 和 } 之间的内容。在此例子中，此程序会执行第 7～14 行之间的内容。

C 语言的基本结构是：

```
main( )
{
}
```

4. 程序的内容

在程序范例中，第 8～13 行都属于程序内容，如下所示：

```
08    int  i;
09
10    i=1;
11    printf("An Introduction in UNIX C\n");
12    printf("Program exercise %d\n",i);
13    printf("C is fantastic  \n");
```

注意： *在每一个完整的语句后面，一定要加上分号(;)，表示每一个语句的结束。*

5. 变量的声明

在 C 语言中，所有的变量在使用前，一定要加以声明，以便 C 编译程序为这个变量安排一个存储空间。以后使用此变量时，C 编译程序就会自行到这个存储空间存取数据。

范例：在程序代码的第 8 行中，声明了[int i;]以后，整个存储器如图 1.3 所示。

范例：在程序代码的第 10 行中定义[i=1;]时，整个存储器将如图 1.4 所示。

至于其他更详细的变量声明的原则，下一章再做讨论。

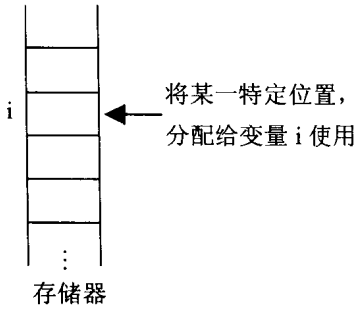


图1.3　变量声明与内存空间

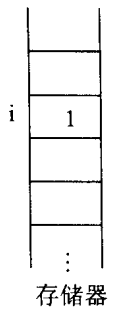


图1.4　变量使用与内存空间

6. 程序范例ch1.c的解释说明

在程序范例中，预留了很多空白行，例如：第 5 行及第 9 行。这样是为了使程序看起来不拥挤。C 编译程序在看到这些行时，会将其忽略。

以下是整个程序的说明：

- 第 1～4 行，是程序的注释。
- 第 6 行说明这是主程序。
- 第 7 行“{”(左大括号)，表示主程序从这里开始。
- 第 8 行是把变量 i 声明为整数(int)。
- 第 10 行是将变量 i 的值设置为 1。
- 第 11 行是打印下列字符串：

```
An Introduction in UNIX c
```

- 第 12 行是打印下列字符串：

```
Program Exercise 1
```

- 第 13 行是打印下列字符串：

```
C is fantastic
```

- 第 14 行“}”(右大括号)，在此表示主程序结束。

有关程序输入与输出的原则，将留待第 3 章再进行说明。

7. 附注

如果有下列指令编译程序：

```
cc   -o  outch1  ch1.c
```

可以自行设置任何可执行文件名

则编译完本程序之后，程序的可执行文件将被存放在 outch1 内，此时可以直接输入 outch1 执行此文件。

第 2 章　C 语言数据处理的基本概念

程序语言最基本的数据处理对象就是变量和常量，本章将对所有的变量和常量进行解释说明。另外，C 语言程序拥有的很多不同于其他高级语言的运算符，本章也将一一说明。

本章学习目标

- 了解变量的使用和声明方式
- 了解 C 语言的各种数据类型和基本运算
- 了解 C 语言的一些特殊运算

2.1　变量名称的使用

C 语言对变量名称的使用是有一些限制的，变量名必须以 3 种字符作为开头即大写字母、小写字母和下划线(_)；至于变量名称的组成则是由 4 种字符所构成：大写字母、小写字母、下划线(_)和阿拉伯数字 0～9。

范例：下列均是合法的变量名称:

```
SUM
hung
sum_1
_fg
x5
y61
```

范例：下列均是不合法的变量名称:

```
sum,1    ←变量名称不可有逗号(，)
3y       ←变量名称不可由阿拉伯数字开头
x$2      ←变量名称不可含有$符号
```

另外，需要注意的是，在 C 语言中大写字母和小写字母代表不同的变量。

范例：下列 3 个字串，分别代表 3 个不同的变量。

```
sum
Sum
SUM
```

有关变量使用另一点需要注意的是，有些字为系统保留字(也称为关键字)，在 C 语言