



普通高等教育“十一五”国家级规划教材

# C语言 程序设计教程

朱鸣华 刘旭麟 杨微 等编著

为教师提供电子课件



机械工业出版社  
China Machine Press

TP312  
2241  
2007



普通高等教育“十一五”国家级规划教材

# C语言 程序设计教程

朱鸣华 刘旭麟 杨微 编著  
李慧 孙大为 罗晓芳 赵晶



机械工业出版社  
China Machine Press

本书介绍利用C语言进行计算机程序设计的基本知识。全书共分13章，详细介绍了C语言的基本概念、输入和输出、选择结构、循环结构、数组、函数、编译预处理、指针、结构体与共用体、文件等内容，同时介绍了面向对象程序设计语言C++的基本概念和数据结构的基本内容。另外，每章配有大量例题和习题，便于读者巩固所学知识，掌握程序设计的基本方法与编程技巧。

本书力求概念叙述准确、严谨，语言简练通俗，条理清晰易读。本书可作为高等院校理工科非计算机专业的“C语言程序设计”课程的教材，同时也可作为工程技术人员学习C语言的自学用书。

版权所有，侵权必究。

本书法律顾问 北京市展达律师事务所

## 图书在版编目 (CIP) 数据

C语言程序设计教程 / 朱鸣华等编著. —北京: 机械工业出版社, 2007.2  
(普通高等教育“十一五”国家级规划教材)  
ISBN 978-7-111-20683-5

I. C… II. 朱… III. C语言—程序设计—高等学校—教材 IV. TP312

中国版本图书馆CIP数据核字 (2007) 第002715号

机械工业出版社 (北京市西城区百万庄大街22号 邮政编码 100037)

策划编辑: 武恩玉

责任编辑: 迟振春

北京慧美印刷有限公司印刷 · 新华书店北京发行所发行

2007年2月第1版第1次印刷

184mm × 260mm · 17.5印张

定价: 26.00元

凡购本书, 如有倒页、脱页、缺页, 由本社发行部调换

本社购书热线: (010) 68326294

# 前 言

掌握一门高级程序设计语言是高等院校理工科学生利用计算机解决实际问题的必要条件之一。目前,国内各高等院校都把计算机程序设计课程作为重要的计算机基础课程。

C语言是国内外广泛使用的计算机程序设计语言之一,是国内外大学普遍开设的程序设计课程。C语言功能丰富,表达能力强,使用灵活方便,程序执行效率高,它不但具有高级语言的功能,而且还可以实现汇编语言的许多功能。C语言程序具有完善的模块程序结构,可移植性好,而且可以直接实现对系统硬件及外部设备接口的控制,具有较强的系统处理能力。

本书是根据教育部《关于进一步加强高等学校计算机基础教学的意见》的教学基本要求和高等院校计算机基础教学改革的需要,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的。全书共分13章,详细介绍了C语言的基本语法结构,同时介绍了面向对象程序设计语言C++的基本概念和数据结构的基本内容。另外,每章均配有精心安排的例题和习题,书后还给出了习题参考答案。

本书内容丰富,实用性强,侧重结合一般数值计算介绍利用C语言进行计算机程序设计的基本知识,使读者掌握C语言的基本内容及程序设计的基本方法和编程技巧,了解进行科学计算的一般思路与方法,为进一步学习和应用计算机打下基础。本书具有如下特点:

1) 内容全面。教材内容体系、前后内容的衔接、知识点的安排由浅入深,循序渐进,全书围绕结构化程序设计的方法,详细介绍C语言的语法规则和程序设计的基本方法。另外,C++和数据结构的内容有利于学生扩充知识及后续课程的学习。

2) 通俗易懂。本书在编写中力求做到概念叙述准确、严谨,语言简练通俗。为便于学生更好地学习和理解C语言,在一些较难理解的部分尽量使用图解的方法,在例题中加入了部分代码注释,使读者低起点、高效率、高目标地掌握C语言。

3) 实践性强。书中示例丰富,习题难易适中,覆盖面广,既包括理解语法的内容,又有增加兴趣和提高编程能力的例子。例题和习题强调了编程能力训练的重要性,淡化了部分语法细节。

本书的叙述以ANSI C为基础,书中的程序是在Turbo C环境下调试通过的,使用其他C语言编译系统的读者请注意参考有关手册。

本书适合作为高等院校理工科非计算机专业的“C语言程序设计”课程的教材和工程技术人员学习C语言的参考书。另外,本书还可作为自学用书。

本书适合于学时较少,并与多媒体教学方法相匹配的教学方式。对于本书的学时数,建议课堂教学32~48学时,上机实践24~32学时。为了克服学时少、内容多的矛盾,建议在教学中注重学生程序设计能力的培养,采用任务驱动、案例教学相结合的教学模式,以提高学生学习的兴趣和主动性,语法问题则留给学生课后自学。

本书第1、2、3章由罗晓芳编写，第4、7章由李慧编写，第5、9章和附录由刘旭麟编写，第6、11章由孙大为编写，第8、10章由杨微编写，第12章由赵晶编写，第13章由朱鸣华和赵晶编写。全书由朱鸣华、刘旭麟统稿。

在本书的编写过程中得到了许多老师的帮助，在此表示诚挚的谢意。由于作者水平有限，书中难免有疏漏和不当之处，敬请读者和专家指正。

作者  
2006年12月

# 目 录

## 前言

第1章 C语言概述 .....	1
1.1 C语言发展简史 .....	1
1.2 C语言的特点 .....	2
1.3 简单C语言程序举例 .....	2
1.4 C语言程序的组成与结构 .....	4
1.5 C语言程序的开发步骤 .....	5
小结 .....	6
习题 .....	6
第2章 数据类型、运算符与表达式 .....	8
2.1 C语言的基本符号 .....	8
2.1.1 标识符 .....	8
2.1.2 常量 .....	8
2.1.3 变量 .....	9
2.1.4 关键字 .....	10
2.2 C语言的数据类型 .....	10
2.2.1 整型数据 .....	11
2.2.2 实型数据 .....	13
2.2.3 字符型数据 .....	14
2.3 运算符和表达式 .....	16
2.3.1 算术运算符和算术表达式 .....	17
2.3.2 赋值运算符和赋值表达式 .....	18
2.3.3 逗号运算符和逗号表达式 .....	19
2.4 数据类型转换 .....	20
2.4.1 不同数据类型的数据间的混合运算 .....	20
2.4.2 强制类型转换 .....	21
2.5 自增运算和自减运算 .....	21
2.6 位运算 .....	22
小结 .....	24
习题 .....	25
第3章 数据的输入和输出 .....	27
3.1 数据的输出 .....	27
3.1.1 格式输出函数printf .....	27

3.1.2 字符输出函数putchar .....	31
3.2 数据的输入 .....	31
3.2.1 格式输入函数scanf .....	31
3.2.2 字符输入函数getchar .....	34
3.3 应用举例 .....	34
小结 .....	35
习题 .....	35
第4章 选择结构 .....	38
4.1 关系运算符和关系表达式 .....	38
4.1.1 关系运算符 .....	38
4.1.2 关系表达式 .....	38
4.2 逻辑运算符和逻辑表达式 .....	38
4.2.1 逻辑运算符 .....	38
4.2.2 逻辑表达式 .....	39
4.3 选择语句 .....	40
4.3.1 if语句 .....	40
4.3.2 if语句的嵌套 .....	42
4.3.3 switch语句 .....	43
4.4 条件运算符和条件表达式 .....	45
4.5 应用举例 .....	46
小结 .....	50
习题 .....	50
第5章 循环结构 .....	53
5.1 goto语句与标号 .....	53
5.2 while循环语句 .....	54
5.3 do-while循环语句 .....	56
5.4 for循环语句 .....	57
5.5 三种循环语句的比较 .....	60
5.6 循环嵌套 .....	60
5.7 break语句、continue语句和空语句 .....	62
5.7.1 break语句 .....	62
5.7.2 continue语句 .....	63
5.7.3 空语句 .....	64
5.8 应用举例 .....	65
小结 .....	68

习题 .....	68	小结 .....	118
第6章 数组 .....	74	习题 .....	118
6.1 数组的概念 .....	74	第8章 编译预处理 .....	124
6.2 一维数组 .....	74	8.1 宏定义 .....	124
6.2.1 一维数组的定义和引用 .....	74	8.1.1 无参宏定义 .....	124
6.2.2 一维数组的初始化 .....	75	8.1.2 带参宏定义 .....	126
6.2.3 一维数组应用举例 .....	76	8.2 文件包含 .....	128
6.3 二维数组 .....	77	8.2.1 文件包含命令的形式 .....	128
6.3.1 二维数组的定义和引用 .....	77	8.2.2 文件包含的实现过程 .....	129
6.3.2 二维数组的初始化 .....	78	8.3 条件编译 .....	130
6.3.3 二维数组应用举例 .....	78	8.3.1 条件编译命令的形式 .....	130
6.4 字符数组 .....	80	8.3.2 条件编译命令的嵌套 .....	132
6.4.1 字符数组的定义 .....	80	小结 .....	133
6.4.2 字符数组的初始化 .....	80	习题 .....	133
6.4.3 字符数组的引用 .....	80	第9章 指针 .....	136
6.4.4 字符数组与字符串 .....	81	9.1 指针的基本概念及指针变量的定义 .....	136
6.4.5 字符数组的输入和输出 .....	82	9.1.1 指针的基本概念 .....	136
6.4.6 字符串处理函数 .....	83	9.1.2 指针变量的定义方法 .....	136
6.4.7 字符数组应用举例 .....	85	9.2 指针运算 .....	137
小结 .....	86	9.2.1 赋值运算 .....	137
习题 .....	87	9.2.2 取地址运算 .....	137
第7章 函数 .....	91	9.2.3 取内容运算 .....	138
7.1 函数的基本概念 .....	91	9.2.4 指针表达式与整数相加、减运算 .....	139
7.1.1 函数的概念 .....	91	9.2.5 自增、自减运算 .....	139
7.1.2 函数的定义 .....	92	9.2.6 同类指针相减运算 .....	140
7.1.3 函数的调用 .....	94	9.2.7 关系运算 .....	140
7.1.4 函数参数的传递方式 .....	95	9.2.8 强制类型转换运算 .....	141
7.1.5 函数的返回值 .....	96	9.2.9 空指针 .....	141
7.1.6 函数的原型声明 .....	97	9.3 指针变量与一维数组 .....	142
7.2 数组作为函数参数 .....	99	9.3.1 指针变量与一维数组之间的 联系和区别 .....	142
7.2.1 一维数组作为函数参数 .....	99	9.3.2 字符串指针与字符串 .....	143
7.2.2 二维数组作为函数参数 .....	101	9.4 指针与函数 .....	144
7.2.3 数组作为函数参数的函数 调用方式 .....	102	9.4.1 指针作为函数参数 .....	144
7.3 函数的嵌套调用和递归调用 .....	104	9.4.2 返回指针的函数 .....	147
7.3.1 函数的嵌套调用 .....	104	9.4.3 函数的指针和指向函数的指针 变量 .....	148
7.3.2 函数的递归调用 .....	105	9.5 指针与二维数组 .....	150
7.4 变量的作用域和存储方法 .....	108	9.5.1 二维数组的结构 .....	150
7.4.1 局部变量和全局变量 .....	108	9.5.2 二维数组元素及其地址 .....	150
7.4.2 变量的存储方法 .....	111	9.5.3 指针数组 .....	152
7.5 内部函数和外部函数 .....	114	9.5.4 指针与字符串数组 .....	153
7.6 应用举例 .....	115		

9.5.5 指向数组的指针变量 .....	154	11.3 文件的打开与关闭 .....	197
9.6 二级指针 .....	156	11.3.1 文件的打开函数fopen .....	197
9.7 内存空间的动态分配 .....	158	11.3.2 文件的关闭函数fclose .....	198
9.7.1 指向void的指针 .....	158	11.4 文件的读写 .....	198
9.7.2 常用内存管理函数 .....	159	11.4.1 格式化读写函数fscanf和fprintf .....	198
9.8 main函数的参数 .....	159	11.4.2 向文件读取或写入字符函数 fgetc和putc .....	199
9.8.1 命令行参数 .....	160	11.4.3 读写数据块函数fread和fwrite .....	200
9.8.2 指针数组作为main函数的形参 .....	160	11.4.4 读写字符串函数fgets和puts .....	200
9.9 应用举例 .....	161	11.5 文件的定位 .....	201
小结 .....	167	11.5.1 文件指针重定位函数rewind .....	201
习题 .....	167	11.5.2 随机读写函数fseek .....	201
第10章 结构体与共用体 .....	174	11.5.3 取文件指针的位置函数ftell .....	202
10.1 结构体类型和结构体变量 .....	174	11.5.4 检测调用文件是否出错的 函数ferror .....	203
10.1.1 结构体类型的定义 .....	174	11.5.5 检测文件指针函数feof .....	203
10.1.2 结构体变量的定义 .....	175	小结 .....	203
10.1.3 结构体变量的引用 .....	176	习题 .....	203
10.1.4 结构体变量的初始化 .....	177	第12章 面向对象程序设计与C++基础 .....	207
10.2 结构体数组 .....	177	12.1 面向对象程序设计的基本思想 .....	207
10.2.1 结构体数组的定义 .....	178	12.1.1 结构化程序设计与面向对象 程序设计的区别 .....	207
10.2.2 结构体数组的引用 .....	178	12.1.2 面向对象程序设计的基本概念 .....	209
10.2.3 结构体数组的初始化 .....	179	12.2 面向对象程序设计语言C++简介 .....	211
10.3 应用举例 .....	179	12.2.1 C++程序结构 .....	211
10.4 结构体指针变量 .....	181	12.2.2 C++对C的补充 .....	213
10.4.1 结构体指针变量的定义 .....	181	12.2.3 C++中的类 .....	216
10.4.2 结构体数组指针 .....	182	12.2.4 C++中的构造函数和析构函数 .....	218
10.5 结构体类型数据在函数间的传递 .....	183	12.2.5 C++中的继承 .....	220
10.5.1 结构体变量作为函数参数 .....	183	12.2.6 C++中的多态性、函数重载 和虚函数 .....	221
10.5.2 结构体指针变量作为函数参数 .....	184	小结 .....	223
10.5.3 结构体数组作为函数参数 .....	185	习题 .....	224
10.6 共用体 .....	185	第13章 数据结构基础 .....	226
10.6.1 共用体类型的定义 .....	185	13.1 概述 .....	226
10.6.2 共用体变量的定义 .....	186	13.1.1 数据结构的基本概念 .....	226
10.6.3 共用体变量的引用和初始化 .....	186	13.1.2 算法的基本概念 .....	228
10.7 枚举类型 .....	189	13.2 线性表 .....	229
10.7.1 枚举类型的说明 .....	189	13.2.1 线性表概述 .....	229
10.7.2 枚举型变量的定义 .....	189	13.2.2 线性表的存储及运算 .....	229
10.8 用typedef定义类型 .....	191	13.3 栈和队列 .....	233
小结 .....	192	13.3.1 栈 .....	233
习题 .....	193		
第11章 文件 .....	196		
11.1 文件概述 .....	196		
11.2 文件类型的指针 .....	197		



13.3.2 队列 .....	235	13.7 排序.....	252
13.4 树与二叉树.....	239	小结 .....	253
13.4.1 树的定义 .....	239	习题 .....	254
13.4.2 二叉树 .....	240	习题参考答案 .....	255
13.4.3 哈夫曼树 .....	244	附录A Turbo C的主要文件 .....	259
13.5 图.....	246	附录B C语言的关键字 .....	261
13.5.1 图的定义 .....	246	附录C 双目算术运算中两边运算量 类型转换规律 .....	262
13.5.2 图的存储结构 .....	247	附录D 运算符的优先级和结合性 .....	263
13.5.3 图的应用——最小生成树 .....	248	附录E 常用字符与ASCII码对照表 .....	264
13.6 查找.....	249	附录F 常用库函数 .....	266
13.6.1 顺序查找 .....	249	参考文献 .....	270
13.6.2 折半查找 .....	250		
13.6.3 散列查找 .....	251		

# 第1章 C语言概述

## 1.1 C语言发展简史

在计算机科学应用领域，人与计算机之间交流最通用的手段是程序设计语言。当人们想利用计算机解决某个问题时，必须用程序设计语言安排好处理步骤，并存入计算机内供计算机执行，这些用程序设计语言安排好的处理步骤称为计算机程序。程序是计算机操作指令的集合。用程序设计语言编制计算机程序的过程叫做程序设计。

目前有很多程序设计语言，而且新的语言不断涌现。常见的有Fortran、BASIC、COBOL、Pascal、C、C++、Java、Delphi等。各类语言都有其各自的特点和适用的领域。

C语言是一种通用的程序设计语言，由于它很适合用来编写编译器和操作系统，因此被称为“系统编程语言”，但它同样适用于编写不同领域中的大多数程序。

### 1. C语言的产生

C语言是一种被广泛应用的计算机高级程序设计语言，是在B语言的基础上发展起来的，它经历了不同的发展阶段。

早期的系统软件设计均采用汇编语言，例如，大家熟知的UNIX操作系统。尽管汇编语言在可移植性、可维护性和描述问题的效率等方面远远不及高级程序设计语言，但是一般的高级语言有时难以实现汇编语言的某些功能。

那么，能否设计出一种集汇编语言与高级语言的优点于一身的语言呢？这种思路促成了UNIX系统的开发者——美国贝尔实验室的Ken Thompson——于1970年设计出了既简单又便于硬件操作的B语言，并用B语言写了第一个UNIX操作系统，这个操作系统先在PDP-7上实现，1971年又在PDP-11/20上实现。

B语言的前身是BCPL (Basic Combined Programming Language)，它是英国剑桥大学的Martin Richards在1967年基于CPL语言设计的，而CPL语言又是在1963年基于ALGOL 60产生的。

1972~1973年，贝尔实验室的D. M. Ritchie在B语言的基础上设计出C语言，该语言弥补了B语言过于简单、功能有限的不足。

1973年，Ken Thompson和D. M. Ritchie合作将90%以上的UNIX代码用C改写。随着改写UNIX操作系统的成功，C语言也逐渐被人们接受。

1987年以后，C语言已先后被移植到大、中、小、微型机上，并独立于UNIX和PDP，从而得到了广泛应用。

### 2. C语言的发展和应用

1978年，B. M. Kernighan 和D. M. Ritchie合写了一本经典著作——《C程序设计语言》(The C Programming Language，中文版、影印版已由机械工业出版社引进出版)，它奠定了C语言的基础，被称为标准C。

1983年，美国国家标准学会 (ANSI) 根据C语言问世以来的各种版本对C的发展和扩充，制订了新的标准，称为ANSI C。1987年又公布了新标准，称为87 ANSI C。目前流行的多种版

本的C语言编译系统都是以此为基础的。例如，在微机上广泛使用的Microsoft C、Turbo C、Quick C等。

### 3. C语言和C++语言交融发展

由于C语言是面向过程的结构化和模块化的程序设计语言，当处理的问题比较复杂、规模庞大时，就显现出一些不足，由此面向对象的程序设计语言C++应运而生。C++的基础是C，保留了C的所有优点，增加了面向对象机制，并且与之完全兼容。绝大多数C语言程序可以不经修改直接在C++环境中运行。

## 1.2 C语言的特点

C语言产生的历史并不长，它之所以后来居上，成为深受欢迎的高级语言之一，是因为和其他早期的高级语言相比，它具有以下优点：

1) 兼具高级、低级语言的双重能力。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，所以又被称为中级语言。

2) 生成的目标代码质量好，程序执行效率高。C语言具有汇编语言的许多特性，一般只比汇编程序生成的目标代码效率低10%~20%，可以开发出执行速度很快的程序。

3) 语言简洁，结构清晰。C程序通常是由若干函数组成的，强大的函数功能为程序的模块化和结构化提供了保证，因此程序简洁清晰，可读性强。

4) 语言表达能力强。C语言运算符丰富，例如，在C语言中，把括号、赋值、强制类型转换等都作为运算符处理。C语言具有现代化语言的各种数据结构，如整型、字符型、数组型、指针型、结构体和共用体等，而且具有结构化的控制语句。

5) 程序通用性、可移植性好。C语言没有依赖于硬件的输入/输出语句，而采用系统的库函数进行输入/输出操作，因此C语言不依赖于任何硬件系统，这种特性使得用C语言编写的程序很容易移植到其他环境中。

当然，C语言也有自身的不足，和其他高级语言相比，其语法限制不太严格，例如，对变量的类型约束不严格，影响程序的安全性，对数组下标越界不作检查等。从应用的角度来看，C语言比其他高级语言较难掌握。

总之，C语言既具有高级语言的特点，又具有汇编语言的特点；既是一个成功的系统设计语言，又是一个实用的程序设计语言；既能用来编写不依赖计算机硬件的应用程序，又能用来编写各种系统程序。它是一种深受欢迎、应用广泛的程序设计语言。

## 1.3 简单C语言程序举例

在这一节中，我们通过两个简单的C语言程序例子来介绍C语言的程序结构，并对C语言的基本语法成分进行相应的说明，以便使读者对C语言程序有一个大致了解。

### 【例1-1】 计算矩形的面积。

```
#include "stdio.h"          /*1:编译预处理*/
main()                     /*2:主函数*/
{                           /*3:函数体开始*/
    float h, w, area;      /*4:声明部分，定义变量*/
    h=10.5;                /*5:以下4条C语句为执行部分，给变量h和w赋值*/
    w=20.5;
    area=h*w;              /*7:计算矩形的面积*/
```

```
    printf(" area=%6.2f", area); /*8:输出area的值*/  
}
```

运行结果:

```
area=215.25
```

每行中的 /\*……\*/表示程序的注释内容。

第1行: 是一个编译预处理, 在程序编译前执行, 指示编译程序如何对源程序进行处理。它以“#”开头, 结尾不加分号, 以示和C语句的区别。

第2行: main表示主函数, 每一个C程序都必须有一个主函数。

函数体由第3行和第9行的一对花括号括起来。

第4行: 是变量声明部分, 定义变量h、w和area为实型变量。

第5、6行: 是两条赋值语句, 给变量h赋值10.5, w赋值20.5。

第7行: 将算术表达式h\*w的值赋予变量area。

第8行: 调用函数printf输出矩形面积值。

**【例1-2】** 计算两个矩形的面积之和。

```
#include "stdio.h" /*1:编译预处理*/  
float area(float h, float w) /*2:定义函数area*/  
{  
    float s;  
    s=h*w;  
    return s; /*6:返回s的值, return是关键字*/  
}  
main() /*8:主函数*/  
{  
    float h1, h2, w1, w2, s1, s2; /*10:声明部分, 定义变量*/  
    h1=10.5; w1=20.5;  
    h2=1.5*h1; w2=1.5*w1; /*12:计算变量h2, w2的值*/  
    s1=area(h1, w1); /*13:调用area函数, 将得到的返回值赋给变量s1*/  
    s2=area(h2, w2);  
    printf("area=%6.2f ", s1+s2); /*15:输出两个矩形的面积之和*/  
}
```

运行结果:

```
area=699.56
```

本程序包括主函数、被调用函数 area 和一个编译预处理指令。

第2行: 从该行开始到第6行定义函数area, 包括函数类型、函数名和函数体等部分。

第13、14行: 调用函数area, 将两次调用的返回值分别赋给变量s1和s2。

第15行: 计算并输出两个矩形的面积之和。

上述两个函数构成了一个完整的程序, 称为源程序。它以文件的方式存在, 文件中包含函数的源程序代码。可以把这两个函数放在一个文件中, 当程序语句多的时候也可以分别以函数为单位放在两个以上的文件中, C语言规定保存C源程序文件的扩展名为“.c”。

## 1.4 C语言程序的组成与结构

通过以上两个例子，我们对C语言程序的组成和结构有了初步和直观的了解，总结如下：

1) 一个C语言程序由一个或若干个函数构成。这些函数的代码以一个或若干个文件的形式保存。这些函数中必须有且只能有一个名为main的主函数。

2) 主函数main是程序的入口，它可以出现在程序的任何位置。一个C程序总是从主函数main开始执行。

3) C程序中的函数包括：主函数main，用户自定义函数（例如，例1-2中的area），系统提供的库函数（例如，输出函数printf）。

4) 函数由函数头和函数体两部分组成，函数头由函数类型的定义、函数名和参数表组成，函数体由声明部分（所使用变量和函数的说明）和若干执行语句组成。

5) 语句由关键字和表达式组成，每个语句和声明部分的结尾都必须加分号。复合语句的开头和结尾使用左、右花括号{ }。

关键字是C语言系统为了清晰地表达程序的功能而使用的一些英文单词或单词缩写。例如，例1-2中的return就是关键字。

用运算符将操作对象连接起来，符合C语言语法的式子称为表达式。表达式的组成元素有：变量、常量、函数调用、运算符。这些组成元素是以标识符和关键字的形式存在的。例如，例1-2中的 $h2=1.5*h1$ 和 $w1=20.5$ 都是表达式。

6) 程序中“/\* \*/”内的文字是程序的注释部分，是便于阅读理解程序的解释性附加文本，程序编译器完全忽略注释部分的内容。此外，在程序调试时，也可以将一部分代码转换为注释保留，而不必删除，以提高程序调试的效率。

从以上的分析可以发现，C程序的组织和构造与我们日常的文章的结构很类似，如表1-1所示。

表1-1 文章和C语言对应的层次结构

文章的层次结构	对应C程序的层次结构
文章	程序
章节	文件
段落	函数
开头第一段	主函数
句子	语句
词组	表达式
字	常量、变量、关键字、运算符等

在一般语言的学习过程中，我们是先学字、词组，然后造句，阅读范文，最后写作文。现在学习计算机语言，我们也同样遵循这个规律，即先学习常量、变量的类型和定义方法，然后依次学习表达式、语句和函数等，同时阅读一些程序范例，最后编写程序。当然二者也有本质上的区别，一般语言的学习以形象思维为主，而计算机语言的学习是以逻辑思维为主。C语言的程序结构如图1-1所示。

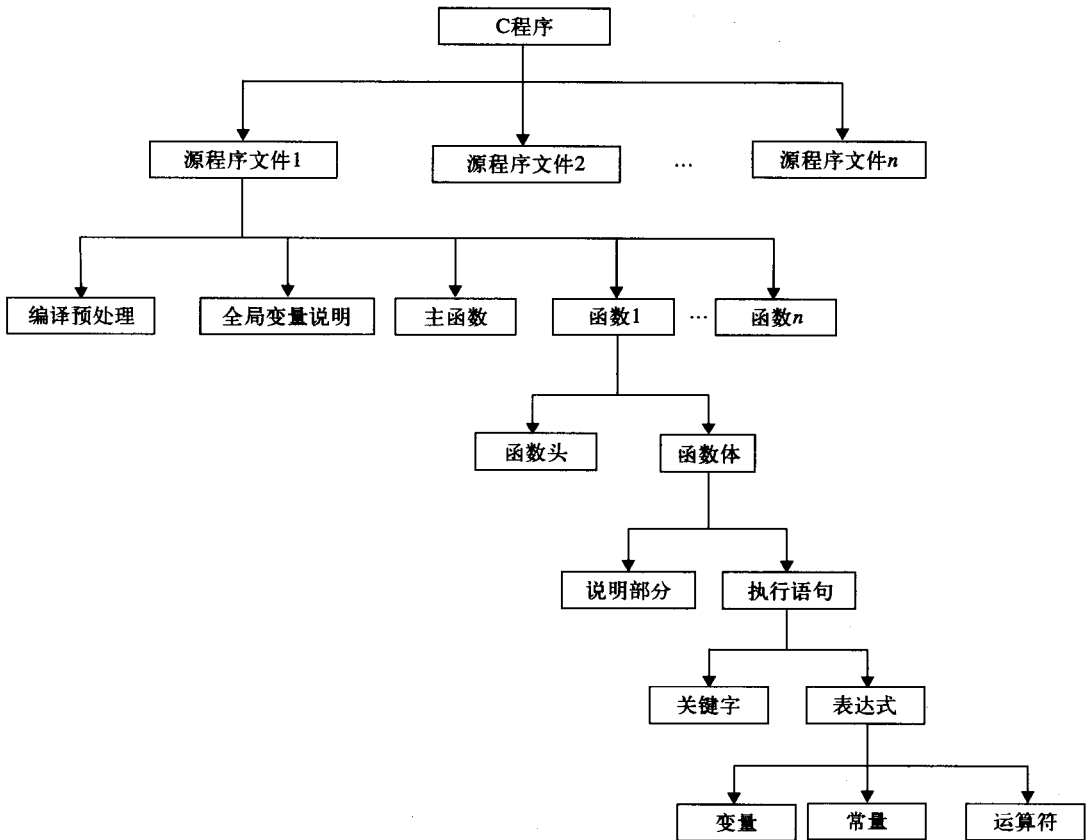


图1-1 C语言程序的层次结构图

## 1.5 C语言程序的开发步骤

一个C语言程序从最初编写到得到最终结果，大致经过以下几个步骤：

1) 编辑源程序。选择一种C语言开发工具，输入编写好的程序代码，称之为源程序，它以文件的方式存在，文件的扩展名为“.c”。

2) 编译源程序。为了使计算机能执行高级语言源程序，必须把源程序转换为二进制形式的目标程序，这个过程称为编译源程序。

编译是以源程序文件为单位分别进行的，每一个源程序文件对应生成一个目标文件，目标文件的扩展名为“.obj”。

编译过程中对源程序的全部内容进行检查，检查程序中关键字的拼写是否正确，根据程序的上下文检查语法是否有错，编译结束后系统显示所有的编译出错信息。

一般编译系统的出错信息有两种：一种是错误（error）信息，这类错误出现后，系统不生成目标文件，必须改正后重新编译；另一种是警告（warning）信息，是指一些不影响程序运行的不合理现象或轻微错误。例如，程序中定义了一个变量，却一直没有使用，这类警告信息出现后，系统可以生成目标文件。

3) 连接目标文件。编译结束，得到一个或多个目标文件，此时要用系统提供的“连接程序（linker）”将一个程序的所有目标文件和系统的库文件以及系统提供的其他信息连接起来，最

终形成一个可执行的二进制文件，可执行文件的扩展名为“.exe”。

4) 运行程序。运行最终形成的可执行文件，得到运行的结果。

5) 结果分析。分析程序的运行结果，如果发现结果不对，应检查程序或算法是否有问题，修改程序后再重复上面的步骤。其过程如图1-2所示。

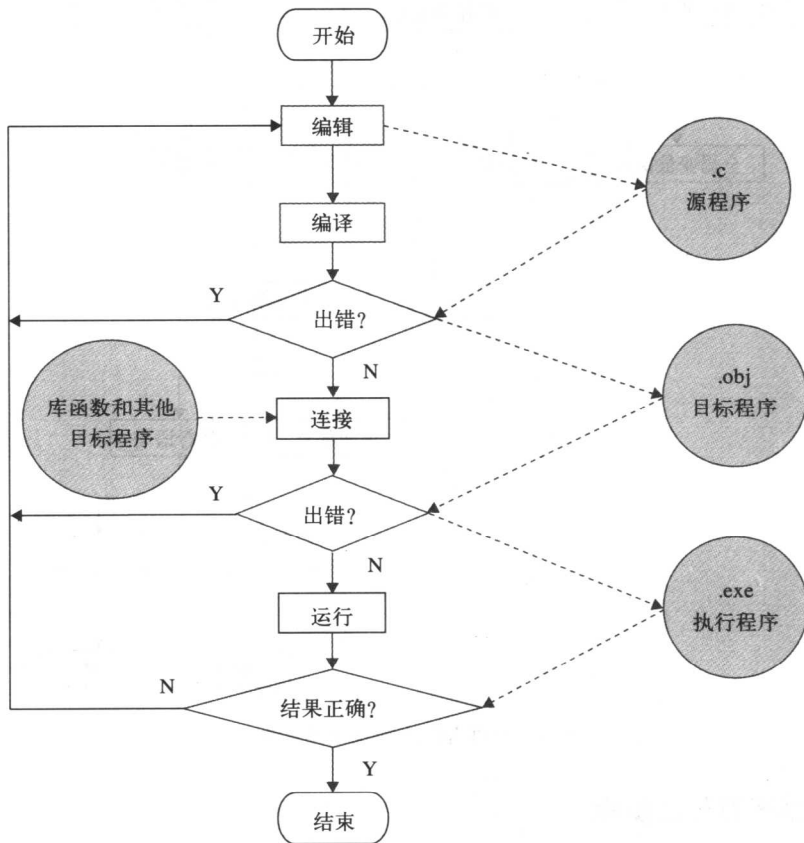


图1-2 C语言程序的开发步骤

## 小结

本章首先概述了程序设计的基本概念和C语言产生和发展的历史过程，然后与其他高级语言进行对比，列举了C语言的特点，再通过两个简单的C程序的实例，描述了C语言程序的基本组成和结构特点，最后介绍了C语言程序开发各个步骤的内容。

通过本章的学习，读者应该对程序设计的概念有初步的认识，对C语言总体结构和开发步骤有初步的了解。

## 习题

### 一、简答题

简要回答下列问题。

1. 程序的定义是什么？程序主要由几部分组成？
2. C语言的主要特点有哪些？

3. C语言程序是由哪些部分组成的, 各部分的作用是什么?

## 二、选择题

以下各题在给定的四个答案中选择一个正确答案。

1. 以下叙述正确的是( )。

- A. C语言允许直接访问物理地址, 可以直接对硬件进行操作
- B. C语言程序不用编译, 即可被计算机识别运行
- C. C语言不允许直接访问物理地址, 不可以直接对硬件进行操作
- D. C语言程序只需编译, 不需连接即可被计算机运行

2. 在一个C程序中( )。

- A. main函数出现在所有函数之前。C程序不一定都有main函数
- B. main函数可以在任何地方出现。一个C程序必须有且仅有一个main函数
- C. main函数必须出现在所有函数之后。一个C程序只能有一个main函数
- D. main函数出现在固定位置。一个C程序可以有多个main函数

## 三、填空题

1. C语言开发工具直接输入的程序代码是 A 文件, 经过编译后生成的是 B 文件, 经过连接后生成的是 C 文件。
2. C语言源文件的后缀是 A, 经过编译后生成的文件的后缀是 B, 经过连接后生成的文件的后缀是 C。

## 四、编写程序题

输入下面的程序, 上机调试并运行。

```
main()
{ float r, s;
  r=15.5;
  s=2*3.14*r;
  printf("r=%4.2f, s=%f", r, s);
}
```



## 第2章 数据类型、运算符与表达式

本章主要介绍C语言中的基本符号、各种数据类型及运算符和表达式。

作为一种程序设计语言，C语言规定了一套严密的语法规则和字符集，程序设计就是根据这些语法规则和基本字符按照实际问题的需要编制出相应的C语言程序。因此，程序设计中不能违反这些语法规则，也不能使用字符集以外的字符。

### 2.1 C语言的基本符号

任何一种高级语言，都有自己的基本词汇表。C语言的基本词汇表有下列几部分：

- 数字：10个 (0~9)。
- 英文字母：大、小写各26个 (A~Z, a~z)。
- 下划线字符 “\_”。
- 运算符：是指为表达程序基本操作使用的一些符号。
- 关键字：是指为表达程序功能使用的一些英文单词或单词缩写。

#### 2.1.1 标识符

程序中用来为符号常量、变量、函数、数组、类型、文件命名的有效字符序列称为标识符。标识符的命名规则如下：

- 只能由字母、数字和下划线组成。
- 第一个字符必须为字母或下划线。
- 不能使用关键字。
- 区分大小写字符，因此Sum、sum和SUM被认为是不同的标识符。
- ANSI C 没有规定标识符的长度，各个编译系统允许的标识符的长度（标识符中字符的个数）不同。例如，Turbo C规定最长为32个字符。建议标识符的长度不超过8个字符。

下面是合法的标识符：

area, Day, DATE, lesson\_1, s1, a23

下面是不合法的标识符：

Mr.ret, 2a, #sum, a>b, int

在选择标识符时，应该注意做到“见名知意”，即选择有含义的英文单词或缩写作为标识符，以增加程序的可读性，例如用标识符“length”表示长度变量，用“PI”表示圆周率的符号常量。

#### 2.1.2 常量

常量又称为常数，是在程序运行过程中其值不能被改变的量。常量的类型是由本身隐含决定的。例如，25、0、-13为整型常量，-55.23、0.57、3.8e3为实型常量，'B'、'9'为字符型常量。