

GENERAL
EDUCATION

高等学校通识教育系列教材

C语言程序设计

(基于CDIO思想) (第2版)

郑晓健 布瑞琴 李向阳 主编
李向阳 主审



清华大学出版社



GENERAL EDUCATION

高等学校通识教育系列教材

C语言程序设计 (基于CDIO思想) (第2版)

郑晓健 布瑞琴 李向阳 主编

高世健 周 波 方娇莉 副主编

清华大学出版社

北 京

内 容 简 介

本书采用 CDIO 的思想和模式编写,注重培养读者的 C 语言程序设计能力和独立解决问题的能力。与第 1 版相比,第 2 版在内容上进行了充实、修改和删减,并在每章增加了实验实训,目的是提高读者的综合运用能力。

全书内容分为 3 个层次:第 1~4 章是基础篇,介绍 C 语言的顺序结构、选择结构和循环结构的程序设计,以及相关语法知识;第 5~7 章是提升篇,进一步介绍程序设计方法、相关算法和技巧,内容包括数组、函数、变量的存储类型和编译预处理、用户定制数据类型等;第 8~9 章是深入篇,介绍指针、文件的程序设计知识。第 10 章介绍 Visual C++ 6.0 开发环境。

本书内容丰富,注重实用,强调基础,目的是为计算机专业建设精品课程打下良好的基础。本书既可作为计算机及相关专业的教材,也可作为非计算机专业公共课程的教材。

本书封面贴有清华大学出版社防伪标签,无标签者不得销售。

版权所有,侵权必究。侵权举报电话:010-62782989 13701121933

图书在版编目(CIP)数据

C 语言程序设计:基于 CDIO 思想/郑晓健,布瑞琴,李向阳主编.—2 版.—北京:清华大学出版社,2017
(高等学校通识教育系列教材)

ISBN 978-7-302-46509-6

I. ①C… II. ①郑… ②布… ③李… III. ①C 语言—程序设计 IV. ①TP312.8

中国版本图书馆 CIP 数据核字(2017)第 025415 号

责任编辑:魏江江 王冰飞

封面设计:文 静

责任校对:梁 毅

责任印制:李红英

出版发行:清华大学出版社

网 址: <http://www.tup.com.cn>, <http://www.wqbook.com>

地 址:北京清华大学学研大厦 A 座 邮 编:100084

社总机:010-62770175 邮 购:010-62786544

投稿与读者服务:010-62776969, c-service@tup.tsinghua.edu.cn

质量反馈:010-62772015, zhiliang@tup.tsinghua.edu.cn

课 件 下 载: <http://www.tup.com.cn>, 010-62795954

印 装 者:清华大学印刷厂

经 销:全国新华书店

开 本:185mm×260mm

印 张:20.25

字 数:492 千字

版 次:2012 年 4 月第 1 版

2017 年 6 月第 2 版

印 次:2017 年 6 月第 1 次印刷

印 数:1~2000

定 价:44.50 元

产品编号:071242-01

第2版前言

进入 21 世纪,出现了一批高校向应用型大学转型的大趋势,根据这一特点,我们在本书第 1 版的基础上进行了修订,结合应用型高校学生的特点及学时设置努力使本书第 2 版达到兼顾专业、通俗、实用和与时俱进的目标,使之成为应用型高等院校计算机专业和非计算机专业的 C 语言程序设计课程的教材和参考手册。

本书采用通俗而亲和的描述风格,比较适合 C 程序设计的初学者、大学一年级学生,也适合想更深入掌握 C 语言程序设计的读者。本书中每个知识点的讲解由浅入深、循序渐进,教学内容以案例驱动方法进行,案例实用、有趣、易懂,贴近普通大学生的实际生活,让学生备感亲切。本书在潜移默化中帮助学生了解应用程序设计方法和过程,掌握程序设计语言知识和实用技巧,培养他们的编程素养。

本书围绕把问题讲清楚、讲明白、讲透彻这个中心目标,放弃一些陈旧的、以炫技为目的的传统设计习惯,追求更直接、更简练的设计风格,尽量把我们多年来在教学和实际应用开发实践中积累的经验贯穿于书中,同时也将程序设计领域最新、最有价值的思想和方法渗透到书中,使本书以焕然一新的面貌呈现在各位亲爱的读者面前。

本书仍然延续第 1 版所采用的 CDIO 的思想和模式,即每章节的基本模式如下:

- (1) CDIO 引例示范(教师的“CDIO”);
- (2) 相关知识点描述;
- (3) 在教师引导下,课堂实例演练 CDIO(学生参与的 CDIO);
- (4) 小结涵盖本部分的知识点、容易混淆的概念及容易出错的问题;
- (5) 实战演练和习题要求学生通过练习实现自主的 CDIO;

(6) 从第 2 章开始,每章均增加“综合设计”一节,作为本章所讲述知识的综合运用模式,引导学生提升分析和设计能力。

本书第 2 版在第 1 版的基础上进行了很多修订和大量扩编,增加了例题、习题,将第 1 版中的第 10 章删去。全书共分 10 章,内容包括 C 语言概述、C 程序的运算符和表达式、选择结构、循环结构、数组、函数、自定义数据类型、指针、文件以及 Visual C++ 6.0 开发环境。

为了进一步强化学生在应用开发中的实操能力,本书每章的练习中都增加了实验实训部分。学生可以模仿章节综合设计的程序实例完成自己的设计,还可以创造性地对问题进行扩展设计,形成具有个人风格的独立程序。本书习题有编程题,也有等级考试中常见的选择题、填空题、读程序写结果题,题型较多,并附有参考答案,对于备考二级考试的同学也有一定的帮助。

本书的第 1、2 章由布瑞琴修订;第 3、5、6、9 章由郑晓健修订;第 4 章由周波修订;第

7、8 章由高世健修订;第 10 章由冯瑶修订。郑晓健负责全书的统稿。全书由李向阳教授主审。在此,我们还必须由衷地感谢本书第 1 版的作者李向阳教授、方娇莉副教授、郭琳老师和王云泽老师,没有他们之前所做的大量工作,我们的修订工作不会如此顺利。

建议本书授课时数为 48 学时,基本上机练习 24 学时,学生自主上机练习 64 学时左右,与本书配套的相关教学资料请到清华大学出版社网站(www.tup.com.cn)下载。由于编者水平和学识所限,书中难免存在疏漏与不足,敬请读者指正。

编 者

2017 年 3 月

目 录

第 1 章 C 语言概述	1
1.1 C 语言的产生与发展	1
1.1.1 程序设计语言简述	1
1.1.2 C 语言的历史	2
1.2 C 语言的特点	2
1.3 一个简单的 C 程序	3
1.4 C 语言程序的格式	6
1.5 标识符、变量与常量	7
1.5.1 标识符	7
1.5.2 变量	8
1.5.3 常量	8
1.6 数据类型	9
1.6.1 整型、实型和字符型	9
1.6.2 C 程序语句	13
1.6.3 格式输出/输入函数	14
1.6.4 实战演练	17
1.7 小结	17
习题 1	18
本章实验实训	21
第 2 章 C 程序的运算符和表达式	23
2.1 计算 a/b 和 $a\%b$ 的值	23
2.1.1 分析与设计	23
2.1.2 运算符与表达式	24
2.2 算术运算符	24
2.2.1 算术表达式	24
2.2.2 数据类型与运算结果的关系	25
2.2.3 实战演练	25
2.3 位运算符	25
2.3.1 按位与运算	26

2.3.2	按位或运算	26
2.3.3	按位异或运算	27
2.3.4	按位取反运算	27
2.3.5	左移运算	28
2.3.6	右移运算	28
2.3.7	实战演练	28
2.4	赋值运算符及表达式	29
2.4.1	赋值运算符	29
2.4.2	赋值表达式	29
2.4.3	复合的赋值运算符	29
2.5	自增自减运算符	30
2.5.1	自增 1 运算实例	30
2.5.2	实战演练	31
2.6	其他运算符	31
2.6.1	逗号运算符	31
2.6.2	求字节数运算符	32
2.7	运算符的优先级和结合性	32
2.7.1	C 语言的运算符分类	32
2.7.2	运算符的优先级	33
2.7.3	结合方向	33
2.8	不同类型数据之间的转换	33
2.8.1	自动类型转换	34
2.8.2	强制类型转换	34
2.9	综合设计	35
2.10	小结	36
	习题 2	36
	本章实验实训	39
第 3 章	选择结构	41
3.1	工程师岗位面试(关系运算符和逻辑运算符)	41
3.1.1	分析与设计	41
3.1.2	关系运算符和关系表达式	42
3.1.3	逻辑运算符和逻辑表达式	43
3.1.4	条件运算符和条件表达式	44
3.2	判断身材是否标准(if 语句)	45
3.2.1	分析与设计	46
3.2.2	if 语句	46
3.2.3	if 语句的嵌套	51
3.2.4	实战演练	54

3.3 顾客点餐(switch 语句)	55
3.3.1 分析与设计	55
3.3.2 switch 语句	56
3.3.3 使用 switch 语句的注意事项	58
3.3.4 多路选择结构的比较	59
3.3.5 实战演练	59
3.3.6 综合设计(简单界面设计)	60
3.4 小结	61
习题 3	62
本章实验实训	67
第 4 章 循环结构	69
4.1 输出 100 个数(for 语句)	69
4.1.1 分析与设计	69
4.1.2 for 循环语句	70
4.1.3 for 语句的几点说明	72
4.1.4 实例分析与设计	73
4.1.5 实战演练	76
4.2 统计英语成绩(while 语句)	77
4.2.1 分析与设计	77
4.2.2 while 循环语句	79
4.2.3 实例分析与设计	79
4.2.4 实战演练	80
4.3 整数逆序输出(do-while 语句)	81
4.3.1 分析与设计	81
4.3.2 do-while 循环语句	82
4.3.3 实例分析与设计	82
4.3.4 用 while 语句和用 do-while 语句的比较	83
4.3.5 实战演练	84
4.4 打印矩形(循环嵌套)	85
4.4.1 分析与设计	85
4.4.2 循环嵌套	86
4.4.3 死循环	88
4.4.4 实战演练	88
4.5 找最小数(break 和 continue 语句)	89
4.5.1 分析与设计	89
4.5.2 break 语句	90
4.5.3 continue 语句	91
4.5.4 用 for 和 while 循环实现 do-while 循环功能	92

4.5.5 实战演练	92
4.5.6 综合设计	92
4.6 小结	94
习题 4	94
本章实验实训	103
第 5 章 数组	104
5.1 厨师选鱼(一维数组)	104
5.1.1 分析与设计	104
5.1.2 一维数组	105
5.1.3 实战演练	112
5.2 果园里的竞赛(二维数组)	113
5.2.1 分析与设计	113
5.2.2 二维数组	115
5.2.3 实战演练	120
5.3 古诗词填空(字符数组)	121
5.3.1 分析与设计	121
5.3.2 字符数组	121
5.3.3 字符串处理函数	126
5.3.4 实战演练	129
5.4 综合设计	130
5.4.1 解决数据的存储问题	131
5.4.2 找摘桃子最多的人	132
5.4.3 计算选手的总成绩	133
5.5 小结	135
习题 5	136
本章实验实训	141
第 6 章 函数	142
6.1 阶乘之和(函数的定义、调用和声明)	142
6.1.1 分析与设计	142
6.1.2 函数的定义和调用	144
6.1.3 函数原型、函数声明与函数调用	146
6.1.4 实战演练	149
6.2 成绩统计(函数的参数传递)	150
6.2.1 分析与设计	150
6.2.2 函数的参数传递	152
6.2.3 实战演练	155
6.3 计算三角形面积(嵌套调用和递归调用)	156

6.3.1	分析与设计	156
6.3.2	嵌套调用	157
6.3.3	递归调用	158
6.3.4	实战演练	159
6.4	迎接第 15 亿个婴儿(变量的作用域)	160
6.4.1	分析与设计	161
6.4.2	局部变量和全局变量	161
6.4.3	实战演练	164
6.5	构造整数(变量的存储类型)	165
6.5.1	分析与设计	165
6.5.2	局部变量的存储类型	167
6.5.3	全局变量的存储类型	169
6.5.4	实战演练	171
6.6	快速计算(编译预处理)	172
6.6.1	分析与设计	172
6.6.2	宏定义命令	173
6.6.3	文件包含	174
6.6.4	实战演练	175
6.7	综合设计(诗词十二宫格游戏)	175
6.7.1	分析与设计	175
6.7.2	完整的源程序代码	175
6.8	小结	177
	习题 6	178
	本章实验实训	183
第 7 章	自定义数据类型	184
7.1	今天是星期几(枚举类型)	184
7.1.1	分析与设计	184
7.1.2	枚举类型的定义与引用	184
7.2	模拟显示数字时钟(结构体类型)	185
7.2.1	分析与设计	186
7.2.2	结构体类型的定义与引用	187
7.2.3	结构体数组及其使用	190
7.2.4	结构体变量做参数	191
7.3	学生成绩表的制作(共用体类型)	191
7.3.1	分析与设计	191
7.3.2	共用体类型的定义与引用	192
7.4	实战演练	193
7.5	综合设计	196

7.5.1 分析与设计	196
7.5.2 完整的源程序代码	197
7.6 小结	201
习题 7	203
本章实验实训	205
第 8 章 指针	207
8.1 用函数实现变量值的交换	207
8.1.1 分析与设计	207
8.1.2 指针的定义及运算	209
8.2 数组与指针	210
8.2.1 指向一维数组的指针	211
8.2.2 指针指向数组时的运算	212
8.2.3 指向二维数组的指针	213
8.3 用指针操作字符串	213
8.3.1 分析与设计	213
8.3.2 使用字符数组与字符指针变量的区别	215
8.4 指针与函数	217
8.4.1 用指向函数的指针实现函数调用	217
8.4.2 返回指针值的函数	219
8.5 指针数组和指向指针的指针	219
8.5.1 指针数组的概念	219
8.5.2 指向指针的指针	221
8.5.3 指针数组做 main() 函数的参数	222
8.6 实战演练——验证卡布列克运算	223
8.7 综合设计——用指针实现数据的动态管理	225
8.7.1 分析与设计	225
8.7.2 程序	226
8.7.3 动态数据管理在插入、删除操作中的优点	229
8.8 小结	230
习题 8	232
本章实验实训	235
第 9 章 文件	237
9.1 学生数据文件的创建与读取	237
9.1.1 分析与设计	237
9.1.2 文件操作入门	239
9.2 文件的概念	240
9.2.1 文件的定义	240

9.2.2	文件的分类	240
9.2.3	文件缓存区	241
9.2.4	文件类型与文件指针	241
9.2.5	文件的操作过程	242
9.3	文件的打开和关闭	242
9.3.1	文件的打开	242
9.3.2	文件的关闭	244
9.4	文件的读/写操作	244
9.4.1	字符读/写函数 fgetc 和 fputc	245
9.4.2	字符串读/写函数 fgets 和 fputs	246
9.4.3	格式化读/写函数 fscanf 和 fprintf	248
9.4.4	数据块读/写函数 fread 和 fwrite	250
9.5	文件的定位操作	252
9.6	文件的出错检测	255
9.7	实战演练	257
9.8	综合设计	259
9.8.1	分析与设计	259
9.8.2	完整的源程序代码	260
9.9	小结	263
	习题 9	264
	本章实验实训	268
第 10 章	Visual C++ 6.0 开发环境	269
10.1	开发环境概述	269
10.2	菜单栏简介	271
10.3	开发环境的工具栏	273
10.4	VC++ 6.0 的主要窗口	275
10.5	新建、编辑、编译、连接、运行一个 C 程序	276
10.6	常见问题处理	278
	习题参考答案	280
	附录 A ASCII 码表	300
	附录 B C 语言中的关键字	302
	附录 C C 标准库函数	303
	参考文献	309

第 1 章

C 语言概述

自从计算机诞生以来,用计算机完成的每件工作都必须用计算机语言编写程序,将“程序”输入计算机内存后,计算机才能按程序要求完成任务。也就是说,要学习程序设计,就必须学习计算机语言。计算机语言随着计算技术的发展和应用的不断发展,先后产生了机器语言、汇编语言、高级语言、面向对象的高级语言等计算机语言。

我们应了解为什么要选择 C 语言,以及它有哪些特性。只有了解了 C 语言的历史和特性,我们才会更深刻地了解这门语言,并且增加今后学习 C 语言的信心。C 语言是目前广泛使用的一种程序设计语言,其既可用于编写系统软件,也可用来编写应用软件。由于 C 语言具有所有高级语言都支持的数据类型、控制结构等,还可以对位、字节和地址这些计算机功能中的基本成分进行操作。随着计算机科学的不断发展,C 语言的学习环境也在不断变化,刚开始学习 C 语言时大多数人会选择一些相对简单的编译器,如 Turbo C 2.0,现在更多的人选择了由 Microsoft 公司推出的 Visual C++ 6.0 编译器。

本章致力于使读者了解 Visual C++ 6.0 的开发环境,掌握其中各个部分的使用方法,并能编写一个简单的应用程序以练习使用开发环境。

1.1 C 语言的产生与发展

1.1.1 程序设计语言简述

在介绍 C 语言的发展历程之前,我们先对程序语言进行大概的了解。计算机语言是什么?计算机语言是人与计算机交流的“桥梁”。学习计算机语言能对计算机的原理及功能有深入的了解。

1. 机器语言

机器语言是低级语言,也称为二进制代码语言。计算机使用的是由 0 和 1 组成的二进制数组成的一串指令来表达计算机操作的语言。机器语言的特点是计算机可以直接识别,不需要进行任何的翻译。

2. 汇编语言

汇编语言是面向机器的程序设计语言。为了减轻程序员使用机器语言编程的痛苦,用英文字母或符号串来代替机器语言的二进制码,这样就把不易理解和使用的机器语言变成了汇编语言,使得使用汇编语言比使用机器语言更便于用户阅读和理解程序。

3. 高级语言

由于汇编语言依赖于硬件体系,并且该语言中的助记符号数量比较多,所以其运用起来仍然不够方便。为了使程序语言能更贴近人类的自然语言,同时又不依赖于计算机硬件,于

是产生了高级语言。这种语言的语法形式类似于英文,并且因为远离对硬件的直接操作而易于被普通人所理解与使用,其中影响较大、使用普遍的高级语言有 Fortran、ALGOL、Basic、COBOL、LISP、Pascal、PROLOG、C、C++、C#、VC、VB、Java 等。

1.1.2 C 语言的历史

从程序语言的发展过程可以看到,以前的操作系统等系统软件主要是用汇编语言编写的。但由于汇编语言依赖于计算机硬件,程序的可读性和可移植性都不是很好,为了提高可读性和可移植性,人们开始寻找一种语言,这种语言应该既具有高级语言的特性,又不失低级语言的优点,于是 C 语言产生了。

C 语言是在由 UNIX 的研制者丹尼斯·里奇(Dennis Ritchie)和肯·汤普逊(Ken Thompson)于 1970 年研制出的 BCPL 语言(简称 B 语言)的基础上发展和完善起来的。19 世纪 70 年代初期,AT&T Bell 实验室的程序员丹尼斯·里奇(见图 1.1)第一次把 B 语言改为 C 语言。



图 1.1 丹尼斯·里奇

最初,C 语言运行于 AT&T 的多用户、多任务的 UNIX 操作系统上,后来,丹尼斯·里奇用 C 语言改写了 UNIX C 的编译程序,UNIX 操作系统的开发者肯·汤普逊又用 C 语言成功地改写了 UNIX,从此开创了编程史上的新篇章,UNIX 成为第一个不是用汇编语言编写的主流操作系统。

1983 年,美国国家标准委员会(ANSI)对 C 语言进行了标准化,于 1983 年颁布了第一个 C 语言草案(83ANSI C),后来于 1987 年又颁布了另一个 C 语言标准草案(87ANSI C),最新的 C 语言标准 C99 于 1999 年颁布,并在 2000 年 3 月被 ANSI 采用。但是由于未得到主流编译器厂家的支持,C99 并未得到广泛使用。

尽管 C 语言是在大型商业机构和学术界的研究实验室研发的,但是当开发者们为第一台个人计算机提供 C 编译系统之后,C 语言就得以广泛传播,并为大多数程序员所接受。对 MS-DOS 操作系统来说,系统软件和实用程序都是用 C 语言编写的。Windows 操作系统大部分也是用 C 语言编写的。

C 语言是一种面向过程的语言,同时具有高级语言和汇编语言的优点。C 语言可以广泛应用于不同的操作系统,如 UNIX、MS-DOS、Microsoft Windows 及 Linux 等。

在 C 语言的基础上发展起来的有支持多种程序设计风格的 C++ 语言、网络上广泛使用的 Java、JavaScript,以及 Microsoft 公司的 C# 语言等。也就是说,学好 C 语言之后再学习其他语言时就会比较轻松。

1.2 C 语言的特点

C 语言是一种通用的程序设计语言,主要用来进行系统程序设计,具有以下特点:

1. 高效性

谈到高效性,不得不说 C 语言是“鱼与熊掌”兼得。从 C 语言的发展历史也可以看到,

它继承了低级语言的优点,产生了高效的代码,并具有友好的可读性和编写性。一般情况下,C语言生成的目标代码的执行效率只比汇编程序低 10%~20%。

2. 灵活性

C语言中的语法不拘一格,可在原有语法的基础上进行创造、复合,从而给程序员更多的想象和发挥空间。

3. 功能丰富

除了C语言中所具有的类型,程序员还可以使用丰富的运算符和自定义的结构类型来表达任何复杂的数据类型,完成所需要的功能。

4. 表达力强

C语言的特点体现在它的语法形式与人们所使用的语言形式相似,书写形式自由,结构规范,并且只需简单地控制语句即可轻松控制程序流程,完成烦琐的程序要求。

5. 移植性好

由于C语言具有良好的移植性,从而使得C程序在不同的操作系统下只需要简单地修改或者不用修改即可进行跨平台的程序开发操作。

正是由于C语言拥有上述优点,使得它在程序员选择语言时备受青睐。

1.3 一个简单的C程序

在通往C语言程序世界之前,我们不要对C语言产生恐惧感,觉得这种语言应该是学者或研究人员的专利。C语言是人类共有的财富,是普通人只要努力学习就可以掌握的知识。下面通过一个简单的程序来看一看C语言程序是什么样子。

例 1-1 一个简单的C程序。

本程序实现的功能是显示一条信息“Hello, welcome to C world!”,通过这个程序我们可以初窥C程序样貌。虽然这个简单的小程序只有7行,却充分说明了C程序是从什么位置开始、在什么位置结束。

```
#include "stdio.h"
main()
{
    printf("Hello, welcome to C world!\n");    /* 输出要显示的字符串 */
}
```

运行结果:

Hello, welcome to C world!

运行程序,显示效果如图 1.2 所示。



图 1.2 一个简单的C程序

下面分析一下该程序。

1. #include 指令

```
#include "stdio.h"
```

这个语句的功能是进行有关的预处理操作。include 被称为文件包含命令,后面双引号中的内容被称为头部文件或首文件。有关预处理的内容将会在后面的章节中进行详细讲解,读者在此只需对此概念有所了解即可。

2. 空行

C 语言是一种较灵活的语言,因此格式并不是固定不变、拘于一格的。也就是说,空格、空行、跳格并不会影响程序。有的读者会问:“为什么要有这些多余的空格和空行呢?”其实这就像在纸上写字一样,虽然拿来一张白纸就可以在上面写字,但是通常还会在纸的上面印上一行一行的方格或段落,隔开每一段文字,自然就更加美观和规范。合理、恰当地使用这些空格、空行可以使编写出来的程序更加规范,对日后的阅读和整理发挥着重要的作用。

3. main 函数的声明

```
main()
```

在函数中这一部分称为函数头部分。在每一个程序中都会有一个 main 函数,那么 main 函数的作用是什么呢? main 函数就是一个程序的入口部分。也就是说,程序都是从 main 函数头开始执行的,然后进入到 main 函数中,执行 main 函数中的内容。

4. 函数体

```
{  
    printf("Hello, welcome to C world! \n");           /* 输出要显示的字符串 */  
}
```

在上面介绍 main 函数时提到了一个名词——函数头。通过这个词可以进行一下联想:既然有函数头,那也应该有函数的身体吧? 没错,一个函数分为两个部分,一个是函数头,另一个是函数体。

程序代码中的两个大括号构成了函数体,函数体也可以称为函数的语句块。

5. 执行语句

```
printf("Hello, welcome to C world! \n");           /* 输出要显示的字符串 */
```

执行语句就是函数体中要执行的动作内容。这一行代码是这个简单的例子中最复杂的。该行代码虽然看似复杂,其实也不难理解,printf 是产生格式化输出的函数,可以简单地理解为向控制台输出文字或符号。括号中的内容称为函数的参数,在括号内可以看到输出的字符串“Hello, welcome to C world!”,还可以看到“\n”这样一个符号,称之为转义字符。

6. 代码的注释

```
printf("Hello, welcome to C world! \n");           /* 输出要显示的字符串 */
```

对代码的解释描述称为代码的注释。

其语法格式如下：

/* 其中为注释内容 */

例 1-2 一个完整的 C 程序。

本实例要实现这样的功能：有一个长方体，它的高已经给出，然后输入这个长方体的长和宽，通过输入的长、宽以及给定的高度计算出长方体的体积。

```
#include <stdio.h>          /* 包含头文件 */
#define Height 10           /* 定义常量 */
void main()                 /* 主函数 main */
{
    int longcf;             /* 定义整型变量,表示长度 */
    int width;              /* 定义整型变量,表示宽度 */
    int volume;             /* 定义整型变量,表示长方体的体积 */
    printf("长方形的高度为: %d\n",Height); /* 显示提示 */
    printf("请输入长度\n");    /* 显示提示 */
    scanf("%d",&longcf);      /* 输入长方体的长度 */
    printf("请输入宽度\n");    /* 显示提示 */
    scanf("%d",&width);       /* 输入长方体的宽度 */
    volume = longcf * width * Height; /* 计算体积 */
    printf("长方体的体积是: "); /* 显示提示 */
    printf("%d\n",volume);     /* 输出体积大小 */
}
```

运行结果：

```
长方形的高度为: 10
请输入长度
35
请输入宽度
40
长方体的体积是: 14000
```

运行程序，显示效果如图 1.3 所示。



图 1.3 完整的 C 程序

1. 定义常量

```
#define Height 10          /* 定义常量 */
```