



高等职业教育计算机类专业“十二五”规划教材

C语言程序设计 项目教程

主编 陈翠红 黄玲

CDIO教育理念

· 面向就业
· 能力培养
· 项目导向
· 任务驱动



国防工业出版社

National Defense Industry Press

高等职业教育计算机类专业“十二五”规划教材

C 语言程序设计项目教程

主 编 陈翠红 黄 玲

副主编 陆金江 蔡传军 庄 彦

参 编 芮素文 胡 煜 吴 俊

国防工业出版社

·北京·

内 容 简 介

本书详细介绍了 C 语言的基础知识。全书共分为 12 章,主要内容包括 C 语言概述,数据类型、运算符和表达式,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,结构体与共用体,位运算,文件等相关知识。全书以项目 + 案例的模式编写。以“学生成绩管理系统”项目贯穿始终,将该项目的子目标分解到各个章节,并在最后一章进行整合。本着理论以“够用为主”的原则,在每章中,将关联案例提炼到章节开始处,然后再给出相关知识链接。

本书内容丰富,重点突出,实用性较强,适合作为高职高专院校计算机相关专业教材,也可作为其他非专业本科生自学 C 语言教材。

图书在版编目(CIP)数据

C 语言程序设计项目教程/陈翠红,黄玲主编. —北京:国防工业出版社,2012.7

高等职业教育计算机类专业“十二五”规划教材

ISBN 978-7-118-08112-1

I. ①C... II. ①陈... ②黄... III. ①C 语言 - 程序设计 - 高等职业教育 - 教材 IV. ①TP312

中国版本图书馆 CIP 数据核字(2012)第 108642 号

※

国防工业出版社出版发行

(北京市海淀区紫竹院南路 23 号 邮政编码 100048)

北京奥鑫印刷厂印刷

新华书店经售

*

开本 787 × 1092 1/16 印张 17 $\frac{3}{4}$ 字数 435 千字

2012 年 7 月第 1 版第 1 次印刷 印数 1—4000 册 定价 35.00 元

(本书如有印装错误,我社负责调换)

国防书店:(010)88540777

发行邮购:(010)88540776

发行传真:(010)88540755

发行业务:(010)88540717

前 言

在众多的程序设计语言中,C 语言以其灵活性和实用性受到了广大计算机应用人员的喜爱。C 语言是既得到美国国家标准化协会(ANSI)标准化,又得到工业界广泛支持的计算机语言之一,几乎任何一种机型、任何一种操作系统都支持 C 语言开发;C 语言在巩固其原有应用领域的同时,又在拓展新的应用领域,支持大型数据库开发和 Internet 应用。一旦掌握了 C 语言,就可以较为轻松地学习其他任何一种程序设计语言,为后续的面向对象程序设计 Java 程序设计、C#程序设计等程序设计语言,以及数据结构、操作系统、数据库编程、网站开发、单片机编程等课程的学习打下基础。

目前关于 C 语言方面的书籍和教程很多,但是适合高职教育的教材大多采用传统的编写方式,教师教学及学生自学普遍认为枯燥乏味。本书总结众多从事 C 语言教学的一线教师的教学经验,推出项目+案例教学的模式编写。

全书共分为 12 章,主要内容包括 C 语言概述,数据类型、运算符和表达式,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,结构体与共用体,位运算,文件等相关知识。全书以“学生成绩管理系统”项目贯穿始终,将该项目的子目标分解到各个章节,并在最后一章进行整合。本着理论以“够用为主”的原则,在每章中,将关联案例提炼到章节开始处,而后再给出相关知识链接。内容丰富,重点突出,实用性较强。

本书由陈翠红、黄玲任主编,负责全书的统稿、修改和定稿工作;由陆金江、蔡传军、庄彦任副主编;由芮素文参编。其中第 1、4 章由芮素文(安徽国防科技职业学院)编写,第 2、6 章由陆金江(安徽财贸职业学院)编写,第 3 章由庄彦(安徽工商职业学院)编写,第 5、7 章由陈翠红(安徽工商职业学院)编写,第 8、10 由黄玲(安徽工商职业学院)编写,第 9、11 章由蔡传军(安徽医学高等专科学校)编写,第 12 章由陈翠红、黄玲共同编写;另外,参加本书编写及检验等工作的还有胡煜(安徽国际商务职业学院),吴俊(安徽医学高等专科学校)等老师。

本书所有案例都已在真实环境中调试通过,读者可索取课件,电子邮箱:44189762@qq.com 或 cch_ahbvc0801@sina.com。也恳请广大读者对书中的错误及不足之处给予指正。

编者

目 录

[第一部分 项目提出]

第 1 章 C 语言概述	2
【学习目标】	2
【项目介绍】	2
1.1 C 语言的发展历史及特点	2
1.1.1 C 语言的发展历史	3
1.1.2 C 语言的特点	4
1.2 C 程序的运行	5
1.2.1 C 程序的结构	5
1.2.2 C 程序的运行步骤	6
1.2.3 Turbo C 简介	7
1.2.4 VC + +6.0 简介	9
【常见问题集锦】	11
【实验与实训】	11

[第二部分 项目分解]

第 2 章 数据类型、运算符和表达式	14
【学习目标】	14
【项目子目标】	14
【关联案例】	14
【知识链接】	18
2.1 数据类型	18
2.1.1 整型	18
2.1.2 浮点型	19
2.1.3 字符型	20
2.2 常量和变量	20
2.2.1 常量	20
2.2.2 变量	23

2.2.3 符号常量	25
2.3 运算符和表达式	25
2.3.1 算术运算符及算术表达式	25
2.3.2 赋值运算符及赋值表达式	27
2.3.3 逗号运算符及逗号表达式	28
2.4 数据类型转换	28
【常见问题集锦】	30
【实验与实训】	31
第3章 顺序结构程序设计	34
【学习目标】	34
【项目子目标】	34
【关联案例】	34
【知识链接】	37
3.1 C 语言语句	37
3.2 赋值语句	38
3.3 程序的三种基本结构	39
3.4 数据的输入/输出	40
3.4.1 数据的格式化输出函数 printf	40
3.4.2 数据的格式化输入函数 scanf	42
3.4.3 字符数据的输入/输出	43
【常见问题集锦】	43
【实验与实训】	44
第4章 选择结构程序设计	47
【学习目标】	47
【项目子目标】	47
【关联案例】	47
【知识链接】	51
4.1 关系运算符与关系表达式	51
4.1.1 关系运算符	51
4.1.2 关系表达式	51
4.2 逻辑运算符与逻辑表达式	52
4.2.1 逻辑运算符	52
4.2.2 逻辑表达式	52
4.3 if 语句	53

4.3.1	if 语句的三种形式	53
4.3.2	if 语句的嵌套	55
4.4	switch 语句	56
	【常见问题集锦】	57
	【实验与实训】	58
第 5 章	循环结构程序设计	62
	【学习目标】	62
	【项目子目标】	62
	【关联案例】	62
	【知识链接】	68
5.1	概 述	68
5.2	while 语句	69
5.3	do - while 语句	69
5.4	for 语句	71
5.5	循环的嵌套	73
5.6	break 语句和 continue 语句	73
5.6.1	break 语句	73
5.6.2	continue 语句	73
5.7	goto 语句	74
	【常见问题集锦】	74
	【实验与实训】	76
第 6 章	数 组	81
	【学习目标】	81
	【项目子目标】	81
	【关联案例】	81
	【知识链接】	93
6.1	一维数组	93
6.1.1	一维数组的声明	93
6.1.2	一维数组的初始化	94
6.1.3	一维数组元素的引用	94
6.2	二维数组	95
6.2.1	二维数组的声明	96
6.2.2	二维数组的初始化	96
6.2.3	二维数组元素的引用	97

6.3	字符数组与字符串	98
6.3.1	字符数组	98
6.3.2	字符串	99
6.3.3	字符串的输入和输出	100
6.3.4	字符串处理函数	101
	【常见问题集锦】	103
	【实验与实训】	104
第7章	函数	109
	【学习目标】	109
	【项目子目标】	109
	【关联案例】	109
	【知识链接】	119
7.1	概述	119
7.2	函数的定义、声明与调用	121
7.2.1	函数的定义	121
7.2.2	函数的调用	122
7.2.3	函数参数和函数返回值	124
7.3	函数的参数传递	125
7.3.1	值传递	125
7.3.2	地址传递	126
7.4	数组作为函数参数	126
7.4.1	数组元素作为函数参数	126
7.4.2	数组名作为函数参数	127
7.4.3	多维数组名作为函数参数	127
7.5	函数的嵌套调用和函数的递归调用	128
7.5.1	函数的嵌套调用	128
7.5.2	函数的递归调用	128
7.6	变量的作用域和存储类型	128
7.6.1	变量的作用域	129
7.6.2	变量的存储类别	130
7.7	内部函数和外部函数	134
7.8	编译预处理	134
7.8.1	宏定义	135
7.8.2	文件包含	136
7.8.3	条件编译	137

【常见问题集锦】	138
【实验与实训】	139
第8章 指针	147
【学习目标】	147
【项目子目标】	147
【关联案例】	147
【知识链接】	167
8.1 指针与指针变量	167
8.1.1 指针的概念	167
8.1.2 指针变量的定义与引用	169
8.1.3 指针变量作为函数参数	172
8.1.4 多级指针	173
8.2 指针与数组	174
8.2.1 指针与一维数组	174
8.2.2 指针与二维数组	177
8.2.3 指针数组	180
8.3 指针与字符串	183
8.3.1 指向字符串常量的指针变量	183
8.3.2 指向字符数组的指针变量	184
8.3.3 字符指针做函数参数	185
8.4 指针与函数	185
8.4.1 指向函数的指针	185
8.4.2 返回指针值的函数	186
8.4.3 带参数的 main 函数	187
【常见问题集锦】	187
【实验与实训】	189
第9章 结构体与共用体	195
【学习目标】	195
【项目子目标】	195
【关联案例】	195
【知识链接】	203
9.1 结构体	203
9.1.1 结构体类型的定义	203
9.1.2 结构体类型变量的定义	204

9.1.3	结构体类型变量的初始化	205
9.1.4	结构体类型变量的引用	206
9.1.5	结构体数组	206
9.1.6	结构体指针	207
9.1.7	用结构体变量和指向结构体的指针作为函数参数	208
9.1.8	结构体嵌套	208
9.2	共用体	208
9.2.1	共用体类型的定义	208
9.2.2	共用体类型变量的定义	209
9.2.3	共用体类型变量的引用	209
9.3	枚举类型	210
9.3.1	枚举型的定义	210
9.3.2	枚举型变量的定义	210
9.3.3	枚举型变量的引用	210
9.4	用户自定义类型	211
9.4.1	基本类型自定义	211
9.4.2	数组类型自定义	211
9.4.3	结构型自定义	211
9.4.4	指针型自定义	212
	【常见问题集锦】	212
	【实验与实训】	212
第 10 章	位运算	216
	【学习目标】	216
	【项目子目标】	216
	【关联案例】	216
	【知识链接】	220
10.1	原码、反码和补码	220
10.2	位运算符与位运算	220
10.2.1	按位与	221
10.2.2	按位或	221
10.2.3	按位异或	221
10.2.4	按位取反	222
10.2.5	左位移	222
10.2.6	右位移	222
	【常见问题集锦】	223
	【实验与实训】	223
第 11 章	文件	226
	【学习目标】	226

【项目子目标】	226
【关联案例】	226
【知识链接】	231
11.1 C 文件概述	231
11.1.1 文件的概念	231
11.1.2 文件的分类	232
11.1.3 文件类型指针	233
11.2 文件的打开与关闭	233
11.2.1 文件的打开操作	233
11.2.2 文件的关闭操作	235
11.3 文件的读写	235
11.3.1 字符读/写操作	235
11.3.2 字符串读/写操作	236
11.3.3 数据块读/写操作	236
11.3.4 格式化读/写操作	237
11.4 文件的定位	237
11.4.1 置文件位置指针于文件开头的 rewind 函数	237
11.4.2 改变文件位置指针位置的 fseek 函数	237
11.5 出错检测	238
11.5.1 ferror 函数	238
11.5.2 clearerr 函数	238
【常见问题集锦】	239
【实验与实训】	239

[第三部分 项目整合]

第 12 章 成绩管理系统设计与实现	242
12.1 数据结构设计	242
12.2 功能模块设计	242
12.3 main 函数的执行流程	243
12.4 程序实现	243
12.5 运行结果	262
附录 1 常用字符与 ASCII 代码对照表	263
附录 2 关键字及其用途	265
附录 3 运算符的优先级和结合性	266
附录 4 Turbo C2.0 常用库函数	267
参考文献	272

第一部分 项目提出

第1章 C语言概述

【学习目标】

1. 了解 C 语言的形成和发展的过程。
2. 了解程序设计语言的概念。
3. 理解 C 语言的特点。
4. 熟悉 C 语言程序的运行步骤。
5. 熟练 Turbo C 和 VC++6.0 开发环境。

【项目介绍】

学生成绩管理系统利用计算机对学生成绩进行管理，可以进一步提高学校办学效益和现代化水平，提高广大教师的工作效率，实现学生成绩管理 workflows 的系统化、规范化和自动化。本系统涉及程序设计各个方面的知识，通过该项目的实现使读者能对 C 语言有一个全面的认识，能够综合利用所学知识解决实际问题。

学生成绩管理系统包括信息的录入、显示、查找、插入、删除、修改、排序、统计、存盘等常规功能，如图 1-1 所示。

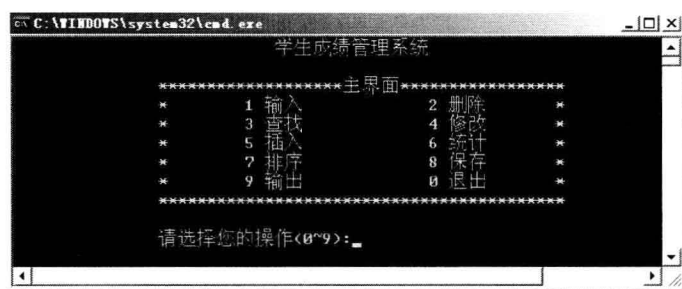


图 1-1 学生成绩管理系统功能菜单

本书将本项目的各个功能目标分解到第二部分的每个章节，最后在第三部分提供详细的设计思路和解决方案，读者可以通过对 C 语言知识的系统学习，最终能独立完成本项目的所有功能。

1.1 C语言的发展历史及特点

计算机语言是进行程序设计的工具，故而也称为程序设计语言，可分为机器语言、汇编语言和高级语言。C 语言是目前国际上广泛使用的一种计算机高级语言，适合编写系统软件，

也可以编写应用软件。

1.1.1 C 语言的发展历史

C 语言的发展过程与 UNIX 相辅相成，紧密地联系在一起。

早期的操作系统等系统软件主要是用汇编语言编写的，如 UNIX 操作系统。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比较差。为了提高可读性和可移植性，最好改用高级语言，但一般高级语言难以实现汇编语言可以直接对硬件进行操作的功能，如对内存地址的操作、位(bit)操作等。人们设想能否找到一种既具有一般高级语言特性，又具有低级语言特性的语言，集它们的优点于一身。于是，C 语言就在这种情况下应运而生了。

C 语言是在 B 语言的基础上发展起来的，它的根源可以追溯到 ALGOL 60。1960 年出现的 ALGOL 60 是一种面向问题的高级语言，它离硬件比较远，不宜用来编写系统程序。1963 年英国的剑桥大学推出了 CPL(Combined Programming Language)。CPL 在 ALGOL 60 的基础上接近硬件一些，但规模比较大，难以实现。1967 年英国剑桥大学的 Martin Richards 对 CPL 作了简化，推出了 BCPL(Basic Combined Programming Language)。1970 年美国贝尔实验室的 Ken Thompson 以 BCPL 为基础，又作了进一步简化，它使得 BCPL 能挤压在 8KB 内存中运行，这个很简单的而且很接近硬件的语言就是 B 语言(取 BCPL 的第一个字母)，并用它写了第一个 UNIX 操作系统，在 DEC PDP-7 上实现。1971 年在 PDP-11/20 上实现了 B 语言，并写了 UNIX 操作系统。但 B 语言过于简单，功能有限，并且和 BCPL 都是“无类型”的语言。

1972 年至 1973 年间，贝尔实验室的 D. M. Ritchie 在 B 语言的基础上设计出了 C 语言(取 BCPL 的第二个字母)。C 语言既保持了 BCPL 和 B 语言的优点(精练、接近硬件)，又克服了它们的缺点(过于简单、数据无类型等)。最初的 C 语言只是为描述和实现 UNIX 操作系统提供一种工具语言而设计的。1973 年，K. Thompson 和 D. M. Ritchie 两人合作把 UNIX 的 90% 以上用 C 语言改写，即 UNIX 第 5 版。原来的 UNIX 操作系统是 1969 年由美国的贝尔实验室的 K. Thompson 和 D. M. Ritchie 开发成功的，是用汇编语言写的，这样，UNIX 使分散的计算系统之间的大规模联网以及互联网成为可能。

C 语言发展过程如图 1-2 所示。

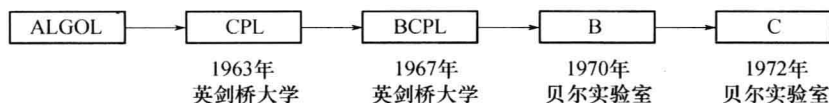


图 1-2 C 语言发展过程

后来，C 语言作了多次改进，但主要还是在贝尔实验室内部使用。直到 1975 年 UNIX 第 6 版公布后，C 语言的突出优点才引起人们普遍注意。1977 年出现了不依赖于具体机器的 C 语言编译文本《可移植 C 语言编译程序》，使 C 语言移植到其他机器时所需做的工作大大简化了，这也推动了 UNIX 操作系统迅速地在各种机器上实现。随着 UNIX 的日益广泛使用，C 语言也迅速得到推广。1978 年以后，C 语言已先后移植到大、中、小、微型机上，如 IBM System/370、Honeywell 6000 和 Interdata 8/32，已独立于 UNIX 和 PDP 了。以 1978 年由美国电话电报公司(AT&T)贝尔实验室正式发表的 UNIX 第 7 版中的 C 编译程序为基础，Brian W. Kernighan 和 Dennis M. Ritchie(里奇)合著了影响深远的名著《The C Programming Language》，常常称它为“K&R”，也有人称为“K&R 标准”或“白皮书”(white book)，它成

为后来广泛使用的 C 语言版本的基础。但在“K&R”中并没有定义一个完整的标准 C 语言。为此，1983 年，美国国家标准化协会(ANSI)X3J11 委员会根据 C 语言问世以来各种版本对 C 语言的发展和扩充，制定了新的标准，称为 ANSI C，ANSI C 比原来的标准 C 语言有了很大的发展，1987 年，ANSI 又公布了新标准——87 ANSI C。1988 年 K&R 修改了他们的经典著作《The C Programming Language》，按照 ANSI C 标准重新写了该书。现在 C 语言已风靡全世界，成为世界上应用最广泛的几种计算机语言之一，后来的 Java、C++、C#都是以 C 语言为基础发展起来的。

1.1.2 C 语言的特点

一种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 语言的主要特点如下：

(1) 语言简洁、紧凑，使用方便、灵活。C 语言一共只有 32 个关键字，9 种控制语句，程序书写形式自由，主要用小写字母表示，压缩了一切不必要的成分。

(2) 运算符丰富。C 语言的运算符包含的范围很广泛，共有 34 种运算符。C 语言把括号、赋值、强制类型转换等都作为运算符处理，从而使 C 语言的运算类型极其丰富，表达式类型多样化。灵活使用各种运算符可以实现在其他高级语言中难以实现的运算。

(3) 数据结构丰富，具有现代化语言的各种数据结构。C 语言的数据类型有整型、实型、字符型、数组类型、指针类型、结构体类型、共用体类型等。能用来实现各种复杂的数据结构(如链表、树、栈等)的运算。尤其是指针类型数据，使用起来比 PASCAL 更为灵活、多样。

(4) 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、for 语句)。用函数作为程序的模块单位，便于实现程序的模块化。C 语言是良好的结构化语言，符合现代编程风格的要求。

(5) 语法限制不太严格，程序设计自由度大。例如对数组下标越界不做检查，由程序编写者自己保证程序的正确。对变量的类型使用比较灵活，例如整型数据与字符型数据可以通用。一般的高级语言语法检查比较严，能检查出几乎所有的语法错误。而 C 语言允许程序编写者有较大的自由度，因此，放宽了语法检查。程序员应当仔细检查程序，保证其正确，而不要过分依赖 C 语言编译程序去查错。“限制”与“灵活”是一对矛盾。限制严格，就失去灵活性；而强调灵活，就必然放松限制。一个不熟练的编程人员，编一个正确的 C 语言程序可能会比编一个其他高级语言程序难一些。也就是说，对用 C 语言的人，要求对程序设计更熟练一些。

(6) C 语言能进行位(bit)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，又具有低级语言的许多功能，可用来写系统软件。C 语言的这种双重性，使它既是成功的系统描述语言，又是通用的程序设计语言。

有人把 C 语言称为“高级语言中的低级语言”或“中级语言”，意为兼有高级和低级语言的特点。一般仍习惯将 C 语言称为高级语言，因为 C 程序也要通过编译、连接才能得到可执行的目标程序，这是和其他高级语言相同的。

C 语言的以上特点，读者现在也许还不能深刻理解，待学完 C 语言以后再回顾一下，就会有比较深的体会。总之，C 语言对程序员要求较高。程序员使用 C 语言编写程序会感

到限制少，灵活性大，功能强，可以编写出任何类型的程序。学习和使用 C 语言的人已越来越多。

1.2 C 程序的运行

计算机只能执行机器语言，C 语言是一种高级程序设计语言，而任何高级语言源程序都要“翻译”成机器语言，才能在机器上运行。“翻译”的方式有两种：一种是解释方式，即对源程序解释一句执行一句；另一种是编译方式，即先把源程序“翻译”成目标程序(用机器代码组成的程序)，再经过连接装配后生成可执行文件，最后执行可执行文件而得到结果。

C 语言是一种编译型的程序设计语言，它采用编译的方式将源程序翻译成目标程序(机器代码)。运行一个 C 程序，从输入源程序开始，要经过编辑源程序文件(.c)、编译生成目标文件(.obj)、连接生成可执行文件(.exe)和执行四个步骤。

1.2.1 C 程序的结构

```
示例： /* 第一个 C 程序*/           /*注释*/
#include "stdio.h"                   /*编译预处理命令*/
void main(void)                       /*函数*/
{
    printf("Hello. \n");             /*语句*/
}
```

运行结果如图 1-3 所示。

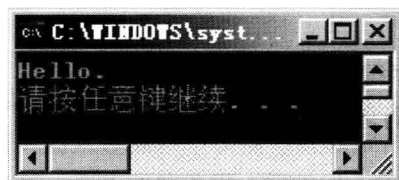


图 1-3 示例运行结果

从此示例可以看出，C 语言程序的结构大体可分为以下几个组成部分。

1. 函数

函数是程序的基本单位，是程序的主体，程序由一个或多个函数组成，但是每个 C 程序有且仅有一个主函数 `main()`。主函数的位置可以在程序的任意位置，它是程序的入口，程序执行总是从 `main` 开始，在 `main` 中结束，其它函数通过调用得以执行。

C 函数由函数名、形式参数和函数体三部分组成，其格式如下：

```
返回值类型 函数名 ([形式参数])
{
    函数体                               /*变量定义语句、实现函数功能的语句等*/
}
```

例如：

```
int Add(int x,int y)                   /*函数的首部*/
```



```

{
    int sum;
    sum=x+y;
    return(sum);
}

```

/* “{” 代表函数体开始*/
/*变量定义语句*/
/*具体实现函数功能的语句*/
/* “}” 代表函数体结束*/

2. 语句

语句是组成程序的基本单元，以分号(;)作为结束标志。C程序的书写格式较为自由，一条语句可以写在多行上，一行上也可以写若干条语句。C程序区分大小写，C语言的基本语句都是用小写字母表示的。

3. 编译预处理命令

每个以符号“#”开头的行，称为编译预处理行，是C语言提供的一种模块工具。

4. 注释

用/*……*/括起的内容，可对程序进行注释，其作用是给程序设计者一种提示或记号。注释使用时不能嵌套，且注释内容不参加程序的执行，主要作用是提高程序的可读性，故而不产生编译代码。

1.2.2 C程序的运行步骤

C语言程序的运行步骤包括以下过程。

(1) 上机输入或修改已有的源程序(.c文件)，称为编辑。

(2) 通过“编译程序”对源程序进行编译(包括预处理)，将其翻译成计算机能识别和执行的二进制指令形式的“目标程序”(obj文件)，称为编译。

(3) 将目标程序与系统函数库以及其他目标程序连接起来，形成可执行的目标程序(.exe文件)称为连接。

(4) 运行可执行的目标程序得到结果，称为运行。

对得到的运行结果需要进行验证，如果结果不正确，则需重新调试程序。C程序的运行步骤可用图1-4表示。

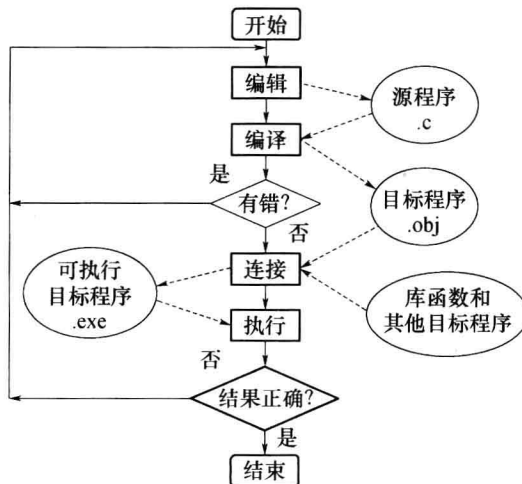


图 1-4 C语言程序运行步骤