教育部大学计算机课程改革项目规划教材

C程序设计精编

主审 孙淑霞
主编 肖阳春 魏琴 李思明

高等教育出版社

教育部大学计算机课程改革项目规划教材

C程序设计精编

主审 孙淑霞
主编 肖阳春 魏琴 李思明

高等教育出版社·北京

内容提要

本书作为C语言程序设计课程的教材，共由9章组成。其主要内容包括：C程序设计基础、分支结构、循环结构、函数、数组、指针、结构体、文件、综合案例。每章后面都有实验内容及指导，并配有一定数量的习题。全书以案例进行知识点的讲解，内容安排紧凑、简明扼要、由浅入深，尤其适用于学时少、内容不减的教学，一本教材涵盖了理论教学、实验教学和课后练习三方面的内容，为教与学提供了极大的方便。

本书可作为本科院校非计算机专业本科生、研究生的相关课程教材，也可作为计算机专业学生学习C语言程序设计的教材，同时还可作为自学者参考用书。

图书在版编目（CIP）数据

C程序设计精编/肖阳春，魏琴，李思明主编. --
北京：高等教育出版社，2019.9
ISBN 978-7-04-052120-7

Ⅰ. ①C… Ⅱ. ①肖… ②魏… ③李… 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（2019）第116151号

策划编辑 刘 娟	责任编辑 刘 娟	封面设计 李卫青	版式设计 马 云
插图绘制 于 博	责任校对 刘娟娟	责任印制 尤 静	

出版发行	高等教育出版社	网 址	http://www.hep.edu.cn
社 址	北京市西城区德外大街4号		http://www.hep.com.cn
邮政编码	100120	网上订购	http://www.hepmall.com.cn
印 刷	北京明月印务有限责任公司		http://www.hepmall.com
开 本	850mm×1168mm 1/16		http://www.hepmall.cn
印 张	13.75		
字 数	290千字	版 次	2019年9月第1版
购书热线	010-58581118	印 次	2019年9月第1次印刷
咨询电话	400-810-0598	定 价	33.30元

物料号 52120-00

C程序设计精编

孙淑霞

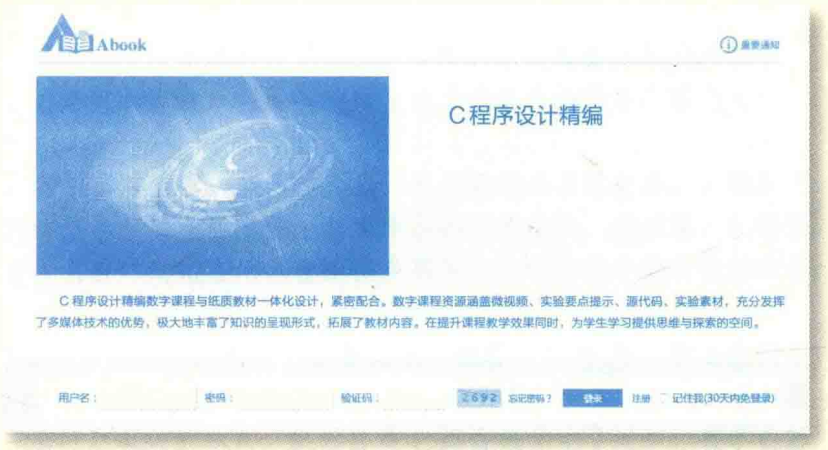
肖阳春 魏琴
李思明

1 计算机访问http://abook.hep.com.cn/186048，或手机扫描二维码、下载并安装Abook应用。

2 注册并登录，进入“我的课程”。

3 输入封底数字课程账号（20位密码，刮开涂层可见），或通过Abook应用扫描封底数字课程账号二维码，完成课程绑定。

4 单击“进入课程”按钮，开始本数字课程的学习。



课程绑定后一年为数字课程使用有效期。受硬件限制，部分内容无法在手机端显示，请按提示通过计算机访问学习。

如有使用问题，请发邮件至abook@hep.com.cn。

前言

程序设计课程一直以来都是高等院校重要的计算机基础课程之一，其重点是培养学生的计算思维能力，掌握程序设计的思想和方法，利用一种程序设计语言去编写程序解决实际问题，从而提高问题求解的能力。

C 语言由于其结构简单、数据类型丰富、表达能力强；既有高级语言的优点，又兼具低级语言直接操作计算机硬件的特点，使用灵活方便；其程序也具有速度快、效率高、代码紧凑、可移植性强等优点，一直以来被广泛地应用，也是很多高等院校作为程序设计课程的首选语言。

作者开设的“C/C++程序设计”课程先后被评为国家级精品课程（2008 年）和国家精品资源共享课（2014 年），经过多年的课程建设和教学实践，在教学内容、教学方法、教学手段和考试方法方面已经形成了一套行之有效的体系。针对目前 C 语言教学中学时少、教学内容多的情况，本教材以案例编写为主导思想，旨在通过具体案例教学达到举一反三的效果。

教材编写思路及其特点：

（1）每章精选若干案例进行知识点的讲解，强调编程思想，注重问题求解思维方式的培养，程序设计基本方法的引导。案例的选取不仅要考虑本章所涉及的知识，更重要的是突出程序设计思想的典型案例和有可进行举一反三的思考与练习题配套。

（2）不同于一般教材的编写思路，本书编写不再是大而全。本书内容简洁实用（适合课程学时少的教学需求），同时考虑到了知识点的覆盖面；本书采用新形态教材编写方式，读者使用手机扫描教材中的二维码即可观看到相关知识点的微视频。

（3）为了做到学以致用举一反三，每一章都提供了实验内容及指导，还提供了相应的习题，使教材更方便学生使用。

（4）每章每节的第一小节都给出一个案例，往后的各小节针对该案例中涉及的知识点及其算法进行讲解。

（5）容易出现的错误也分别在各章相应知识点处进行讲解。

（6）本书将理论教学与实验教学的内容合编一册，为教与学提供了极大便利。

本书由 9 章组成。每章的基本内容如下：

- 第 1 章 C 程序设计基础。介绍简单程序的编写。
- 第 2 章 分支结构。介绍分支结构程序的编写。

- 第 3 章 循环结构。介绍如何使用 3 种循环语句编写循环程序。
- 第 4 章 函数。讲解函数的定义和调用，局部变量和全局变量的使用。
- 第 5 章 数组。讲解如何用一维、二维数组进行批量数据处理的程序设计。
- 第 6 章 指针。讲解函数的定义、调用及参数的传递。
- 第 7 章 结构体。介绍结构体的定义、结构成员的引用以及单链表的程序设计。
- 第 8 章 文件。介绍文件操作的方法，数据文件的读/写。
- 第 9 章 综合案例。通过 3 个综合案例，达到进一步提升程序设计能力的目的。

（7）书中标注有☒的内容表示容易出现的错误；标注有*的内容表示选做项；标注有※的内容表示说明。

本教材在编写中努力做到概念清晰、通俗易懂、实用性强，力求激发读者的学习兴趣，使其能够通过案例举一反三真正掌握程序设计的思想和方法。

要想学好程序设计课程，需要教师和学生的共同努力。对于学习者来说，需要多动手、多实践、多思考。一分耕耘，一分收获，坚持耕耘定会有意想不到的收获。

本书第 1、6、9 章由肖阳春编写，第 2、5、7 章由魏琴编写，第 3、4、8 章由李思明编写。孙淑霞审阅了全书并对全书的编写提出了宝贵意见。

由于作者水平有限，书中难免存在不妥之处，请读者批评指正。

最后要感谢高等教育出版社刘茜编辑对本书编写给予的帮助和建议，感谢为本书提出宝贵意见的教师和读者，以及在本书撰写和出版过程中给予支持和帮助的人。

作　者

2019 年 9 月

目录

第 1 章　C 程序设计基础

1.1　简单 C 程序的编写

本节通过一个简单程序说明 C 程序的基本特征与语法。

1.1.1　案例

源代码 1-1：sumofnumber.cpp

【例 1-1】　编写一个计算并输出 $a+b$（其中 $a=5$，$b=10$）之值的程序。

例 1-1　　sumofnumber.cpp

```
1   #include<stdio.h>
2   int main(void)              /* 定义 main( )函数 */
3   {
4     int a=5, b=10, c;         /* 定义 3 个变量 a、b、c，并给变量 a 和 b 赋初值 */
5
6     c = a + b;                /* 计算 a+b */
7     printf ("The sum of two number\n");/* 输出“The sum of two number” */
8     printf ("a + b =%d", c);  /* 输出“a+b=15” */
9     return 0;
10  }
```

程序运行结果：

```
The sum of two number
a + b =15
```

程序及其知识点解析

（1）语句前的数字。例 1-1 每行前都添加了一个数字是为了便于讲解。本书部分程序前面加了一个代表行号的数字，实际编写程序时不能添加该数字，否则会报错。

（2）程序中的空行。一般可在说明性语句和可执行语句之间，或相对独立的功能模块之间插入一空行，便于阅读程序，例如例 1-1 中的第 5 行。

（3）main( )函数。每一个C程序都有且只有一个main( )函数，函数体由一对花括号{ }括起来，如例1-1的第3、10行。

（4）输出。C程序例1-1的第7、8行都是调用格式化输出函数printf( )进行数据输出，其讲解参见第1.1.4节。

（5）第1行的讲解参见第1.1.4节。

（6）注释。C的注释内容是括在/* */之中的。它可以放在任何位置，也可以跨多个行，但/和*之间不允许留有空格。

在C++环境中可以使用双斜线//注释，也称为行注释，即从//起到行的末尾都将被看作注释，通常用来说明程序段的功能、变量的作用等，使用非常灵活。//注释不能跨行，如果一行写不完注释内容，下一行需要继续使用//。

添加注释是为了增加程序的可读性和便于维护，在必要的位置加写注释是一个好习惯。在C++中可以用//或/* */注释。

（7）C语句。C语言的每一条语句都以分号“;”结束，如例1-1的第6～9行。

（8）C程序的书写。为了清晰地显示程序的结构，程序的书写应该采用缩进格式，一行只书写一条语句。例如，例1-1的第4行较之第3行略有缩进。

微视频1-1：C程序的组成

由例1-1这个简单的C程序可以得到C程序的组成具有如下特点。

① 一个C源程序由函数构成，其中有且只有一个主函数main( )。

② C程序总是由main( )函数开始执行，且结束于main( )函数。

③ 分号“;”是C语句的一部分，每一条语句均以分号结束。

④ C程序书写格式自由，一行内可写多条语句。

⑤ 在适当的位置添加注释有利于程序的理解和维护。

1.1.2 数据类型、常量与变量

在C程序中有常量和变量，常量是程序运行过程中不会改变的量，如例1-1中的数字5和10；变量是程序运行过程中可以改变的量，如例1-1中的a、b、c。程序中出现的变量必须先定义后使用，定义变量的一般格式为：

```
[存储类型] 数据类型 变量名1[, 变量名2, 变量名3,…, 变量名n];
```

其中：

（1）存储类型决定程序执行过程中变量占用内存的情况。例如，存储类型为static（静态类型）的变量，在程序执行过程中始终占用内存，直到结束执行。另外，系统会为static类型的变量赋初值0，例如：

```
static int x,y,z;                /* 系统为变量x，y，z赋初值0 */
```

变量存储类型除static外，大量使用auto（动态类型），关键字auto在使用时可以省略（C语言的关键字参见附录B）。动态变量在程序执行过程中只作用于一个函数或函数中的某个结构。

（2）数据类型决定其变量的存储空间大小、取值范围和允许进行的操作。编译系统为其定义的变量分配固定大小的内存单元。

（3）变量名的命名遵循标识符命名规则。为了和符号常量区别，变量名一般用

小写字母，并采用见名知意的方法命名。

同类型变量可定义在同行内，用逗号隔开，如例 1-1 第 4 行定义的 3 个变量 a、b、c：

```
int a, b, c;                    /* 定义整型变量 */
```

也可以定义在不同行中，定义在不同行中的等价形式为：

```
int a;
int b;
int c;
```

微视频 1-2：数据类型

程序中的数据以变量或常量的形式出现，每个变量或常量都有其数据类型。表 1.1 和表 1.2 分别列出了不同整型数和实型数在 C++中所占的字节数和取值范围等。

表 1.1 不同整型数在计算机中所占的字节数和取值范围

类 型	关 键 字	长度（字节）	取 值 范 围
基本整型	int	4	−2 147 483 648～2 147 483 647 $(-2^{31}\sim 2^{31}-1)$
长整型	long [int]	4	−2 147 483 648～2 147 483 647 $(-2^{31}\sim 2^{31}-1)$
短整型	short [int]	2	−32 768～32 767 $(-2^{15}\sim 2^{15}-1)$
无符号整型	unsigned [int]	4	0～4 294 967 295 $(0\sim 2^{32}-1)$
无符号长整型	unsigned long	4	0～4 294 967 295 $(0\sim 2^{32}-1)$
无符号短整型	unsigned short	2	0～65 535 $(0\sim 2^{16}-1)$

注：表 1.1 中的[*]是可省略的部分，编程时可以不写。

整型常量可以是正的或负的自然数，可以用八进制、十进制、十六进制数表示，其八进制数用 0 作为引导符，十六进制数用 0x 作为引导符，例如 0123、010、077 表示八进制数，0x123、0xAF、0xFF 表示十六进制数。在整型常量后面加上字母 l 或 L，表示该常量是 long 型的。加上字母 u 或 U，表示该常量是 unsigned 型的。

表 1.2 不同实型数在计算机中的字节数和取值范围

类 型	关 键 字	长度（字节）	有 效 位	取 值 范 围
单精度	float	4	7	$-3.4*10^{-38}\sim 3.4*10^{38}$
双精度	double	8	15	$-1.7*10^{-308}\sim 1.7*10^{308}$
长双精度	long double	8	15	$-1.7*10^{-308}\sim 1.7*10^{308}$

C 语言除了整、实型常量外，还有字符型常量又称为字符常量，它是用一对单引号' '括起来的一个字符。例如，'3' '0' 'a' '?' '*' 'A'等。一个字符常量用 1 个字节的内存单元存储。计算机存储的并不是字符本身，而是该字符的 ASCII 码（参见附录 A）。例如，字符'a'的 ASCII 码值为 97，转换成二进制数为 01100001，因此把字符'a'存放到内存中，实际上是把 01100001 存放在内存中。

C 语言中除用一对单引号括起来的普通字符外，还可以使用以“\”作为引导符的特殊字符常量，用于表示 ASCII 码中不可打印的控制字符和特定功能的字符，这种字符称为转义字符，如例 1-1 中第 7 行中的\n 代表换行。

使用字符常量时应当注意：

① 大小写英文字符代表不同的字符，如'A'不同于'a'。

② 空格也是一个字符，如' '。

③ 字符常量的单引号中只能有一个字符，也不能用双引号（" "）。例如，'ab'、"a"、"ab"都不是正确的字符常量。

转义字符常用在格式输出函数 printf( )中，起控制输出格式的作用。常用转义字符及其含义见表 1.3。

表 1.3 常用转义字符及其作用

字符形式	含义	ASCII 代码
\0	输出空值，无实际意义，表示一个字符串结束	0
\n	换行，将光标移到下一行的开始位置	10
\t	光标横向移动一个 Tab 键位（一般为 8 列）	9
\b	光标向前移动一列（一个字符）	8
\r	光标移到本行的开头	13
\f	光标移到下一页的开头	12
\\	输出反斜线字符“\”	92
\'	输出单引号字符“'”	39
\"	输出双引号字符“"”	34
\ddd	输出 1 到 3 位八进制数代表的字符	
\xhh	输出 1 到 2 位十六进制数代表的字符	

📖 提示

（1）转义字符用在 printf( )函数中，一般不在 scanf( )函数中使用，否则可能导致输入错误。

（2）转义字符代表一个字符。

（3）反斜线“\”后的八进制数可以不用 0 开头。如'\101'代表字符常量'A'，'\141'代表字符常量'a'。即在一对单引号内，可以用反斜线跟一个八进制数来表示一个字符常量。

（4）反斜线“\”后的十六进制数只能以小写字母 x 开头，不允许用大写字母 X 或 0x 开头。如'\x41'代表字符常量'A'，'\x61'代表字符常量'a'。也可以在一对单引号内，用反斜线跟一个十六进制数来表示一个字符常量。

1.1.3 运算符、表达式和语句

微视频 1-3：运算符

C 语言提供了丰富的运算符，可以满足不同类型数据的运算。通过运算符将变量、常量等连接起来形成表达式，用于解决各种简单或复杂的问题。

例 1-1 中的第 6 行用到了加法运算符“+”和赋值运算符“=”，C 语言提供了如表 1.4 所示的 13 类，约 50 个运算符。

表 1.4 C 语言的运算符

序号	种类	运算符
1	算术运算符	+ - * / %
2	赋值运算符	=及其扩展（复合）赋值运算符 ++ -- += *=等

续表

序　号	种　类	运　算　符
3	关系运算符	> < == >= <= !=
4	逻辑运算符	! && |
5	位运算符	<< >> ~ | ^ &
6	条件运算符	? :
7	逗号运算符	,
8	指针运算符	* &
9	求字节运算符	sizeof（类型）
10	强制类型转换运算符	（类型）
11	分量运算符	. ->
12	下标运算符	[ ]
13	其他	如函数调用运算符( )

一个运算符能连接的对象（包括常量、变量、函数等）个数称为“目”。运算符按目分为3类：

（1）单目运算符：即只能连接一个操作对象的运算符。如：++、--、！、&等。

（2）双目运算符：必须连接两个操作对象的运算符。如+、-、*、/、=、>、>=、!=、+=等。

（3）三目运算符：连接3个操作对象的运算符。C语言只有一个三目运算符，即条件运算符“ ？ :”。

表达式是由运算符和运算对象按C语言语法规则且具有实际意义的式子组成。根据运算符的不同，可以构成算术表达式、关系表达式、逻辑表达式、赋值表达式等。当表达式中出现多个运算符时，系统会按运算符的优先级（运算符执行的先后顺序）进行运算。当运算符具有相同优先级时，则运算顺序由结合性（从左向右或从右向左运算）决定。绝大部分运算符都具有左结合性，即从左向右计算。运算符的优先级和结合性可参见附录C。

算术运算符、关系运算符、逻辑运算符与其他运算符的优先级关系为：

```
!→ 算术运算符→关系运算符→&&→|→赋值运算符（=）
高 ——————————————————————————→ 低
```

算术运算符的运算级别是：+、-运算同级别，*、/、%运算同级别，后3种优先级高于前两种。使用运算符“/”时，当两个运算对象都是整数时，其运算结果是去掉小数点后面的数，不采用四舍五入。例如，2/4的结果为0。%只能连接两个整数，其结果的符号由左边整数的符号决定，与右边整数的符号无关。例如，8%-3的结果是2，-8%3的结果是-2。算术运算符的运算方向都是从左向右。

☒ 使用求余运算时，忽略了变量的类型，进行了不合法的运算。例如：

```
float a, b;
printf("%d", a%b);
                /* 求余运算符%的操作数只能是整型数，而a和b是浮点型变量 */
```

当一个表达式中具有不同数据类型的操作数时，编译系统会按照如图1.1所示

的规则自动转换其数据类型，即精度较低的转换为精度较高的类型。例如：

```
int  x;
x= 'A'/10.0+10*1.5+'A'+1.53;
```

其中，'A'/10.0→65.000000/10.0→6.500000；10*1.5→10.000000 *1.5→15.000000；'A'→65.000000，整个表达式的计算结果为88.029999，因此最后赋给整型变量 x 的值是88。

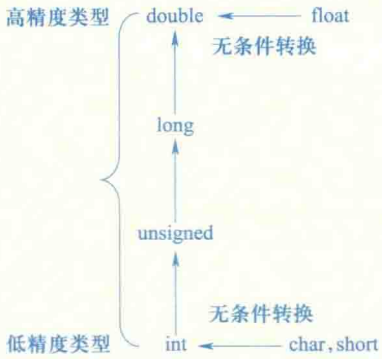


图 1.1 不同数据类型转换规则

关系运算符按“从左到右”的方向进行运算，其优先级为：

<	小于	优先级相同（高）
<=	小于或等于	
>	大于	
>=	大于或等于	
==	等于	优先级相同（低）
!=	不等于	

※ 在关系运算符中，==、>=、<=、!= 运算符之间不能出现空格，不能写成= =、> =、< =、！ =；当两个符号不同时，不能写反位置，例如，不能把“>=”写成“=>”。

逻辑运算符的优先级为：!>&&>||。逻辑“与”和逻辑“或”运算符的运算方向都是“自左向右”，而逻辑非运算符的运算方向是“自右向左”。

1.1.4 格式化输出函数 printf( )

例1-1的第7、8行用到了C语言提供的格式化输出函数printf( )进行数据的输出。由于printf( )是C标准库中提供的函数并在系统文件stdio.h中声明，所以在使用该函数的源程序文件开始位置处要添加如下编译预处理命令：

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

※ #include是预处理指令，而不是C语言的语句，所以在其后面不能加分号。

1. printf( )函数的一般调用格式

printf( )函数的作用是把数据按指定格式输出到屏幕上，其一般调用格式为：

```
printf(格式控制字符串[, 变量列表])
```

例如：

printf("I=%d, F=%f, C=%c\n", i, f, c)

格式控制字符串　　输出列表

其中：

（1）格式控制字符串：用于控制输出数据格式，必须以双引号（""）引导，内容由一个或多个格式控制字符（如表 1.5 所示）组合而成，也可以含有非格式控制字符，如例 1-1 的第 8 行。非格式控制字符称为普通字符，输出时按原样输出在相应位置上。

（2）输出列表：用于指定输出对象，如变量名，表达式等。输出多个对象时，各对象之间用逗号隔开。输出列表是可选项，如果没选，则 printf( )函数的格式控制字符串就完全由非格式控制字符或转义字符组成，执行结果将输出这些非格式控制字符，如例 1-1 的第 7 行输出了非格式控制字符：The sum of two number。

表 1.5　printf( )函数中常用的格式控制字符

数据类型	格式字符	格式控制字符	含　义
整型	d 或 i	%d 或%i	输出带符号的十进制整数
	u	%u	输出无符号的十进制整数
	o	%o	输出八进制无符号整数
	x 或 X	%x 或%X	输出十六进制无符号整数（大小写作用相同）
字符	c	%c	输出一个字符
字符串	s	%s	输出一个字符串
实型	f	%f	以小数形式或指数输出实数
	e 或 E, g 或 G	%e 或%E %g 或%G	与 f 格式作用相同，e 与 f, g 可以相互替换（大小写作用相同）

2．格式控制字符串的使用

printf( )函数中的格式控制字符串用于控制输出，为了使输出数据排列美观合理，常常在格式引导符（%）和格式符（如 d、f、c、s、e）之间插入一些附加的格式修饰符，如 m、l、m.n 等。

（1）%d 是按整型数据的实际长度输出，而%md 则按 m 个英文字母宽度输出。如果数据的位数小于 m，则输出数据左端补空格；若大于 m，则按实际宽度输出。例如：

```
printf("A=%4d, B=%3d", 123, 1234);
```

则输出结果为：

```
A= ␣123, B=12345
```

（2）%f 的输出宽度由系统自动确定，输出实数中的全部整数和 6 位小数。单精度浮点数有效位数一般为 7 位，双精度浮点数有效位数为 15 位。这里的 7 位或 15 位包括整数位和小数位之和，不是有效小数位。例如下面的程序段：

```
float a=123456.123, b=654321.321;
double c=55444333222111.1122, d=11222333444555.4733;
```

```
printf("%f\n", a+b);
printf("%f\n", c+d);
```

输出结果为：

```
777777.437500              （只有前7位数据有效）
66666666666666.578000      （只有前15位数据有效）
```

（3）%m.nf 则指定输出数据的宽度占 m 位（包含小数点本身），其中小数占 n 位，多于 n 位的小数部分，最高位四舍五入输出。如果数值长度小于 m，则左端补空格；如果数值长度大于 m，则整数部分原样输出，小数占 n 位。“－”表示左对齐，否则右对齐。例如下面的程序段：

```
float a=123.456, b=12.4567, c=1234.123, d=1.1;
printf("a=%8.2f\nb=%8.2f\nc=%8.2f\nd=%-8.2f\n", a, b, c, d);
```

输出结果如下：

```
a=␣␣123.46
b=␣␣␣12.46
c=␣1234.12
d=1.10
```

1.1.5 C程序的编写与执行

C 程序从编写到执行要经过 5 个阶段：编辑、预处理、编译、连接、运行，其中运行阶段可以对程序进行跟踪调试。可以选择在 Turbo C 或者 Visual C++（本教材采用 Visual C++2010 Express）集成环境中完成。

1. 编辑

用 C 语言编写的程序文件叫源程序文件，其文件的扩展名可以为“C”或“CPP”，在 C++中的默认扩展名为 CPP。无论新编写一个程序，还是修改一个原有的程序，在编辑器中输入或修改 C 程序的过程都称为编辑。

2. 预处理

预处理是指在编译之前，C 预处理程序执行 C 程序中的专门命令，即预处理指令。例如，例 1-1 第 1 行的 include 命令就是将其后面的 stdio.h 文件的内容插入到当前文件的当前位置。

3. 编译

编译是指用 C 语言提供的编译器将编辑好的源程序翻译成二进制形式的目标代码文件的过程。目标代码文件的扩展名为“obj”，又称为 OBJ 文件。

在编译过程中，编译器将检查源程序每一条语句的词法和语法错误。

编译错误分为两种性质：Error（致命）错误和 Warning（警告）错误。

- Error 错误将终止程序继续编译，不会生成 OBJ 文件，必须修改程序重新编译。
- Warning 错误是编译程序不能百分之百确定的错误，即源程序在这里可能有错。如果程序中只有 Warning 错误，则可以连接生成可执行程序。警告错误有两种，一种不会影响程序运行结果，如定义了多余的变量；另一种则会影响程序运行结果，这时需要分析具体情况，找到并修改错误。

值得注意的是，编译时，当信息窗口中列出了很多行的错误信息时，并不表示