

高等学校计算机基础教育规划教材

C语言程序设计

(第3版)

向 艳 周天彤 主编

清华大学出版社



高等学校计算机基础教育规划教材

C语言程序设计

(第3版)

向 艳 周天彤 主 编
潘亚平 程起才 副主编

清华大学出版社
北京

内 容 简 介

为了切合当前 C 语言的发展和教学的需要,对《C 语言程序设计(第 2 版)》进行修订而形成了本书。本书以程序设计为中心,由浅入深地介绍了 C 语言和程序设计的基本概念和要点,把语言和算法紧密结合。依照“适用”加“实用”的原则,适当调整了教材内容,重点更加突出。调整后的主要内容包括: C 程序设计入门、顺序结构程序设计、选择结构程序设计、循环结构程序设计、函数、数组、指针、结构体与共用体、动态数组与链表、文件、综合应用案例等。

本书体系合理,篇幅适中,重点突出,文字通俗易懂,内容由浅入深,知识点言简意赅,是初学者学习 C 语言程序设计的理想教材,可作为高等院校学生学习 C 语言程序设计课程的主教材,也适合计算机培训班或自学的读者使用。

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/向艳,周天彤主编. —3 版. 北京:清华大学出版社,2018

(高等学校计算机基础教育规划教材)

ISBN 978-7-302-50771-0

I. ①C… II. ①向… ②周… III. ①C 语言—程序设计—高等学校—教材 IV. ①TP312

中国版本图书馆 CIP 数据核字(2018)第 177391 号

责任编辑:袁勤勇

封面设计:常雪影

责任校对:时翠兰

责任印制:宋 林

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:三河市君旺印务有限公司

经 销:全国新华书店

开 本:185mm×260mm 印 张:24.5 字 数:568 千字

版 次:2008 年 9 月第 1 版 2018 年 10 月第 3 版 印 次:2018 年 10 月第 1 次印刷

定 价:59.00 元

产品编号:080637-01

前言

C 语言是当今世界上广泛流行的一门程序设计语言,深受广大程序员和编程爱好者的喜爱。C 语言不仅适用于开发系统软件,而且也是用于开发应用软件和进行大规模科学计算的常用编程语言。

由于 C 语言的基本概念复杂,内容丰富,使用灵活,一些初学者会发现,学习 C 语言的过程是一个充满挫折的艰难过程。一方面觉得学习 C 语言内容枯燥,难度大;另一方面即便学完了 C 程序设计课程,但一旦要用 C 语言来独立编写一些解决实际问题的程序时会感到无从下手。为此,作者通过认真分析和研究,并结合长期从事 C 程序设计课程教学的实践经验,于 2008 年编写了《C 语言程序设计》一书,由清华大学出版社出版,至今共出版了两个版次。

《C 语言程序设计》一书在使用中得到了广大读者的肯定,并提出了不少宝贵的意见,在此表示感谢。此次对本书做第二次修订,主要基于以下原因:

(1) 现代计算机技术不断发展,C 语言及编程技术也在发展中,教材内容要与时俱进,推陈出新。

(2) 作者在教学实践和教学改革中积累了一些新的经验,对学生的学习心理有了更深刻的认识和了解。

(3) 广大读者和同仁对本书提出了一些新的建议与期望。

为此,在继续保持前两版写作风格和特色的基础上,对本书主要做了以下修订:

(1) 以 VC++ 2010 为编程环境,所有例题、习题都在 VC++ 2010 环境下调试通过。

(2) 考虑到 C 语言的发展和系统兼容性问题,增加了部分 C99 标准的新规定,所有程序风格均采用如下所示的 C99 标准形式:

```
int main()  
{  
  
    return 0;  
}
```

(3) 去掉了“预处理”和“位运算”两章,把宏常量和基本位运算符的内容放在了第 1 章;考虑到内容的关联性,把关系运算和逻辑运算的内容调整到第 3 章。

(4) 为了进一步巩固和综合应用各章知识,了解和掌握大型程序的设计方法,按照软件工程的方法编写了综合应用案例一章,并将项目管理和文件包含等内容融入其中。

(5) 重新组织课后习题,依照“巩固基础、综合应用、拓展创新”三个层次,呈阶梯式递进形式。

(6) 结合作者实践经验,在附录中增加了 VC++ 2010 环境 C 程序的调试运行方法,强化编程训练。

经修订后,本书内容共分为以下 11 章:

第 1 章 C 程序设计入门 介绍 C 语言的特点、程序结构、C 语言的基本数据类型、基本运算符和表达式、宏常量和常变量、基本位运算。

第 2 章 顺序结构程序设计 介绍程序设计的基本方法、C 语句的分类、基本的输入输出处理以及顺序结构程序设计的基本方法。

第 3 章 选择结构程序设计 介绍关系运算和逻辑运算、C 语言中实现选择结构的语句以及选择结构程序设计的基本方法。

第 4 章 循环结构程序设计 介绍 C 语言中实现循环结构的语句和循环结构程序设计的基本方法,还介绍了应用循环的一些常用算法,如级数求和问题、穷举法、递推法以及求素数方法等。

第 5 章 函数 介绍函数的定义、调用以及函数参数的传递方式,函数的嵌套调用和递归调用方法,全局变量和局部变量,变量的存储类别等。

第 6 章 数组 介绍一维数组、二维数组以及字符数组的定义、引用和初始化方法,数组名作为函数参数的调用方式,以及应用数组的一些算法,如排序、查找、求最大最小值、矩阵运算等。

第 7 章 指针 介绍指针的概念,指针变量的定义和引用,以及指向变量的指针、指向数组的指针、指向字符串的指针、指向函数的指针和多级指针等概念和应用。

第 8 章 结构体与共用体 介绍结构体类型、共用体类型和枚举类型的定义,以及相关变量的定义、引用、初始化和应用方法。

第 9 章 动态数组与链表 介绍 C 语言中实现动态存储分配的标准函数,以及动态数组和链表的概念和应用。

第 10 章 文件 介绍 C 语言中文件的基本概念,文件的打开和关闭方法以及文件的读写和定位方法。

第 11 章 综合应用案例 介绍综合应用各章知识,按照软件工程的方法开发一个股票交易系统程序设计和实现的全过程。

本书由向艳、周天彤担任主编并由向艳统稿,第 1、2、10 章由潘亚平和程起才共同编著,第 3、4、5、6、8、9、11 章由向艳编著,第 7 章由周天彤编著,附录由周天彤和程起才共同编著。希望通过此次修订,使本书内容更丰富,条理更清晰,实用性更强,更有利于读者学习。由于作者水平有限,书中不足在所难免,再次恳请读者批评指正。

学习 C 语言程序设计是一个循序渐进的过程。建议读者在学习中依照“一读、二仿、三写”三个步骤,即先多读一些好的程序;然后模仿实例编写相似的程序,并能举一反三;最后独立编写程序,提高编程能力。

编 者

2018 年 3 月

目 录

第 1 章 C 程序设计入门	1
1.1 概述	1
1.1.1 程序和程序设计语言	1
1.1.2 C 语言的起源和发展	2
1.1.3 C 语言的特点	3
1.2 简单的 C 程序	4
1.2.1 简单的 C 程序举例	4
1.2.2 C 程序的结构	8
1.2.3 C 程序的调试与运行环境	9
1.3 基本数据类型	11
1.3.1 整型数据	12
1.3.2 实型数据	16
1.3.3 字符型数据	18
1.4 运算符和表达式	23
1.4.1 运算符优先级及结合性	23
1.4.2 数据类型转换	24
1.4.3 算术运算符和算术表达式	26
1.4.4 赋值运算符和赋值表达式	27
1.4.5 逗号运算符和逗号表达式	29
1.4.6 ++和--运算符	30
1.4.7 sizeof 运算符	31
1.5 宏常量与常变量	32
1.5.1 宏常量	32
1.5.2 常变量	33
1.6 位运算	34
复习与思考	38
习题 1	38

第 2 章 顺序结构程序设计	41
2.1 编程逻辑与技术	41
2.1.1 算法及算法的描述工具	41
2.1.2 程序设计的基本过程	44
2.1.3 结构化程序设计方法	45
2.2 C 语言的语句类型	47
2.3 数据的输入与输出	48
2.3.1 字符输入与输出函数	49
2.3.2 格式输入与输出函数	50
2.4 常用计算函数	60
2.4.1 数学库函数	60
2.4.2 伪随机函数	61
2.5 程序举例	62
复习与思考	67
习题 2	67
第 3 章 选择结构程序设计	70
3.1 关系运算符和关系表达式	70
3.1.1 关系运算符	70
3.1.2 关系表达式	71
3.2 逻辑运算符和逻辑表达式	72
3.2.1 逻辑运算符	72
3.2.2 逻辑表达式	72
3.3 if 语句	74
3.3.1 if 语句的三种形式	74
3.3.2 if 语句的嵌套	78
3.4 条件运算符和条件表达式	79
3.5 switch 语句	81
3.6 程序举例	84
复习与思考	91
习题 3	91
第 4 章 循环结构程序设计	98
4.1 当型循环与直到型循环	98
4.1.1 当型循环结构	98
4.1.2 直到型循环结构	98
4.2 循环语句	99

4.2.1	while 语句	99
4.2.2	do-while 语句	101
4.2.3	for 语句	103
4.3	循环的嵌套	105
4.4	break 语句和 continue 语句	108
4.4.1	break 语句	108
4.4.2	continue 语句	109
4.5	程序举例	110
	复习与思考	122
	习题 4	122
第 5 章	函数	130
5.1	函数的定义与调用	130
5.1.1	函数的定义	131
5.1.2	函数的返回值	132
5.1.3	函数的调用	134
5.2	函数间的参数传递	140
5.2.1	实参与形参的传递方式	140
5.2.2	局部变量与全局变量	142
5.2.3	局部变量的存储类别	145
5.2.4	全局变量的存储类别	147
5.3	函数的嵌套调用	148
5.4	函数的递归调用	150
	复习与思考	155
	习题 5	156
第 6 章	数组	165
6.1	一维数组	165
6.1.1	一维数组的定义与引用	165
6.1.2	一维数组的初始化	168
6.1.3	一维数组应用举例	169
6.2	二维数组	174
6.2.1	二维数组的定义与引用	174
6.2.2	二维数组的初始化	176
6.2.3	二维数组应用举例	178
6.3	字符数组	182
6.3.1	字符数组的定义与初始化	182
6.3.2	字符串与字符数组	184

6.3.3	字符数组的输入与输出	185
6.3.4	常用字符串处理函数	188
6.3.5	字符数组应用举例	192
6.4	数组作为函数参数	195
6.4.1	用一维数组名作为函数参数	195
6.4.2	用二维数组名作为函数参数	201
	复习与思考	203
	习题 6	203
第 7 章	指针	211
7.1	指针的概念	212
7.2	指针的操作	214
7.2.1	取地址运算符 & 与指针运算符 *	214
7.2.2	使用指针实现地址传递	216
7.2.3	使用简单指针的例子	219
7.3	数组和指针	220
7.3.1	指向数组元素的指针	220
7.3.2	指针与整数的加减法	221
7.3.3	指针的类型与指针间的减法	222
7.3.4	指向字符串的指针	223
7.3.5	使用指针处理一维数组的应用举例	226
7.4	指针数组与多级指针	229
7.4.1	指针数组	229
7.4.2	二级指针及多级指针	231
7.4.3	使用指针数组作为 main 函数的参数	233
7.5	数组的指针与函数的指针	235
7.5.1	指向数组的指针	235
7.5.2	行指针与列指针	236
7.5.3	函数指针与指针函数	237
	复习与思考	240
	习题 7	241
第 8 章	结构体与共用体	249
8.1	结构体类型与结构体变量	249
8.1.1	结构体类型的声明	249
8.1.2	结构体类型变量的定义	250
8.1.3	结构体类型变量的引用	252
8.1.4	结构体类型变量的初始化	254

8.2	结构体数组	255
8.2.1	结构体数组的定义与引用	255
8.2.2	结构体数组应用举例	256
8.3	结构体指针	259
8.3.1	指向结构体变量的指针	259
8.3.2	指向结构体数组的指针	260
8.3.3	将结构体指针作为函数参数	262
8.4	共用体	266
8.5	枚举类型	269
8.6	用 typedef 自定义类型名	272
	复习与思考	274
	习题 8	274
第 9 章	动态数组与链表	282
9.1	内存动态分配与动态数组	282
9.1.1	常用的内存动态分配函数	282
9.1.2	动态数组	284
9.2	链表	286
9.2.1	链表的基本概念	286
9.2.2	创建动态链表	286
9.2.3	输出动态链表	289
9.2.4	动态链表的删除操作	290
9.2.5	动态链表的插入操作	291
9.2.6	动态链表的应用举例	292
	复习与思考	296
	习题 9	296
第 10 章	文件	300
10.1	文件的基本概念	300
10.1.1	字节流	300
10.1.2	缓冲文件系统	300
10.1.3	文件类型指针	301
10.1.4	文件位置指针及文件打开方式	301
10.2	文件的打开与关闭	302
10.2.1	文件的打开	302
10.2.2	文件的关闭	303
10.3	文件的读写	304
10.3.1	字符读写函数	304

10.3.2 格式读写函数.....	307
10.3.3 数据块读写函数.....	310
10.3.4 其他读写函数.....	312
10.4 文件的定位.....	313
复习与思考.....	314
习题 10	315
第 11 章 综合应用案例——股票交易系统	322
11.1 功能模块设计.....	322
11.2 数据结构设计.....	323
11.3 函数功能描述.....	324
11.4 编制程序.....	331
11.5 运行程序.....	342
11.5.1 VC++ 2010 下的多文件管理	342
11.5.2 文件包含.....	343
11.5.3 调试运行.....	344
附录 A 基本 ASCII 码表.....	346
附录 B 运算符和结合性	348
附录 C C 语言关键字	350
附录 D VC++ 2010 环境下的常用库函数	352
附录 E 基于 VC++ 2010 环境下的 C 语言程序运行步骤与方法	361
附录 F VC 英文版中通用错误代码分析	378
参考文献.....	381

C 程序设计入门

1.1 概 述

1.1.1 程序和程序设计语言

1946 年 2 月 14 日,在美国宾夕法尼亚大学,诞生了众所周知的世界上第一台电子数字计算机 ENIAC。从第一台计算机产生至今的半个多世纪里,计算机的发展日新月异,已成为人们工作生活中必不可少的工具。

计算机之所以能按照人们的意愿有条不紊地工作,是因为人们将想要计算机做的事情用程序的形式存储到计算机的存储器中,并由 CPU(Central Processing Unit,中央处理器)一条一条地执行来实现的。所谓程序(Program)是一组指示计算机执行动作或做出判断的指令,通常用某种程序设计语言编写,运行于某种目标体系结构上。

程序设计(Programming)是给出解决特定问题程序的过程,是软件构造活动中的重要组成部分。程序设计往往以某种程序设计语言为工具,给出这种语言下的程序。

程序设计语言(Programming Language)是用于书写计算机程序的语言。正如人与人之间进行交流,要使用语言(汉语、英语等)来表达自己的思想。人与计算机交流信息,也需要使用语言来交流,这种语言就是程序设计语言。计算机每做的一次动作,一个步骤,都是按照事先已经用程序设计语言编好的程序来执行的。在编制程序时,要严格遵循这种程序设计语言的语法和语义规则。

自 20 世纪 60 年代以来,世界上公布的程序设计语言已有上千种之多,但是只有很小一部分得到了广泛的应用。从发展历程来看,程序设计语言可以分为三代。

1. 机器语言

机器语言(Machine Language)是由二进制 0 和 1 构成,计算机能直接识别和执行的一种指令系统。机器语言具有灵活、直接执行和速度快等特点。

在计算机发展的初期,程序员要让计算机知道自己的意图,就要编写许多条由 0 和 1 组成的指令,这是一件十分烦琐的工作。这些全是 0 和 1 的指令代码,直观性、可读性非常差,还很容易出错。另外,对于不同的 CPU 具有不同的指令系统。不同型号的计算机

其机器语言是不相通的,按照一种计算机的机器指令编制的程序,不能在另一种计算机上执行。

2. 汇编语言

由于机器语言的种种缺点,人们采用了一些助记符号来编写程序代码。例如采用 ADD 代表“加”,SUB 代表“减”,MUL 代表“乘”,DIV 代表“除”等。汇编语言(Assembly Language)相对于枯燥的机器代码易于读写、易于调试和修改。但汇编语言编写的程序代码(源程序),计算机并不能直接识别和执行,需要借助汇编语言编译器,将源程序翻译成计算机能直接执行的机器指令。

汇编语言保持了机器语言的优点,具有直接和简洁的特点,可有效地访问、控制计算机的各种硬件设备,如磁盘、存储器、CPU、I/O 端口等,且占用内存少,执行速度快,是高效的程序设计语言。在今天的实际应用中,汇编语言通常被应用在计算机底层,应用在对硬件操作和要求程序优化的场合。驱动程序、嵌入式操作系统和实时运行程序都需要汇编语言。

3. 高级语言

汇编语言在编写复杂程序时具有明显的局限性,汇编语言依赖于具体的机型,不能通用,也不能在不同机型之间移植。相对于第三代高级语言(High-Level Programming Language)来讲,人们将机器语言和汇编语言都称为“低级语言”。

高级语言是面向用户的、基本上独立于计算机种类和结构的语言。高级语言与计算机的硬件结构及指令系统无关,它有更强的表达能力,可方便地表示数据的运算和程序的控制结构,能更好地描述各种算法,而且容易学习掌握。但高级语言编译生成的程序代码一般比用汇编语言设计的程序代码要长,执行的速度也慢。高级语言最大的优点是:形式上接近于自然语言,使编写程序的过程更符合人类的思维习惯。高级语言的一个命令可以代替几条、几十条甚至几百条汇编语言的指令。因此,高级语言易学易用,通用性强,应用广泛。

高级语言并不是特指某一种具体的语言,而是包括很多编程语言,如流行的 Java、C、C++、C#、Pascal、Python、Lisp、Prolog 以及 FoxPro 等,这些语言的语法、命令格式都不相同。

1.1.2 C 语言的起源和发展

C 语言最早的原型是 ALGOL 60 语言。

1963 年,剑桥大学将其发展成为一种称之为 CPL(Combined Programming Language)的语言。

1967 年,剑桥大学的 Martin Richards 对 CPL 语言进行了简化,产生了 BCPL(Base Combined Programming Language)语言。

1970 年,美国贝尔实验室的 Ken Thompson 将 BCPL 进行了修改,并命名为 B 语言,

并用 B 语言写了第一个 UNIX 操作系统。

1972 年,美国贝尔实验室的 Dennis Ritchie 在 BCPL 和 B 语言的基础上设计出了一种新的语言,取 BCPL 中的第二个字母为名,这就是大名鼎鼎的 C 语言。

1978 年,Dennis Ritchie 和 Brian Kernighan 合作推出了 *The C Programming Language* 的第一版(简称 K&R),成为那时 C 语言事实上的标准,通常人们也称之为 K&R 标准。随着 C 语言在多个领域的推广和应用,一些新的特性不断被各种编译器实现并添加进来。于是,当务之急就是建立一个新的“无歧义、与具体平台无关的 C 语言定义”。

1983 年,美国国家标准委员会(ANSI)对 C 语言进行了标准化,并于 1989 年发布,通常称为 C89 标准。随后,*The C Programming Language* 第二版开始出版发行,书中内容根据 ANSI C(C89)进行了更新。

1990 年,国际标准化组织(ISO)批准了 ANSI C 成为国际标准。于是 ISO C(通常称为 C90)诞生了。ISO C(C90)和 ANSI C(C89)在内容上完全一样。之后,ISO 在 1994 年和 1996 年分别出版了 C90 的技术勘误文档,更正了一些印刷错误,并在 1995 年通过了一份 C90 的技术补充,对 C90 进行了微小的扩充,经过扩充后的 ISO C 被称为 C95。

1999 年,ANSI 和 ISO 又通过了最新版本的 C 语言标准和技术勘误文档,该标准被称为 C99。这基本上是目前关于 C 语言的最新、最权威的定义。现在,各种主流 C 编译器都提供了 C89(C90)的完整支持,但对 C99 还只提供了部分支持。

2011 年,ISO 正式公布 C 语言新的国际标准草案 ISO/IEC9899: 2011,即 C11。

1.1.3 C 语言的特点

一种程序设计语言之所以能存在和发展,并具有生命力,总是有其不同于或优于其他语言的特点。C 语言是一个用途广泛的、可移植、结构化的高级语言,目前其主要的用途之一是编写“嵌入式系统程序”,还有许多开发者用 C 语言编写文字处理器、数据库以及图形应用程序等。C 语言的主要特点为:

(1) 语言简洁、紧凑,使用方便、灵活。

C 语言一共有 37 个关键字(见附录 C)、9 种控制语句。程序书写形式自由。

(2) 运算符丰富。

C 语言的运算符包含的范围很广,一共有 34 种运算符(见附录 B)。

(3) 数据结构丰富,具有现代化语言的各种数据结构。

C 语言提供了丰富的数据类型,包括:整型(int、short int、long int、unsigned int、unsigned short int、unsigned long int 六种)、浮点型(float、double 两种)、字符型(char)、数组类型、指针类型、结构体类型、共用体类型、枚举类型等。C99 标准又扩充了复数浮点类型、超长整型(long long int、unsigned long long int)、布尔类型(bool)等。

(4) 支持自顶向下、结构化、模块化的编程。

C 语言提供了结构化的程序控制语句:如 if 语句(if...if...else...、if...else if...else if...else...三种形式)、while 语句、do...while 语句、for 语句等。C 语言采用函数作为程序的基本模块,实现了程序的模块化编程。

(5) 源代码的可移植性高。

C 语言的编译系统非常简洁,不需要修改源代码,可以直接编译“标准链接库”中的大部分功能,用 C 语言编写的程序可以很方便地移植到新的系统上。

(6) 语法限制不太严格,程序设计自由度大。

对数组下标越界不进行检查,由程序编写者自己保证程序的正确。程序员可以灵活使用变量的类型,如整型量与字符型数据和逻辑型数据之间可以通用。

(7) C 语言面向程序员,可以对硬件进行操作,允许直接访问物理地址。

C 语言允许直接访问物理地址,允许进行位操作,因此 C 语言既有高级语言的功能,又具有低级语言的功能。

(8) 生成目标代码质量高,程序执行效率高。

1.2 简单的 C 程序

1.2.1 简单的 C 程序举例

为了让 C 语言初学者对 C 语言编程有个感性的认识,首先介绍几个简单的 C 语言程序。

【例 1-1】 在屏幕上输出如下所示的一行信息。

```
hello world!
```

程序如下:

```
#include <stdio.h>           /* 预处理命令 */
int main()                   /* 定义 main 函数 */
{
    printf("hello world!\n"); //输出 hello world!
    return 0;                //表示程序正常结束
}
```

运行结果:

```
hello world!
请按任意键继续...
```

上面的运行结果是在 Visual C++ 2010 环境下运行程序时屏幕上的显示结果。其中第一行的输出“hello world!”是“printf(“hello world! \n”);”语句运行的结果。第二行的输出“请按任意键继续…”或“Press any key to continue…”是 Visual C++ 2010 系统在输出完运行结果后自动输出的信息,它告诉用户:“若想继续进行下一步,请按任意键。”当用户按下键盘上的任意键后,输出窗口关闭,并立刻返回到程序窗口,以便用户进行下一步的工作。为了使运行结果更纯粹,后面的运行结果截图不再包含这段文字。

程序分析:

(1) 该 C 程序由一个预处理命令 `#include` 和一个主函数 `main` 构成。

(2) 预处理命令 `#include` 称为文件包含命令, `stdio.h` 是系统提供的一个标准输入输出的头文件, `stdio` 是 `standard input & output` 的缩写, 扩展名, `h` 表示该文件的类型是头文件(header file)。如果在程序中用到标准库函数中的输入(`scanf` 函数)和输出函数(`printf` 函数), 则在源程序的开头一定要写上 `#include <stdio.h>` 语句, 请注意此 `#include` 语句的末尾没有分号。

(3) `int main()` 称为函数首部。其中, `int` 为 `main` 函数返回值类型符, `main` 是主函数的名称。每一个 C 程序有且仅有一个 `main` 函数, 每个 C 程序都从 `main` 函数开始执行, 在 `main` 函数中结束。

(4) 紧接函数首部之后是一对大括弧, 所有的语句都放在其内, 这一部分称为函数体。

(5) 在本程序函数体内, 包含一条调用 `printf` 函数的语句。 `printf` 函数功能是把要输出的内容(由一对双引号括起来的字符串“hello world!”)送到显示器去显示。请读者注意, 这一对双引号本身并不显示。该字符串末尾的 `\n` 是换行符, 在输出 `hello world!` 后, 显示器上的光标位置移动到下一行的开头。在 `printf` 函数调用后面紧接的“;”是 C 语言语句的结束标志, 每一条 C 语句末尾都有一个分号。

(6) 最后一条语句“`return 0;`”表示当主函数正常结束时, 得到的函数值为 0。之所以加上这条语句, 是为了使程序更加规范。

(7) 上述代码中出现的 `/* ... */` 和 `//` 都是 C 语言中的注释语句。注释是用来对程序的有关部分进行必要的说明和解释。良好的编程风格是在写程序时要多用注释, 以方便自己和他人理解程序各部分的作用。在程序进行编译时注释部分不产生目标代码, 注释对运行不起作用。注释只是给人看的, 计算机并不执行。

C 语言有两种形式的注释, 一种是以 `//` 开始的单行注释。这种注释可以写在一行的开头, 也可以写在一行其他内容的右侧。从 `//` 开始, 到这行末尾, 之间所有的内容都作为注释。另一种是以 `/*` 开始, 以 `*/` 结束的块注释。它可以单独占一行, 也可以包含多行。编译系统会把从 `/*` 开始, 到 `*/` 结束之间的所有内容都作为注释。在 Visual C++ 2010 系统中, 注释语句默认是以绿色字体显示的。

【例 1-2】 求两个整数的和。

```
#include <stdio.h>           /* 预处理命令 */
int main()                   /* 定义 main 函数 */
{
    int x, y, sum;           // 定义 3 个变量 x, y 和 sum
    x=1;                     // 对变量 x 赋值
    y=2;                     // 对变量 y 赋值
    sum=x+y;                 // 进行 x+y 的运算, 并把运算结果存放到变量 sum 中
    printf("%d+%d=%d\n", x, y, sum); // 输出结果
    return 0;
}
```

运行结果:

1+2=3

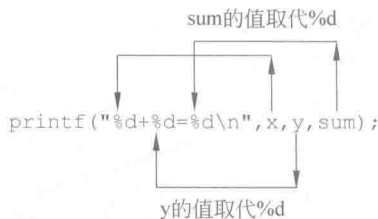
程序分析:

(1) main 函数内的“int x,y,sum;”称为变量定义,用来定义变量 x、y 和 sum 的数据类型。int 是 C 语言中的整数数据类型,在 Visual C++ 2010 系统中,int 是以蓝色字体显示的。x、y、sum 是三个变量名称,它们的数据类型都是整型。C 语言规定,源程序中所有用到的变量都必须“先定义,后使用”,否则将会出错。

(2) “x=1;”是一条赋值语句。C 语言中的=表示赋值,表示将等号右边的常量 1 赋给左边的变量 x,赋完值之后,x 变量的值就是 1。

(3) “sum=x+y;”也是一条赋值语句。将变量 x 的值与变量 y 的值相加之后的结果赋给变量 sum,赋完值之后,sum 变量的值就是 3。

(4) “printf(“%d+%d=%d\n”,x,y,sum);”是 C 语言中的输出语句。printf 函数圆括号内有 4 个参数,这 4 个参数之间用逗号分隔。第一个参数是“%d+%d=%d\n”,它是以一对双引号括起来的输出格式字符串。%d 表示以十进制整数的格式输出结果。这里有三个%d,依次来控制变量 x、变量 y、变量 sum 的输出格式。“%d+%d=%d\n”中的+和=称为普通字符,出现在 printf 的格式控制字符串中,输出时,是原样输出这些普通字符。\\n 是转义字符,出现在 printf 的格式控制字符串中,输出时,是进行换行,显示器上的光标进入到下一行。变量 x、变量 y、变量 sum 称为 printf 函数的输出项。输出时,是把这三个变量的值按照十进制整数的格式输出。



【例 1-3】 输入学生的学号和年龄,并在屏幕上显示出来。

```
/* the program displays your roll number and age */
#include <stdio.h>                                /* 预处理命令 */
int main()                                       /* 定义 main 函数 */
{
    int i,j;                                    /* 定义两个变量,i 存放学号,j 存放年龄 */
    printf("please input your roll number:");
    scanf("%d",&i);                             /* 输入学生的学号 */
    printf("please input your age:");
    scanf("%d",&j);                             /* 输入学生的年龄 */
    printf("My roll number is NO.%d",i); /* 输出学生的学号 */
    printf("\\n");
    printf("I am %d years old.\\n",j); /* 输出学生的年龄 */
    return 0;
}
```