计算机“十三五”规划教材
“互联网+计算机教育”立体化教材

C语言程序设计教程

主编 王剑峰 马涛 刘浪

含微课

航空工业出版社

计算机“十三五”规划教材

“互联网＋计算机教育”立体化教材

C语言程序设计教程

主编　王剑峰　马　涛　刘　浪

航空工业出版社

北　京

内容提要

本书从初学者的角度出发，以通俗易懂的语言、丰富多彩的实例，详细地介绍了使用C语言进行程序开发所需掌握的知识和技术。本书共分为12章，内容包括C语言概述、算法、C程序设计基础、顺序结构程序设计、选择结构程序设计、循环结构程序设计、数组、函数、指针、预处理命令、用户自定义数据类型和文件。

本书可作为各类院校和培训机构C语言程序设计课程的专用教材，也可供程序设计爱好者自学使用。

图书在版编目（CIP）数据

C语言程序设计教程 / 王剑峰，马涛，刘浪主编. -- 北京 : 航空工业出版社，2018.2
ISBN 978-7-5165-1555-6

Ⅰ. ①C… Ⅱ. ①王… ②马… ③刘… Ⅲ. ①C语言－程序设计－高等学校－教材 Ⅳ. ①TP312.8

中国版本图书馆CIP数据核字(2018)第037378号

C语言程序设计教程
C Yuyan Chengxu Sheji Jiaocheng

航空工业出版社出版发行
（北京市朝阳区北苑2号院 100012）
发行部电话：010-84936597 010-84936343
北京谊兴印刷有限公司印刷 全国各地新华书店经售
2018年2月第1版 2018年2月第1次印刷
开本：787×1092 1/16 印张：19.75 字数：444千字
印数：1—2800 定价：49.80元

前言

C 语言作为一种计算机编程语言，具有功能强大、使用灵活、可移植性好等特点，同时兼备高级语言和低级语言的优点，深受广大编程人员的喜爱。另外，由于 C 语言的概念简洁、语句紧凑、表达能力强、程序结构性和可读性好，因此很多院校开设了“C 语言程序设计”这一课程作为第一门计算机语言课程。

随着社会的发展，传统的教学模式已难以满足就业的需要。本书根据应用型本科教学模式的改革需要，结合作者多年讲授 C 语言程序设计课程的教学经验编写而成。

本书内容

本书共分为 3 篇：

（1）第 1 篇为基础知识篇，主要介绍了 C 语言基础知识，包括 C 语言概述、算法、C 程序设计基础、顺序结构程序设计、选择结构程序设计和循环结构程序设计等内容。

（2）第 2 篇为核心技术篇，主要介绍了数组、函数和指针这三大核心技术。

（3）第 3 篇为高级应用篇，主要介绍了预处理命令、用户自定义数据类型和文件等内容。

本书特色

（1）由浅入深，循序渐进。本书首先介绍了 C 语言的基础知识，然后介绍了 C 语言的基本程序结构，接着介绍了 C 语言的核心技术，最后介绍了 C 语言的高级应用，由浅入深，符合读者的认知规律和学习流程，帮助读者循序渐进学习相关知识。

（2）精讲多练，举一反三。本书采用“一个知识点，一个例子”的模式，安排了大量例题，难易适中，针对性强。除此之外，每个实例都给出了合理的讲解步骤，包括以下几个方面：

- 问题分析：给出解决问题的思路和算法。
- 参考代码：给出关键代码，并进行详细注释。
- 运行结果：使用直观的运行界面截图以验证程序运行结果。
- 程序说明：对运行结果进行分析，并对关键技术进行总结。

另外，本书大部分章节还增加了“应用举例”小节，让读者能够通过对实例的学习巩固所学的知识，做到融会贯通，举一反三。

（3）错误分析，高效学习。本书中大部分章节都增加了“常见错误分析”小节，指出了初学者在学习过程中可能遇到的一些常见问题，并给出了正确的解决方法，帮助初学者有效提高学习效率。

（4）精彩栏目，贴心提醒。本书根据需要在各章使用了“提示”“名师点睛”“高手点拨”“拓展阅读”等栏目，帮助读者轻松掌握关键技术的应用技巧。

（5）配套微课，扫码即得。本书采用最新的二维码技术，读者借助手机或其他移动设备扫码即可获取相关知识的微课视频，从而更方便地理解和掌握相关内容。

（6）配套资源，丰富多彩。本书配有课件、微课视频、习题答案等教学资源，方便老师教学和学生学习。

本书读者对象

本书可作为各类院校和培训机构C语言程序设计课程的专用教材，也可供程序设计爱好者自学使用。

教学资源下载

本书配有精心制作的教学课件，书中的全部例题和习题答案都已整理和打包，读者可到北京金企鹅联合出版中心网站（www.bjjqe.com）下载。如果读者在学习过程中有什么疑问，也可登录该网站寻求帮助，我们将会及时解答。

尽管我们在编写本书时已竭尽全力，但书中存在的疏漏及错误之处，敬请广大读者批评指正。

编　者

2018年1月

本书编委会

主　编　王剑峰　马　涛　刘　浪

副主编　马　东　陈淑萍　李建鹏

赵晓容　范智华　闫　政

韩欣洲　凡成福　越　缙

杨业娟　胡　翰　侯玉香

王雅峰　王增刚

目录 Contents

基础知识篇

第 1 章　C 语言概述 ……2
学习目标 ……2
1.1　C 语言的发展 ……2
1.2　C 语言的特点 ……3
1.3　第一个 C 语言程序 ……4
1.3.1　最简单的 C 语言程序结构 ……4
1.3.2　C 语言程序的格式特点 ……6
1.4　C 程序的开发步骤 ……7
1.5　开发环境 ……9
1.5.1　Visual C++ 6.0 ……9
1.5.2　Dev-C++ ……15
1.6　常见错误分析 ……20
本章小结 ……22
思考与练习 ……23

第 2 章　算法 ……25
学习目标 ……25
2.1　算法的概念 ……25
2.2　算法的特点 ……26
2.3　算法的表示 ……26
2.3.1　自然语言 ……27
2.3.2　流程图 ……27
2.3.3　N-S 流程图 ……30
2.4　应用举例 ……32
2.4.1　判断是否闰年 ……32
2.4.2　斐波那契数列 ……33
本章小结 ……35
思考与练习 ……36

第 3 章　C 程序设计基础 ……37
学习目标 ……37
3.1　C 程序组成元素 ……37
3.1.1　关键字 ……38
3.1.2　变量与常量 ……38
3.1.3　标识符 ……38
3.1.4　数据类型 ……39
3.2　常量 ……39
3.2.1　整型常量 ……39
3.2.2　实型常量 ……40
3.2.3　字符型常量 ……40
3.2.4　符号常量 ……41
3.3　变量 ……42
3.3.1　整型变量 ……43
3.3.2　实型变量 ……45
3.3.3　字符型变量 ……47
3.4　运算符和表达式 ……48
3.4.1　算术运算符与算术表达式 ……49
3.4.2　赋值运算符与赋值表达式 ……51
3.4.3　逗号运算符和逗号表达式 ……53
3.4.4　位运算符与位运算 ……53
3.5　数据类型转换 ……55
3.5.1　隐式类型转换 ……55
3.5.2　强制类型转换 ……56
3.6　常见错误分析 ……57
本章小结 ……60
思考与练习 ……60

第 4 章　顺序结构程序设计 ……63
学习目标……63
4.1　C 语句概述……63
4.2　格式输入输出……65
4.2.1　格式输出——printf 函数 ……65
4.2.2　格式输入——scanf 函数……68
4.3　字符输入输出……71
4.3.1　字符输出——putchar 函数 ……71
4.3.2　字符输入——getchar 函数 ……72
4.4　应用举例……73
4.4.1　输出字符表情……73
4.4.2　大小写字母转换……74
4.4.3　求三角形的面积……75
4.5　常见错误分析……76
本章小结……78
思考与练习 ……78

第 5 章　选择结构程序设计 ……81
学习目标……81
5.1　关系运算符和关系表达式……81
5.1.1　关系运算符……81
5.1.2　关系表达式……82
5.2　逻辑运算符和逻辑表达式……83
5.2.1　逻辑运算符……83
5.2.2　逻辑表达式……84
5.3　if 语句的基本形式 ……85
5.3.1　if 语句形式 ……85
5.3.2　if-else 语句形式 ……86
5.3.3　if-else-if 语句形式……88
5.4　嵌套的 if 语句 ……90
5.5　条件运算符和条件表达式…… 92
5.6　switch 语句 …… 92
5.7　应用举例 …… 96
5.7.1　判断是否闰年 …… 96
5.7.2　加油站加油问题 …… 97
5.7.3　求解一元二次方程 …… 99
5.8　常见错误分析 ……100
本章小结……104
思考与练习……104

第 6 章　循环结构程序设计……107
学习目标……107
6.1　while 循环语句……107
6.2　do-while 循环语句……109
6.3　for 循环语句……113
6.3.1　for 语句的一般形式……113
6.3.2　for 语句的说明……114
6.4　三种循环语句的比较……118
6.5　循环嵌套……119
6.6　转移语句……122
6.6.1　break 语句……122
6.6.2　continue 语句……123
6.6.3　goto 语句和标号……125
6.7　应用举例……126
6.7.1　斐波那契数列……126
6.7.2　四方定理……127
6.7.3　百钱买百鸡问题……128
6.8　常见错误分析……129
本章小结……131
思考与练习……131

核心技术篇

第 7 章　数组 …… 136
学习目标…… 136
7.1　一维数组…… 136
7.1.1　一维数组的定义…… 136
7.1.2　一维数组的引用…… 138
7.1.3　一维数组的初始化…… 139
7.1.4　一维数组应用举例…… 139
7.2　二维数组…… 142
7.2.1　二维数组的定义…… 142
7.2.2　二维数组的引用…… 142
7.2.3　二维数组的初始化 ……143
7.2.4　二维数组应用举例 ……143
7.3　字符数组和字符串 ……145
7.3.1　字符数组的定义和引用 ……145
7.3.2　字符数组的初始化 ……146
7.3.3　字符串 ……147
7.3.4　字符数组的输入输出 ……147
7.3.5　字符串处理函数 ……150
7.3.6　字符数组应用举例 ……153
7.4　常见错误分析 ……155

本章小结 ……157
思考与练习 ……158
第 8 章 函数 ……160
学习目标 ……160
8.1 函数概述 ……160
8.1.1 库函数 ……161
8.1.2 用户自定义函数 ……162
8.2 函数定义 ……163
8.3 函数的调用 ……164
8.3.1 函数的调用形式 ……164
8.3.2 函数声明 ……166
8.3.3 函数的参数传递 ……168
8.3.4 返回语句和函数返回值 ……169
8.4 函数的嵌套调用 ……170
8.5 函数的递归调用 ……172
8.6 数组作为函数的参数 ……173
8.6.1 数组元素作为函数参数 ……173
8.6.2 数组名作为函数参数 ……174
8.7 局部变量和全局变量 ……176
8.7.1 局部变量 ……176
8.7.2 全局变量 ……178
8.8 变量的存储类别 ……180
8.8.1 局部变量的存储类别 ……180
8.8.2 全局变量的存储类别 ……182
8.8.3 存储类别小结 ……184
8.9 内部函数和外部函数 ……185
8.9.1 内部函数 ……185
8.9.2 外部函数 ……185
8.10 应用举例 ……185
8.10.1 哥德巴赫猜想 ……186
8.10.2 快速排序 ……187
8.11 常见错误分析 ……189
本章小结 ……192
思考与练习 ……192
第 9 章 指针 ……195
学习目标 ……195
9.1 指针的概念 ……195
9.2 指针变量 ……196
9.2.1 指针变量定义 ……196
9.2.2 指针变量初始化 ……197
9.2.3 指针变量引用 ……197
9.2.4 空指针与 void 指针 ……199
9.3 指针与数组 ……200
9.3.1 指针与一维数组 ……200
9.3.2 指针与二维数组 ……203
9.3.3 指针与字符串 ……209
9.4 指针与函数 ……210
9.4.1 指针变量作为函数参数 ……211
9.4.2 指针作为函数的返回值 ……214
9.4.3 函数的指针 ……215
9.5 指针数组和指向指针的指针 ……216
9.5.1 指针数组 ……216
9.5.2 指向指针的指针 ……218
9.5.3 指针数组作为 main 函数的形参 ……220
9.6 应用举例 ……221
9.6.1 删除有序数组中重复元素 ……221
9.6.2 卡布列克常数 ……223
9.7 常见错误分析 ……224
本章小结 ……228
思考与练习 ……228

高级应用篇

第 10 章 预处理命令 ……234
学习目标 ……234
10.1 宏定义 ……234
10.1.1 不带参数的宏定义 ……234
10.1.2 带参数的宏定义 ……235
10.1.3 撤销宏定义命令 ……238
10.2 文件包含命令 ……238
10.3 条件编译命令 ……240
10.4 常见错误分析 ……242
本章小结 ……243
思考与练习 ……243
第 11 章 用户自定义数据类型 ……246
学习目标 ……246
11.1 结构体 ……246
11.1.1 结构体类型声明 ……247

11.1.2 结构体变量定义 …… 248
11.1.3 结构体变量初始化 …… 250
11.1.4 结构体变量引用 …… 251
11.2 结构体数组 …… 253
11.2.1 结构体数组定义 …… 253
11.2.2 结构体数组初始化 …… 254
11.2.3 结构体数组应用举例 …… 254
11.3 结构体指针 …… 255
11.3.1 指向结构体变量的指针 …… 255
11.3.2 指向结构体数组的指针 …… 256
11.3.3 结构体作为函数参数 …… 258
11.3.4 结构体指针的应用举例 …… 259
11.4 链表 …… 261
11.4.1 链表概述 …… 261
11.4.2 处理动态链表的函数 …… 263
11.4.3 建立动态链表 …… 264
11.5 共用体 …… 267
11.6 枚举类型 …… 268
11.7 用 typedef 声明新类型名 …… 270
11.8 常见错误分析 …… 271
本章小结 …… 273
思考与练习 …… 273
第 12 章 文件 …… 277
学习目标 …… 277
12.1 文件概述 …… 277
12.2 文件基本操作 …… 278
12.2.1 文件类型指针 …… 278
12.2.2 文件打开——fopen 函数 …… 279
12.2.3 文件关闭——fclose 函数 …… 281
12.3 文件读写 …… 281
12.3.1 字符读写——fgetc 和 fputc 函数 …… 281
12.3.2 字符串读写——fgets 和 fputs 函数 …… 284
12.3.3 格式化读写——fscanf 和 fprintf 函数 …… 285
12.3.4 数据块读写——fread 和 fwrite 函数 …… 287
12.4 文件定位 …… 289
12.5 应用举例 …… 291
12.5.1 文件内容复制 …… 291
12.5.2 将计算结果保存到文件 …… 292
12.6 常见错误分析 …… 294
本章小结 …… 294
思考与练习 …… 295
附录 …… 297
附录 1 常用字符与 ASCII 代码对照表 …… 297
附录 2 运算符的优先级和结合性表 …… 297
附录 3 常用标准库函数 …… 298
参考文献 …… 304

基础知识篇

第1章　C语言概述

在现实生活中，人与人之间的交流用的是语言，如汉语、英语、法语等。那么，人与计算机之间交流用的是什么呢？答案是编程语言。比如，要让计算机帮助人们做一些事情，此时可利用编程语言编写程序来告诉它怎么做，而C语言就是一种通用的、过程式的编程语言。

本章首先介绍了C语言的发展和特点，然后通过一个简单的C语言程序实例，让读者了解C语言程序结构及其格式特点，最后介绍了C程序的开发步骤及开发环境，其目的是使读者能够通过编写一个简单的C语言程序练习使用开发环境。

学习目标

- 了解C语言的发展
- 了解C语言的特点
- 熟悉C语言程序结构及格式特点
- 熟悉C程序的开发步骤
- 掌握使用Visual C++ 6.0编译器开发C程序的方法
- 了解使用Dev-C++编译器开发C程序的方法

1.1　C语言的发展

在介绍C语言的发展之前，我们先来看一下计算机语言经历的几个发展阶段。

机器语言阶段　机器语言是低级语言，是用二进制代码表示的一种机器指令的集合。其特点是计算机可以直接识别和执行，不需要进行任何的解释和翻译。

汇编语言阶段　汇编语言是机器语言的升级版，它用英文字母或符号串代替了机器语言中的二进制码，因此，汇编语言也称符号语言，它比机器语言程序便于阅读和理解。

高级语言阶段　高级语言是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而产生的。这种语言的语法形式类似英文，易于被人们理解和使用。早期使用较多的高级语言主要有FORTRAN、ALGOL、BASIC、COBOL、LISP、Pascal、VB等，当然也包括C语言。不过，随着时代的发展，早期流行的很多语言（如FORTRAN、COBOL等）都已被淘汰。目前，C语言主要用于教学及嵌入式系统的应用程序开发，实际开发中使用较多的编程语言主要有Java、C++、C#、PHP、JavaScript、Python等。

C语言的原型是ALGOL 60语言（也称为A语言）。1963年，剑桥大学将ALGOL 60

语言发展成为 CPL（Combined Programming Language）语言。1967 年，剑桥大学的 Martin Richards 对 CPL 语言进行了简化，于是产生了 BCPL（Basic Combined Programming Language）语言。1970 年，美国 AT&T 贝尔实验室的 Ken Thompson 以 BCPL 语言为基础，设计出了简单且接近硬件的 B 语言（取 BCPL 的第一个字母）。但 B 语言过于简单，功能有限。1972～1973 年间，美国贝尔实验室的 D.M.Ritchie 在 B 语言的基础上设计出了一种新的语言，他取了 BCPL 的第二个字母作为这种语言的名字，这就是 C 语言。

C 语言与 UNIX 操作系统有着密切的联系，开发 C 语言的最初目的是为了更好地描述 UNIX 操作系统。C 语言的出现，促进了 UNIX 操作系统的开发，同时随着 UNIX 的日益广泛使用，C 语言也迅速得到推广。

1983 年，美国国家标准协会（ANSI）对 C 语言进行了标准化，经过多次修订，C 语言标准于 1988 年完成并在 1989 年 12 月正式通过，我们把这一 C 语言版本称为 C89。最新的 C 语言标准 C99 于 1999 年颁布，并在 2000 年 3 月被 ANSI 采用。

C 语言对现代编程语言有着巨大的影响，许多现代编程语言都借鉴了大量的 C 语言特性。如 C++包括了所有 C 特性，但增加了类和其他特性以支持面向对象编程；又如，Java 是基于 C++的，所以也继承了许多 C 语言的特性；而 C#是由 C++和 Java 发展起来的一种语言，所以它也继承了 C 语言的许多特性。

1.2　C 语言的特点

C 语言之所以能够流行起来，归功于它有以下一些主要特点。

1．简洁

C 语言是现有程序设计语言中规模最小的语言之一。C 语言只有 37 个关键字（见附录 3）、9 种控制语句，压缩了一切不必要的成分。C 语言表达方法简洁，使用一些简单的方法就可以构造出相当复杂的数据类型和程序结构。

2．灵活

C 语言的语法限制不太严格，程序设计的自由度比较大，程序的书写格式与变量的类型使用都很灵活，如整型数据、字符型数据和逻辑型数据可以通用。

3．功能丰富

C 语言具有丰富的数据类型，包括整型、实型、字符型、数组类型、指针类型、结构体类型、共用体类型、枚举类型等。C 语言还具有多种运算符（共 34 种，见附录 4），灵活使用各种运算符可以实现其他高级语言难以实现的运算。

4．模块化和结构化

C 语言用函数作为程序模块，以实现程序的模块化；C 语言具有结构化的控制语句，如 if…else 语句、while 语句、do…while 语句、switch 语句和 for 语句等。

5．具有低级语言的功能

C 语言允许直接访问物理地址，能进行位操作，能实现汇编语言的大部分功能，可以直接对硬件进行操作。因此，C 语言既具有高级语言的功能，又具有低级语言的功能，故

有人把 C 语言称为“高级语言中的低级语言”或“中级语言”。由于 C 语言的双重性，所以它既可以用来编写应用软件，又可以用来编写系统软件。

6. 执行效率高

C 语言生成的目标代码质量高，程序执行效率高。C 语言一般只比汇编程序生成的目标代码效率低 10%～20%。

7. 可移植性好

C 语言是通过编译和连接来得到可执行代码的，由于 C 语言的编译系统相当简洁，因此很容易移植到新的系统。

提　示

这一节的内容相对比较抽象和不好理解，读者待学习完这本教材后再来回顾一下，会有更好的领悟。

1.3　第一个 C 语言程序

与用其他语言编写的程序相比，C 语言较少要求“形式化的内容”。一个完整的 C 程序可以只有寥寥数行。

1.3.1　最简单的 C 语言程序结构

由于读者刚开始接触 C 语言，在这里我们先不长篇论述 C 程序的全部组成部分，只介绍 C 程序的基本组成部分。在读者会写简单的 C 程序之后上，通过后面章节的学习再逐步深入掌握 C 程序的完整结构。

下面以一个简单的例子说明 C 程序的基本结构。

【例 1-1】　只有一行输出的 C 程序。

```
#include <stdio.h>                    /*编译预处理指令*/
int main()                            /*主函数的函数头*/
{                                     /*函数体的开始标记*/
    printf("你好，C 语言！\n");       /*输出要显示的字符串*/
    return 0;                         /*程序返回值 0*/
}                                     /*函数的结束标记*/
```

【运行结果】　程序运行结果如图 1-1 所示。

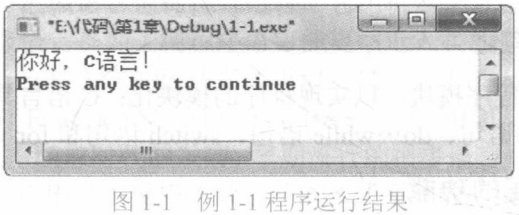

图 1-1　例 1-1 程序运行结果

【程序说明】　该程序运行结果的第 1 行是程序运行后输出的结果，第 2 行是任何一个 C 程序在 Visual C++ 6.0 环境下运行都会自动输出的一行信息，告诉用户：“如果想继续，

请按任意键”。当用户按任意键后，屏幕上不再显示运行结果，返回程序窗口。

小技巧

运行界面默认情况下显示的是黑底白字，为了美观可以进行设置。方法如下：

（1）右击运行窗口的标题栏，在弹出的快捷菜单中选择“属性”选项，如图 1-2 所示。

（2）这时会弹出“属性”对话框，在属性对话框中选择最后一个选项卡“颜色”，在这里可以根据需求设置“屏幕文字”和“屏幕背景”的颜色，如图 1-3 所示。在本书中所有运行结果截图均采用白底黑字模式。

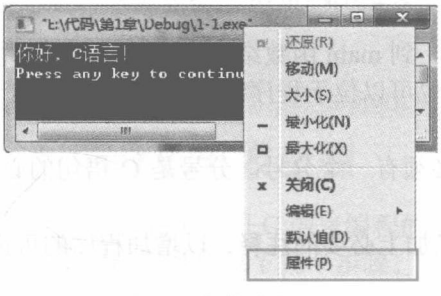

图 1-2　快捷菜单

图 1-3　“属性”对话框

例 1-1 的程序是一个由头文件和主函数组成的简单 C 语言程序，下面分别解释一下各行程序的意义。

第 1 行：

```
#include <stdio.h>
```

这是一个预处理操作。include 称为文件包含命令，后面尖括号中的内容称为头文件，stdio.h 是 C 语言的系统文件，stdio 是“standard input & output（标准输入输出）”的缩写，.h 是文件的扩展名。由于程序的第 4 行使用了库函数 printf，编译系统要求程序提供有关此函数的信息（例如对这些输入输出函数的声明和宏的定义、全局变量的定义等），所以此处需要这条命令。

第 2 行：

```
int main()
```

这一行代码是函数头，其中 main 是函数的名字，表示“主函数”，main 前面的 int 表示函数的返回值是 int 类型（整型）。每个 C 语言程序都必须有一个 main 函数。

第 3 行到第 6 行：

```
{
    printf("你好，C 语言！\n");
```

```
    return 0;
}
```

由花括号{}括起来的部分是函数体，该程序主函数的函数体由两条语句构成，每条语句后都要加分号，表示语句结束。其中 printf 是 C 编译系统提供的函数库中的输出函数，用来在屏幕输出内容，输出语句中双引号中间可以是字母、符号及中文字符等；“return 0;”的作用是当 main 函数执行结束前将整数 0 作为函数值，返回到调用函数处。

在程序各行的右侧都可以看到一段关于这行代码的文字描述（用/*　*/括起来），称为代码注释。其作用是对代码进行解释说明，为日后自己阅读或者他人阅读源程序时方便理解程序代码含义和程序设计思路。

通过对例 1-1 的介绍，我们可以看到一个 C 语言程序的结构主要有以下特点：

（1）一个 C 程序由一个或多个源程序文件组成。一个规模较小的程序，往往只包括一个程序文件。本书中的例子都是基于一个源程序文件的。

（2）C 程序是由函数构成的，函数是 C 程序的基本单位。任何一个 C 语言源程序必须包含一个且仅包含一个 main 函数，可以包含零个或多个其他函数。

（3）一个函数由两部分组成：函数头和函数体。函数头如例 1-1 中的 int main()。函数体为函数头下面花括号{}内的部分。若一个函数内有多个大括号，则最外层的一对{}为函数体的范围（关于函数的组成可参见第 8 章）。

（4）一个 C 程序总是从 main 函数开始执行，到 main 函数结束，与 main 函数所处的位置无关（main 函数可以位于程序的开始位置，也可以位于程序的末尾，还可以位于一些自定义函数的中间）。

（5）C 程序的每个语句和数据定义的最后必须有一个分号。分号是 C 语句的必要组成部分，必不可少。

（6）一个好的、有使用价值的源程序都应当加上必要的注释，以增加程序的可读性。

知识库

C 语言允许用两种注释方式：

① 以//开始的单行注释。这种注释可以单独占一行，也可以出现在一行中其他内容的右侧。此种注释的范围从//开始，以换行符结束，即这种注释不能跨行。若注释内容一行内写不下，可以用多个单行注释。如：

```
printf("你好，C 语言！\n");          //输出要
//显示的字符串
```

② 以/*开始，以*/结束的块式注释。这种注释可以单独占一行，也可以包含多行。编译系统在发现一个/*后，会开始找注释结束符*/，把二者间的内容作为注释，如例 1-1 中的注释。

1.3.2　C 语言程序的格式特点

通过上面的实例可以看出，C 语言编写有一定的格式特点，具体如下：

（1）函数体中的大括号用来表示程序的结构层次，需要注意的是：左右大括号要成对使用。

小技巧

在编写程序时，为了防止对应大括号的遗漏，建议先将两个对应的大括号输入程序中，再往括号中添加代码。

（2）在程序中，可以使用英文的大写字母，也可以使用小写字母。但要注意的是，在 C 语言中是区分字母大小写的，也就是说，大写字母和小写字母代表不同的字符，如‘a’和‘A’是两个完全不同的字符。一般情况下，C 语言程序中使用小写字母较多，但在定义常量时会使用大写字母。

（3）在程序中的空格、空行、跳格并不会影响程序的执行。合理地使用这些空格、空行，可以使编写出来的程序更加规范，有助于日后的阅读和整理。

（4）C 程序书写格式自由，一行内可以写多个语句，一个语句可以分写在多行上。但为了有良好的编程风格，最好将一条语句写在一行。

（5）代码缩进统一为 4 个字符。建议不使用空格，而用【Tab】键，如例 1-1 中的第 4 行和第 5 行。例如，以下程序完全是正确的，但是由于书写格式不规范，因而不利于阅读和理解，应参照例 1-1 的格式来进行书写。

```
#include <stdio.h>
int main()
{printf("你好，C 语言！\n");return 0;}
```

1.4　C 程序的开发步骤

学习 C 语言就是学习 C 语言编程的过程。C 程序的开发从确定任务到得到结果一般要经历以下几个步骤。

1. 需求分析

需求分析是指对要解决的问题进行详细的分析，弄清楚问题的要求，包括需要输入什么数据，要得到什么结果，最后应输出什么等。这个过程好比是考试时候的审题，一定要领会题目的要求，如若不然解题过程再漂亮也是无济于事的。

2. 算法设计

算法设计是指对要解决的问题设计出解决问题的方法和具体步骤。例如，要求解一个 1 到 100 的累加问题，首先要选择用什么方法求解（直接累加计算还是用等差数列的求和公式进行计算）；然后把求解的每一步清晰地描述出来。描述算法有多种方式，详见第 2 章。

3. 编写程序

编写程序是把算法设计的结果变成一行行代码，输入到程序编辑器中，然后将此源程序以文件形式保存到自己指定的文件夹内。文件用.c 作为后缀，如 file.c。

4. 编译程序

编译程序需要利用编译器把送入的源程序翻译成机器语言，也就是编译器对源程序进行语法检查并将符合语法规则的源程序语句翻译成计算机能识别的语言。如果经编译器检