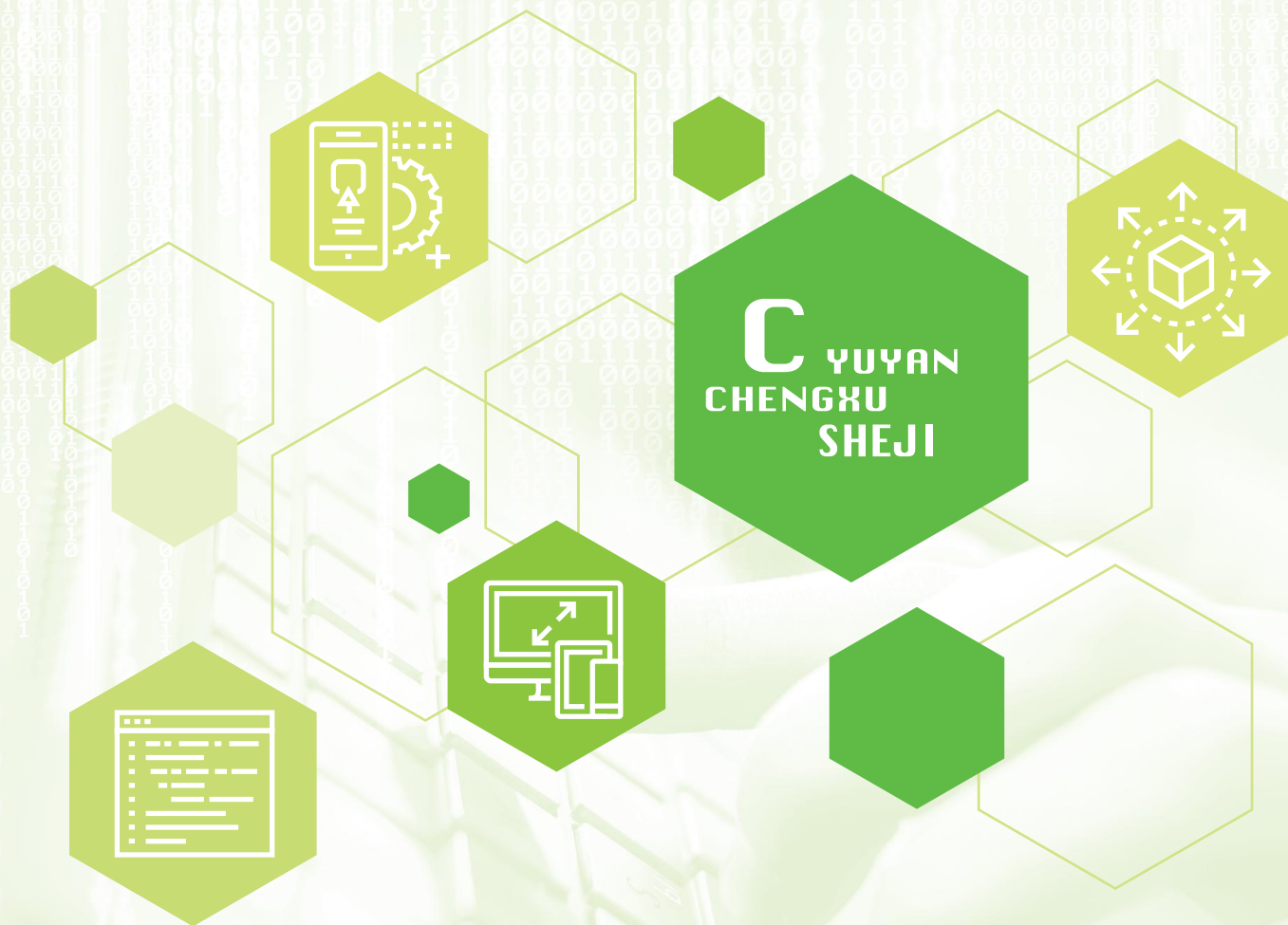


辽宁省“十四五”职业教育规划教材

C 语言 程序设计

主编 韩冬艳 范咏红 史迎新



上海交通大学出版社
SHANGHAI JIAO TONG UNIVERSITY PRESS

内容提要

本书系统介绍了 C 语言的基本概念、特点和结构化程序设计方法。全书共分为 11 个项目，包括 C 语言概述、数据类型及顺序结构程序设计、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、文件、预处理、综合实战。

本书内容全面，涵盖了 C 语言中的所有知识，可作为职业院校 C 语言程序设计课程的教材，也可作为相关技术人员的参考用书。

图书在版编目 (CIP) 数据

C 语言程序设计 / 韩冬艳，范咏红，史迎新主编. —

上海：上海交通大学出版社，2021.8（2023.4 重印）

ISBN 978-7-313-25229-6

I. ① C… II. ①韩… ②范… ③史… III. ① C 语言—
程序设计—高等职业教育—教材 IV. ① TP312.8

中国版本图书馆 CIP 数据核字（2021）第 153770 号

C 语言程序设计

C YUYAN CHENGXU SHEJI

主 编：韩冬艳 范咏红 史迎新

出版发行：上海交通大学出版社

邮政编码：200030

印 制：北京荣玉印刷有限公司

开 本：889 mm × 1194 mm 1/16

字 数：359 千字

版 次：2021 年 8 月第 1 版

书 号：ISBN 978-7-313-25229-6

定 价：49.80 元

地 址：上海市番禺路 951 号

电 话：6407 1208

经 销：全国新华书店

印 张：17.5

印 次：2023 年 4 月第 2 次印刷

版权所有 侵权必究

告读者：如发现本书有印装质量问题请与印刷厂质量科联系

联系电话：010-6020 6144



计算机科学发展的每一步几乎都在软件设计和程序设计语言中得到充分体现。C 语言作为一种程序设计语言具有高级语言和汇编语言的特点。C 语言的应用非常广泛，既可以用于编写系统软件，也可以用于编写应用软件，还可以应用到有关单片机及嵌入式系统的开发中。“C 语言程序设计”是针对职业院校计算机专业及非计算机专业理工科学生开设的一门重要的编程基础课程，具有很强的操作性、实用性和应用性。该课程主要培养学生计算机编程基本思想、编程基本技能及逻辑思维能力。

本书贯彻落实党的二十大精神，在内容中融入思政教育，针对学生的特点和培养目标，从学生的思维方式、理解能力、接收能力等因素出发，结合多年的教学经验，深入浅出地介绍 C 语言程序设计的基础知识和编程方法，遵循以学生为本的教育理念，注重全面提高学生的职业实践能力和职业素养。

本书系统介绍了 C 语言的基本概念、特点和结构化程序设计方法。全书共分 11 个项目。每个项目都融入思政元素，在讲解 C 语言程序设计知识的同时让学生潜移默化地受到思政教育。项目 1 介绍了 C 语言的发展和特点、C 程序的构成及上机环境的使用方法；项目 2 介绍了编程语言基础，包括数据类型、常量和变量、运算符、表达式、数据的输入与输出、顺序结构程序设计，掌握这些基础知识就可以进行简单编程了；项目 3 和项目 4 分别介绍了选择结构和循环结构，均结合丰富的例题分析进行讲解，这两个项目是学习 C 语言的重点，是培养编程思想的部分；项目 5 介绍了构造类型数组，包括一维、二维数组的定义和引用，以及字符和字符串数组；项目 6 介绍了 C 语言的函数，包括函数的定义、调用，变量的存储类型，内部函数和外部函数；项目 7 介绍了指针类型的定义及使用方法，是全书的难点项目，也是 C 语言的精髓；项目 8 介绍了结构体与共用体；项目 9 介绍了文件的概念及基本操作；项目 10 介绍了预处理的相关知识；项目 11 是综合实战，综合应用前面所学知识完成项目实战。

本书特点是知识点深入浅出，语言通俗易懂，采用项目任务式书写格式，层次分明，逻辑清晰，例题丰富，实用性强。本书的作者都是长期从事一线教学的计算机专业教师，擅长从不同角度审视问题，在教学和实践中与学生广泛接触，深知学生在学习上的特点和规律，并根据多年教学经验逐一进行了归纳总结。此外，本书作者还为广大一线教师提供了服务于本书的教学资源库，有需要者可致电 13810412048 或发邮件至 2393867076@qq.com。

本书内容全面，涵盖了 C 语言中的所有知识，可作为职业院校 C 语言程序设计的教材，也可作为一般工程技术人员的参考用书。

由于作者水平有限，编写时间仓促，书中存在的错误和疏漏之处，恳请广大读者批评指正。

目 录



项目 1 C 语言概述 / 1

项目描述	2	1.6 运行 C 语言程序的步骤	8
相关知识	2	1.7 C-Free 编译软件介绍	8
1.1 计算机语言	2	1.7.1 C-Free 的下载	8
1.2 C 语言的发展过程	3	1.7.2 C-Free 的安装	8
1.2.1 C 语言的产生背景	3	1.7.3 C-Free 的使用	10
1.2.2 C 语言的发展历程	4	项目实训	12
1.3 C 语言能干什么	4	项目总结	13
1.4 C 语言的特点	4	项目考核	14
1.5 C 语言程序的格式与构成	5		



项目 2 数据类型及顺序结构程序设计 / 15

项目描述	16	2.5 字符型数据	21
相关知识	16	2.5.1 字符常量	21
2.1 C 语言数据类型	16	2.5.2 字符串常量	22
2.2 标识符、常量和变量	17	2.5.3 字符变量	22
2.2.1 标识符	17	2.6 运算符和表达式	22
2.2.2 常量	18	2.6.1 算术运算符与算术表达式	23
2.2.3 变量	18	2.6.2 自增、自减运算符	23
2.3 整型数据	18	2.6.3 关系运算符与关系表达式	24
2.3.1 整型常量	18	2.6.4 逻辑运算符和逻辑表达式	25
2.3.2 整型变量	19	2.6.5 条件运算符和条件表达式	26
2.3.3 整型变量的定义及使用	19	2.6.6 赋值运算符和赋值表达式	26
2.4 实型数据	20	2.6.7 复合赋值运算符	26
2.4.1 实型常量	20	2.6.8 逗号运算符和逗号表达式	27
2.4.2 实型变量	21	2.6.9 类型转换	27
2.4.3 实型变量的定义	21		

2.7 数据的输入与输出	28	2.8.1 C 语句	31
2.7.1 printf() 函数	28	2.8.2 顺序结构程序设计	33
2.7.2 printf 函数的格式符说明	30	项目实训	33
2.7.3 scanf() 函数	30	项目总结	35
2.8 顺序结构程序设计	31	项目考核	36



项目 3 选择结构程序设计 / 39

项目描述	40	3.4 switch 语句	44
相关知识	40	项目实训	47
3.1 简单 if 语句	40	项目总结	51
3.2 if-else 语句	41	项目考核	52
3.3 选择结构的嵌套	43		



项目 4 循环结构程序设计 / 57

项目描述	58	4.6 break 语句	66
相关知识	58	4.7 continue 语句	68
4.1 循环结构	58	4.8 循环的嵌套	69
4.2 while 语句	58	项目实训	71
4.3 do-while 语句	61	项目总结	82
4.4 for 语句	63	项目考核	83
4.5 for 语句的变形	64		



项目 5 数 组 / 89

项目描述	90	5.3 字符和字符数组	96
项目分析	90	5.3.1 字符数组的定义和初始化	96
5.1 一维数组的定义与引用	90	5.3.2 字符数组的引用	97
5.1.1 一维数组的定义	90	5.3.3 字符数组与字符串	98
5.1.2 一维数组的引用	90	5.3.4 字符数组的输入输出	99
5.1.3 一维数组的初始化	91	5.3.5 字符及字符串处理函数	100
5.2 二维数组的定义与引用	93	项目实训	103
5.2.1 二维数组的定义	93	项目总结	112
5.2.2 二维数组的引用	93	项目考核	112
5.2.3 二维数组的初始化	94		



项目 6 函 数 / 117

项目描述	118	6.6 变量的作用域	135
相关知识	118	6.6.1 局部变量	135
6.1 C 语言中函数的类别	118	6.6.2 全局变量	137
6.2 函数的声明及函数的调用	119	6.7 变量的存储类型	139
6.2.1 无参函数的一般格式	119	6.7.1 自动变量	139
6.2.2 有参函数的一般格式	120	6.7.2 外部变量	141
6.2.3 函数的调用	121	6.7.3 静态变量	142
6.3 函数的参数和函数的值	123	6.7.4 寄存器变量	144
6.3.1 函数的参数	123	6.8 内部函数和外部函数	144
6.3.2 函数的值	125	6.8.1 内部函数	144
6.3.3 函数原型声明	127	6.8.2 外部函数	145
6.3.4 数组作为函数参数	129	项目实训	145
6.4 函数的嵌套调用	132	项目总结	152
6.5 函数的递归调用	133	项目考核	152



项目 7 指 针 / 157

项目描述	158	7.4.1 指针与一维数组	164
项目分析	158	7.4.2 数组指针做函数的参数	166
7.1 指针与指针变量	158	7.4.3 指针与字符数组	168
7.2 指针变量的定义与引用	159	7.5 指向数组的指针和指针数组	170
7.2.1 指针变量的定义	159	7.5.1 指向数组的指针	170
7.2.2 指针变量的引用	159	7.5.2 指针数组	172
7.3 指针运算符与指针表达式	160	项目实训	173
7.3.1 指针运算符与指针表达式	160	项目总结	178
7.3.2 指针变量做函数的参数	162	项目考核	179
7.4 指针与数组	163		



项目 8 结构体与共用体 / 183

项目描述	184	8.2.3 结构体类型定义的方法	188
相关知识	184	8.2.4 结构体数组	189
8.1 结构体类型的引入	184	8.2.5 结构体指针	191
8.2 结构体类型的定义及使用	184	8.3 共用体	192
8.2.1 基本结构体类型的定义	184	8.3.1 共用体类型声明及引用	192
8.2.2 结构体变量的引用	186	8.3.2 共用体的实例测试	194

8.4 用 typedef 定义类型	195	项目总结	199
项目实训	196	项目考核	200



项目 9 文 件 / 203

项目描述	204	9.4 一般文件读写函数及其使用	206
相关知识	204	9.5 文件定位函数	218
9.1 概述	204	项目实训	219
9.2 文件指针	205	项目总结	220
9.3 文件打开与关闭函数	205	项目考核	220



项目 10 预处理 / 223

项目描述	224	10.2 “文件包含” 处理	229
相关知识	224	10.3 条件编译	230
10.1 宏定义	224	项目实训	231
10.1.1 不带参数的宏定义	224	项目总结	232
10.1.2 带参数的宏定义	227	项目考核	233



项目 11 综合实战 / 236

项目描述	236	项目总结	256
项目实训	236		



附 录 / 257

附录 A 常用字符与 ASCII 代码对照表	257	附录 D C 语言常用语法提要	262
附录 B C 语言中的关键字	258	附录 E C 语言库函数	265
附录 C 运算符和结合性	259		

参考文献	272
------------	-----

项目 1

C 语言概述

项目目标

- 1 了解 C 语言的发展史。
- 2 掌握 C 语言的特点。
- 3 掌握 C 语言程序的基本组成结构。
- 4 熟悉 C-Free 软件的使用方法。
- 5 熟练使用 C 语言程序的调试及运行步骤。
- 6 激发科技报国的家国情怀。
- 7 激发对软件行业的热情，提高编程兴趣。

知识导图



项目描述

C 语言概述讲解了 C 语言的应用和计算机语言的发展历程，从中可以发现我国在专业软件上的差距，更可以发现西方国家的科技是数代人上百年累积的结果。我们既要正视差距，更要正视改革开放以来的伟大科技成就，树立文化自信、制度自信。党的二十大指出“教育、科技、人才是全面建设社会主义现代化国家的基础性、战略性支撑”。在新一轮科技革命和人才竞争的时代背景下，期待广大青年才俊勇攀科技高峰，实现自己的价值，完成自己使命。相信中国人民一定能，中国一定行！

小李是今年考入某大学的计算机网络专业的一名学生，入校后了解到计算机网络专业是培养具有一定计算机网络基本理论和开发技术，具备从事程序设计、Web 软件开发、计算机网络组建、网络设备配置、网络管理和安全维护能力的网络高技术应用型人才。大学一年级有一门很重要的专业必修课程“C 语言程序设计”，小李对这门课程不是十分了解，很想尽快掌握入门的方法。

【分析】

程序是一组计算机能识别和执行的指令。源程序是用高级语言编写的程序，人们把 C 语言写的程序称为 C 源程序。程序设计是用户告诉计算机按程序员的思路一步一步地工作。那么 C 语言程序设计就是通过 C 语言与计算机通信，告诉计算机应如何工作。

相关知识

1.1 计算机语言

语言是人与人之间交流的工具，人和计算机怎么沟通呢？由于计算机硬件的器件特性决定了计算机本身只能接受由 0 和 1 编码的二进制指令和数据，因此人们为了让计算机工作，必须编写出由“0”和“1”组成的一系列指令，通过它指挥计算机工作，这就产生了计算机语言。计算机语言有低级语言和高级语言之分。

1. 低级语言时代（1946—1953 年）

低级语言包括机器语言和汇编语言，这两种语言都是面向机器的语言。

机器语言（machine language）也称二进制语言，是第一代计算机语言。机器语言是机器能直接识别的程序语言或指令代码，无须经过翻译，每一个操作码在计算机内部都由相应的电路来完成。程序由“0”和“1”组成，这种用二进制编写的程序也称目标程序。人们读写和识别很困难，特别是程序需要修改时更是非常麻烦。不同型号的计算机都有各自的机器语言，即指令系统，所以用某种机器语言编写的程序只能在相应的计算机上执行，无法移植到其他计算机上。但计算机能直接识别和执行程序中的指令，所以机器语言的运算效率是所有语言中最高的。

汇编语言（assembly language）也称符号语言，是第二代计算机语言。汇编语言把机器指令转换成易于阅读的助记符，例如，使用“ADD”表示加法，“MOV”表示数据传递。通过这种助记符，人们就能较容易地读懂程序，调试和维护也更方便了。但这些助记

符号计算机无法识别，需要一个专门的程序将其翻译成机器语言，这种翻译程序称为汇编程序。因为汇编语言只是将机器语言做了简单编译，其通用性和移植性仍很差，但是保持了机器语言优秀的执行效率，因为它的可阅读性和简便性，至今仍是一种常用的软件开发工具。它通常被应用在底层以及硬件操作和高要求的程序优化场合。

2. 高级语言时代（1954 年至今）

高级语言（high-level language）是第三代计算机语言，它接近于人类自然语言和数学语言，基本脱离了机器的硬件系统，用人们更易理解的方式编写程序，便于广大用户掌握和使用。高级语言的通用性强，兼容性好，便于移植。由于机器语言是计算机唯一识别的语言，使用高级语言编写的程序必须事先翻译成机器语言程序计算机才能执行。这个“翻译”工作是由称为“编译系统”的软件来实现的。

从 1954 年出现的第一个高级语言 FORTRAN 以来，全世界先后出现了 2500 种以上的高级语言，编程语言并不像人类自然语言发展变化那样缓慢而持久，其发展相当迅速，这主要是计算机硬件、互联网和 IT 业的发展促进了编程语言的发展。每种高级语言都有其特定的应用领域。其中应用比较广泛的有 100 多种，常见的高级编程语言有 FORTRAN（数值计算）、ALGOL（第一个结构化程序设计语言）、C、C++、Basic、Pascal、LISP、Prolog（人工智能语言）、Visual Basic（支持面向对象程序设计的语言）、Visual C++、Java（适用于网络的语言）和 Python（Web 和 Internet 开发语言）等。

1.2 C 语言的发展过程

C 语言诞生于 1972 年，由美国电话电报公司（American Telephone & Telegraph Company, AT&T）贝尔实验室的丹尼斯·里奇（Dennis M. Ritchie，见图 1-1）发明。为了更好地开发新版本的 UNIX，丹尼斯·里奇在 B 语言的基础上设计了 C 语言，并在一台使用 UNIX 操作系统的 DEC PDP-11 计算机上实现。

1.2.1 C 语言的产生背景

C 语言的原型是 ALGOL 60 语言（也称 A 语言）。1963 年，剑桥大学将 ALGOL 60 语言发展成为 CPL（combined programming language）。

1967 年，英国剑桥大学的马丁·理查德（Martin Richard）对 CPL 语言进行了简化，于是产生了 BCPL 语言。

1970 年，美国贝尔实验室的肯·汤普逊（Ken Thompson）（见图 1-1）以 BCPL 语言为基础，设计出一种既简单又接近于硬件的语言，取名为“B 语言”，意思是将 CPL 语言煮干，提炼出它的精华。他用 B 语言写了第一个 UNIX 操作系统，并在 PDP-7 计算机上实现。

1972 年，B 语言也被“煮”了一下，美国贝尔实验室的丹尼斯·里奇（Dennis M. Ritchie）在 B 语言的基础



图 1-1 肯·汤普逊（左）和丹尼斯·里奇（1941—2011）（右）



笔记



上最终设计出了一种新的语言，这就是 C 语言。C 语言既保持了 BCPL 语言和 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）。

1.2.2 C 语言的发展历程

C 语言问世后，在应用中不断改进。1975 年，UNIX 版本 6 公布以后，C 语言开始引起人们的注意，它的优点逐步被人们所认识。1977 年，出现了与具体机器无关的 C 语言编译文本，推动了 UNIX 操作系统在各种机器上的迅速地实现。C 语言最初是为描述和实现 UNIX 操作系统提供一种工作语言而设计的（原来的 UNIX 操作系统是 1969 年由肯·汤普逊与丹尼斯·里奇开发成功的，是用汇编语言编写的），但随着 UNIX 日益广泛的使用，C 语言也得到了推广。1978 年以后，C 语言先后被移植到大、中、小和微型机上，它很快成为世界上应用最广泛的计算机语言之一。

1978 年，UNIX 版本 8 推出后，以该版本中的 C 编译程序为基础，布莱恩·柯尼汉（B. W. Kernighan）和丹尼斯·里奇合作（合称 K&R）出版了 *The C Programming Language*，被称为标准 C。1983 年，美国国家标准化协会（American National Standards Institute, ANSI）对 C 语言的各种版本进行了扩充，推出了新的标准，被称为 83 ANSI C。它比原来的标准 C 有了改进和扩充。1987 年，ANSI 又公布了新标准——87 ANSI C。目前流行的各种 C 语言编译系统的版本大多以此为基础，但各有不同。

1.3 C 语言能干什么

C 语言一般应用在与底层硬件有接触的程序中，因为 C 语言可以直接对硬件寄存器进行操作和控制：

（1）C 语言最著名的应用领域就是操作系统，基本上大多数常见的操作系统内核都是用 C 语言编写的，如 Linux、Windows、UNIX、BSD（UNIX 的衍生系统）等。

（2）C 语言可以做服务器开发，现在的游戏服务器端大多都是基于 C 或 C++ 语言开发的。

（3）单片机、ARM 处理器等都用 C 语言，这都是目前比较热门的方向。另外，C 语言除了用在单片机上外，还可以用在很多其他嵌入式系统上。

（4）家里使用的小家电，例如，电冰箱控制显示板、洗衣机控制显示板，以及家里的电视机遥控器大部分也是用 C 语言来编写的。

（5）现在的微控自动化、航空制导、人工智能、云技术及服务器操作系统内核几乎全是用 C 语言编写的。

可见 C 语言的功能非常强大，效率也很高。

1.4 C 语言的特点

C 语言不仅可以发挥高级编程语言的功用，还具有汇编语言的优点，因此相对于其他

编程语言，C 语言具有以下几个特点：

(1) 简洁、紧凑，使用方便灵活。掌握任何程序设计语言都需要掌握一些关键字才能编写程序，C 语言一共有 32 个关键字、9 种控制语句，程序书写格式自由，且区分大小写。它把高级语言的基本结构和语句与低级语言的实用性结合起来，可以像汇编语言一样对位、字节和地址进行操作，而这三者是计算机基本的工作单元。

(2) 运算符丰富。C 语言有 34 种运算符，除了算术运算符、关系运算符和逻辑运算符外，还有自增、自减、复合赋值和位运算符等，使 C 程序的表达式类型和运算符类型均非常丰富。

(3) 数据类型丰富，具有现代语言的各种数据结构。C 语言不仅包含传统的字符型、整型、实型、数组，还有指针、结构体和共用体等数据类型，使用十分灵活和多样化，还可以实现诸如链表、堆栈、队列、树等各种复杂的数据结构。其中，指针使参数的传递简单、迅速，并节省了内存。

(4) 具有结构化的控制语句。C 语言具有 if-else、switch、while、do-while 和 for 等流程控制语句，C 语言是以函数格式提供给用户的，这些函数可方便地调用，并且具有多种循环、条件控制程序流向，从而使程序结构化。

(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，对变量的类型使用比较灵活，整型与字符型数据及逻辑型数据可以通用。对于初学者来说，编写一个正确的 C 语言程序可能会比编一个其他高级语言程序难一些，C 语言要求编写程序的人员对程序设计更熟练些。

(6) C 语言允许直接访问物理地址，能进行位操作，实现汇编语言的大部分功能，可以直接对硬件操作。因此，C 语言既具有高级语言功能，又具有低级语言的许多功能，可用来编写系统软件和应用软件。

(7) 生成目标代码质量高，程序执行效率高。C 语言编写的程序生成目标代码质量高，一般只比汇编程序生成的目标代码效率低 10%~20%，这是其他高级语言无法相比的。

(8) 可移植性好。用 C 语言编写的程序基本不做修改就能用于各种操作系统和各种型号的计算机，如 DOS、UNIX 和 Windows，也适用于多种机型。

上面介绍的是 C 语言最容易理解的一些特点，C 语言内部的其他特点将结合以后各项目的内容做介绍。C 语言的以上特点，初学者一时可能难以深刻理解，待学完 C 语言以后再回头看这些特点会有更深的体会。

1.5 C 语言程序的格式与构成

为了说明 C 语言源程序结构的特点，先看以下几个程序实例。

【例 1-1】最简单的 C 语言程序，输出“你好，中国”。

```
#include<stdio.h>
main()
```



笔记

```
{  
    printf("你好，中国\n"); /* 输出函数，运行后在屏幕上输出“你好，中国” */  
}
```

程序解析：

在使用标准函数库中的输入/输出函数时，编译系统要求程序提供有关的信息（如对输入/输出函数的声明），程序第一行“`#include<stdio.h>`”就是用来提供这些信息的。`include`是文件包含命令，其含义是把尖括号或双引号内指定的文件包含到本程序中，成为本程序的一部分。`stdio.h`是C语言编译系统提供的一个文件名，`stdio`是“standard input output”的缩写，即有关“标准输入输出”的信息，这里要用到`stdio.h`中的`printf()`格式化输出函数。有关C语言库函数的内容可以参见附录E。对于这部分内容在项目10预处理中有详细介绍，暂不讨论，大家只需知道在编写程序时第一行要写上`#include<stdio.h>`即可。

一个C语言程序有且仅有一个主函数`main()`，`main`是主函数的名称，一个程序的执行是从主函数开始，执行`main()`下面的左大括号“{”后面的语句到右大括号“}”结束。由大括号“{”括起的部分称为函数体，表示函数具体实现某种功能的一段代码。本例中函数体只有一条语句。

`printf`为格式化输出函数，该函数定义在标准输入输出函数库中，本例中双引号是原样输出里面的内容。“`\n`”是换行符，在执行程序时，输出“你好，中国”后光标跳转到下一行的开始处。`printf`函数在项目2中有具体介绍。

分号“`;`”表示语句的结束，即一个分号表示一个具有完整意义的C语句的结束。

`/*.....*/`为C语言的注释内容，程序中的注释是给阅读程序的人看的，为了提高程序的可读性而加的说明。程序运行时不会执行注释的内容。

其程序的运行结果如图1-2所示。

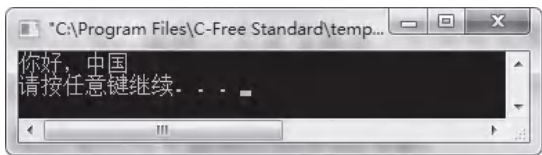


图 1-2 “你好，中国”程序的运行结果

【例 1-2】求两个数的和。

```
#include<stdio.h>  
main()  
{  
    int a,b,sum; /* 定义 3 个整型变量 */  
    a=123; b=456; /* 为变量赋值 */  
    sum=a+b;  
    printf("sum is %d\n",sum);  
}
```

其程序运行结果如下：

```
sum is 579
```

这里涉及变量的定义和使用、printf 函数的用法，详细介绍见项目 2。

【例 1-3】输入两个整数，输出其中的最大值。

```
#include<stdio.h>
main()
{
    int max(int x,int y);
    int a,b,c;
    printf("请输入两个整数，以逗号分隔，输入结束按回车键 :\n");
    scanf("%d,%d",&a,&b); /* 输入函数，输入变量 a,b 的值 */
    c=max(a,b);          /* 主函数调用 max() 函数 */
    printf("max=%d",c);
}
int max(int x, int y) /* 用户自己定义的函数 max() */
{
    int z;
    if (x>y) z=x;
    else z=y;
    return(z);
}
```

其程序的运行结果如图 1-3 所示。

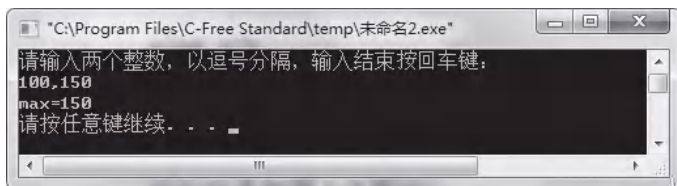


图 1-3 最大值程序的运行结果

例 1-2 和例 1-3 只需关注它的程序组成格式，具体含义会在后面项目中学习。通过以上 3 个程序可以看出 C 源程序的构成有以下特点：

(1) C 语言程序主要由函数构成。C 语言程序中有主函数 main()、系统提供的库函数（如 printf()，scanf() 等）及编程人员自己设计的自定义函数（如例 1-3 中的 max() 函数）3 种类型函数。

(2) 一个函数由两部分组成。

函数首部：函数的第一行，包括函数名、函数类型、函数参数等信息。

函数体：函数首部下面大括号“{ }”内的部分。



笔记

函数的具体内容将在项目 6 中进行介绍。

(3) 一个 C 语言程序总是从 `main()` 函数开始执行的, 无论 `main()` 函数在什么位置。

(4) C 语言程序是区分大小写的, 一般程序中都使用小写字母书写。

(5) 在 C 语言程序中, 所有标点符号都使用英文标点符号。

(6) C 程序书写自由, 一行可以写多个语句, 一个语句也可以分多行书写。

(7) 每一个说明, 每一个语句都必须以分号 “;” 结尾。注意, 这里分号是英文状态下的分号。

(8) C 语言本身没有输入输出语句, 输入和输出的操作是由库函数 `scanf` 和 `printf` 等函数来完成的。

(9) 用 `/*.....*/` 对 C 语言程序中的任何一行或数行做注释。

1.6 运行 C 语言程序的步骤

编写完一个 C 语言程序后, 通常要经过如下运行步骤:

(1) 编辑源程序: 在编译软件下 (如 C-Free) 输入源程序并保存文件 (扩展名为 `.c`)。

(2) 编译源程序: 将源程序翻译为目标文件 (扩展名为 `.obj`), 将源程序翻译成二进制目标代码。

(3) 连接目标程序: 编译后产生的目标文件输入内存与系统提供的库函数进行连接, 生成可执行文件 (扩展名为 `.exe`)。

(4) 运行: 最后把 `.exe` 文件调入内存并使其运行, 得到运行结果。

1.7 C-Free 编译软件介绍

为了编译、连接和运行 C 语言程序, 必须有相应的 C 语言编译系统, 常用的有 Turbo C/C++、Visual C++、win-Tc 和 C-Free 等。本书中的程序是在 C-Free 环境下调试运行的。

1.7.1 C-Free 的下载

启动浏览器, 在地址中输入 http://www.programarts.com/cfree_ch/index.htm, 进入 C-Free 网站, 按照提示即可下载 C-Free 软件。

1.7.2 C-Free 的安装

(1) 双击下载的 C-Free 软件, 进入其欢迎界面, 如图 1-4 所示, 单击 “Next” 按钮。



cfree 安装



笔记



图 1-4 C-Free 欢迎界面

(2) 按照提示安装完成后，单击“Finish”按钮即可，如图 1-5 所示。



图 1-5 C-Free 安装完成



1.7.3 C-Free 的使用

(1) 双击 C-Free 桌面快捷图标，进入如图 1-6 所示的界面。

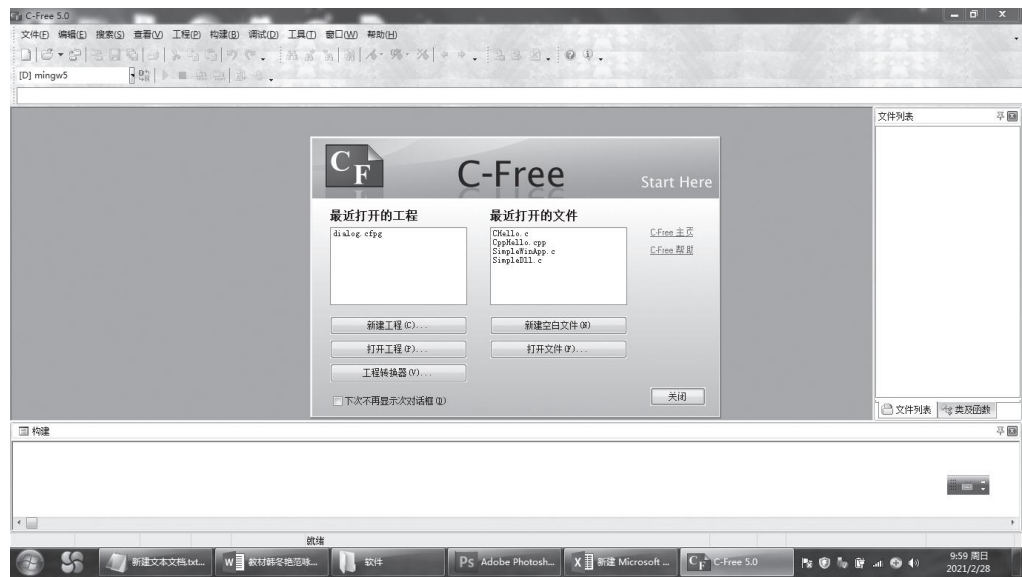


图 1-6 C-Free 启动界面

勾选“下次不要显示此对话框”复选框，并单击“关闭”按钮。

(2) 工程的建立。选择“工程”菜单中的“新建”选项，弹出“新建工程”对话框，如图 1-7 和图 1-8 所示。

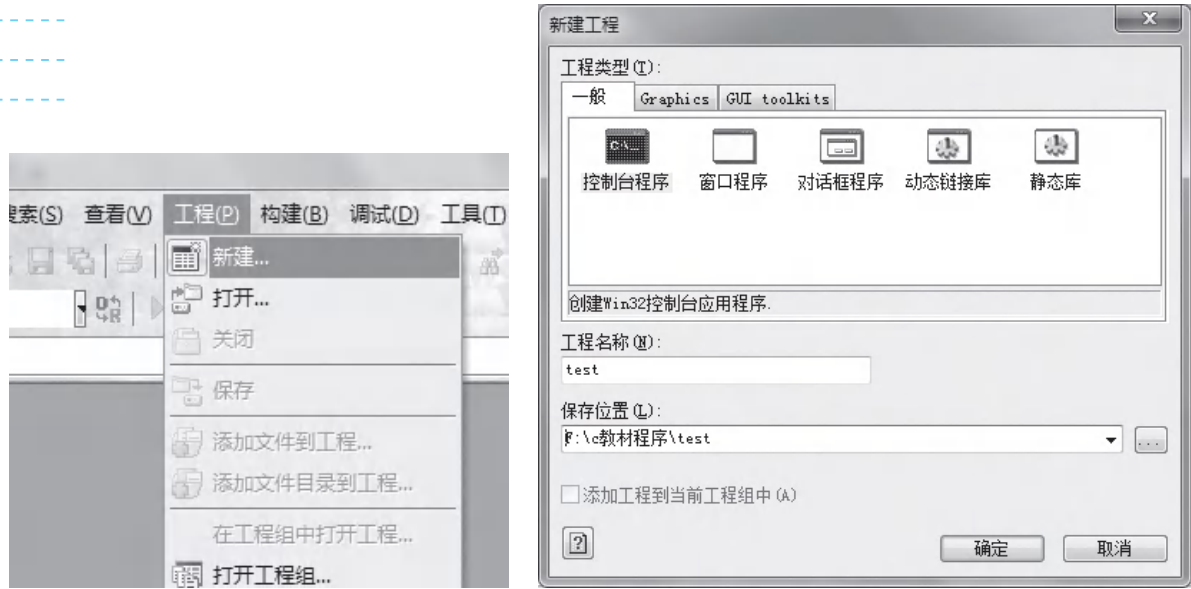


图 1-7 “新建”命令

图 1-8 “新建工程”对话框

在“新建工程”对话框“一般”选项卡中选择“控制台程序”选项，输入工程名称，单击“保存位置”右侧的按钮，选择工程保存位置。设置好保存位置后单击“确定”按钮，弹出“程序类型”对话框，选择“空的程序”，单击“下一步”按钮。在弹出的“语言选择”对话框中，直接单击“下一步”按钮，弹出“构建配置”对话框，单击“完成”按钮

即可建立一个新的工程，如图 1-9 所示。

(3) 工程创建完成后，右侧会显示“文件列表”窗格，如图 1-10 所示，其中 test 为工程名称。如果“文件列表”窗格关闭后还要再显示出来，可以选择“查看”菜单中的“文件列表窗口”选项。



笔记



图 1-9 工程建立完成



图 1-10 “文件列表”窗格

(4) 建立 C 源程序文件。选择“文件”菜单中的“新建”选项，或者按组合键“Ctrl+N”，都可以立即建立一个空白的源程序文件。单击工具栏上的“保存”按钮，或者按组合键“Ctrl+S”，弹出“文件保存”对话框。

在该对话框中的“文件名”文本框中输入程序文件的名称，本处为“demo”；设置“保存类型”为“C 语言文件 (*.c)”，单击“保存”按钮，如图 1-11 所示，弹出“加入工程”对话框。

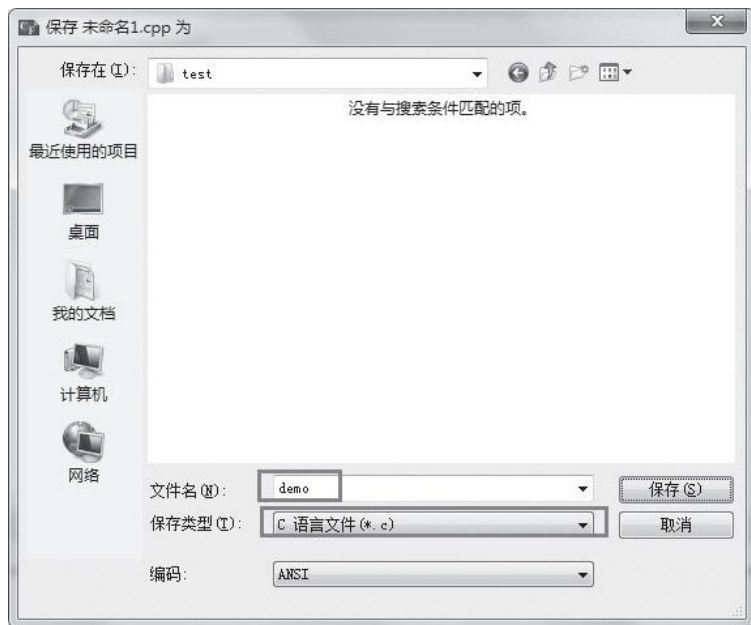


图 1-11 保存新建文件

单击“确定”按钮，弹出“添加到工程中”对话框，如图 1-12 所示，选择“Source Files”文件夹，单击“确定”按钮，即建立一个 C 源程序文件。



注意：在使用“C-Free”软件时，也可以不建立工程，直接建立文件。在第一次保存文件时，直接选择文件存储位置和名称，对于初学者来说是可以的。如果使用 C 语言编写多文件的大程序，以及日后学习 C++ 语言等，就必须建立工程，使用工程来分类管理各种文件。

(5) 编写程序。如图 1-13 所示，在程序文件中编写程序，需要注意字母的大小写和语句后面的分号。

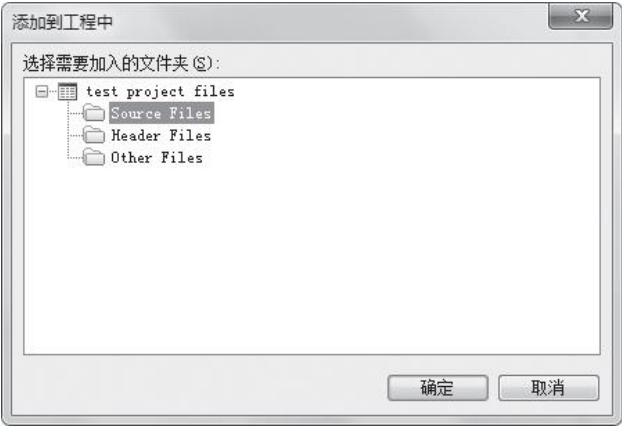


图 1-12 “添加到工程中”对话框

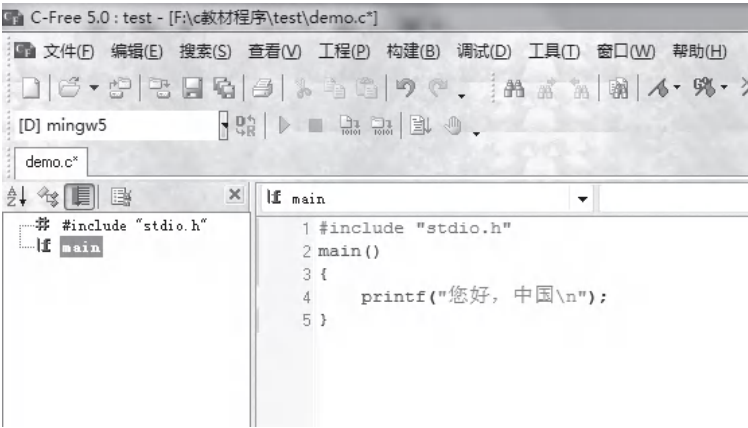
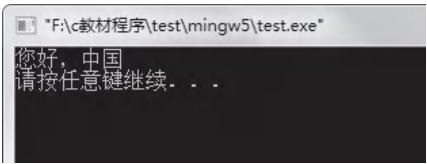


图 1-13 编写程序

(6) 程序的运行。程序写好后，单击工具栏上的绿色三角按钮或按快捷键“F5”都可以编译并执行程序。其程序运行结果如图 1-14 所示。



(7) 程序的调试。在写代码过程中，难免会出现一些语法错误，此时如果执行程序，会出现相应的错误提示。例如，将上面程序第 4 行的“;”删除掉，再次执行程序，下方会有如图 1-15 所示的错误提示。

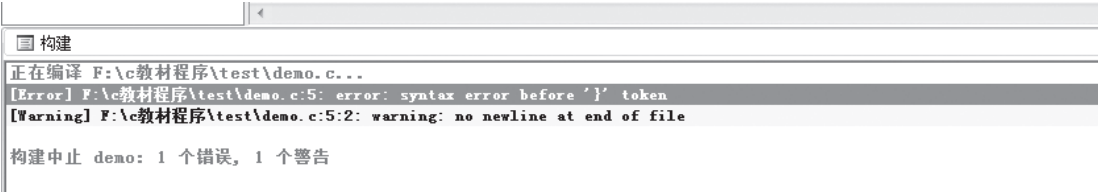


图 1-15 错误提示

提示在第 5 行的上方有错误。大家将其改正后即可正常运行。

项目实训

实训一：输出“hello !”

```
#include<stdio.h>
main()
{
    printf("hello ! \n");
}
```

利用 printf 函数输出英文，其程序运行结果为在屏幕输出“hello!”，光标跳转到下一行。

实训二：求 10 以内两个整数的乘积



```
#include<stdio.h>
main()
{
    int a,b,mult;
    a=3; b=5;
    mult=a*b;
    printf("%d\n",mult);
}
```

这里 a^① 的值是 3，b 的值是 5，mult 的值为 3 和 5 的乘积，结果输出 15。

实训三：输出如下菱形块

```

      *
    * *
  *   *
 *   *
 *
```

分析：要输出用星号组成的菱形块，用 printf 中的双引号括起来就可以原样输出。每输出一行要换行，用“\n”换行。

```
#include<stdio.h>
main()
{
    printf(" * \n");
    printf(" * * \n");
    printf(" *   * \n");
    printf(" *   * \n");
    printf(" *   * \n");
}
```

项目总结

本项目主要介绍了 C 语言的基础知识、C 语言的发展过程、C 语言的格式与构成、C 源程序的结构特点，以及如何运行 C 语言程序和 C-Free 软件的下载、安装及使用方法。

(1) 计算机是由程序控制的，人们为了让计算机工作，必须编写程序指挥计算机工作。程序设计语言有机器语言、汇编语言和高级语言 3 种。

(2) C 语言由一个或多个函数构成，必须有一个主函数 main()，主函数是程序运行的

① 为与程序一致，此处用正体表示，余同。

笔记

入口。函数体内可以包含多条语句，语句以“;”作为结束标志。C 语言书写自由，一行可以写多个语句，一个语句可以分多行写。

(3) 一个 C 程序要经过 4 个步骤才能得到运行结果：编辑源程序、编译源程序、连接目标程序和运行。

(4) C 语言常用的上机环境有 Turbo C/C++、Visual C++、Win-Tc 和 C-Free 等。

C 语言的核心在于函数，每一个函数都可以实现一个小功能，若干个小功能组合起来，就能实现一个大功能，进而构成一个复杂的应用。

天下难事，必作于易；天下大事，必作于细。我们在学习 C 语言时，一定要掌握基本语句的写法、基本结构的应用，多读程序，多实践，多累积各种问题的解决思路及算法。只要我们注重点滴的积累，打牢基础，天长日久，必将汇聚成知识的海洋。

项目考核

一、选择题

1. 一个 C 语言程序是由 () 组成。
A. 主函数 B. 子程序 C. 函数 D. 过程
2. 一个 C 语言程序总是由 () 开始执行的。
A. 主函数 B. 子程序 C. 主程序 D. 主过程
3. 以下说法正确的是 ()。
A. 在 C 程序中，主函数必须位于程序的最前面
B. C 程序的每行只能写一条语句
C. C 语言本身没有输入输出语句
D. 在对一个 C 程序编译时，可发现注释中的拼写错误
4. 以下不属于 C 语言编译环境的是 ()。
A. C-Free B. Turbo C C. UNIX D. Visual C++

二、编程题

1. 编写一个 C 语言程序，输出以下信息。

```
*****  
Welcome to China!  
*****
```

2. 编写程序，实现 3 个整数进行加减运算并输出。
3. 上机运行本项目中涉及的程序。

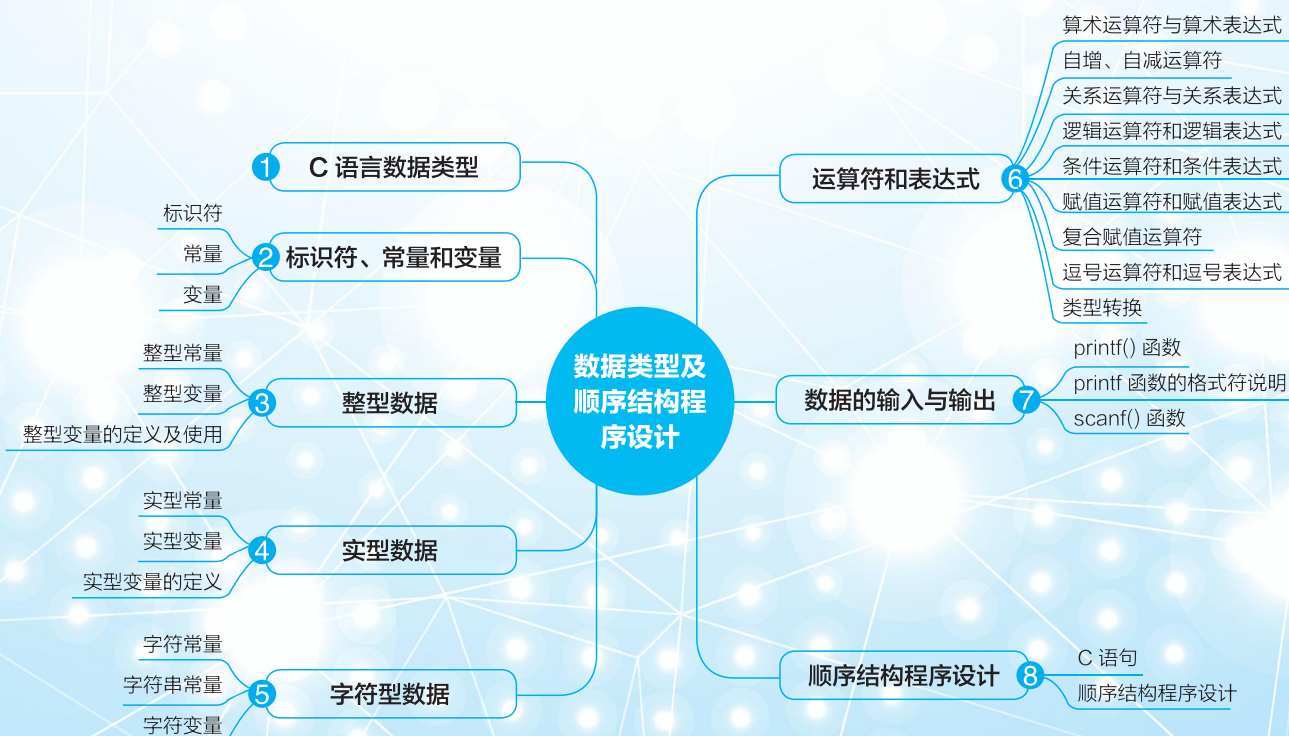
项目 2

数据类型及顺序结构程序设计

项目目标

- 1 掌握 C 语言的基本数据类型及定义方法。
- 2 掌握 C 语言运算符的种类、优先级和结合性。
- 3 学会格式化输入 / 输出函数的使用方法。
- 4 认真体会顺序结构程序设计流程。
- 5 弘扬工匠精神，培养刻苦钻研、精益求精的职业精神。
- 6 培养思考能力，激发创新思维，增强解决问题的能力。

知识导图



笔记

项目描述

数据类型以及顺序结构程序设计部分主要介绍了 C 语言的语法规则、变量的定义与应用。要想学好 C 语言，必须明确 C 语言的运行规则；要想用 C 语言解决问题，必须明确问题的处理思路。同样，也要养成良好的学习习惯，掌握学习方法，才能扎实地学好科技知识。未来走上工作岗位，要从思想上树立大局意识、担当意识，把握所从事工作的发展思路，才能把工作做好，为国家的发展添砖加瓦。

大学毕业生小李参加某网络公司招聘考试，公司研发部经理要求小李编写一组程序，将一些研发数据输出并展示，用于考核。

【分析】

在 C 语言中，所有数据的输入 / 输出都是由库函数和函数语句完成的，思考片刻的小李准备使用函数语句来完成研发数据的输入 / 输出及显示。

相关知识

2.1 C 语言数据类型

数据是程序处理的对象，程序设计的过程就是数据加工和处理的过程。在 C 程序中每一个数据都被赋予一个确定的数据类型，编译程序为不同数据类型分配不同大小的存储空间，并规定了每种类型能进行的运算。C 语言的数据类型可分为基本数据类型、构造数据类型（复合数据类型）、指针类型和空类型四大类。C 语言数据类型如图 2-1 所示。

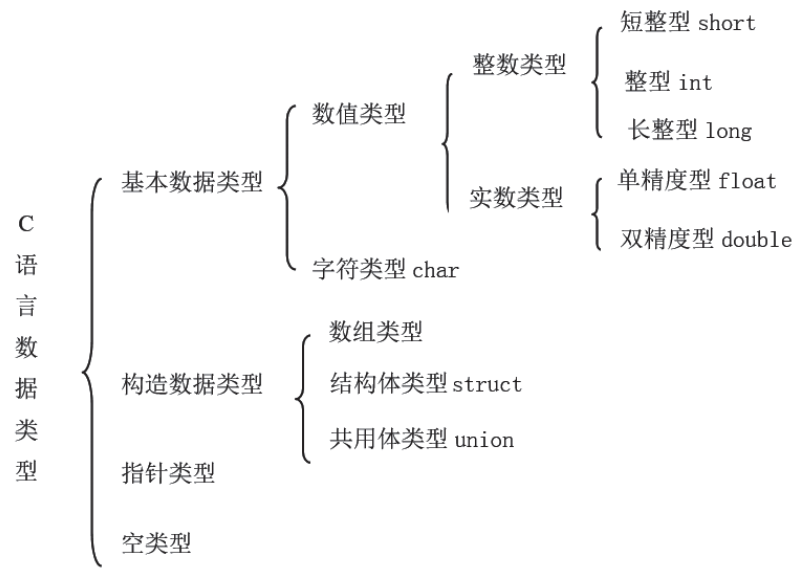


图 2-1 C 语言数据类型

其中，基本数据类型是由系统事先定义好的不可再分割的类型，可以直接利用这些类型定义数据；构造数据类型是由基本数据类型组成的复杂的类型；指针类型是用来表示某个量在内存中的地址；空类型主要用于说明特殊指针变量和无返回值函数。

2.2 标识符、常量和变量



C 语言的数据类型，分为常量和变量。常量又分成普通常量和符号常量，在定义符号常量和变量时，需要定义符号常量和变量的名称，它们的名称要符合标识符的命名规则。

2.2.1 标识符

标识符是用来标识对象（包括符号常量、变量、类型、函数、数组、结构体及文件）名称的有效字符序列。简单地说，标识符就是一个名称。

标识符有三类：关键字、预定义标识符和用户标识符。

（1）关键字。关键字是已被 C 语言编译系统所占用的标识符，每个关键字在 C 语言程序中都有特定的作用，关键字不能作为用户标识符。C 语言的关键字共有 32 个，分为以下几类：

①数据类型定义：typedef。

②数据类型：char、double、enum、float、int、long、short、struct、union、unsigned、void、signed、volatile、auto、extern、register、static、const。

③运算符：sizeof。

④语句：break、case、continue、default、do、else、for、goto、if、return、switch、while。

（2）预定义标识符。C 语言系统提供的库函数名（如 printf、scanf）和编译预处理命令（如 include、define）等构成了预定义标识符。

（3）用户标识符。即用户根据需要自己定义的标识符。一般用来给变量、函数、数组等命名。与人的取名一样，用户定义标识符的命名也有一定的规则：

①标识符由字母、数字和下划线组成，如 x1、ab、sum。

②第一个字符必须是字母或下划线，不能以数字开头。

③字母区分大小写，即 Name、NAME、name 为三个不同的名称。

④不允许使用关键字，如 int、if 等。

⑤标识符的长度不能超过 32 个字符，如果超过，只有前 32 个字符有效（在系统默认状态下）。

【例 2-1】判断下列标识符命名的合法性。

a1	/* 合法 */
abc.c	/* 不合法，点 (.) 不是合法字符 */
a_b	/* 合法 */
a-b	/* 不合法，连接线 (-) 不是下划线 */
abc123	/* 合法 */
3a	/* 不合法，不能以数字开头 */
int	/* 不合法，不能使用关键字 */



2.2.2 常量

在程序运行过程中，其值恒定不变的量称为常量。常量有不同的类型，例如，12、0、-3 是整型常量；4.6、-1.23 是实型常量；'a'、'd'、'A' 是字符常量。

常量也可以用一个名称代表，称为符号常量。符号常量的定义格式如下：

```
#define 符号常量名 常量
```

例如：

```
#define PRICE 30
```

则在后面的程序语句中，PRICE 就是 30。

习惯上符号常量名用大写字母表示。使用符号常量的优点是在需要改变一个常量值时能做到“一改全改”，例如，将 PRICE 改为 40，则在程序中所有 PRICE 的值都自动改为 40。

2.2.3 变量

在程序运行过程中，其值可以被改变的量称为变量。

变量有三个基本属性：

(1) 变量名。变量的命名要符合标识符命名规则。习惯上变量名用小写字母表示，变量命名时要尽量做到“见名知意”，这样有助于记忆，又可提高程序的可读性。

(2) 变量的数据类型。每一个变量都被指定为一个确定的数据类型，在定义时系统会为其分配一定的存储空间，以便在编译时检测该变量所进行的运算是否合法。

(3) 变量的值。变量对应的存储空间中所存放的数值。

在 C 语言中，要求对所有用到的变量做强制定义，也就是“先定义，后使用”。变量的定义和使用将在 2.3.3 节进行介绍。

2.3 整型数据

2.3.1 整型常量

整型常量用来表示整数，C 语言允许使用十进制、八进制和十六进制整型常量。

(1) 十进制整数。十进制整数是由数字 0~9 组成的数字串，多位数时最左第一个数字不能为 0，前面可以有表示正负的符号“+”或“-”，“+”可以省略，如 738、-50^①、0、+39 是合法的整数。

(2) 八进制整数。八进制整数是由数字 0~7 组成的数字串，必须以 0 开头，它是八进制数的前导符号。例如，023、0245、060 是合法的八进制整数，0281、281 是不合法的八

① 为与程序一致，本书负号和减号均采用“-”表示。

进制数。

八进制可以与十进制相互转换：

把八进制数按权展开、相加即得十进制数。

例如， $0137 = 1 \times 8^2 + 3 \times 8^1 + 7 \times 8^0 = 95$ 。

则与八进制数 0137 所对应的十进制数为 95。

若将十进制数 95 转换为八进制数，可以用十进制数除以 8 取余，直到商为 0 为止，将所有余数倒序相连，得到八进制数。

$$95 \div 8 = 11 \cdots 7$$

$$11 \div 8 = 1 \cdots 3$$

$$1 \div 8 = 0 \cdots 1$$

与十进制数 95 所对应的八进制数为 0137。

(3) 十六进制整数。十六进制整数是由数字 0~9 和字母 a~f (或 A ~ F) 组成的符号串，符号串必须以 0x 或 0X 开头，它是十六进制数的前导符号。例如，0x13a、0xc32、0xff、0xb800、0xdc 都是合法的十六进制整数，013a、x13a、13a 是不合法的十六进制数。

2.3.2 整型变量

按占用内存大小的不同，整型变量可以分为如下 4 类：

(1) int：为基本整型，通常在内存中占用 2 字节（16 个二进制位），取值范围为 $-2^{15} \sim (2^{15}-1)$ ，即 -32768~32767。如果数据超过这个范围，则不能准确地表示。在有些 C 语言编译软件中（Visual C++），int 类型占用 4 字节（32 个二进制位），此时取值范围更大。

(2) short int：可以简写为 short，为短整型，在内存中占用 2 字节，取值范围为 $-2^{15} \sim (2^{15}-1)$ ，即 -32768~32767。

(3) long int：可以简写为 long，为长整型，在内存中占用 4 字节，取值范围为 $-2^{31} \sim (2^{31}-1)$ ，即 -2147483648~2147483647。

(4) unsigned int：可以简写为 unsigned，为无符号整型，在内存中占用 2 字节，只能表示正数，取值范围为 $0 \sim (2^{16}-1)$ ，即 0~65535。

另外还有 unsigned short 和 unsigned long，即无符号短整型和无符号长整型。

2.3.3 整型变量的定义及使用

每个变量在使用前都必须先定义，否则在编译过程中，程序就会报错。变量在内存中占一定的存储单元，例如，定义一个变量 a，内存中就给 a 分配一个空间，用来存储数据，如图 2-2 所示。

变量的一般定义格式如下：

类型名 变量名 1, 变量名 2, ..., 变量名 n;

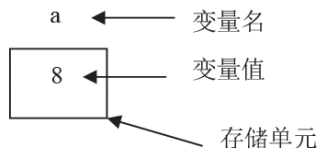


图 2-2 定义变量



笔记



例如：

```
int a;
long c;
unsigned d;
int a,b,c; /* 定义多个变量时，变量之间用“,”分隔 */
```

定义变量后，就可以使用了。给变量赋初值有以下两种方式。

一是通过赋值运算符“=”赋值，其一般格式如下：

```
变量名 = 表达式；
```

例如：

```
int x; x=6;
int a,b,c; a=6+3;b=7;c=8;
```

二是在定义变量时给出初值，称为初始化。例如：

```
int a=6; /* 在定义变量同时，给变量初始化 */
int x,y=2,z=6;
```

变量和常量不同的是，变量在整个程序运行过程中，其值可以改变。

例如，下列程序段：

```
int a=5,b=6;      /*b 的值为 6*/
b=a+b;            /*b 的值为 11*/
b=a+b;            /*b 的值为 16*/
```

这里的 b 从开始的 6，到执行 `b=a+b`，最后变成 11。第二次执行 `b=a+b` 时，赋值运算符“=”右侧的 b 是 11，再加上 a 后变成 16，然后将 16 赋给赋值运算符“=”左侧的 b，使 b 的值变为 16。

2.4 实型数据

2.4.1 实型常量

实型常量即实数，在 C 语言中又称为浮点数，其值有以下两种表达形式：

- (1) 十进制形式：由正负号、小数点、数字 0~9 组成。例如，3.14、9.8 等。
- (2) 十进制指数形式：由数字、E（或 e）和整数组成。例如，123e3、123.4E2 等。

注意：E 后面必须有整数，E 前面必须要有数字，如 0.2E、0.2E2.3、e-5 不合法。

实型常量不分 float 型和 double 型。一个实型常量，可以赋给一个实型变量（float 或

double)。



2.4.2 实型变量

根据占用内存空间的不同，实型变量可以分为单精度型 float、双精度型 double 和长双精度型 long double 三种。单精度型在内存中占用 4 字节（32 个二进制位），可以表示的数据范围为 $-3.4 \times 10^{-38} \sim 3.4 \times 10^{38}$ ，单精度型可以精确到小数点后 6~7 位；双精度型在内存中占用 8 字节（64 个二进制位），可以表示的数据范围为 $-1.7 \times 10^{-308} \sim 1.7 \times 10^{308}$ ，双精度型可以精确到小数点后 15~16 位。

2.4.3 实型变量的定义

定义实型变量的方法与整型变量类似。对每一个实型变量都应在使用前加以定义。例如：

```
float x,y;
double z;
long double t;
```

2.5 字符型数据

2.5.1 字符常量

C 语言的字符常量是用一对单引号括起来的一个字符，例如，'A'、'1'、'\$' 等都是字符常量。这里要注意，英文字母区分大小写的，'a' 和 'A' 是不同的字符常量。

一个字符型常量代表 ASCII 码字符集中的一个字符。字符数据在内存中占用一个字节，存放该字符相应的 ASCII 码值。例如，'A' 的 ASCII 值为 65、'a' 的 ASCII 值为 97、'0' 的 ASCII 值为 48。所以一个字符常量可以像整数一样参与一些运算，如加法、减法等。常用字符与 ASCII 代码对照见附录 A。

除上述字符常量外，C 语言中还允许一种特殊字符常量，即以反斜杠“\”开头的字符序列，称为转义字符，是将“\”后面的字符转变成另外的意义。转义字符占 1 个字节，算 1 个字符，它们一般起控制作用。常见的转义字符及其含义如表 2-1 所示。

表 2-1 转义字符及其含义

转义字符	含义
\n	换行，光标由当前位置移到下一行开头
\t	横向跳格，跳到下一个输出区，每区占 8 列
\v	竖向跳格

(续表)

笔记

转义字符	含义
\b	退格，光标回退一个位置
\r	回车，光标由当前位置移到本行开头处
\\	反斜杠字符 “\”
\'	单引号字符 “'”
\"	双引号字符 “”
\0	空字符
\ddd	1~3 位八进制数，例如，'\101' 表示字母 A
\xhh	1 位和 2 位十六进制数，例如，'\x41' 也表示字母 A

2.5.2 字符串常量

字符串常量是一对双引号括起来的字符序列，例如，"How do you do"。

注意：不要混淆字符常量与字符串常量。

- (1) "a" 与 'a' 的区别："a" 分配 2 个字节，存放字母 a 和字符串结束符 "\0"，而 'a' 只分配一个字节。
- (2) 字符串常量不能赋给一个字符变量，例如，char c1;c1="a"; 是错误的。

2.5.3 字符变量

字符变量使用关键字 char 来声明，占用 1 个字节的内存空间。字符变量用来存储字符常量，一个字符型数据，既可以使用字符形式输出，也可以使用整数形式输出。

```
char ch1, ch2, ch3;
ch1='a'; ch2='A'; /* 给字符变量 ch1、ch2 赋值 */
ch3=98;           /*ch3 为 98，表示存放的字符对应的 ASCII 码值是 98，
                  即字母 b*/
```

2.6 运算符和表达式

在 C 语言中，有很丰富的运算符，如下所示：

- 算术运算符：+、-、*、/、%
- 自增 1，自减 1 运算符：++、--
- 关系运算符：<、<=、==、>、>=、!=
- 逻辑运算符：!、&&、||



自增自减运算符

位运算符：<<、>>、~、|、&、^

条件运算符：?:

指向结构体成员运算符：->

结构体成员运算符：.

赋值运算符：=、+=、-=、*=、/=、%=、^=、!=

逗号运算符：,

指针运算符：*、&

求字节数：sizeof()

强制类型转换：(类型)

下标运算符：[]

其他：() -

现将部分常用运算符分类介绍如下。



2.6.1 算术运算符与算术表达式

算术运算符有 5 种，分别为加“+”、减“-”、乘“*”、除“/”、取余数“%”。在使用基本算术运算符时要注意以下几点：

(1) 两个整数相除，结果为整型，如果其中一个为实型，则结果为 double 型。例如， $5/3=1$ 、 $3/5=0$ 、 $3.0/5=0.6$ 。

(2) % 运算符只适用于整数（表示两个整数求余数）。例如， $5\%3=2$ 、 $3\%5=3$ 、 $8\%4=0$ 、 $-7\%4=-3$ 、 $7\%(-4)=3$ 是合法的；而 $5.2\%3$ 不合法。

用算术运算符、括号，把数值型常量、变量等连接起来符合 C 语法规则的式子称为算术表达式。例如：

```
a*b/c+d-e;
```

C 语言规定了运算符的优先级和结合性。在求表达式的值时，先按运算符的优先级别高低次序执行，当进行混合运算时先计算乘除，后计算加减。例如，表达式 $a+b*c$ 有两个运算符“+”和“*”，先算乘法 $b*c$ ，再算 a 加上 $b*c$ 的积。

如果两个运算符的优先级别相同，如 $a*b/c$ ，则按规定的结合方向处理。C 语言规定了各种运算符的结合方向（结合性），算术运算符的结合性为“自左至右”， $a*b/c$ 里面的乘法和除法按从左到右的顺序运算，即先算 $a*b$ ，再把 $a*b$ 的积除以 c 。

C 语言各种运算符的优先级和结合性见附录 C。

2.6.2 自增、自减运算符

自增、自减运算符有两种，分别为 ++ 和 --。它们只能作用于整型变量，在使用时既可以放在变量的前方（左侧），也可以放在变量的后方（右侧）。例如， $i++$ 、 $++i$ 、 $i--$ 、 $--i$ ，它们都能给变量 i 增 1 或减 1。

$x=i++$; 相当于 $x=i$; $i=i+1$;



笔记

$x=++i$; 相当于 $i=i+1$; $x=i$;

即“ $++$ ”是先使用变量 i 的原值，然后再使 i 增 1，“ $++i$ ”是先使变量 i 增 1，然后再使用 i 的值。 $--$ 的用法及含义与 $++$ 相同。

【例 2-2】自增、自减运算符的应用。

```
#include<stdio.h>
main()
{
    int i,j,k;
    i=5;
    j=i++;
    printf("i=%d,j=%d\n",i,j);
    j=++i;
    printf("i=%d,j=%d\n",i,j);
    i=5;
    j=--i;
    printf("i=%d,j=%d\n",i,j);
    j=i--;
    printf("i=%d,j=%d\n",i,j);
}
```

其程序运行结果如下：

```
i=6, j=5
i=7, j=7
i=4, j=4
i=3, j=4
```

自增、自减运算符的优先级高于基本算术运算符。“ $++$ ”“ $--$ ”运算符是自右至左结合的。例如， $-n++$ 表示 $-(n++)$ ，而不是 $(-n)++$ 。

自增、自减运算符参与运算的变量只能是整型、字符型和指针型变量。自增和自减运算符更多用于循环语句中，使控制循环的变量加 1 或减 1。此外，也常用于指针变量中，使指针上移或下移一个位置。

2.6.3 关系运算符与关系表达式

关系运算就是比较运算，将两个值进行比较，判断比较的结果是否符合给定的条件。用关系运算符将两个表达式连接起来的式子称为关系表达式。C 语言有 6 种关系运算符：小于“ $<$ ”、小于等于“ $<=$ ”、大于“ $>$ ”、大于等于“ $>=$ ”、等于“ $==$ ”、不等于“ $!=$ ”。

关系运算符中 $<$ 、 $<=$ 、 $>$ 、 $>=$ 的优先级高于 $==$ 、 $!=$ ，关系运算符为左结合性。

关系表达式的值是一个逻辑值，即“真”或“假”。在 C 语言中，没有专门的逻辑类型，作为关系运算的结果，以数字“1”表示逻辑真，以数字“0”表示逻辑假。例如， $a=3$ 、 $b=4$ 、 $c=5$ ，则：

$a>b$

$(a>b)!=c$

$a<b<c$

$d=a>b==b>c$

值为 0

值为 1

值为 1

值为 1

在 C 语言中，一个等号“=”是赋值运算符，两个等号连写“==”是等于运算符。例如， $a=3+2$ ，是将赋值号右侧表达式的值 5 赋值给左侧变量，执行后 a 的值为 5； $a==3+2$ ，是判断 a 的值是否和 5 相等。如果 a 的值是 5，那么这个表达式是“真”；否则是“假”的。

2.6.4 逻辑运算符和逻辑表达式

C 语言的逻辑运算符有逻辑非“!”、逻辑与“&&”、逻辑或“||”，其优先级从左向右依次降低。其中，! 具有右结合性，&& 和 || 具有左结合性。如表 2-2 所示。

表 2-2 逻辑运算符的运算规则

运算符	运算对象 1	运算对象 2	运算结果
与运算符 &&	真 (非 0)	真 (非 0)	真 (1)
	真 (非 0)	假 (0)	假 (0)
	假 (0)	真 (非 0)	假 (0)
	假 (0)	假 (0)	假 (0)
或运算符 	真 (非 0)	真 (非 0)	真 (1)
	真 (非 0)	假 (0)	真 (1)
	假 (0)	真 (非 0)	真 (1)
	假 (0)	假 (0)	假 (0)
非运算符 !(取反)	真 (非 0)		假 (0)
	假 (0)		真 (1)

作为运算结果，逻辑表达式也只有“真”和“假”两个值，用 1 表示“真”，用 0 表示“假”。在逻辑表达式的求解中，并不是所有的逻辑运算符都被执行，只有在必须执行下一个逻辑运算符才能求出表达式的解时，才执行该运算符。例如， $a\&\&b\&\&c$ 只有当 a 为“真”时才判断 b 的值，只有当 a 和 b 都为“真”时才判断 c 的值。如果 a 为“假”，则不必判断 b 和 c，后面两个 && 运算没有执行。 $a||b||c$ 只有当 a 为“假”时才判断 b 的值，只有当 a 和 b 都为“假”时才判断 c 的值。如果 a 为“真”，则不必判断 b 和 c，后面两个 || 运算没有执行。

例如，设“ $a=12,b=8;$ ”，则：

(1) $!a\&\&b$ ：a 为 12，非 0 值，a 结果为逻辑真，“!a”为假，&& 运算不必判断 b 值



笔记 

的真假，表达式值为假，值为 0。

(2) $a \geq 1 \&\& b \leq 31$: $a \geq 1$ 为真，则继续判断 $b \leq 31$ ，也为真，结果为真，表达式值为 1。

(3) $a \parallel b > 31$: a 为 12, a 为真, $\parallel$ 运算不必判断 $b > 31$ ，结果为真，表达式值为 1。

(4) $a < -1 \parallel b < 12$: $a < -1$ 为假，则继续判断 $b < 12$ ，结果为真，表达式值为 1。

2.6.5 条件运算符和条件表达式

$?:$ 是条件运算符，由条件运算符将一些常量、变量、表达式连接起来构成的式子为条件表达式，条件表达式的一般格式如下：

表达式 1 ? 表达式 2 : 表达式 3

条件表达式的求解过程为：先求解“表达式 1”的值，如果“表达式 1”的值为“真”（非 0），求解“表达式 2”的值，并将表达式 2 的值作为整个条件表达式的值；如果“表达式 1”的值为“假”（为 0），求解“表达式 3”的值，并将表达式 3 的值作为整个条件表达式的值。例如： $8 > 5 ? (3+2) : (9+6)$ ，因为 8 大于 5 成立，其值为真，所以整个条件表达式的值为 $(3+2)$ 的值 5。

要注意条件表达式中的表达式 2 和表达式 3 只执行一个，当与 $x++$ 、 $x--$ 配合时，只有执行到的 $x++$ 中 x 才会变化。

2.6.6 赋值运算符和赋值表达式

C 语言中“ $=$ ”称为赋值运算符，它的作用是将赋值号右侧表达式的值赋值给左侧的变量。赋值运算符的一般格式为：变量 = 赋值表达式，例如： $x = 5$ 、 $y = 5.0/2$ 等。在使用时如果左侧变量的类型与右侧结果的类型不一致，系统会将结果的类型转换成被赋值变量的类型，然后进行赋值操作。

由赋值运算符，将一个变量和一个表达式连接起来的式子，称为赋值表达式。

任何表达式都有值，赋值表达式也不例外。赋值号右侧表达式的值，就是整个赋值表达式的值。例如，“ $a = 5+2$ ”这个赋值表达式，赋值号（ $=$ ）右侧表达式（ $5+2$ ）的结果值为 7，所以赋值表达式的值就是 7。

赋值运算符为右结合性，即从右向左计算。

2.6.7 复合赋值运算符

为简化代码书写的工作量，在 C 语言中还有五种复合运算符： $+=$ 、 $-=$ 、 $*=$ 、 $/=$ 、 $\%=$ 。它们的具体含义如下：

$x += 3$ 等价于 $x = x + 3$

$y *= x + 6$ 等价于 $y = y * (x + 6)$ ，而不是 $y = y * x + 6$

2.6.8 逗号运算符和逗号表达式



C 语言提供一种用逗号运算符 “,” 连接起来的式子，称为逗号表达式。逗号运算符又称为顺序求值运算符。逗号表达式的一般格式如下：

```
表达式 1, 表达式 2, ..., 表达式 n
```

逗号表达式的求解过程如下：自左至右，依次计算各表达式的值，“表达式 n” 的值即整个逗号表达式的值。

【例 2-3】求逗号表达式 (a=3*5,a*4),a+5 的值。

分析：本例逗号表达式中，表达式 1 为 (a=3*5,a*4)，表达式 2 为 a+5。

(1) 求解表达式 1: (a=3*5,a*4)，表达式 1 也是一个逗号表达式，a=3*5，a 的值为 15，则左侧表达式值为 15；右侧 a*4，值为 60，也即表达式 1 值为 60。

(2) 求解表达式 2: a+5，表达式 1 中 a=15，则 a+5 结果为 20。

整个逗号表达式的值为表达式 2 的值，即 20。

注意：并不是任何地方出现的逗号，都是逗号运算符。很多情况下，逗号仅用作分隔符。

2.6.9 类型转换

当不同类型的数据进行混合运算、跨类型赋值时，通常需要进行类型转换。数值型数据的级别及运算时的类型转换如图 2-3 所示。

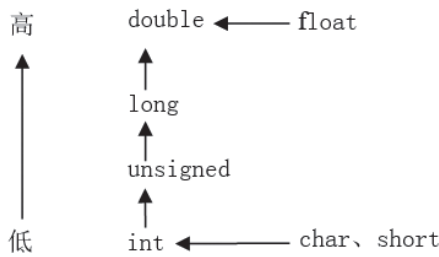


图 2-3 数值型数据的级别及运算时的类型转换

图 2-3 中，横向箭头表示必定的转换，如字符数据 char 必定先转换为整数 int、短整型数据 short 先转换为 int 型、float 型数据在运算时先转换成双精度型 double，以提高运算精度。即使两个 float 型数据相加，也都先转换成 double 型，再进行相加运算。

纵向箭头表示当运算对象为不同类型时转换的方向。例如，int 型与 double 型运算，先将 int 型数据转换为 double 型数据，然后再运算，结果为 double 型。

注意：箭头方向只表示数据类型级别的高低，图 2-3 中 int 型是直接转换成 double 型，而不是 int 型先转换为 unsigned，再转换为 long，最后转换为 double 型。

(1) 两个同类型数的运算结果为原类型。

例如：28+32 的结果为 60；1/2=0 的结果为 0。参与运算的数据都为整型 int，所以结果也为整型 int。

笔记

(2) 两个不同类型数运算时, 系统自动将低级类型数据转换为高级类型数据, 而后进行同类型数据运算, 结果为高级类型数据。

例如: $28+32.5$ 的结果为 60.5 。其中, 28 为整型, 32.5 为 `double` 型 (`float` 型在参与运算时, 无条件变为 `double` 型), 所以 `int` 型的 28 转换为 `double` 型的 28.0 , 然后两个 `double` 型的数据进行运算, 结果为 `double` 型。

$28+'A'$ 的结果为 93 。其中, 28 为整型, `'A'` 为字符型。字符型在参与数值运算时是无条件转换为 `int` 型的, 其值为该字符所对应的 ASCII 码。`'A'` 的 ASCII 值为 65 , 所以本处为 $28+65$, 结果为 93 。

(3) 当赋值运算时, 将右侧表达式结果值的类型转换成左侧变量的类型。

例如:

```
int a;
a=28+32.5;
```

运行结果是 `a` 为 60 。

其中, $28+32.5$ 的结果为 `double` 型 60.5 , 这里变量 `a` 为 `int` 型, 所以系统自动将 `double` 型转换为 `int` 型, 即对 60.5 进行取整, 结果为 60 , 然后将数据 60 赋值给变量 `a`。

(4) 强制类型转换。类型的强制转换是在一个表达式前加一个结果类型的运算符, 将表达式的类型强制转换为所指定的类型。其一般格式如下:

(类型说明符) 表达式

表示将表达式的类型强制转换为 “()” 中所指定的 “类型说明符” 的类型。例如:

```
int a; (double)(a+5);
```

`a` 是 `int` 型变量, 表达式 $a+5$ 的值仍然是 `int` 型的, 通过强制转换后, 将表达式 $a+5$ 的类型转换成为 `double` 型。

这种强制转换并不改变原来变量 (如 `a` 仍是 `int` 型) 的类型, 只是暂时的一次性转换。

2.7 数据的输入与输出

C 语言本身不提供输入 / 输出语句, 输入和输出是由函数来实现的, 这些函数的有关信息都存在 `stdio.h` 标准输入输出头文件中, 在使用标准输入输出函数时, 需要使用预编译命令: `#include<stdio.h>` 或 `#include"stdio.h"`, 编译预处理内容详见项目 10, 具体头文件内容详见附录 E。

2.7.1 printf() 函数

`printf()` 函数是格式化输出函数, 一般用于向标准输出设备 (如显示器、打印机等) 按规定格式输出信息。格式化输出函数的格式如下:



格式化输出函数

```
printf(" 格式控制 ", 输出表列 )
```



例如：

```
printf("hello,%c\n",x);
```

其括号内包括以下两部分：

（1）格式控制是用双引号括起来的一个字符串，称为“转换控制字符串”，简称“格式字符串”。它包括两种信息：

①格式声明。格式声明由“%”和格式字符串组成，如 %d、%c 等。它的作用是将输出的数据转换为指定的格式输出。

②普通字符。普通字符即需要原样输出的字符。上面例子中的“hello”，与换行符要原样输出。

常用的格式化字符如表 2-3 所示。

表 2-3 格式化字符

符 号	作 用
%d	十进制有符号整数
%o(是 “字母 o”，不是 “数字 0”)	无符号以八进制表示的整数
%X	无符号以十六进制表示的整数
%f	浮点数
%c	单个字符
%e	指数格式的浮点数
%ld	长整型
%lf	double 浮点数

（2）输出表列是需要输出的一些数据，可以是常量、变量或表达式。一般来说，前面有几个 % 格式声明，后面就要对应几个输出项。

例如：

```
int a=3,b=4;
printf("a=%d b=%d",a,b);
```

这里两个 %d 是格式声明，a,b 是输出表列，即把 a 的值放在第 1 个 %d 的位置，把 b 的值放在第 2 个 %d 的位置。“a=,b=” 是普通字符，原样输出。

printf 输出结果是 a=3 b=4。

【例 2-4】读程序写结果。

```
#include <stdio.h>
main( )
```

笔记

```
{  
    int a,b,sum;  
    a=123;b=456; sum=a+b;  
    printf("a=%d,b=%d,c=%d\n",a,b,sum);  
}
```

其程序运行结果如下：

```
a=123,b=456,c=579
```

2.7.2 printf 函数的格式符说明

可以在“%”和字母之间加数字表示最大场宽，下面是几种常用的格式：

(1) %md：表示输出的整数宽度是 m 位，右对齐，不够 m 位左侧补空格。如用 %5d 格式输出 123，输出结果为□□ 123。“□”表示空格。

(2) %-md：负号表示左对齐，不够 m 位右侧补空格。如用 %-5d 格式输出 123，输出结果为 123 □□。

(3) %m.nf：输出整数、小数和小数点部分宽度总计占 m 位，n 为小数位数，不足 m 位左侧补空格，小数要进行四舍五入。

例如：用 %8.2f 格式输出 123.456，%8.2f 表示输出宽度是 8 位，其中小数部分是 2 位，小数部分 0.456 保留 2 位，进行四舍五入为 0.46，小数部分输出 46，整数部分占 5 位（%8.2f 输出宽度是 8 位，除掉小数部分 2 位，小数点占 1 位），123 整数占 3 位，左侧补 2 个空格，输出结果为□□ 123.46。

注意：小数点占 1 位宽度。

如果字符串的长度或整型数位超过说明的场宽，将按实际整数位输出。但对浮点数，若整数部分位数超过了说明的整数位宽度，将按实际整数位输出；若小数部分位数超过了说明的小数位宽度，则按说明的宽度进行四舍五入输出。

若想在输出前加一些 0，就应在场宽项前加个 0。例如，%04d 表示在输出一个小于 4 位的数值时，将在前面补 0，使其总宽度为 4。

2.7.3 scanf() 函数

scanf() 函数是格式化输入函数，它从标准输入设备（如键盘、鼠标、扫描仪等）读取输入的信息。格式化输入函数的一般格式如下：

```
scanf (" 格式控制 " , 地址表列 )
```

(1) 格式控制与 printf() 函数中的格式说明基本相同。

(2) 地址表列：由若干个地址组成的表列，可以是变量的地址或字符串的首地址，多个地址用逗号隔开。



格式化输入函数

(3) 变量地址组成: 地址运算符 “&” 后加变量名, 多个变量用 “,” 隔开。

例如:

```
scanf("%d%d",&a,&b);
```

表示从键盘上输入 2 个整数分别赋给变量 a 和变量 b。

注意以下几点:

(1) 在输入多个数值数据时, 若格式控制串中没有非格式字符作为数据间隔符, 如 “%d%d” 格式, 则可用空格键、Tab 键或回车键做间隔。

(2) 非格式控制字符的普通字符要原样对应输入, 如逗号 “,” “a=b=” 等要原样输入。

(3) scanf 地址表列中应为变量地址加变量名, 而不是直接写变量名。如 scanf("%d,%d",a,b); 是错误的。要写成 scanf("%d,%d",&a,&b); 才对。

(4) 输入数值数据时, 前导空格不计。

(5) 用 %c 格式声明输入字符时, 空格、回车和转义字符 (如 “\n”) 都作为有效字符输入。

(6) 一个或多个空格、一个或多个 Tab、一个或多个回车均等效。

【例 2-5】要使整型变量 a 的值为 123, 整型变量 b 的值为 456, 有如下输入语句, 则应该如何输入数值?

(1) scanf("%d%d",&a,&b);

输入: 123 空格 456 ↵

其中, “↵” 表示回车。当输入完数据后, 要用回车结束输入。

(2) scanf("%d,%d",&a,&b);

输入: 123,456 ↵

(3) scanf("a=%d,b=%d",&a,&b);

输入: a=123,b=456 ↵

2.8 顺序结构程序设计

2.8.1 C 语句

前面介绍的常量、变量、运算符、表达式等是编写程序过程中用到的一些基础知识, 它们是构成程序的基本组成部分。

C 程序对数据的处理是通过 “语句” 的执行来实现的。一条语句完成一项操作 (或功能), 一个为实现特定目的的程序应包含若干条语句。语句说明了一种行为, 它是用计算语言编写的控制计算机完成确定操作的指令。一条语句的结束要以 “;” 结尾, 在 C 语言中, “;” 标志着一行指令语句的结束。

C 语句分为以下 5 类:



笔记



1. 表达式语句

表达式语句由表达式加上 “;” 组成。

其一般格式如下：

```
表达式;
```

执行表达式语句就是计算表达式的值。

例如：

```
x=y+z;a=520;
```

2. 函数调用语句

由函数名、实际参数加上 “;” 组成。

其一般格式如下：

```
函数名 ( 实际参数表 );
```

执行函数语句就是调用函数体并把实际参数赋给函数定义中的形式参数，然后执行被调函数体中的语句，求函数值。

例如：

```
printf("%d,%d,%c,%c\n",a,b,x,y);
```

3. 控制语句

控制语句用于控制程序的流程，以实现程序的各种结构方式。

它们由特定的语句定义符组成。C 语言有 9 种控制语句。可分成以下 3 类：

(1) 条件判断语句：if、switch 语句。

(2) 循环执行语句：do while、while、for 语句。

(3) 转向语句：break、goto（此语句尽量少用，因为这不利于结构化程序设计，滥用它会使程序流程无规律、可读性差）、continue 和 return 语句。

4. 复合语句

把多个语句用 “{ }” 括起来组成的一个语句称为复合语句。在程序中应把复合语句看成单条语句，而不是多条语句，例如：

```
{  
    x=y+z;  
    a=b+c;  
    printf("%d%d",x,a);  
}
```

这是一条复合语句。复合语句内的各条语句都必须以 “;” 结尾；此外，在 “}” 外

不能加分号。

5. 空语句

只有一个“;”组成的语句称为空语句。空语句的作用是什么也不执行。在程序中空语句可用来做空循环体。

例如：

```
while(getchar()!="\n");
```

本语句是只要从键盘输入的字符不是回车就重新输入。这里的循环体为空语句。

2.8.2 顺序结构程序设计

C 语言是结构化程序设计语言，结构化程序设计是用顺序结构、选择结构和循环结构来构造程序。大家前面接触到的程序都是顺序结构的，顺序结构是最简单的程序结构，也是最常用的程序结构，只需按照解决问题的顺序写出相应的语句，它是根据程序中语句的书写顺序依次执行的语句序列，不发生流程的跳转。

项目实训

实训一：整型和字符型数据的输入 / 输出

```
#include<stdio.h>
main()
{
    int a,b;
    char x,y;
    scanf("%d%d",&a,&b);
    scanf("%c%c",&x,&y);
    printf("%d,%d,%c,%c\n",a,b,x,y);
}
```

输入：32 空格 56 ↵ cd ↵

输出：

```
32 56
c
```

即 a 的值为 32，b 的值为 56，x 的值为回车，y 的值为 c。

这里本意是要输入 32 56 赋给 a,b，回车结束第一个 scanf() 函数，输入 cd 赋给 x,y。可这里回车又作为字符型数据赋给下一个输入函数的 x 变量。y 变量则取 c 值。

当两个 scanf 函数与两个输入行一一对应且第 2 行首列为字符时，应在第 2 个 scanf 函数的格式控制串开头设一个空格以抵消上一行末尾的回车。上例改写如下：



笔记

笔记

```
#include <stdio.h>
main()
{
    int a,b;
    char x,y;
    scanf("%d%d",&a,&b);
    scanf(" %c%c",&x,&y);/* 注意 %c 前有一个空格 */
    printf("%d,%d,%c,%c\n",a,b,x,y);
}
```

输入：32 空格 56 ↵ cd ↵

输出：32,56,c,d

通过第 2 个 scanf 中 %c 前加一个空格，实现了 a 为 32，b 为 56，x 为 c，y 为 d。

实训二：求三角形的面积

输入三角形的三边，用海伦公式求三角形的面积。

海伦公式：假设三角形的三条边长依次为 a、b、c，设 $t=(a+b+c)/2$ ，则三角形面积 $s=\sqrt{t(t-a)(t-b)(t-c)}$ 。

```
#include <math.h>
#include<stdio.h>
main()
{
    float a,b,c,t,s;
    scanf("%f,%f,%f",&a,&b,&c);
    t=1.0/2*(a+b+c);
    s=sqrt(t*(t-a)*(t-b)*(t-c));
    printf("a=%f,b=%f,c=%f,s=%.3f\n",a,b,c,s); /*.3f 是结果保留 3 位小数 */
}
```

输入：9.89,12.65,8.76

输出：a=9.890000,b=12.650000,c=8.760000,s=43.166

程序中求平方根是调用库函数的 sqrt() 函数来完成，调用方法为在程序的开头加 #include<math.h>。#include<math.h> 指包含 math 头文件，“.h”是头文件（header file）的扩展名，这一句声明了本程序要用到标准库函数中的 math.h 文件。math.h 头文件中声明了常用的一些数学运算函数，如乘方、开方等，详见附录 E。

实训三：大小写字母转换

从键盘输入一个大写字母，要求改用小写字母输出。

分析：存储字母所用的是 ASCII 代码，'A' 的 ASCII 码为 65，'a' 的 ASCII 为 97，'B' 的 ASCII 为 66，'b' 的 ASCII 为 98，从附录 A 可以看到一个大写字母的 ASCII 码比它的

小写字母的 ASCII 码小 32。

```
#include<stdio.h>
main()
{
    char c1,c2;
    scanf("%c",&c1);
    c2=c1+32;
    printf("c1=%c c2=%c\n",c1,c2);
}
```

输入：F ✓

输出：c1=F c2=f

项目总结

本项目主要介绍了 C 语言程序设计的基础知识，包括数据类型、运算符、表达式和输入 / 输出函数，并将顺序结构程序设计流程并入此项目，为下一步深入学习 C 语言奠定了基础。读者需要多编写程序以加深理解。

(1) C 语言所处理的数据都是属于一定类型的，不同类型的数据在计算机中所占的空间大小和存储方式是不同的，变量要先定义后使用。

(2) C 语言中一个等号 “=” 是赋值运算符，如 $a=5$; $a=a+3$ ；含义是把 5+3 的结果 8 赋给 a 变量，a 的值是 8，表达式的值也是 8；两个等号 “==” 是等于运算符，如 $a==5+3$ ，是判断 a 的值是否和 5+3 的值即 8 相等，相等则表达式的值是“真”值，不相等则表达式的值是“假”值。

(3) C 语言中的数据按不同优先级从高到低运算，同一优先级的运算符，运算次序由结合方向决定。

(4) C 语言中所有输入与输出都是调用标准库函数中的输入 / 输出函数来实现的。scanf 和 printf 是格式化输入 / 输出函数，可以处理规定格式的数据，如 %d 为整型数据。getchar 和 putchar 是字符输入 / 输出函数，只能输入或输出单个字符，此部分内容将在项目 5 进行介绍。

(5) 编写顺序结构的程序一般步骤如下：先看程序需要引入哪些库函数，如要用 printf 函数和 scanf 函数引入 `#include<stdio.h>`，需要用数学函数时引入 `#include<math.h>`，库函数写在主函数前面，函数体内部先定义需要处理的所有变量，根据不同数据类型定义相应的变量，同一类型可以放一起定义，如 `int a,b; float c;` 变量的初始值可以赋值或用 scanf 函数输入，接下来处理变量之间的运算，最后用 printf 函数向外输出变量结果。

C 语言中的每个变量都有它的数据类型，就像我们在生活中都有各自的角色。无论担任何种角色，我们都要对自己的角色负责，只要我们坚守初心、砥砺前行，平凡的角色也能创造出不平凡的人生。



笔记

一、填空题

1. 假设已经定义 x 、 i 、 j 、 k 为整型变量, 则计算 $x=(i=3,j=5,k=7)$, x 的值为

_____。

2. 当 $x=0$ 时, 表达式 $!x$ 的值为_____。

3. 下面程序的输出结果为_____。

```
float k=-123.246;
printf("#%4.2f#\n",k);
```

4. 下面程序的输出结果为_____。

```
int m=027;
printf("%d,%x,%o\n",m,m,m);
```

5. 下面程序的输出结果为_____。

```
int i=5,j;
j=i++;
printf("i=%d,j=%d\n",i,j);
```

6. 下面程序的输出结果为_____。

```
int a,b,c=241;
a=c/100%9;
b=(-1)&&(-1);
printf("%d,%d",a,b);
```

7. 为使 a 为 1、 b 为 5, 有以下输入函数:

```
scanf("a=%d,%d:",&a,&b);
```

则应从键盘输入_____以满足要求。

8. 下面程序的输出结果为_____。

```
main()
{
    int i=5;
    i+=i*=i+6;
    printf("i=%d\n",i);
}
```

9. 若要求下面程序的输出结果为 8.00, 则横线上 y 的值应该是_____。

```
main()
{
    int k=2,m=5;
    float s,x=1.2,y=_____;
    s=2/3+k*y/x+m/2;
    printf("%4.2f\n",s);
}
```

10. 下面程序的输出结果为_____。

```
main()
{
    int n=100;
    char c;
    float f=10.0;
    double x;
    x=f*=n/(c=48);
    printf("%d %d %3.1f %3.1f\n",n,c,f,x);
}
```

二、选择题

- 下面四个选项中, 属于合法字符常量的是 ()。

A. "\0" B. 'bc' C. '\084' D. '\x34'
- a、b、c 均为整型变量, 从键盘输入数据, 正确的输入语句是 ()。

A. INPUT a,b,c; B. read("%d%d%d",&a,&b,&c);

C. scanf("%d%d%d",a,b,c); D. scanf ("%d%d%d",&a,&b,&c);
- 以下赋值语句合法的是 ()。

A. z=int(x+y); B. a--;

C. a+b; D. s=r=100;
- 下面四个选项中, 均是正确的八进制数或十六进制数的选项是 ()。

A. -10. B. 0abc C. 0010 D. 0a13

0x8f -017 -0x11 -0x1232

-011 0xc 0xf1 -0xa
- 在 C 语言中, 字符型数据在内存中以 () 形式存放。

A. 原码 B. BCD 码

C. 反码 D. ASCII 码
- 若有 char s1="\067", 则 (); char s2="1", 则 (); char s3='1', 则 ()。

A. 包含 1 个字符 B. 包含 2 个字符

