

面向21世纪大学计算机基础教育系列规划教材

# C 语言程序设计

丁峻岭 主编 余坚 姜德森 副主编



中国铁道出版社  
CHINA RAILWAY PUBLISHING HOUSE

面向 21 世纪大学计算机基础教育系列规划教材

# C 语言程序设计

丁峻岭 主 编

余 坚 姜德森 副主编

**中国铁道出版社**  
CHINA RAILWAY PUBLISHING HOUSE

---

## 内 容 简 介

C 语言是目前流行的通用程序设计语言, 是许多计算机专业人员和计算机爱好者学习程序设计的首选语言。本书分为 12 章, 内容包括 C 语言概述, 数据类型, 运算符与表达式, 数据的输入和输出, 选择型程序设计, 循环控制, 数组, 函数, 编译预处理, 指针, 位运算, 结构体、共用体与枚举类型以及文件操作。

本书遵循结构化程序设计原则, 运用文字、插图等表述方法, 深入浅出地介绍了 C 语言的基础算法和程序设计方法, 具有良好的可读性和适用性。样例程序的编排上由浅入深, 强化知识点、算法、编程方法和技巧, 并给出了详细的解释, 更多地考虑到了初学者的需求。

本书可作为高校本、专科各专业 C 语言教材, 也可作为计算机等级考试二级考试教材和自学者的参考用书。

### 图书在版编目 (CIP) 数据

C 语言程序设计 / 丁峻岭主编. —北京: 中国铁道出版社, 2007.1

(面向 21 世纪大学计算机基础教育系列规划教材)

ISBN 978-7-113-07746-4

I. C… II. 丁… 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 019618 号

书 名: C 语言程序设计

作 者: 丁峻岭 余 坚 姜德森

出版发行: 中国铁道出版社 (100054, 北京市宣武区右安门西街 8 号)

策划编辑: 严晓舟 秦绪好

责任编辑: 苏 茜 杨 勇 姚文娟

封面设计: 付 巍

封面制作: 白 雪

责任校对: 刘彦会

印 刷: 北京鑫正大印刷有限公司

开 本: 787×1092 1/16 印张: 18.5 字数: 426 千

版 本: 2007 年 2 月第 1 版 2007 年 2 月第 1 次印刷

印 数: 1~5 000 册

书 号: ISBN 978-7-113-07746-4/TP·2111

定 价: 25.00 元

版权所有 侵权必究

本书封面贴有中国铁道出版社激光防伪标签, 无标签者不得销售

凡购买铁道版的图书, 如有缺页、倒页、脱页者, 请与本社计算机图书批销部调换。



## 面向 21 世纪大学计算机基础教育系列规划教材

编  
委  
会

主 任： 宁正元

副主任： 钱维民 陈 琼

委 员： (按姓氏字母先后为序)

陈常晖 陈振武 何保锋

黄朝辉 李文芳 卢 英

卢昌荆 王 敏 谢储晖

徐 敏 阎 格 杨 升

姚志强 张思民 郑义健

进入 21 世纪以来, 社会信息化进程不断加速, 信息化范围和各行各业信息化的程度都是空前的且不断地快速推进。高等学校非计算机专业的计算机基础教育也由计算机应用基础教育向着信息技术基础教育迈进, 逐步步入更加科学合理、更加符合 21 世纪人才素质需求的新阶段。

为了推动计算机基础教育的深入发展, 上一届“教育部高等学校非计算机专业计算机基础课程教学指导分委员会”曾在系统调研的基础上提出了三个文件, 即《关于进一步加强高等学校计算机基础教学的意见》、《计算机基础教学内容的知识结构与课程设置》和《高等学校非计算机专业计算机基础课程教学基本要求》, 提出了一门“大学计算机基础”加上几门核心课程的“1+x”的课程设置方案, 直接影响并推动着新一轮计算机基础课程的教学改革向着纵深的方向发展。本届“教育部高等学校非计算机专业计算机基础课程教学指导分委员会”秉承上了一届“教育部高等学校计算机专业计算机基础课程教学指导分委员会”的思路, 将陆续推出针对不同专业类别、不同层次院校的计算机基础课程教学的指导性文件; 许多地区、部门和高校也都积极地推行更深入的计算机基础课程教学改革尝试。

教材是体现教学改革、实现教学要求的重要保证。为了更好地促进计算机基础教育的改革, 中国铁道出版社在全国范围内邀请具有丰富经验的计算机基础教学一线教师, 在深入探讨和研究“教育部高等学校非计算机专业计算机基础课程教学指导分委员会”一系列文件的基础上, 组织编写“1+x”方案的系列教材, 争取在近几年内陆续出版。该套教材将力争具备以下特色:

(1) 体系完整、内容先进, 注重非计算机专业学生的特点, 适应新世纪对人才素质培养的要求。

(2) 体现课程内容的基础性和系统性, 基本内容准确明晰。

(3) 注重应用, 技术性、应用性内容的讲解采用任务驱动和案例教学模式, 适应教师指导下的学生自主学习的教学模式, 保证教学效果。

(4) 强调实践, 重视实验教材和辅导教材建设。

(5) 适应现代教学手段的采用, 除纸质教材外, 还配有多媒体电子教案, 重视教学网站和教学资源库的建设。

(6) 追踪课程改革进展, 适时更新版本, 及时反映计算机信息技术的新发展。

我们希望本套教材的陆续出版对深化计算机基础教学改革能有所助益; 并希望得到来自一线师生在使用过程中的宝贵意见和建议, 以便于教材不断改进和完善。

宁正元

2007 年 1 月

C 语言是目前流行的通用程序设计语言，之所以成为许多计算机专业人员和计算机爱好者学习程序设计的首选入门语言，除了 C 语言的众多优点外，最主要的还是 C 语言的实用性。

C 语言是程序设计的工具，因此学会使用 C 语言并不是学习的唯一目的，掌握计算机处理问题的思维方式和程序设计的基本方法，用以解决实际问题更为重要。因此，本书在详细阐述 C 语言基础知识的基础上，着重讨论了程序设计的基本原理、概念和方法，并通过实例来巩固所学的知识。

本书分为 12 章，内容包括 C 语言概述，数据类型，运算符与表达式，数据的输入和输出，选择型程序设计，循环控制，数组，函数，编译预处理，指针，位运算，结构体、共用体与枚举类型以及文件操作。

本书遵循结构化程序设计原则，运用文字、插图等表述方法，深入浅出地介绍了 C 语言的基础算法和程序设计方法，具有良好的可读性和适用性。样例程序的编排上由浅入深，注重强化各知识点及编程方法和技巧，通过详细地解释，更多地考虑到了初学者的需求。建议读者在学习 C 语言的过程中一定要多读程序，多动手编写程序，抓住“先模仿，在模仿的基础上改进，在改进的基础上提高”的学习规律。本书配套教材《C 语言程序设计习题解析与实验指导》给出了本书各章习题的详细解答，并以较大的篇幅介绍了 Turbo C 环境下的程序调试方法，以及实验安排与指导。强调动手实践，提高 C 语言程序设计的能力。

本书全部例题均由编者在 Turbo C++ 3.0 集成环境下调试通过，也可以在 Turbo C 2.0 系统中运行。在本书编写过程中，得到了福建省计算机基础教育研究会、福州大学阳光学院、华侨大学、泉州师范学院、福建工程学院和中国铁道出版社等单位的大力支持，在此表示衷心的感谢！

由于时间仓促及水平所限，书中不妥或错误之处在所难免，恳请读者批评指正。

编者

2006 年 11 月

第 1 章 C 语言概述.....	1
1.1 程序与程序设计.....	1
1.2 C 语言的特点.....	2
1.3 C 程序的基本结构.....	3
1.4 C 语言的词类.....	5
1.4.1 字符集.....	5
1.4.2 保留字.....	5
1.4.3 标识符.....	5
1.4.4 分隔符.....	5
1.5 C 程序的上机过程.....	5
1.5.1 C 程序的建立和编辑.....	6
1.5.2 C 程序的编译和运行.....	9
1.5.3 C 程序的调试.....	11
课后练习.....	12
第 2 章 数据类型、运算符与表达式.....	13
2.1 C 语言的数据类型.....	13
2.2 常量与变量.....	13
2.2.1 常量.....	13
2.2.2 变量.....	14
2.3 整型数据.....	15
2.3.1 整型常量.....	15
2.3.2 整型变量.....	16
2.3.3 整型数据的存储方式.....	17
2.4 字符类型.....	18
2.4.1 字符与 ASCII 码.....	18
2.4.2 字符常量.....	18
2.4.3 字符变量.....	19
2.4.4 字符数据的存储方式.....	20
2.4.5 字符串常量.....	20
2.5 实数类型.....	21
2.5.1 定点数与浮点数.....	21
2.5.2 实型常量.....	22
2.5.3 实型变量.....	22
2.5.4 sizeof 运算符.....	23
2.5.5 C 语言基本数据类型小结.....	23

2.6	算术运算.....	24
2.6.1	运算符与表达式.....	24
2.6.2	二元算术运算符.....	25
2.6.3	复合算术赋值运算符.....	26
2.6.4	自加和自减运算.....	26
2.6.5	正负号运算符.....	27
2.6.6	赋值类运算符的副作用及限制.....	27
2.7	关系运算与逻辑运算.....	28
2.7.1	关系运算.....	28
2.7.2	逻辑运算.....	29
2.7.3	条件运算符.....	30
2.8	逗号运算符和逗号表达式.....	31
2.9	不同类型数据间的转换.....	31
2.9.1	基本概念.....	31
2.9.2	数据类型的隐式转换.....	33
2.9.3	数据类型的显式转换.....	34
	课后练习.....	35
<b>第 3 章</b>	<b>数据的输入和输出.....</b>	<b>36</b>
3.1	printf 函数.....	36
3.1.1	printf 函数格式.....	36
3.1.2	格式说明.....	37
3.2	scanf 函数.....	42
3.2.1	地址参数.....	42
3.2.2	格式控制参数.....	42
3.2.3	scanf 函数的执行过程.....	44
3.2.4	scanf 函数如何分隔数据项.....	44
3.2.5	scanf 函数的停止与返回.....	47
3.3	getchar 函数和 putchar 函数.....	48
3.3.1	getchar 函数和 putchar 函数的定义.....	48
3.3.2	getchar 函数和 putchar 函数的用法.....	48
3.4	顺序结构程序设计举例.....	49
	课后练习.....	50
<b>第 4 章</b>	<b>选择型程序设计.....</b>	<b>56</b>
4.1	算法和算法的表示.....	56
4.1.1	算法的概念.....	56
4.1.2	算法的基本特征.....	58
4.1.3	算法的表示.....	59
4.1.4	结构化程序的三种基本结构.....	61



4.2 C 语言的语句.....	64
4.2.1 C 语言语句的基本概念.....	64
4.2.2 表达式语句.....	65
4.2.3 流程控制语句.....	65
4.2.4 复合语句.....	66
4.3 选择型程序结构.....	66
4.3.1 条件判断语句.....	66
4.3.2 多分支选择语句.....	69
4.4 选择型程序设计举例.....	71
课后练习.....	73
<b>第 5 章 循环控制.....</b>	<b>75</b>
5.1 先判断后执行循环控制语句 while.....	75
5.2 先执行后判断循环控制语句 do...while.....	77
5.3 循环次数控制语句 for.....	80
5.4 goto 语句以及用 goto 语句构成的循环结构.....	81
5.5 循环的嵌套.....	82
5.6 break 语句和 continue 语句.....	84
5.7 程序举例.....	85
课后练习.....	88
<b>第 6 章 数组.....</b>	<b>91</b>
6.1 一维数组.....	91
6.1.1 一维数组的定义.....	91
6.1.2 一维数组的初始化.....	91
6.1.3 数组元素的引用.....	92
6.2 二维数组.....	94
6.2.1 二维数组的定义.....	94
6.2.2 二维数组元素的引用.....	96
6.2.3 二维数组的初始化.....	97
6.3 字符数组.....	99
6.3.1 字符串和字符串的存储方法.....	99
6.3.2 字符数组的初始化.....	99
6.3.3 字符串的输入.....	100
6.3.4 字符串的输出.....	102
6.3.5 字符串运算函数.....	103
6.3.6 二维字符数组.....	105
课后练习.....	107
<b>第 7 章 函数.....</b>	<b>109</b>
7.1 函数的基础知识.....	110

7.1.1	函数的基本概念.....	110
7.1.2	函数的传值调用.....	115
7.1.3	函数的嵌套调用.....	117
7.1.4	函数的递归调用.....	118
7.1.5	标准库函数.....	120
7.2	变量的存储属性.....	123
7.2.1	存储属性的概念.....	123
7.2.2	自动 (auto) 变量.....	124
7.2.3	寄存器 (register) 变量.....	126
7.2.4	静态 (static) 变量.....	127
7.2.5	外部 (extern) 变量.....	128
7.2.6	存储类别小结.....	132
7.3	画图程序的实现.....	134
7.3.1	Turbo C 图形函数简介.....	134
7.3.2	图形函数调用举例.....	137
	课后练习.....	139
<b>第 8 章</b>	<b>编译预处理.....</b>	<b>141</b>
8.1	宏定义 (#define).....	141
8.1.1	不带参宏定义.....	141
8.1.2	带参宏定义.....	143
8.1.3	书写宏定义命令行应注意的问题.....	146
8.2	文件包含 (#include).....	146
8.2.1	文件包含的格式.....	146
8.2.2	典型举例.....	147
8.2.3	文件包含的使用特点.....	147
8.2.4	标准头文件.....	148
8.3	条件编译.....	149
8.3.1	条件编译的概念.....	149
8.3.2	条件编译的三种具体格式.....	149
8.3.3	使用条件编译的优点.....	150
	课后练习.....	151
<b>第 9 章</b>	<b>指针.....</b>	<b>152</b>
9.1	地址和指针.....	152
9.1.1	指针概述.....	152
9.1.2	指针变量的定义.....	153
9.1.3	指针变量的引用.....	154
9.2	一维数组的指针表示方法.....	155
9.2.1	一维数组的地址表示法 (地址法).....	155

9.2.2	一维数组的指针表示法.....	156
9.2.3	使用指针法应注意的问题.....	157
9.2.4	应用指针法举例.....	157
9.2.5	指针变量的运算.....	158
9.3	二维数组的指针表示方法.....	160
9.3.1	二维数组的地址.....	160
9.3.2	二维数组中的行地址与列地址.....	161
9.3.3	指针在二维数组中的应用举例.....	163
9.4	函数参数与指针、数组.....	164
9.4.1	指针作为函数参数.....	164
9.4.2	数组名作为函数参数.....	166
9.4.3	指向数组的指针作函数参数.....	168
9.5	返回指针值的函数.....	170
9.5.1	返回指针值函数的定义形式.....	170
9.5.2	返回指针值的函数用法举例.....	171
9.6	指向函数的指针变量.....	172
9.6.1	函数的指针.....	172
9.6.2	函数指针的定义.....	172
9.6.3	函数指针的特点.....	173
9.6.4	函数指针用法举例.....	173
9.6.5	使用函数指针的优点.....	176
9.7	指针与字符串.....	177
9.7.1	指向字符串的指针用法举例.....	177
9.7.2	字符数组与字符指针的用法区别.....	178
9.8	指针数组.....	179
9.8.1	引入指针数组的原因.....	179
9.8.2	定义指针数组的形式.....	179
9.8.3	指针数组的应用举例.....	180
9.9	指向指针的指针.....	183
9.9.1	指向指针的指针的定义.....	183
9.9.2	双重指针的用法和举例.....	183
9.9.3	多重指针的用法和举例.....	187
9.10	main 函数中的参数.....	187
9.10.1	main 函数中的参数形式.....	187
9.10.2	main 函数参数的传递过程.....	187
9.10.3	有关 main 函数参数的举例.....	188
9.11	指针数据小结.....	189
9.11.1	常用的指针类型变量归纳.....	189
9.11.2	指向 void 类型的指针.....	190

课后练习 .....	191
<b>第 10 章 结构体、共用体与枚举型 .....</b>	<b>193</b>
10.1 结构体类型的基本知识.....	193
10.1.1 结构体类型的定义.....	193
10.1.2 定义结构体类型变量的方法.....	194
10.1.3 结构体变量的初始化.....	196
10.1.4 结构体变量的引用.....	197
10.1.5 结构体变量的输入和输出.....	199
10.2 结构体数组.....	200
10.2.1 结构体数组的定义.....	200
10.2.2 结构体数组的初始化.....	201
10.2.3 结构体数组的引用.....	202
10.3 结构体变量与函数.....	204
10.3.1 结构体变量作为函数参数.....	204
10.3.2 返回结构体类型值的函数.....	206
10.4 结构体变量与指针.....	207
10.4.1 结构体指针 .....	207
10.4.2 指向结构体数组的指针.....	209
10.4.3 用作函数参数的结构体指针.....	210
10.5 链表.....	211
10.5.1 动态存储分配和链表的概念.....	211
10.5.2 用包含指针项的结构体变量构成结点.....	213
10.5.3 内存动态分配函数.....	214
10.5.4 链表应用举例.....	216
10.6 枚举类型数据.....	224
10.6.1 枚举类型的概念.....	224
10.6.2 枚举类型和枚举变量的定义.....	224
10.6.3 有关枚举常量的说明.....	225
10.7 共同体类型数据.....	227
10.7.1 共同体的概念.....	227
10.7.2 共同体变量的引用.....	229
10.7.3 共同体变量的应用.....	230
10.8 用 typedef 定义类型.....	232
10.8.1 使用 typedef 定义类型.....	232
10.8.2 使用 typedef 定义类型时应注意的问题.....	233
课后练习 .....	234
<b>第 11 章 位运算 .....</b>	<b>235</b>
11.1 位运算与位运算符.....	235
11.1.1 按位与运算符.....	235

11.1.2	按位或运算符.....	237
11.1.3	按位异或运算符.....	238
11.1.4	按位取反运算符.....	239
11.1.5	左移运算符.....	239
11.1.6	右移运算符.....	240
11.2	位运算应用举例.....	241
11.3	位段.....	244
11.3.1	位段的概述.....	244
11.3.2	位段的引用.....	246
	课后练习.....	248
<b>第 12 章</b>	<b>文件.....</b>	<b>249</b>
12.1	缓冲型文件指针变量的定义.....	250
12.2	缓冲型文件的打开与关闭.....	251
12.2.1	文件的打开 (fopen 函数).....	251
12.2.2	文件的关闭 (fclose 函数).....	253
12.3	缓冲型文件的读写.....	254
12.3.1	输入和输出一个字符.....	254
12.3.2	输入和输出一个字符串.....	256
12.3.3	格式化的输入和输出.....	258
12.3.4	按“记录”的方式输入和输出.....	260
12.4	缓冲型文件的定位与随机读写.....	262
12.4.1	文件的定位.....	262
12.4.2	随机读写.....	263
12.5	缓冲型文件操作的出错检测.....	266
12.5.1	ferror 函数.....	266
12.5.2	clearerr 函数.....	266
12.6	非缓冲文件系统 (系统 I/O).....	266
12.6.1	非缓冲文件系统的主要特点.....	266
12.6.2	打开文件.....	268
12.6.3	文件的读写与关闭.....	269
12.6.4	缓冲区的设置.....	270
	课后练习.....	271
附录 A	ASCII 码表.....	272
附录 B	C 常用库函数.....	273
附录 C	C 语言的关键字.....	277
附录 D	运算符的优先级与结合性.....	278
参考文献	.....	280

# 第 1 章 C 语言概述

人与人之间进行交流要用某种能够共同理解的语言，人与计算机进行交流当然也要有“语言”。程序员或操作人员是通过按某种语言规范设计程序来控制计算机的工作，从而完成指定的任务。因此，程序员必须事先掌握与计算机打交道的“计算机语言”。

就像人类的语言，有汉语、英语、日语等多种语种，计算机语言也有许多种，它们各自在不同的场合中使用。例如，FoxBase、FoxPro 这类语言通常用于数据库的管理；汇编语言通常用在同计算机硬件、接口打交道的场合；Fortran 语言用于数学计算；C 语言则常用于系统软件、工程软件的设计等。

## 1.1 程序与程序设计

从自然语言角度讲，程序描述了解决某一问题的方法和步骤。从计算机角度说，程序是用计算机能够理解并可执行的某种计算机语言描述的解决问题的方法和步骤。程序的特点是有始有终，每个步骤都能操作，所有步骤执行完毕，对应问题得到解决。

例如，某个会议的议程安排如下：

第一项 宣布会议开始。

第二项 全体起立唱国歌。

第三项 宣读嘉奖令。

第四项 颁发奖励证书。

……

第 N 项 宣布会议结束

又如，求解一元二次方程  $ax^2+bx+c=0$ （设  $a \neq 0$ ）实数根的步骤如下：

第一步 获得系数  $a$ ,  $b$ ,  $c$ 。

第二步 计算  $b^2-4ac$  并赋予  $d$ （即  $d=b^2-4ac$ ）。

第三步 根据  $d$  值求实数根并输出结果。

① 若  $d>0$ ，则计算两实数根：

$$x_1 = \frac{-b + \sqrt{d}}{2a}, \quad x_2 = \frac{-b - \sqrt{d}}{2a}$$

输出“有两个实根，分别为  $x_1$ 、 $x_2$ ”，然后转到第四步。

② 若  $d<0$ ，输出“没有实根”，然后转到第四步。

③ 若  $d=0$ ，计算  $x_1=x_2=(-b/2a)$ ，

输出“有两个相同的实根  $x_1$ ”，然后转到第四步。

第四步 结束。

程序设计就是将其分析解决问题的方法步骤记录下来过程。

**【例 1.1】**用 C 语言编写求解一元二次方程的程序。

```
#include <stdio.h> /*在程序中使用了输入/输出函数，插入其所在库*/
#include <math.h> /*在程序中使用了数学类的函数，插入其所在库*/
void main() /*主函数名*/
```

```

{ float a,b,c,d,x1,x2; /*提前说明程序中用到的变量*/
  printf("请输入系数 a,b,c:"); /*在程序中显示的提示信息*/
  scanf("%f,%f,%f",&a,&b,&c); /*输入 a, b, c 三个系数*/
  d=b*b-4*a*c; /*计算 b*b-4ac 并赋予 d*/
  if(d>0){ x1=(-b+sqrt(d))/(2*a); /*若 d>0, 计算并输出两个实根*/
          x2=(-b-sqrt(d))/(2*a);
          printf("\nX1=%f,X2=%f",x1,x2);
        }
  else if(d<0)printf("没有实数根! "); /*若 d<0, 输出没有实根*/
  else{ x1=x2=-b/(2*a); /*若 d=0, 计算并输出重根*/
        printf("\nX1=X2=%f",x1);
      }
}

```

运行程序，屏幕显示结果如下：

```

请输入系数 a,b,c:1, 6, 2<回车> /*键盘输入 1, 6, 2 三个系数*/
X1=-0.354249,X2=-5.645751

```

需特别指出的是，初学者在开始编程时就要注重程序的格式，保证其可读性，并加入必要的注释。

初级程序员对于一个简单的问题，若要用计算机解决，具体步骤大致为：

- (1) 接受并分析问题，确定待解决问题的计算或处理方法。
- (2) 将实际问题转变为一个数学问题，用数学模型（表达式）描述。
- (3) 编制程序流程图，确定程序结构。
- (4) 选择计算机语言和它的工作模式。
- (5) 编写程序。
- (6) 上机编辑、调试（包括编译和连接），消除语法错误。

(7) 如果结果正确则生成最终的用户程序，否则返回步骤（1）从头再来，找出逻辑或设计错误所在。

对于一个大型软件来说，在它的设计过程中会涉及到许多复杂的问题，所以仅仅采用上面的步骤是不够的，必须按软件工程的方法进行设计。

## 1.2 C 语言的特点

各类计算机语言的发展历程大致为：先有机器语言，再有汇编语言，最后出现中级语言和高级语言。

机器语言是各类语言在计算机上运行和存储的最终形式。它是由二进制编码组成，各命令和地址均用对应的二进制码表示。汇编语言用具有一定意义的符号代替机器语言中的各条命令和地址。汇编语言必须通过汇编程序把其中的符号还原成对应的二进制码才能运行。由于用上述语言所编写的程序只能同类型的计算机上执行，所以又称它们为“面向机器的语言”或“低级语言”。

为了使程序能够脱离具体的机器硬件，将一般程序员从对机器内部结构的深入了解中解放出来，并使程序的编写接近日常的数学表达习惯，计算机专家们设计出了一系列的高级语言。高级语言程序也必须经过编译，转换成机器语言才可以运行。由于高级语言是面向问题

和算法的过程描述，所以它们又称为“面向问题的语言”。

用高级语言或汇编语言编写的程序称为源程序，必须将其翻译成二进制的机器语言程序后才能执行。翻译有两种方式，一种是通过“解释程序”翻译一句执行一句的方式运行；另一种通过“编译程序”一次翻译产生目标程序，然后执行。

虽然高级语言适应面较宽，但它使程序员对硬件的控制能力大为减弱。因此，随后又出现了一系列既保留高级语言的使用简便、可读性好等特点，又具备汇编语言所具有的对硬件的强大控制能力的“中级语言”，常见的就是C语言。

C语言本身的特点主要有：

- 表达能力强，能实现汇编语言的大部分功能，可直接访问内存物理地址和硬件寄存器，能进行位运算。
- 流程控制结构化、程序结构模块化。有顺序、选择、循环三种控制结构，是理想的结构化语言，符合现代编程的风格。
- 语言简练、紧凑。C语言共有32个关键字，9种控制语句，程序书写形式自由，如“ $i=i+1;$ ”在C中还可写为“ $i++;$ ”。
- 数据结构丰富，具有现代化语言的各种数据结构，能用来实现各种复杂的数据结构的运算。
- 运算符丰富。C语言有34种运算符，把括号、赋值、强制类型转换等都作为运算符处理。

C语言的应用特点主要有：

- C程序代码质量高。编译后生成的目标程序运行速度快，占用存储空间少，几乎与汇编语言相媲美。
- 可移植性好。不作改动或稍加改动就能从一个机器系统移植到另一个机器系统上。

C语言的主要不足是：语法限制不太严格、数据类型无严格的对应关系，既是灵活之处又是不足之处。

### 1.3 C程序的基本结构

下面先介绍几个简单的C程序，然后从中分析、理解C程序的基本结构。

**【例 1.2】**编程显示某一字符串。

```
#include <stdio.h>          /*在程序中使用了输入/输出函数，插入其所在库*/
void main()                 /*主函数*/
{
    printf("同学们，你们好！"); /*显示常量字符串“同学们，你们好！”*/
}
```

运行结果：

同学们，你们好！

**【例 1.3】**求三个实数的平均值。

```
#include <stdio.h>          /*在程序中使用了输入/输出函数，插入其所在库*/
void main()                 /*主函数*/
{ float a,b,c,d;           /*定义存放实数的变量*/
  a=1.23;b=-4.56;c=7.89;   /*给各变量赋值*/
  d=(a+b+c)/3;            /*将三个实数的平均值赋予d*/
}
```



```
printf("d=%f",d);          /*显示运行结果*/
}
```

运行结果:

d=1.520000

**【例 1.4】**由键盘输入三个整数，显示其中的最大值和最小值。

要求：求最大值和最小值用函数实现。

```
#include <stdio.h>          /*在程序中使用了输入/输出函数，插入其所在库*/
/*定义一个名为 max 的函数，功能是返回三个数中的最大值*/
int max(int i1,int i2,int i3)
{ int m;                    /*定义一个存放中间结果的变量*/
  if((i1>=i2)&&(i1>=i3))
    m=i1;
  else if((i2>=i1)&&(i2>=i3))
    m=i2;
  else m=i3;
  return m;                 /*返回变量 m 中获得的最大值*/
}
/*定义一个名为 min 的函数，功能是返回三个数中的最小值*/
int min(int i1,int i2,int i3)
{ int m;                    /*定义一个存放中间结果的变量*/
  if((i1<=i2)&&(i1<=i3))
    m=i1;
  else if((i2<=i1)&&(i2<=i3))
    m=i2;
  else m=i3;
  return m;                 /*返回变量 m 中获得的最小值*/
}
void main()                 /*主函数*/
{ int a,b,c,d;              /*定义存放实数的变量*/
  printf("请输入三个整数值:"); /*显示提示信息*/
  scanf("%d,%d,%d",&a,&b,&c); /*输入变量 a, b, c*/
  d=max(a,b,c);             /*求最大值*/
  printf("\n最大值是:%d",d); /*显示最大值*/
  d=min(a,b,c);             /*求最小值*/
  printf("\n最小值是:%d",d); /*显示最小值*/
}
```

运行程序，屏幕显示结果如下：

```
请输入三个整数值: 7,4,5<回车> /*从键盘输入三个数: 7, 4, 5*/
最大值是: 7
最小值是: 4
```

从上述三个例子可以看出：

- (1) C 程序是由若干个函数构成的，每个 C 程序有且仅有一个主函数（函数名规定为 main）。
- (2) 每个函数（含主函数）的定义分为两部分——函数说明部分和函数体。
- (3) 组成 C 语言源程序的基本单位是语句。C 程序中的语句最后总要有个分号（；）作为每个语句的结束。
- (4) 程序中的“注释”是用“/\*”和“\*/”括起来的任意字符段落。
- (5) C 程序的书写格式很灵活，在一行上可以书写多个语句。