“十四五”职业教育国家规划教材

C 语言程序设计

C YUYAN CHENGXU SHEJI

（第 2 版）

方加娟　赵广复　主编

- 教学课件
- 电子教案
- 练习题库

中原出版传媒集团
中原传媒股份公司
河南科学技术出版社

『十四五』职业教育国家规划教材

C语言程序设计（第2版）　方加娟　赵广复　主编

河南科学技术出版社

出 版 人　乔　辉
策划编辑　王向阳　孟明明
责任编辑　孟明明
责任校对　李　军
封面设计　张　伟
责任印制　徐海东

中原出版 CENTRAL CHINA PUBLISH

ISBN 978-7-5725-2065-5
9 787572 520655 >
定价：46.00 元

C 语言程序设计

C YUYAN CHENGXU SHEJI

（第 2 版）

方加娟　赵广复　主编

河南科学技术出版社
· 郑州 ·

内 容 提 要

C 语言是国内外广泛使用的计算机语言，也是计算机程序员应掌握的一种基本程序设计语言。本书面向程序设计初学者编写，内容包括：初识 C 语言，基本数据类型、运算符与表达式，结构化程序设计，数组，函数，指针，结构体和共用体，文件及学生成绩管理系统。本书针对初学者的特点，以“注重基础、注重方法、注重编程技能、注重应用”为指导思想，灵活运用案例教学、任务驱动、启发式教学的方法，对 C 语言的语法知识和程序的设计思想、设计方法进行了系统的介绍，特别适合将 C 语言程序设计作为第一门程序设计课程的高等院校学生学习使用。

本书可作为高职高专院校各专业的 C 语言课程教材，也可以作为成人教育、培训机构的 C 语言培训教材，还可以作为 C 语言编程爱好者的自学参考书。

图书在版编目（CIP）数据

C 语言程序设计 / 方加娟，赵广复主编. -- 郑州 ：河南科学技术出版社，2025. 5.
-- ISBN 978-7-5725-2065-5

Ⅰ. TP312. 8

中国国家版本馆 CIP 数据核字第 2025HA9982 号

出版发行：河南科学技术出版社
地址：郑州市郑东新区祥盛街 27 号　　邮编：450016
电话：（0371）65788865
网址：www. hnstp. cn
出 版 人：乔　辉
策划编辑：王向阳　孟明明
责任编辑：孟明明
责任校对：李　军
封面设计：张　伟
责任印制：徐海东
印　　刷：河南博之雅印务有限公司
经　　销：全国新华书店
开　　本：787 mm×1092 mm　1/16　　印张：19. 5　　字数：450 千字
版　　次：2025 年 5 月第 2 版　　2025 年 5 月第 1 次印刷
定　　价：46. 00 元

目录

项目一　初识 C 语言 …… (1)
任务一　编写简单的 C 语言程序 …… (1)
一、任务分析 …… (1)
二、必备知识与理论 …… (1)
三、任务实施 …… (4)
四、深入训练 …… (7)
任务二　C 语言程序的上机操作 …… (7)
一、任务分析 …… (7)
二、必备知识与理论 …… (8)
三、任务实施 …… (28)
四、深入训练 …… (41)
拓展阅读——程序与人生 …… (41)
项目实训 …… (41)
一、实训目的 …… (41)
二、实训任务 …… (42)
项目练习 …… (42)
项目二　基本数据类型、运算符和表达式 …… (44)
任务一　求圆的面积和周长 …… (44)
一、任务分析 …… (44)
二、必备知识与理论 …… (44)
三、任务实施 …… (51)
四、深入训练 …… (51)
任务二　计算表达式的值 …… (52)
一、任务分析 …… (52)
二、必备知识与理论 …… (52)
三、任务实施 …… (58)
四、深入训练 …… (59)
任务三　求三角形的面积 …… (59)
一、任务分析 …… (59)
二、必备知识与理论 …… (59)
三、任务实施 …… (63)
四、深入训练 …… (64)
任务四　求长方体的体积 …… (64)

一、任务分析 ……（64）
二、必备知识与理论 ……（64）
三、任务实施 ……（67）
四、深入训练 ……（68）
拓展阅读——程序与人生 ……（68）
项目实训 ……（69）
一、实训目的 ……（69）
二、实训任务 ……（69）
项目练习 ……（70）
项目三　结构化程序设计 ……（73）
任务一　求两整数相除的商和余数 ……（73）
一、任务分析 ……（73）
二、必备知识与理论 ……（73）
三、任务实施 ……（76）
四、深入训练 ……（77）
任务二　顺序输出三个数 ……（77）
一、任务分析 ……（77）
二、必备知识与理论 ……（77）
三、任务实施 ……（81）
四、深入训练 ……（81）
任务三　计算员工工资 ……（82）
一、任务分析 ……（82）
二、必备知识与理论 ……（82）
三、任务实施 ……（85）
四、深入训练 ……（86）
任务四　输出学生成绩最值和平均分 ……（87）
一、任务分析 ……（87）
二、必备知识与理论 ……（87）
三、任务实施 ……（91）
四、深入训练 ……（92）
任务五　水果问题 ……（93）
一、任务分析 ……（93）
二、必备知识与理论 ……（93）
三、任务实施 ……（97）
四、深入训练 ……（98）
拓展阅读——程序与人生 ……（98）
项目实训 ……（99）
一、实训目的 ……（99）

二、实训任务 ……（99）
项目练习 ……（101）
项目四　应用数组进行程序设计……（104）
任务一　冒泡排序 ……（104）
一、任务分析 ……（104）
二、必备知识与理论 ……（104）
三、任务实施 ……（108）
四、深入训练 ……（108）
任务二　求矩阵中元素最值 ……（109）
一、任务分析 ……（109）
二、必备知识与理论 ……（109）
三、任务实施 ……（112）
四、深入训练 ……（113）
任务三　统计字符中的单词 ……（114）
一、任务分析 ……（114）
二、必备知识与理论 ……（114）
三、任务实施 ……（120）
四、深入训练 ……（121）
算法拓展……（121）
拓展阅读——程序与人生……（124）
项目实训 ……（124）
一、实训目的 ……（124）
二、实训任务 ……（125）
项目练习 ……（125）
项目五　应用函数进行程序设计……（129）
任务一　比较整数大小 ……（129）
一、任务分析 ……（129）
二、必备知识与理论 ……（129）
三、任务实施 ……（134）
四、深入训练 ……（135）
任务二　求 x 的 n 次方 ……（135）
一、任务分析 ……（135）
二、必备知识与理论 ……（135）
三、任务实施 ……（139）
四、深入训练 ……（140）
任务三　用递归法求 n! ……（140）
一、任务分析 ……（140）
二、必备知识与理论 ……（140）

三、任务实施 ……（144）
四、深入训练 ……（145）
任务四 选择法排序 ……（145）
一、任务分析 ……（145）
二、必备知识与理论 ……（145）
三、任务实施 ……（148）
四、深入训练 ……（149）
任务五 求数组中成绩的平均分和最值 ……（150）
一、任务分析 ……（150）
二、必备知识与理论 ……（150）
三、任务实施 ……（159）
四、深入训练 ……（160）
五、典型函数程序实例 ……（162）
算法拓展 ……（162）
拓展阅读——程序与人生 ……（165）
项目实训 ……（165）
一、实训目的 ……（165）
二、实训任务 ……（165）
项目练习 ……（166）
项目六 应用指针进行程序设计 ……（170）
任务一 两个整数按顺序输出 ……（170）
一、任务分析 ……（170）
二、必备知识与理论 ……（171）
三、任务实施 ……（175）
四、深入训练 ……（176）
任务二 实现数组元素的存储逆序 ……（177）
一、任务分析 ……（177）
二、必备知识与理论 ……（177）
三、任务实施 ……（180）
四、深入训练 ……（181）
任务三 输出指定学生的学号和成绩 ……（181）
一、任务分析 ……（181）
二、必备知识与理论 ……（181）
三、任务实施 ……（186）
四、深入训练 ……（187）
任务四 编写字符串连接函数 ……（187）
一、任务分析 ……（187）
二、必备知识与理论 ……（187）

三、任务实施 ……（194）
四、深入训练 ……（195）
任务五 用指针函数求学生成绩 ……（195）
一、任务分析 ……（195）
二、必备知识与理论 ……（195）
三、任务实施 ……（198）
四、深入训练 ……（199）
五、典型程序实例 ……（199）
算法拓展 ……（201）
拓展阅读——程序与人生 ……（203）
项目实训 ……（203）
一、实训目的 ……（203）
二、实训任务 ……（203）
项目练习 ……（204）
项目七 结构体和共用体 ……（207）
任务一 使用结构体比较学生成绩 ……（207）
一、任务分析 ……（207）
二、必备知识与理论 ……（207）
三、任务实施 ……（212）
四、深入训练 ……（213）
任务二 使用结构体数组统计不及格人数 ……（213）
一、任务分析 ……（213）
二、必备知识与理论 ……（213）
三、任务实施 ……（215）
四、深入训练 ……（215）
任务三 使用结构体指针求最高成绩 ……（216）
一、任务分析 ……（216）
二、必备知识与理论 ……（216）
三、任务实施 ……（221）
四、深入训练 ……（222）
任务四 利用链表录入及输出学生信息 ……（222）
一、任务分析 ……（222）
二、必备知识与理论 ……（223）
三、任务实施 ……（225）
四、深入训练 ……（226）
任务五 利用共用体处理学生和教师信息 ……（227）
一、任务分析 ……（227）
二、必备知识与理论 ……（227）

三、任务实施 ……（229）
四、深入训练 ……（231）
任务六 利用枚举类型模拟机器人控制系统指令……（231）
一、任务分析 ……（231）
二、必备知识与理论 ……（231）
三、任务实施 ……（235）
四、深入训练 ……（235）
算法拓展……（236）
拓展阅读——程序与人生……（237）
项目实训 ……（238）
一、实训目的 ……（238）
二、实训任务 ……（238）
项目练习 ……（239）
项目八 文件……（243）
任务一 文件的打开与关闭 ……（243）
一、任务分析 ……（243）
二、必备知识与理论 ……（243）
三、任务实施 ……（247）
四、深入训练 ……（248）
任务二 将学生成绩存入文件 ……（248）
一、任务分析 ……（248）
二、必备知识与理论 ……（248）
三、任务实施 ……（255）
四、深入训练 ……（257）
任务三 读取文件中的学生成绩 ……（258）
一、任务分析 ……（258）
二、必备知识与理论 ……（258）
三、任务实施 ……（261）
四、深入训练 ……（262）
算法拓展……（262）
拓展阅读——程序与人生……（263）
项目实训 ……（264）
一、实训目的 ……（264）
二、实训任务 ……（264）
项目练习 ……（265）
项目九 学生成绩管理系统……（268）
任务一 需求分析 ……（268）
任务二 系统设计 ……（269）

任务三　功能设计 …… (270)
一、功能选择界面 …… (270)
二、增加学生成绩信息 …… (272)
三、修改学生成绩信息 …… (274)
四、删除学生成绩信息 …… (276)
五、按姓名或学号查询学生成绩信息 …… (277)
六、按成绩排序 …… (280)
七、将操作的数据写入文件 …… (281)
八、建议 …… (282)
项目案例一：云音乐后台管理系统 …… (282)
项目案例二：俄罗斯方块 …… (286)
拓展阅读——程序与人生 …… (288)
参考文献 …… (290)
附录 …… (291)
附录Ⅰ：常用字符与 ASCII 代码对照表 …… (291)
附录Ⅱ：C 语言中的关键字 …… (292)
附录Ⅲ：C 语言运算符优先级与结合性 …… (293)

项目一　初识 C 语言

C 语言是国际上广泛流行的计算机高级语言，具有很多突出的优点。很多人把 C 语言作为入门的计算机语言来学习。C 语言是学习和掌握更高层次语言的基础，也是 C++/C#、Visual C++和 Java 语言程序设计的基础。本项目主要介绍计算机程序设计语言的发展历史、C 语言的发展史和特点、C 语言程序的结构、简单 C 程序的编写和 C 程序的上机调试过程。

扫码看视频

学习目标

1. 了解计算机语言的发展历史。
2. 了解 C 语言的发展史和特点。
3. 正确运用 C 语言的基本符号、标识符和关键字。
4. 掌握 C 程序的结构，设计简单的 C 语言程序。
5. 熟悉 C 语言的开发环境并熟练调试和运行 C 语言程序。

任务一　编写简单的 C 语言程序

知识要点：认识 C 语言。

一、任务分析

学习 C 语言最好的方法是实践。首先让我们动手设计一个简单的应用程序，来认识 C 语言的特点。本任务是编写一个简单的 Hello 程序，它由一个主函数构成，在主函数中调用了一条标准的格式输出函数，当用户运行程序后，可以在屏幕上输出信息“Hello C!”。

二、必备知识与理论

1. 程序设计语言的发展历史

程序设计语言的发展，经历了从机器语言、汇编语言到高级语言的过程。

（1）机器语言。计算机能够直接识别和执行由“0”和“1”组成的二进制代码，由这些代码组成的机器指令的集合就是机器语言。使用机器语言是十分痛苦的，由于每台计算机的指令系统各不相同，在一台计算机上执行的程序，要想在另一台计算机上执行，必须重新编写程序，程序可移植性很差。因为机器语言是针对特定型号的计算机设计的，故而运算效率是所有语言中最高的。机器语言，是第一代计算机语言。

（2）汇编语言。为了克服机器语言难读、难编、难记和易出错的缺点，人们就用一些简洁的英文字母、符号串来替代一个特定指令的二进制串（如用 ADD 表示运算符号“+”的机器代码），于是就产生了汇编语言。汇编语言是一种用助记符表示的仍然面向机器的

计算机语言。汇编语言亦称符号语言，是第二代计算机语言。汇编语言同样十分依赖机器硬件，移植性不好，使用起来仍然比较烦琐费时，属于低级语言。但是，汇编语言用来编制系统软件和过程控制软件时，其目标程序占用内存空间少，运行速度快，有着高级语言不可替代的优势。

（3）高级语言。从最初与计算机交流的痛苦经历中，人们意识到，应该设计一种更加方便程序员操作的计算机程序设计语言，因此高级语言就发展起来了。高级语言是一种比较接近于人的自然语言和数学表达式的计算机程序设计语言。它不依赖于计算机硬件，编出的程序能在所有机器上通用。一般用高级语言编写的程序称为“源程序”，计算机不能识别和执行源程序，要把源程序翻译成机器指令“目标程序”后才能被计算机识别和执行，通常有编译和解释两种方式。1954 年，第一个完全脱离机器硬件的高级语言——FORTRAN 问世了，随后，出现了很多种高级语言。其中，影响较大、使用较普遍的有 FORTRAN、ALGOL、COBOL、BASIC、LISP、SNOBOL、PL/1、Pascal、C、PROLOG、Ada、C++、Visual B、Visual C++、Delphi、Java 等。

2. C 语言的发展史

C 语言是国际上广泛流行的计算机高级语言。它适合作为系统描述语言，既可用来写系统软件，也可用来写应用软件。

C 语言是 1972 年由美国贝尔实验室的 D. M. Ritchie 在 B 语言的基础上设计发明的，开发 C 语言主要是为了更好地描述 UNIX 操作系统。1973 年，贝尔实验室的 K. Thompson 和 D. M. Ritchie 两人合作将用汇编语言编写的 UNIX 的 90%以上代码用 C 语言改写。后来，C 语言做了多次改进，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 语言的突出优势才引起人们的普遍注意。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 语言程序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。C 语言和 UNIX 可以说是一对孪生兄弟，在发展过程中相辅相成。

随着 C 语言的广泛使用，出现了多种版本，但没有一个统一的 C 语言标准。为了改变这种情况，1983 年，美国国家标准学会（ANSI）建立了一个委员会，根据 C 语言问世以来各种版本对其发展和扩充，制定了新的标准，称为 ANSI C，成为现行的 C 语言标准。

目前，C 语言程序开发人员广泛使用的 C 语言集成开发环境有 Visual Studio Code（简称 VS Code）、Dev-C++（或者叫作 Dev-Cpp）、Visual Studio 2017、Visual C++ 6.0 等。VS Code 是微软公司 2015 年发布的一款免费的跨平台（Windows、macOS 和 Linux）IDE。它具有免费、轻量、插件丰富、调试功能强大等特点，能帮助开发人员根据实际需求定制编辑器，更高效地开发程序代码。本书使用 Windows 11 操作系统下 VS Code 的编译环境。虽然各种编译系统的基本部分都是相同的，但也有一些差异。所以请大家注意自己使用的 C 语言编译系统的特点和规定（可以参阅有关手册）。

3. C 语言的特点

一种语言之所以能够存在和发展，并具有较强的生命力，是因为它具有不同于或优于其他语言的特点。C 语言的主要特点有：

（1）C 语言是处于汇编语言和高级语言之间的一种记述性程序语言。C 语言具有高级

语言面向用户、容易记忆、便于阅读和书写的优点；同时，C 语言允许直接访问物理地址，能进行位（bit）操作以及指定用寄存器存放变量等，能实现汇编语言的大部分功能。C 语言同时具备高级语言和低级语言的特征。因此，有人认为 C 语言是中级语言。

（2）C 语言是结构化程序设计语言，即程序的逻辑结构可以用顺序、选择和循环三种基本结构组成。C 语言具有结构化程序设计所要求的控制语句，如 if…else 语句、while 语句、do…while 语句、switch 语句、for 语句等，十分便于采用自顶向下、逐步细化的结构化程序设计技术。因此，用 C 语言编制的程序，具有容易理解、便于维护的优点。

（3）C 语言支持模块化程序设计。C 语言的程序是由函数构成的，每个函数可以单独编写和调试，用函数作为程序的模块单位，便于实现程序的模块化。因此，对于大型程序，程序员们可以分别编写不同的模块，这使得管理和调试工作变得简单和方便，并且可以实现软件重用，即重复使用那些经常需要使用的程序模块。

（4）C 语言具有丰富的数据类型。C 语言提供的数据类型有：整型、浮点型、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、栈、树等）的运算，尤其是指针类型数据，使用十分灵活和多样化。

（5）C 语言的运算符种类多、功能强大。C 语言共提供了 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使用其他高级语言难以实现的运算。

（6）C 语言有大量标准化的库函数。这些库函数不但包括了各种数学计算的函数，还有用于输入、输出的库函数及系统函数，给程序员编写程序带来了极大的方便。

（7）C 语言生成的目标代码质量高，程序执行效率高。其生成目标代码效率一般只比汇编语言生成的目标代码效率低 10%～20%。

（8）用 C 语言编写的程序可移植性好，应用性广，基本上不用修改就能运行于各种型号的计算机和各种操作系统。

C 语言的优点很多，但也有一些不足之处，了解 C 语言的缺点，有助于我们在编写程序时扬长避短。具体来讲，其缺点有以下两个：

第一，C 语言比较灵活，语法限制不太严格，程序设计自由度大，源程序书写格式自由。在语法检查上不如一般的高级语言严格。因此，程序员应当自行仔细检查程序，保证其正确性，不要过分依赖 C 编译系统去查错。这就给程序的调试带来困难，尤其是对初学者。

第二，如果不加以特别的注意，C 语言程序的安全性将会降低。例如，对指针的使用没有适当的限制，指针设置错误，可能引起内存中的信息被破坏，如果经常出现这种错误，极有可能导致系统的崩溃。

C 语言学习难度较大，特别是函数、指针、地址等内容，需要认真学习才能掌握。尽管 C 语言没有其他语言好掌握，但对于编写系统软件和应用软件，C 语言明显优于其他语言。它还是一些后续课程的先修课，所以 C 语言是一种十分优秀而又十分重要的语言。如果想成为一名优秀的软件工程师，就必须认真地学好 C 语言。

4. C 语言的基本符号

（1）基本符号。C 语言的基本符号是 ASCII 字符集，主要包括：26 个英文字母（C 语言区分大小写，即大写和小写字母表示两种不同的符号）；10 个阿拉伯数字（0、1、2、

…、9）；其他特殊符号，以运算符为主（+、-、*、/、=、%、&、<、>等）。

（2）标识符。程序中用来标识变量名、数组名、函数名和其他由用户自定义的数据类型名称等的有效字符序列称为标识符。

标识符的构成规则如下：①只能由英文字母（A~Z、a~z）、数字（0~9）和下划线（_）3类符号组成，但必须以字母或下划线开头。②严格区分字母大、小写，例如，sum、Sum、SUM表示3个完全不同的标识符。③标准C不限制标识符的长度，但它受各种版本的C语言编译系统限制，同时也受到具体机器的限制。一般的C编译系统只取标识符的前8个字符为有效字符。④不能以关键字作为标识符。⑤通常，命名标识符时应尽量做到“见名知意”，即选用有含义的英文单词或缩写，以及汉语拼音作标识符，如sum、name、max、year等。

（3）关键字。关键字又称为保留字，是C语言编译系统所固有的、具有特定含义的标识符，共有32个。它们主要是一些编制C语言源程序会用到的命令名、类型名等。根据关键字的作用不同，可将其分为控制语句关键字、数据类型关键字、存储类型关键字和其他关键字4类。①控制语句关键字（12个）：break、continue、switch、case、default、if、else、do、while、for、goto、return。②数据类型关键字（12个）：char、int、short、long、double、float、signed、unsigned、struct、union、enum、void。③存储类型关键字（4个）：auto、register、static、extern。④其他类型关键字（4个）：const、sizeof、typedef、volatile。

注意：所有关键字的字母均采用小写。

三、任务实施

本任务是编写一个简单的Hello程序，它由一个主函数构成，在主函数中调用了一条标准的格式输出函数，当用户运行程序后，可以在屏幕上输出信息“Hello C!”。

（1）main()函数：函数是C语言程序的基本单位，每一个C语言程序，不论大小，都是由一个或多个函数组成的。main()函数比较特殊，称为主函数，主函数的类型是int（整型）类型。每一个C语言源程序都必须包含且只能包含一个主函数。C语言程序的执行是从主函数中的第一条语句开始，到主函数中的最后一条语句结束。

（2）函数体：C语言程序使用一对花括号{}将函数体括起来。编程时要注意左、右花括号要成对使用。

（3）函数的调用：由于C语言中没有输入、输出语句，输入、输出操作是通过调用函数来完成的，本任务在主函数中调用了一条标准的格式输出函数printf()，用来输出信息“Hello C!”，字符串“Hello C!”作为printf()函数的实参。printf()是C语言的标准输入、输出函数库中的一条函数，在使用时，要用编译预处理命令“#include”将“stdio.h”文件包含到用户源文件中，即#include <stdio.h>。

（4）分号“;”：是C语言的执行语句和说明语句的结束符。C语言程序的函数体是由一条条语句组成的，书写格式自由，一行内可以写多条语句，一条语句也可以分写在多行上。

（5）注释：return 0；通常出现在main（）函数的末尾，用于向操作系统报告程序的执行状态，表示程序正常结束且没有发生错误。

（6）注释：在“/*”与“*/”之间的字符序列称为注释，用来解释程序或语句的作用。被注释的内容可以是一行文字或者连续的多行文字，使用注释能增强程序的可读性，使程序更易于理解。注释可以在程序中自由地使用，在程序编译时被自动忽略。

下面编写一个简单的 Hello 程序，在屏幕上输出信息“Hello C!”。

```
/*   A program to print Hello C!    */
#include <stdio.h>        /* #打头的是预处理命令,包含标准输入、输出函数库的信息 */
int main()                /* 主函数,主函数的类型是 int(整型)类型 */
{
   printf("Hello C! \n"); /* main()函数中调用库函数 printf() */
   return 0;              /* 结束函数的执行 */
}                         /* 花括号中是主函数的函数体 */
```

程序运行结果如图 1.1 所示。

```
Hello C!
```

图 1.1　Hello 程序的运行结果

【例 1.1】求两个整数的和并在屏幕上显示结果。

```
#include <stdio.h>
int main()
{       int a, b, sum;
        a = 12;
        b = 34;
        sum = a + b;
        printf("sum=%d\n", sum);
        return 0;
}
```

程序运行结果如图 1.2 所示。

```
sum=46
```

图 1.2　【例 1.1】程序的运行结果

【例 1.2】从键盘上输入两个整数，比较两数，将大的数输出。

```
#include <stdio.h>
int max(int x, int y)          /*定义 max()函数,函数值为整型,x,y 为形式参数*/
{
    int z;                     /* max()函数中对用到的变量 z 进行定义 */
    if (x > y)   z = x;
    else   z = y;              /* 条件语句 */
    return (z);                /*将 z 的值返回,通过函数调用将值带回到调用处*/
}
```

```
int main()                    /* 主函数 */
{
    int a, b, c;              /* 定义变量 */
    printf("please input two numbers:\n");
    scanf("%d,%d", &a, &b);/* 输入变量 a 和 b 的值 */
    c = max(a, b);            /* 调用 max()函数,将得到的值赋给 c */
    printf("max=%d\n", c);/* 输出 c 的值 */
    return 0;
}
```

程序运行结果如图 1.3 所示。

```
please input two numbers:
123,456
max=456
```

图 1.3　【例 1.2】程序的运行结果

注意：程序中函数的排列顺序并不决定函数的执行顺序，执行顺序是通过函数调用来决定的。不论 main()函数在程序中的什么位置，C 语言程序总是从 main()函数开始执行。一般而言，main()函数执行完毕，程序也就结束了。也就是说，main()函数是程序的入口和出口。

四、深入训练

1. 编写一个 C 语言程序，要求显示如下结果：

```
**************************
        How are you!
**************************
```

2. 编写一个 C 语言程序，计算半径 R 等于 5 的圆的面积和周长。已知圆的面积公式为 S=3.14*R*R；圆的周长公式为 C=2*3.14*R。

任务二　C 语言开发环境配置

知识要点：Visual Studio Code 安装和 C 语言开发环境配置

一、任务分析

在目前广泛流行的 Visual Studio Code 集成开发环境中建立一个 C 语言源程序并调试运行，显示运行结果。首先认识和安装 Visual Studio Code 这个软件，配置 C 语言开发环境，然后建立源程序，在编辑窗口中输入源程序，查看运行结果。

二、必备知识与理论

1. 开发环境概述

（1）Visual Studio Code 简介

Visual Studio Code（简称 VS Code）是一款由微软开发的轻量级、功能强大的源代码编辑器。它支持多种编程语言，具备强大的插件扩展功能，能够满足不同开发需求。VS Code 不仅支持多种操作系统（Windows、Mac、Linux），而且界面简洁、启动速度快、具有智能代码补全、语法检查、代码导航等功能，能提高编码效率。调试功能强大，可帮助开发者快速定位和解决代码中的问题，是一款适用于多种开发场景的高效工具。

对于初学者来说，VS Code 提供了易于上手的环境，而对于有经验的开发者，VS Code 的高度可定制性和扩展性能够满足各种复杂的开发需求。因此，VS Code 已成为全球开发者的首选工具之一。

（2）MinGW 编辑器简介

在安装完 VS Code 后，还要配置 C/C++语言的开发环境。这里采用的编辑器是 MinGW（Minimalist GNU for Windows）。MinGW 是一个开源的 C/C++编译器工具链，基于 GCC（GNU Compiler Collection）开发，专为 Windows 系统提供编译环境。它免费且开源，用户无需支付费用即可下载、使用和修改，非常适合学生和初学者。MinGW 安装过程简单，轻量级，不会占用过多系统资源，能快速搭建开发环境。它与 Windows 系统兼容性良好，可直接在 Windows 上运行，无需复杂配置。MinGW 支持标准 C/C++语言，能编译运行符合标准的程序，满足学习和开发需求。此外，MinGW 在开发者社区中广泛使用，拥有大量用户和丰富文档资源，遇到问题时容易找到解决方案，便于学习和使用。

2. 环境准备与软件下载

（1）准备操作系统环境：准备一台未安装 VS Code 的 Windows 操作系统，避免已有的环境可能带来的干扰（此处使用初始的 Windows 11 进行演示），如图 1.4 所示。

图 1.4　操作系统环境

（2）创建 VS Code 的工作目录：打开【文件资源管理器】，进入 D 盘，创建一个名为“VSCode”的文件夹，如图 1.5 所示。此文件夹将作为 VSCode 的安装目录，用于存放 VSCode 安装文件、C 语言代码文件以及编译器工具等相关的文件。

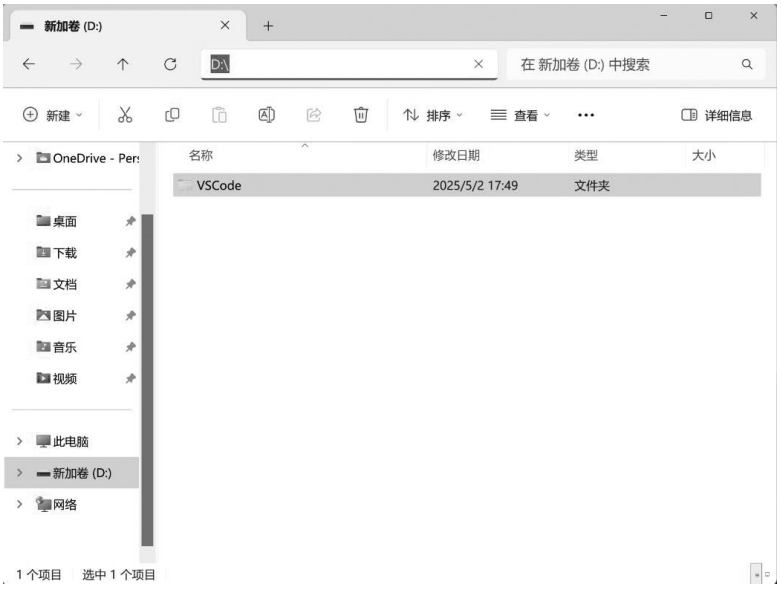

图 1.5　创建“VSCode”文件夹

（3）下载 VSCode：打开浏览器并访问 VSCode 官网：https：//code. visualstudio. com/，单击【Download for Windows】下载安装包，如图 1.6 所示。

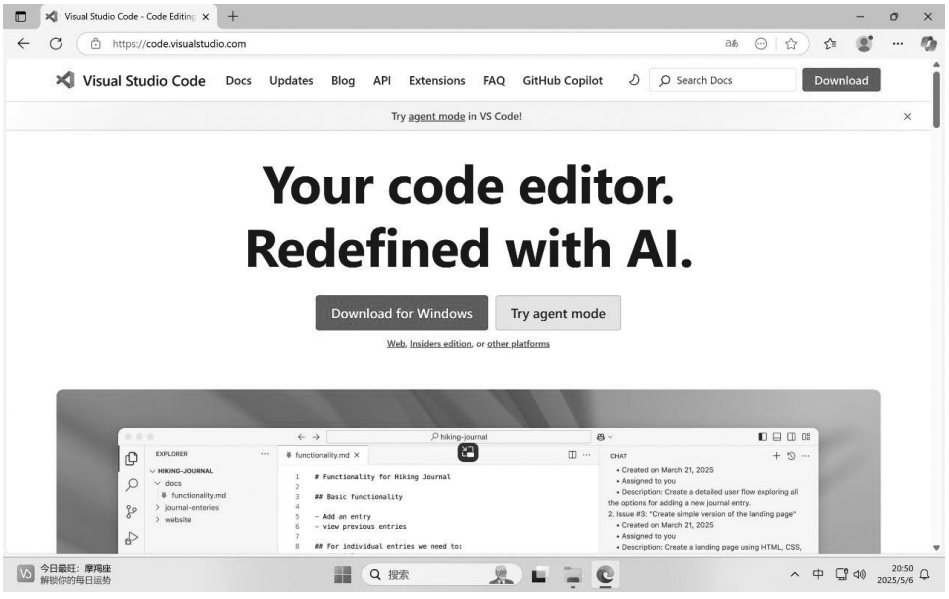

图 1.6　VSCode 官网下载界面

（4）浏览器弹窗提醒下载完成，如图 1.7 所示。

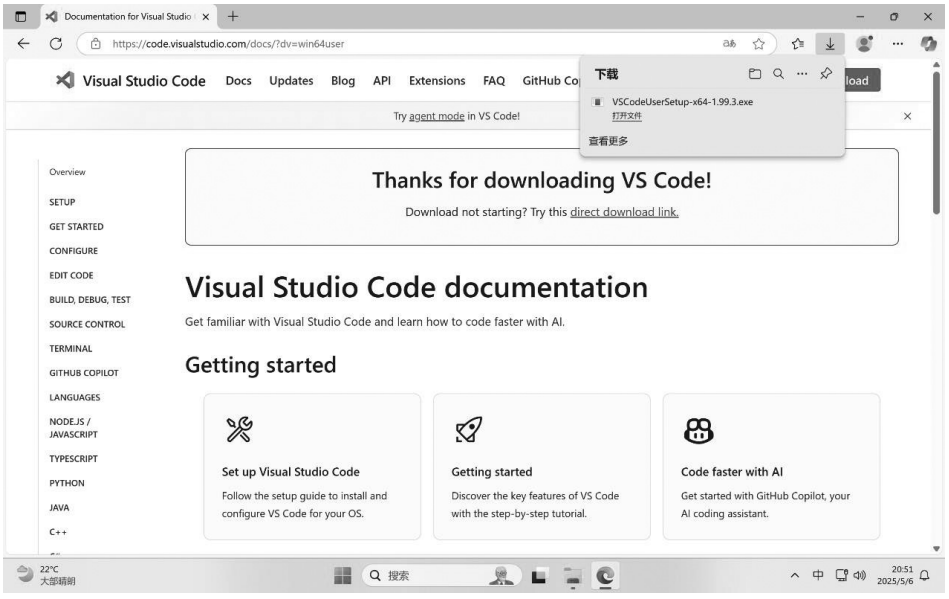

图 1.7　VSCode 安装包下载完成

（5）访问编译器官网：下载 C 语言程序编译和调试工具 mingw，访问官网：https：//www. mingw-w64. org/downloads/，并单击右侧边导航栏中的“MinGW-W64-builds”，如图 1.8 所示。

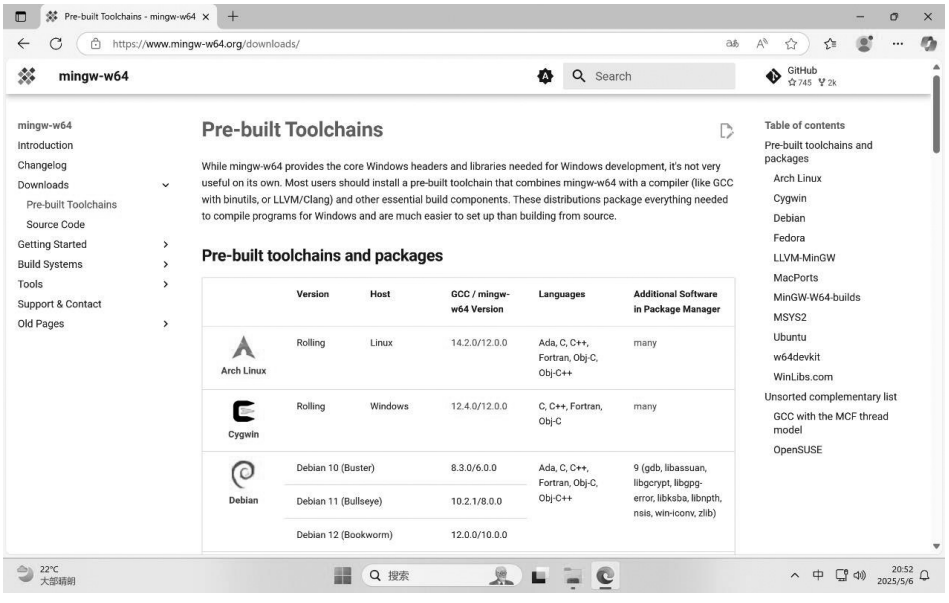

图 1.8　mingw 编译器官网

（6）单击【GitHub】将会跳转到下载页面，如图 1.9 所示。

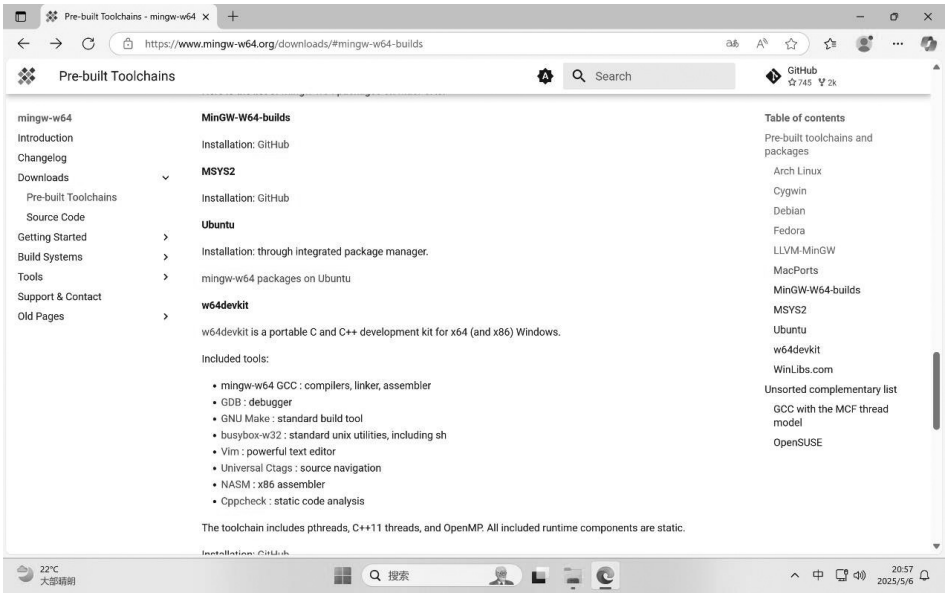

图 1.9　访问编译器下载网址界面

（7）下载编译器压缩包：可以根据不同的操作系统和不同的需求下载相应的编译器压缩包。这里我们选择 MSVCRT（Microsoft C Runtime）x86 64 位的编译器，向下滑动滚动条，找到【x86_ 64-15. 1. 0-release-posix-seh-msvcrt-rt_ v12-rev0. 7z】，如图 1. 10 所示，并单击下载。

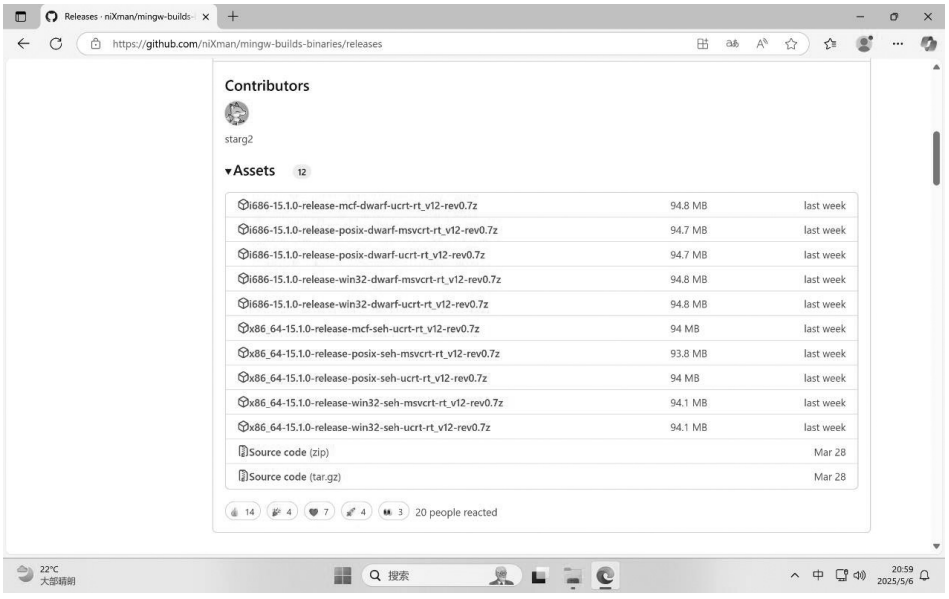

图 1.10　下载编译器压缩包

（8）下载完成后，会在浏览器弹窗中提示，然后单击打开“下载”文件夹，如图1.11所示。

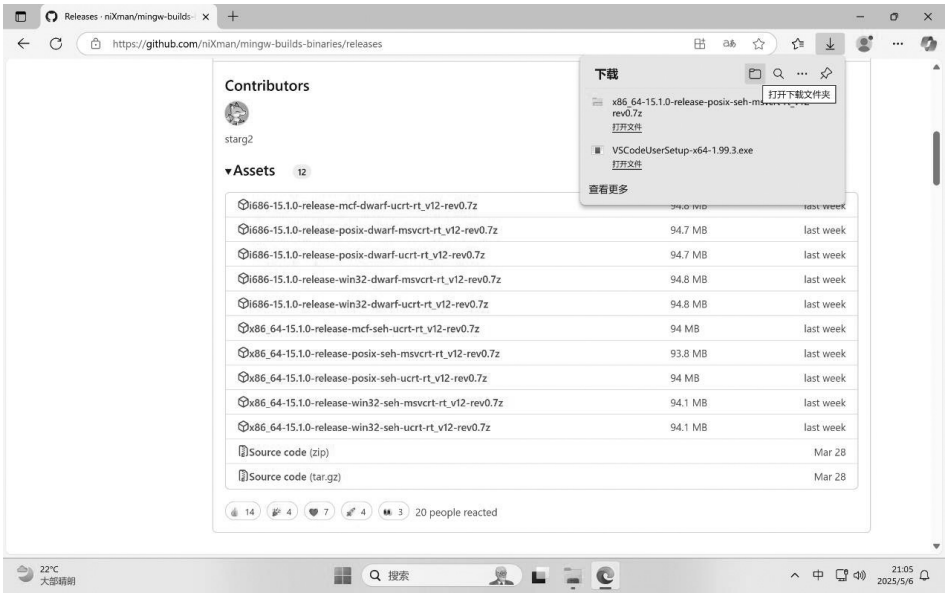

图1.11　打开下载文件夹

（9）在“下载”文件夹中可以看到已经下载好的VS Code安装包【VSCodeUserSetup-x64-1.99.3.exe】和编译器压缩包【x86_64-15.1.0-release-posix-seh-msvcrt-rt_v12-rev0.7z】，如图1.12所示。

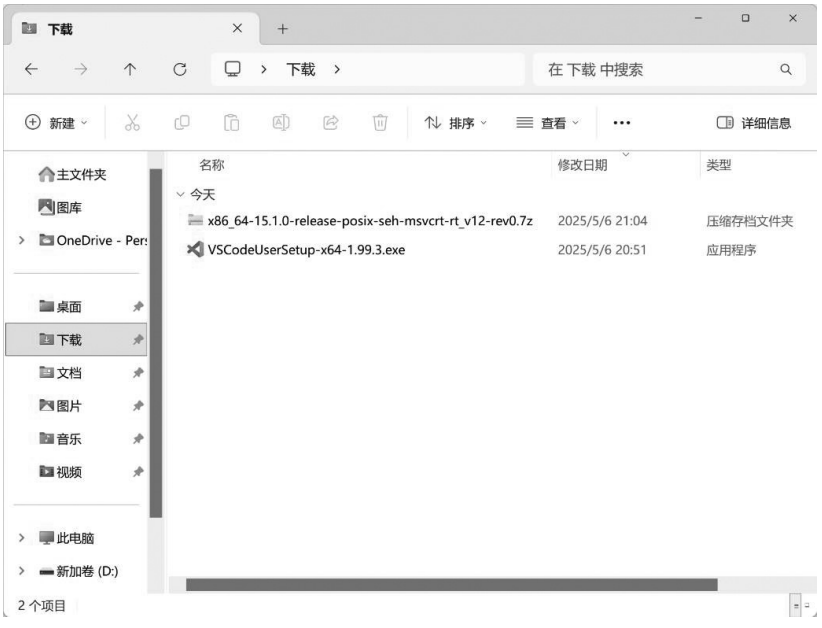

图1.12　安装软件准备就绪

3. 安装 VS Code

（1）双击打开 VS Code 安装包【VSCodeUserSetup-x64-1.99.3.exe】，勾选【我同意此协议】并单击【下一步】，如图 1.13 所示。

图 1.13　VS Code 安装初始界面

（2）默认安装位置在 C 盘，单击【浏览】按钮，如图 1.14 所示，重新选择 VS Code 的安装路径。

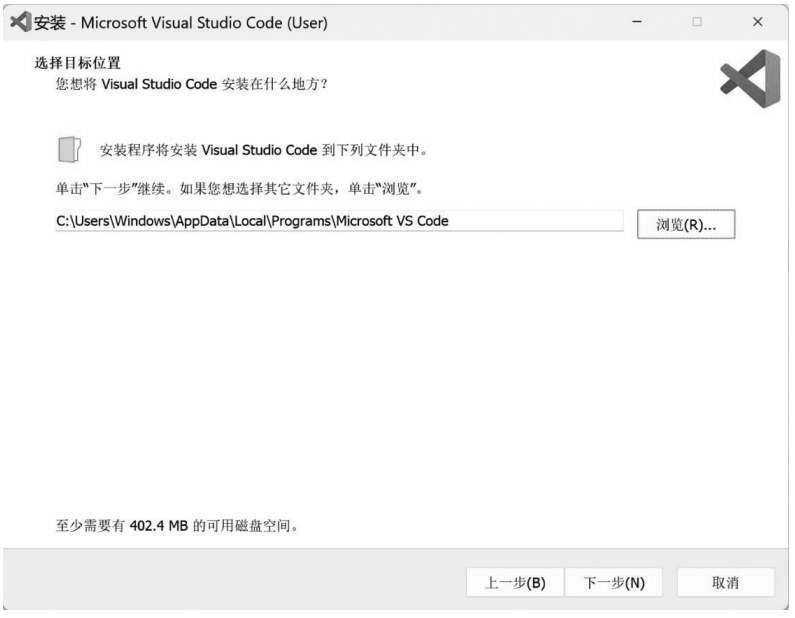

图 1.14　选择安装路径

（3）选择 D 盘下刚刚创建的“VSCode”目录，单击【确定】按钮，如图 1.15 所示。将 VSCode 安装位置选择在 D 盘而不是默认的 C 盘，主要是为了避免占用宝贵的系统

盘空间，因为 C 盘通常会存储大量系统文件和应用程序，空间容易紧张，而 D 盘通常空间更充裕，能更好地容纳 VS Code 及其相关文件。此外，将 VS Code 安装在 D 盘的【VSCode】目录下，可以将所有与之相关的代码文件、编译器工具等集中管理，便于查找、备份和后续操作，同时也能减少因频繁安装软件对 C 盘造成的碎片化，有助于维持系统的稳定性和可靠性。

图 1.15　选择 D 盘 VSCode 文件夹安装 VS Code

（4）自动创建二级目录“Microsoft VS Code”，单击【下一步】按钮，如图 1.16 所示。

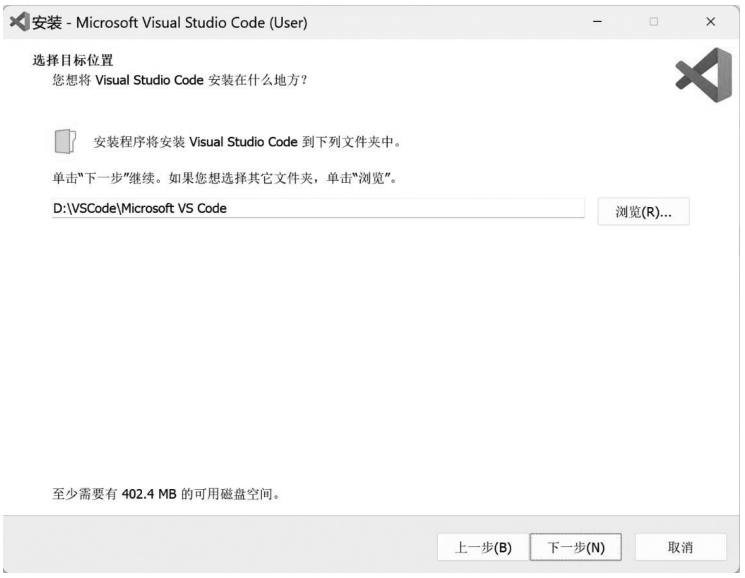

图 1.16　设置 VS Code 完整的安装路径

（5）单击【下一步】按钮，如图 1.17 所示。

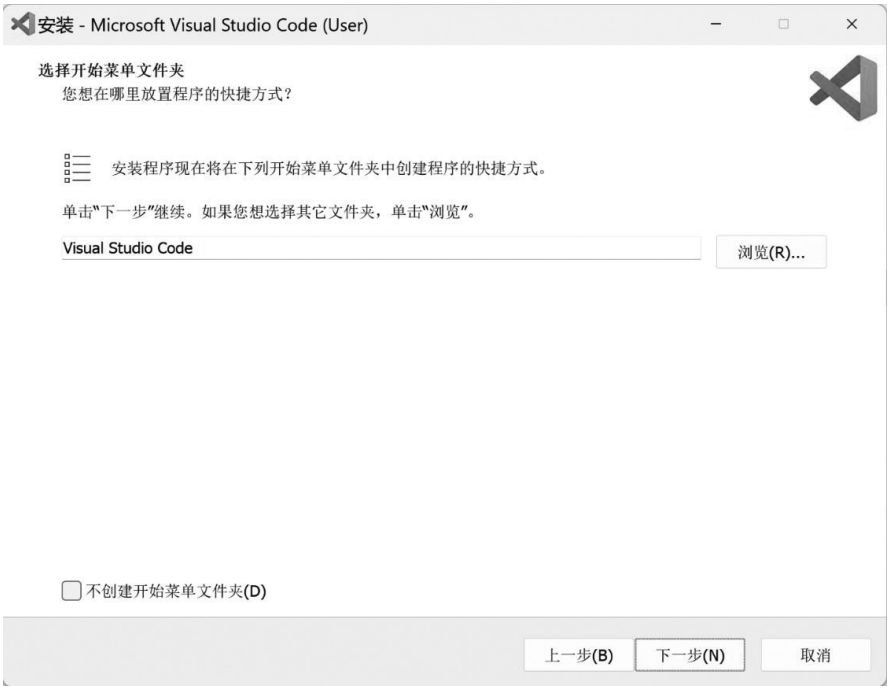

图 1.17　设置 VS Code 快捷方式的路径

（6）勾选所有复选框，单击【下一步】按钮，如图 1.18 所示。

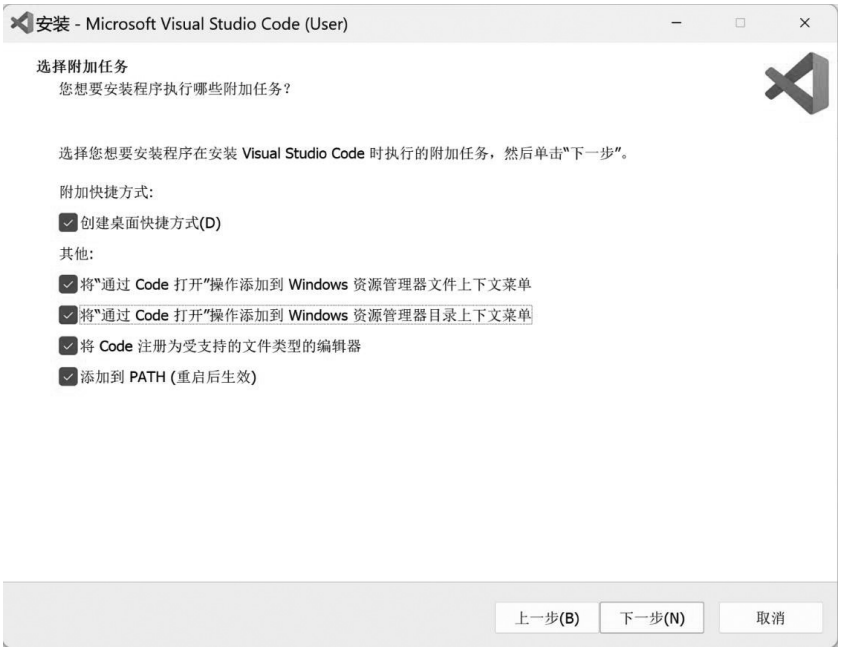

图 1.18　选择安装 VS Code 时执行的附加任务

（7）单击【安装】按钮，如图 1.19 所示。

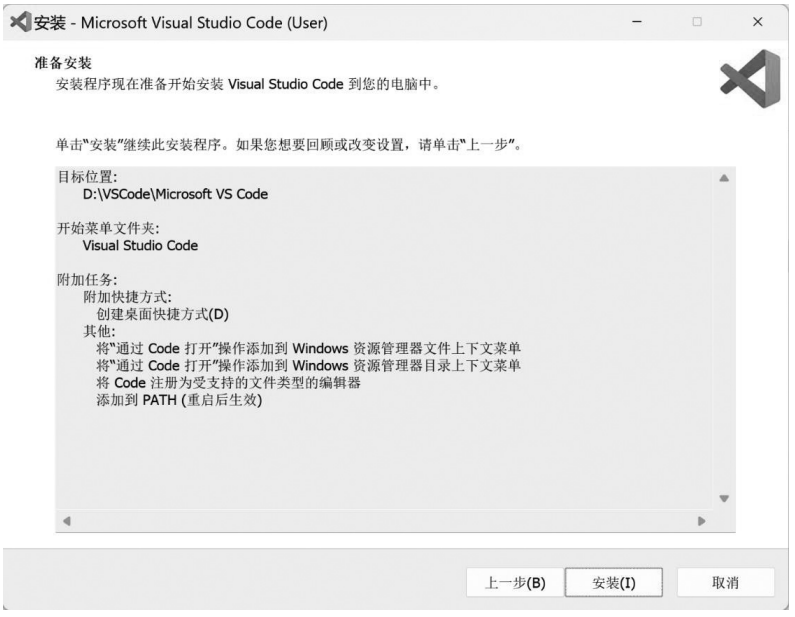

图 1.19　VS Code 安装开始

（8）安装成功后的界面如图 1.20 所示，勾选复选框【运行 Visual Studio Code】，单击【完成】按钮就能进入到软件主界面。

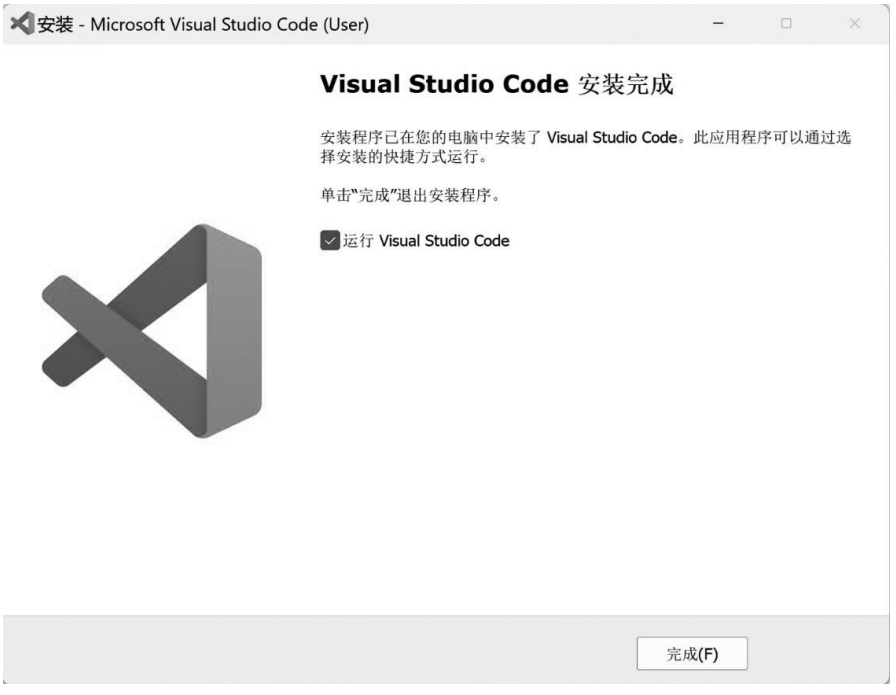

图 1.20　VS Code 安装完成

（9）首次运行 VS Code 的初始界面如图 1.21 所示，左侧为侧边栏（资源管理器、插件管理等），中间为代码编辑区，底部为终端和状态栏。至此，我们已经成功地安装了 VS Code。

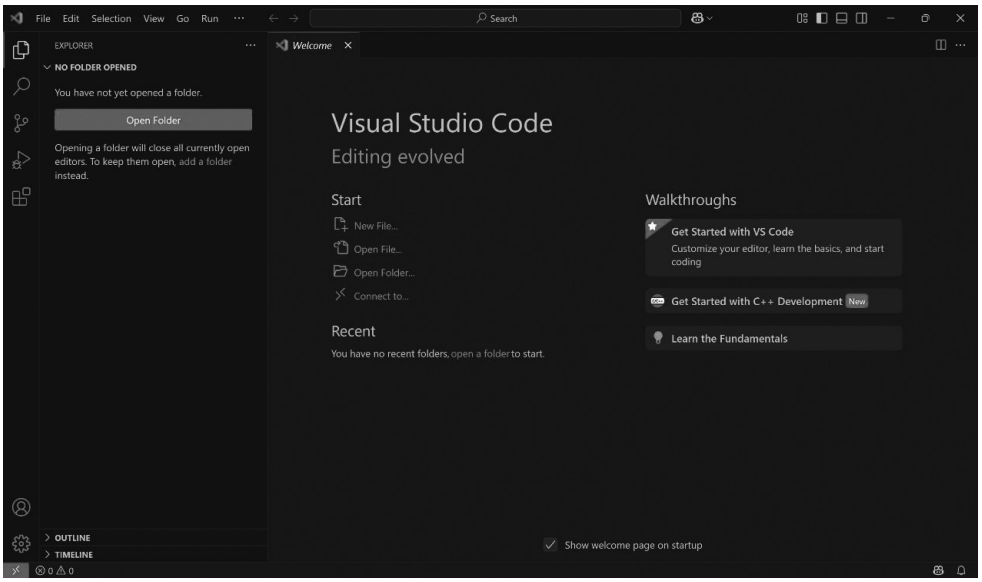

图 1.21　VS Code 运行初始界面

4. 配置中文语言环境以及切换主题

（1）单击侧边导航栏中的  按钮后，在搜索栏中输入“chinese”，选择【Chinese（Simplified）…】插件，单击插件右下角的【Install】按钮，如图 1.22 所示。

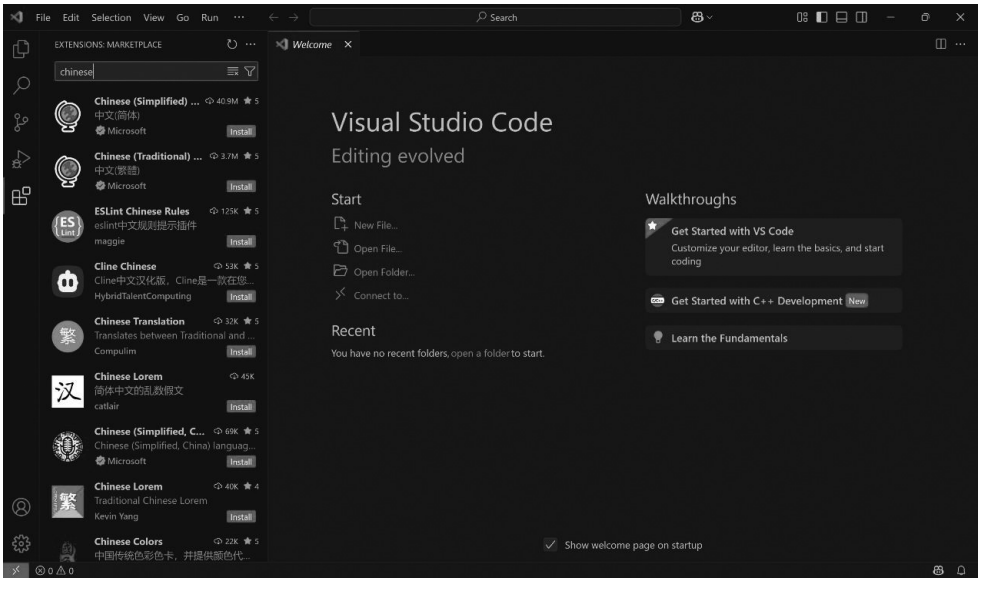

图 1.22　安装中文（简体）语言包插件

（2）安装完成后右下角会弹出提示框，单击【Change Language and Restart】按钮，如图 1.23 所示，将会重启 VSCode。

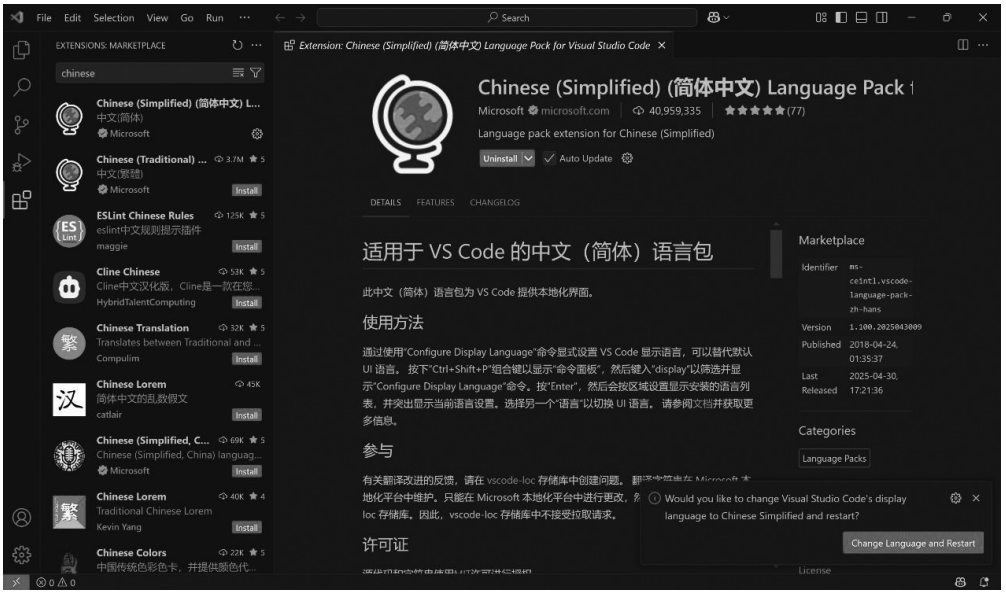

图 1.23　更改语言并重启 VS Code

（3）重启之后即可显示中文，如图 1.24 所示。

图 1.24　VS Code 中文界面

（4）切换主题，单击侧边导航栏中的  按钮，单击【主题】→【颜色主题】选项，如图 1.25 所示。

图 1.25　打开颜色主题界面

（5）选择任意一款自己喜欢的主题即可，例如选中【浅色（Visual Studio）Visual Studio Light】，如图 1.26 所示，VS Code 会立即变更为相应的主题样式。

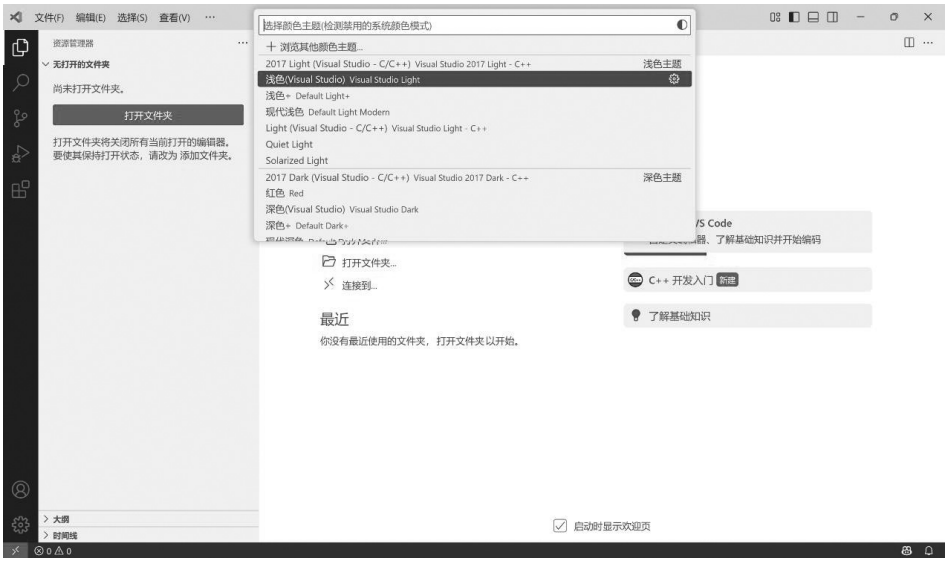

图 1.26　切换颜色主题

5. 配置 C 语言开发环境

（1）安装 C 语言扩展包

这里我们选择安装【C/C++ Extension Pack】C 语言扩展包插件，因其已经包含了【C/C++】插件，所以不用再安装，直接安装 C 语言扩展包插件就行。安装后，他可提供

语法检查、代码补全、智能提示、定义跳转、引用查找、断点调试、变量查看与修改、代码自动格式化及美化、函数文档提示等功能。这些功能有助于新手及时发现语法错误，快速了解函数用法，高效调试和优化代码，使代码风格规范统一，提升开发效率和代码质量。

1）单击侧边导航栏中的  按钮后，在搜索栏中输入“c”，选择【C/C++ Extension Pack】插件，单击插件右下角的【安装】按钮，如图 1.27 所示。

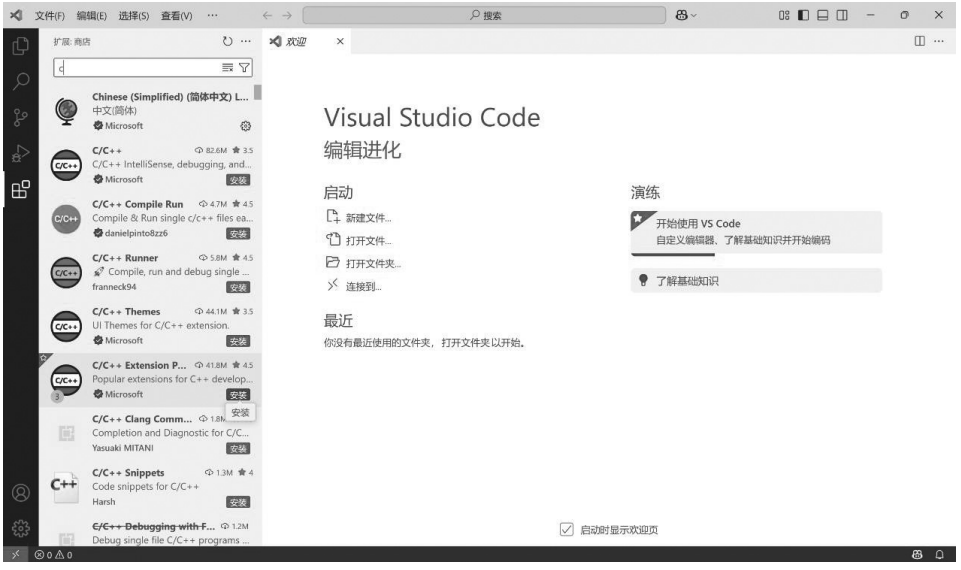

图 1.27　安装 C/C++扩展包插件

2）安装完成后，如图 1.28 所示。

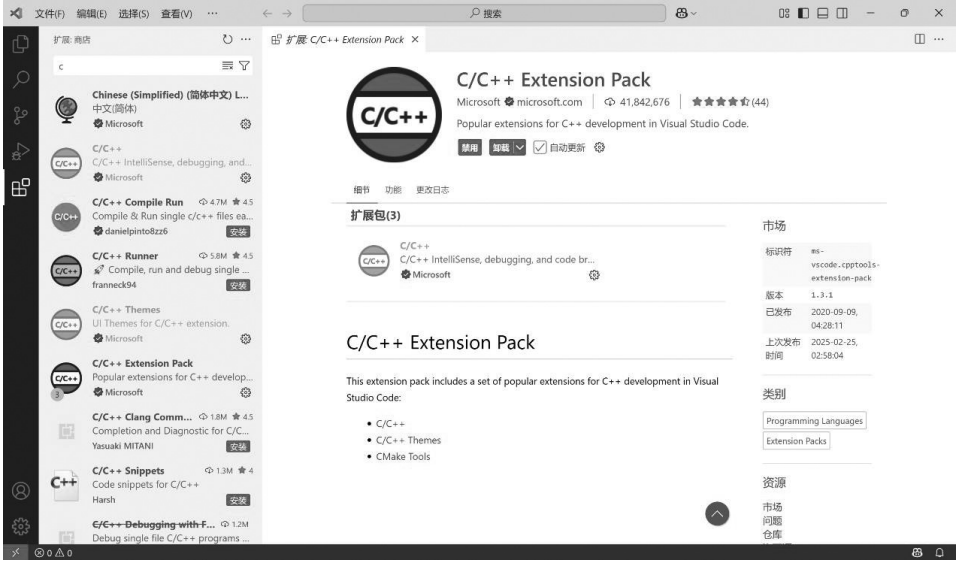

图 1.28　C/C++扩展包安装完成

（2）解压 C 语言程序编译器。

1）切换到“下载”文件夹，右键选中下载好的【x86_ 64-15. 1. 0-release-posix-seh-msvcrt-rt_ v12-rev0. 7z】，单击【全部解压缩】命令，如图 1. 29 所示。

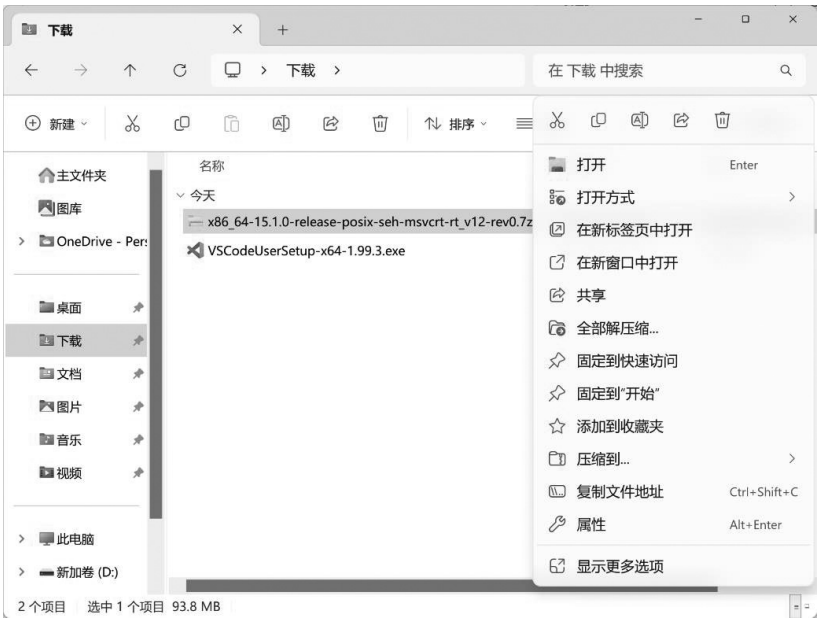

图 1. 29　解压编译器

2）单击【浏览】按钮，如图 1. 30 所示。

图 1. 30　更换解压缩编译器的目标位置

3）切换到“D：\ VSCode”目录下，单击【选择文件夹】按钮，如图 1.31 所示。

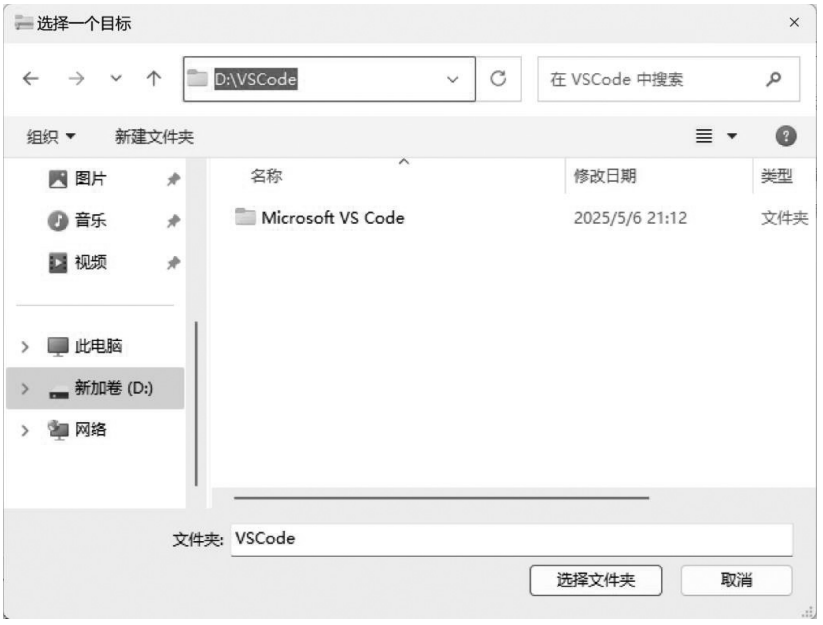

图 1.31　选择 D 盘 VSCode 文件夹作为目标位置

4）单击【提取】按钮，如图 1.32 所示。

图 1.32　单击【提取】按钮

5）提取完成后，在“D：\ VSCode”目录下生成了“mingw64”文件夹，如图 1.33 所示。

（3）配置环境变量。

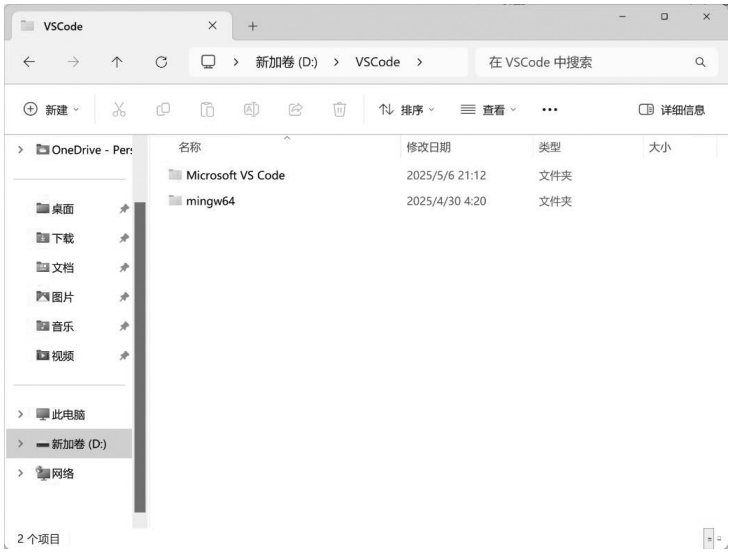

图 1.33　解压“mingw64”文件夹完成

配置环境变量是为了让系统能够识别和找到 MinGW 编译器的路径，从而能够在任何位置的命令行窗口中直接调用编译器命令（如 gcc 和 gdb）来编译和调试程序。通过将 MinGW 的 bin 目录添加到系统的环境变量中，系统会将其纳入可执行文件的搜索路径，使得在任何目录下都能方便快捷地使用这些编译工具，而无需每次都手动指定完整的路径。这大大提高了开发效率，简化了编译和调试过程，为后续的程序开发提供了便利。

1）单击【桌面】上的【搜索】，在搜索框中输入“环境变量”，单击【编辑系统环境变量】选项，如图 1.34 所示。

图 1.34　选择【编辑系统环境变量】选项

2）打开【系统属性】对话框，如图 1.35 所示。

图 1.35　【系统属性】对话框

3）单击【环境变量】按钮，打开【环境变量】对话框，如图 1.36 所示。双击【系统变量】下的【Path】变量。

图 1.36　【环境变量】对话框

4）打开【编辑环境变量】对话框，如图 1.37 所示。

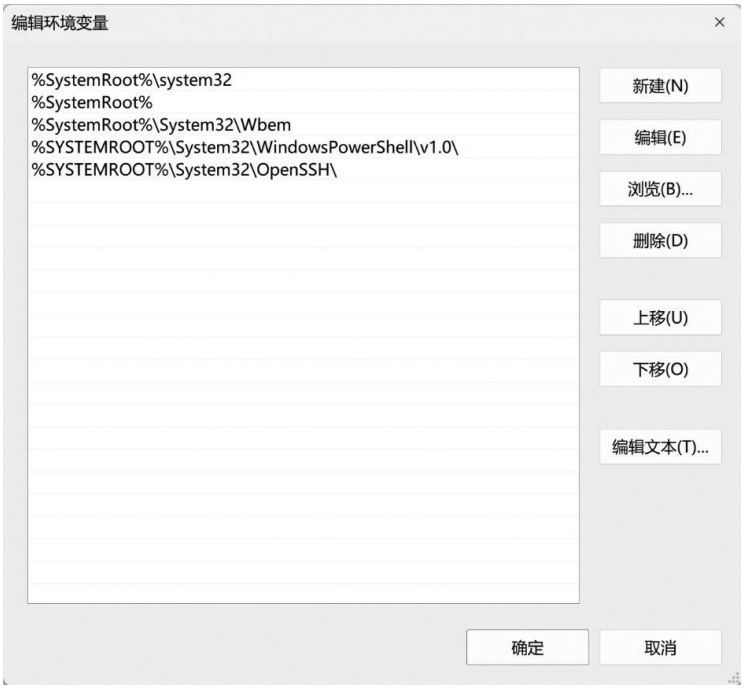

图 1.37　【编辑环境变量】对话框

5）切换到文件资源管理器的“D：\ VSCode”目录中，双击进入到“mingw64”目录，再进入“bin”目录，在地址栏中全选并单击右键，在快捷菜单中单击【复制】选项，如图 1.38 所示。

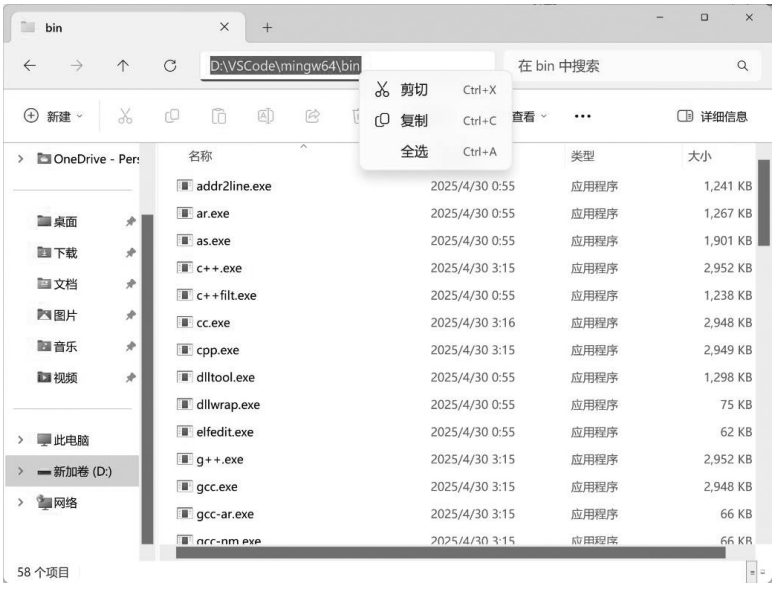

图 1.38　存放编译器可执行文件的目录界面

6）单击【编辑环境变量】对话框中的【新建】按钮，将刚刚复制的路径使用 Ctrl+V 进行粘贴，并单击【确定】按钮，如图 1.39 所示。

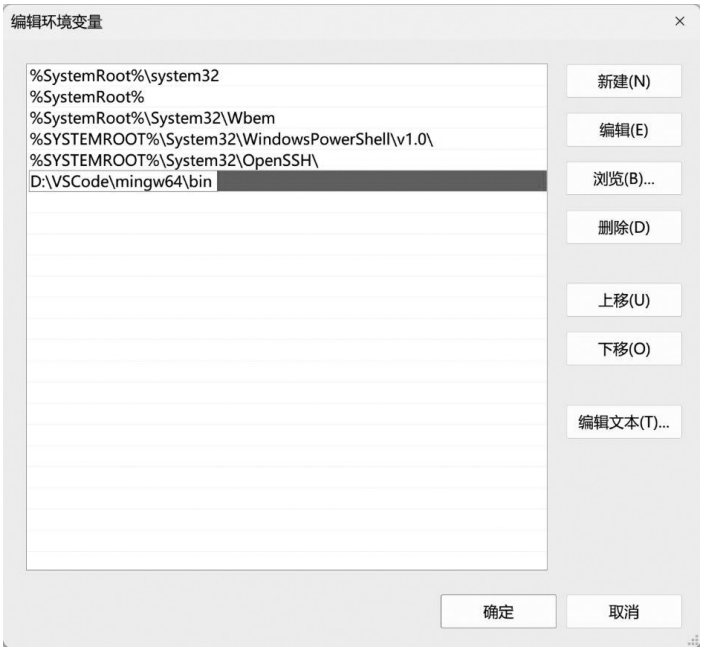

图 1.39　新建路径

7）单击【环境变量】对话框中的【确定】按钮，如图 1.40 所示。

图 1.40　【环境变量】对话框，【确定】按钮

8）单击【系统属性】对话框中的【确定】按钮，如图 1.41 所示，即可完成环境变量配置。

图 1.41　【系统属性】对话框，【确定】按钮

（4）验证环境变量配置。

环境变量配置完成之后，我们可以验证一下环境变量是否配置成功。

1）单击【桌面】上的【搜索】，在搜索框中输入“cmd”，单击【命令提示符】选项，如图 1.42 所示。

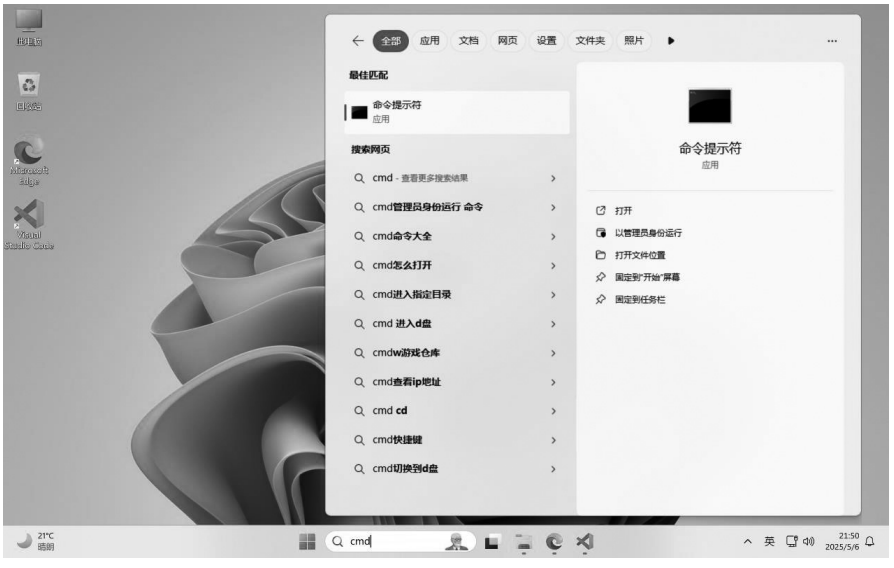

图 1.42　打开【命令提示符】

2）在【命令提示符】窗口中分别输入“gdb -v”以及“gcc -v”命令，如图 1.43 所示。能分别显示对应版本号，则说明环境变量配置成功。

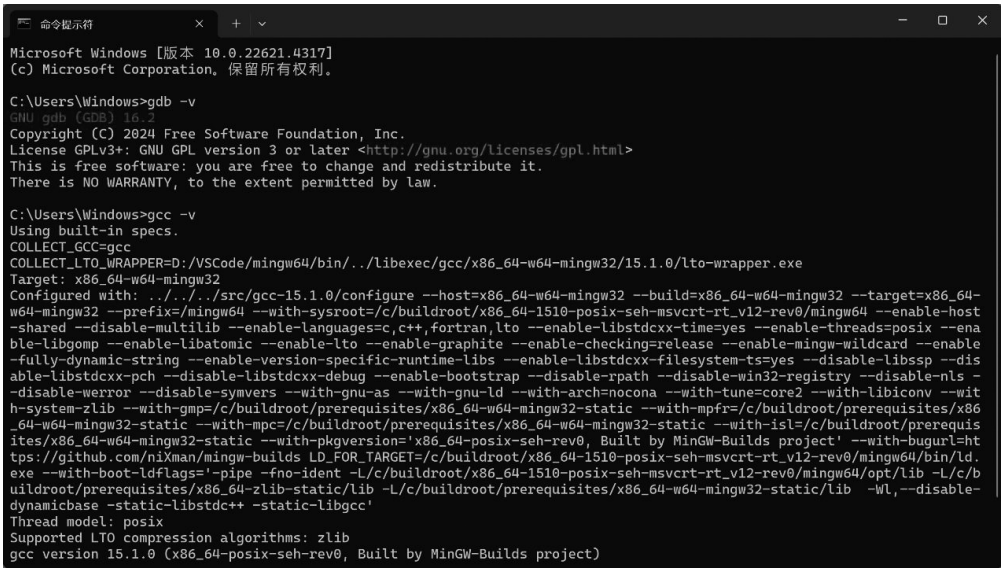

图 1.43　执行命令获取调试器、编译器的版本号

三、任务实施

1. 创建代码文件夹

（1）打开 VS Code 主界面，单击【资源管理器】后再单击【打开文件夹】选项，如图 1.44 所示。

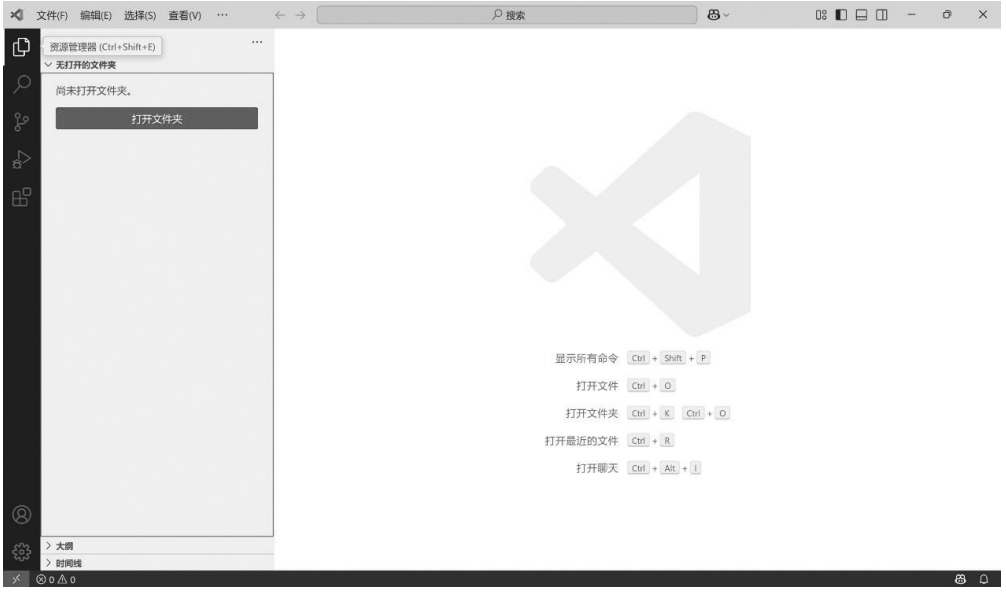

图 1.44　【打开文件夹】选项

（2）在文件资源管理器中的“D：\ VSCode”目标路径下，单击【新建文件夹】，如图 1.45 所示。

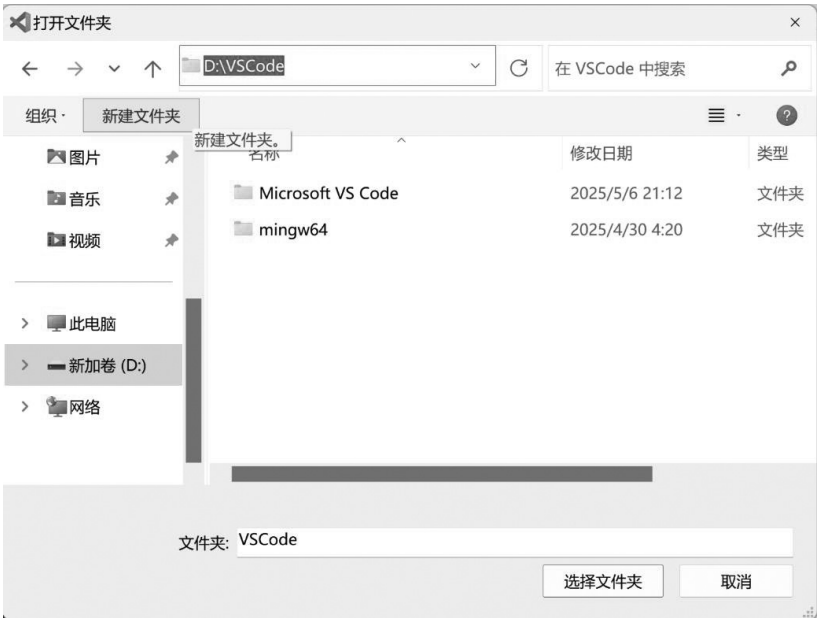

图 1.45　创建存放代码的文件夹

（3）输入“code”作为文件名，单击【选择文件夹】按钮，如图 1.46 所示。

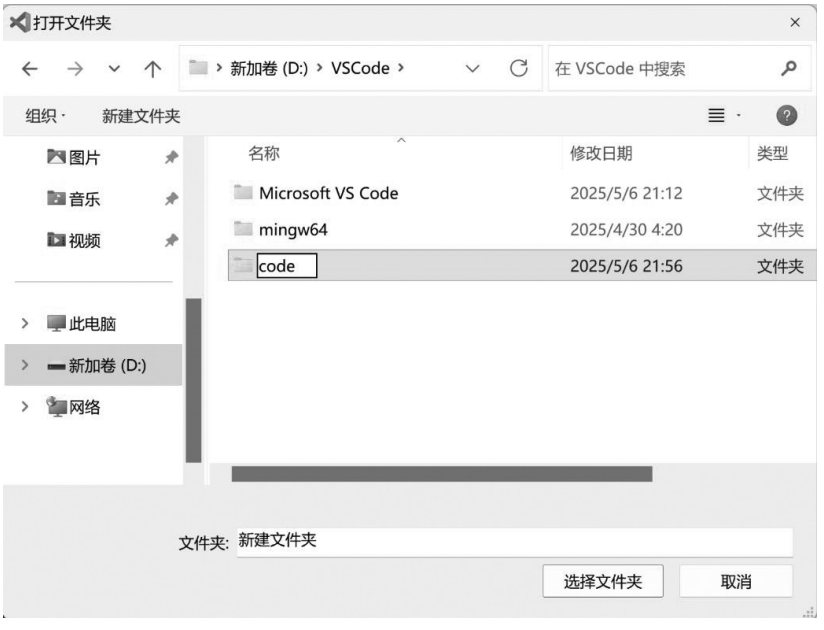

图 1.46　指定文件夹名为“code”

（4）勾选【信任父文件夹“VSCode“中所有文件的作者】复选框，单击【是，我信任此作者】按钮，如图 1.47 所示。

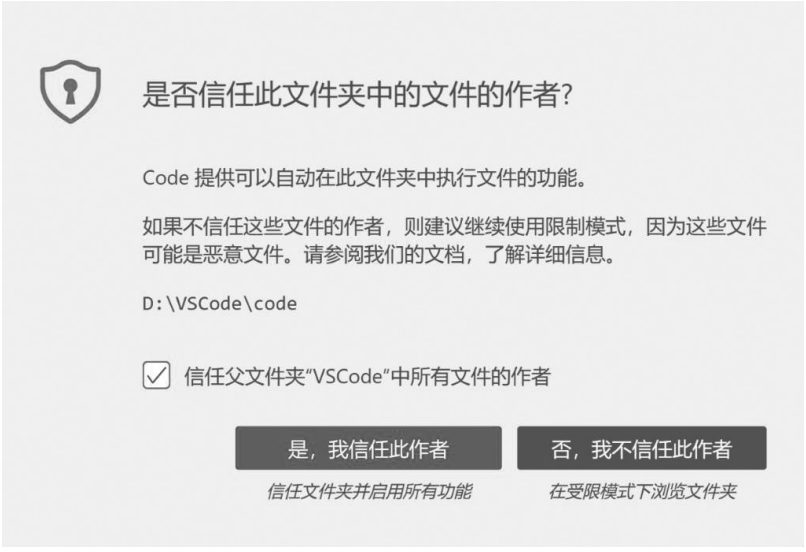

图 1.47　信任文件夹中的文件的作者

2. 编辑器配置

（1）按下键盘的 F1 唤醒命令面板（如果电脑有 FN 键，可能需要先按下 FN 键再按 F1），搜索“c/c++”，单击【Edit Configurations（UI）】选项，打开 C/C++配置，如图 1.48 所示。

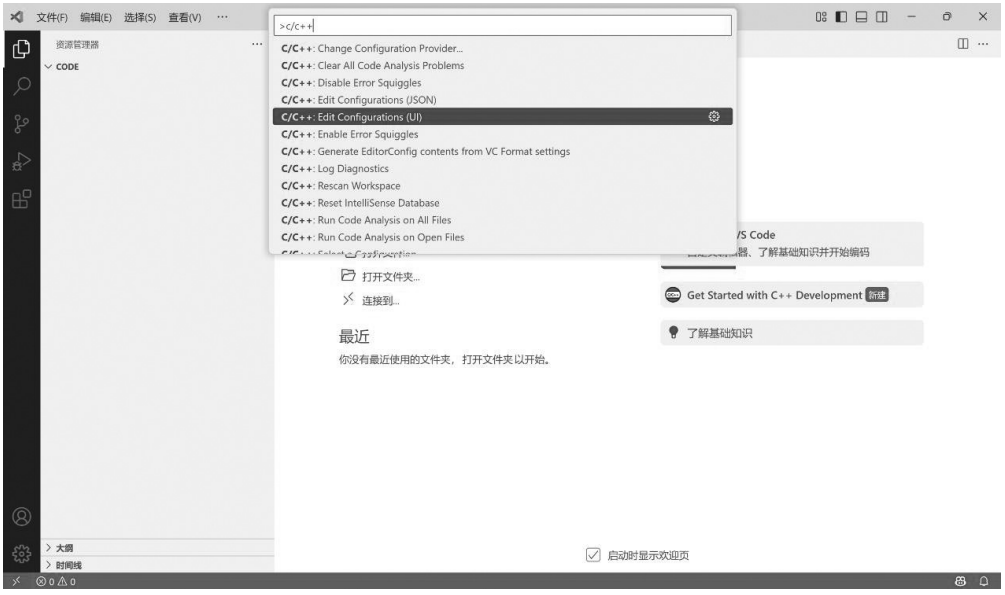

图 1.48　打开 C/C++编辑器配置

（2）指定编译器路径为【D：\ VSCode \ mingw64 \ bin \ gcc. exe】，指定 IntelliSense 模式为【windows-gcc-x64】，无需保存，此时会发现资源管理器【CODE】目录下多了一个【. vscode】目录，如图 1. 49 所示。

图 1. 49　配置编译器路径和 IntelliSense 模式

3. 创建 hello. c 文件

（1）单击【新建文件】按钮，输入文件名为“hello. c”并按下回车键（注意：新建的文件必须和 . vscode 目录在同一层级），如图 1. 50 所示。

图 1. 50　新建“hello. c”文件

（2）在代码区域输入“hello. c”文件的程序代码，如图 1. 51 所示，并使用 Ctrl+S 保存文件！

```
#include <stdio. h>
int main( )
{
    printf( " Hello, World! \n" );
    return 0;
}
}
```

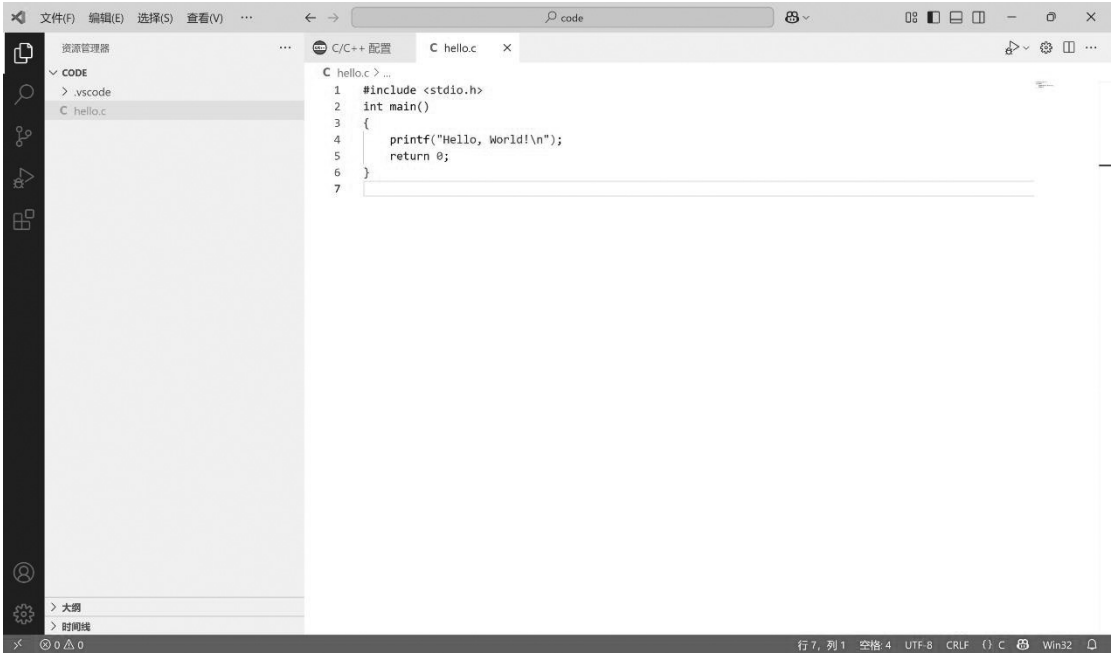

图 1. 51　建立“hello. c”文件

4. 配置构建任务

（1）按下 F1 快捷键，输入“tasks”，在下拉菜单中选择【任务：配置默认测试任务】选项，如图 1. 52 所示。

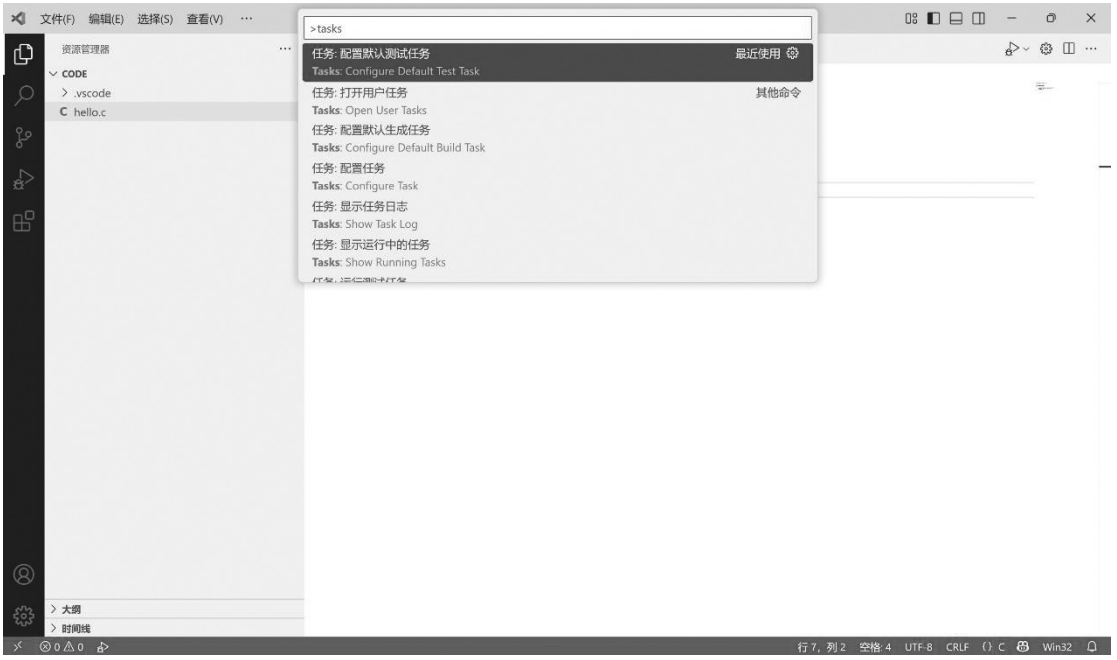

图1.52　选择【任务：配置默认测试任务】选项

（2）单击【C/C++ gcc.exe 生成活动文件】选项，如图 1.53 所示。

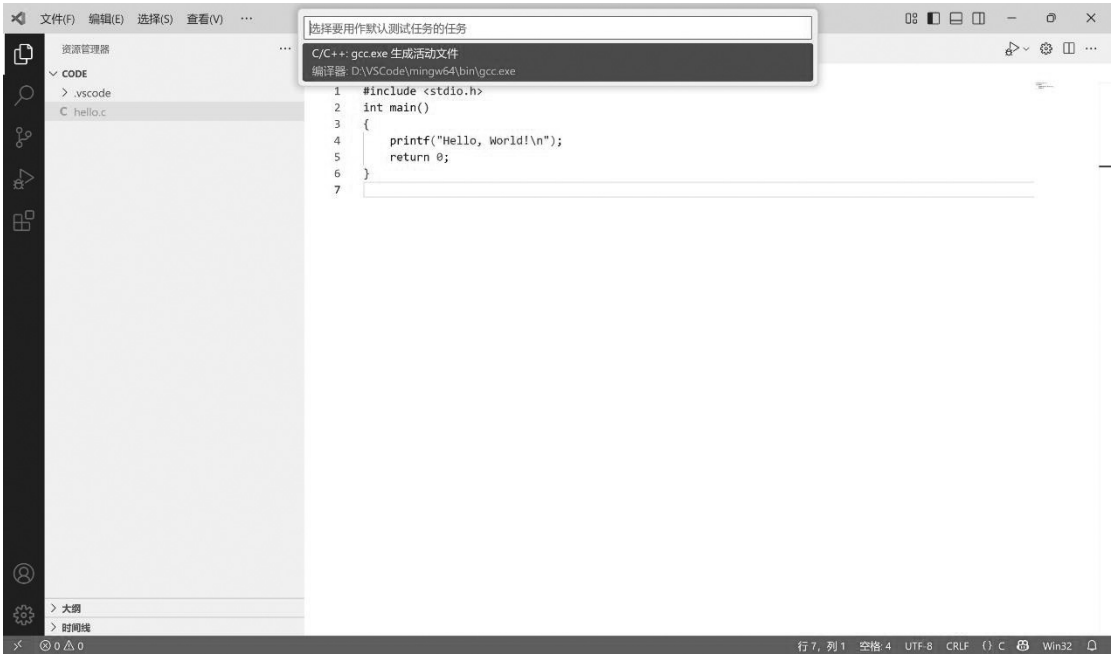

图 1.53　使用 gcc.exe 生成活动文件

（3）自动生成了“tasks.json”文件，如图 1.54 所示。

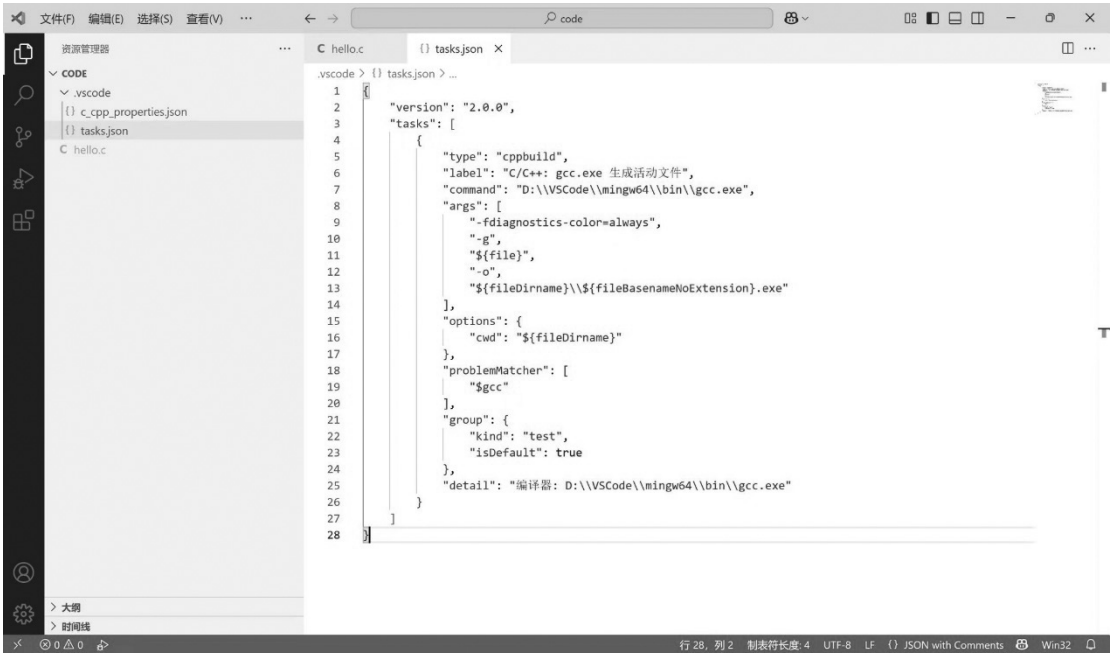

图 1.54　自动生成的“tasks.json”文件

（4）修改“tasks.json”文件中原本的“test”为“build”，如图 1.55 所示，可以更准确地反映这个任务的实际功能，即编译（构建）代码。使用 Ctrl + S 进行保存！“tasks.json”文件完整代码如下：

```
{
    "version": "2.0.0",
    "tasks": [
        {
            "type": "cppbuild",
            "label": "C/C++: gcc.exe 生成活动文件",
            "command": "D:\\VSCode\\mingw64\\bin\\gcc.exe",
            "args": [
                "-fdiagnostics-color=always",
                "-g",
                "${file}",
                "-o",
                "${fileDirname}\\${fileBasenameNoExtension}.exe"
            ],
            "options": {
                "cwd": "${fileDirname}"
            },
```

```
            "problemMatcher": [
                "$gcc"
            ],
                "group": {
                    "kind": "build",
                    "isDefault": true
                },
            "detail": "编译器: D:\\VSCode\\mingw64\\bin\\gcc.exe"
        }
    ]
}
```

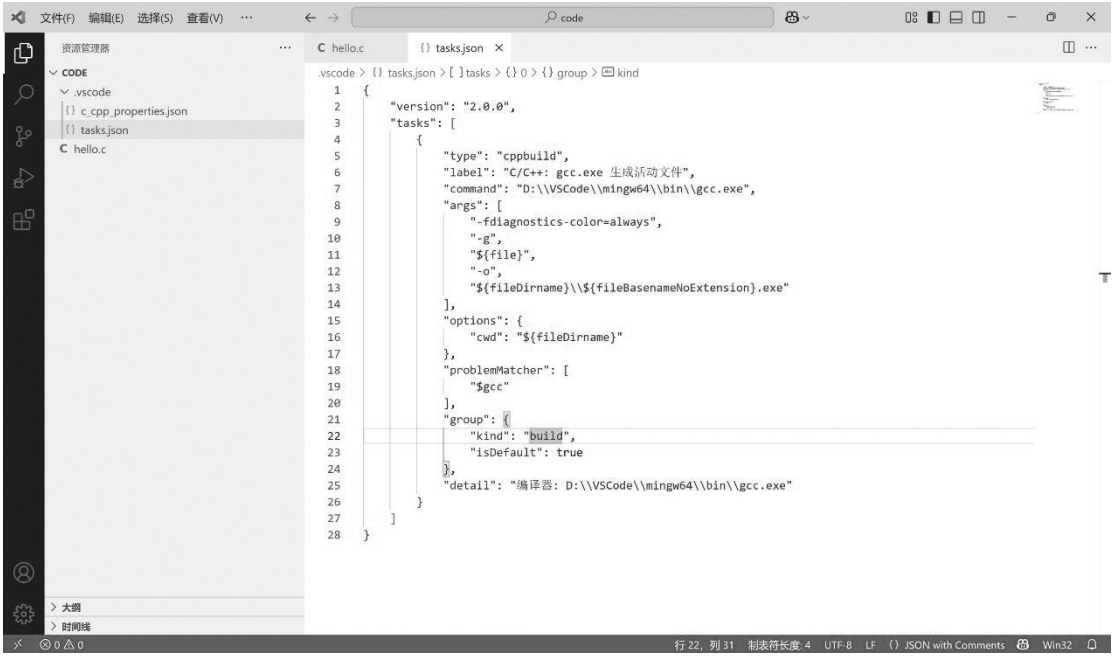

图 1.55　修改后的“tasks.json”文件

5. 配置调试文件

（1）单击侧边导航栏的【运行和调试】按钮，并单击【创建 launch.json 文件】，在弹出的列表中选择【C++（GDB/LLDB）】选项，如图 1.56 所示。

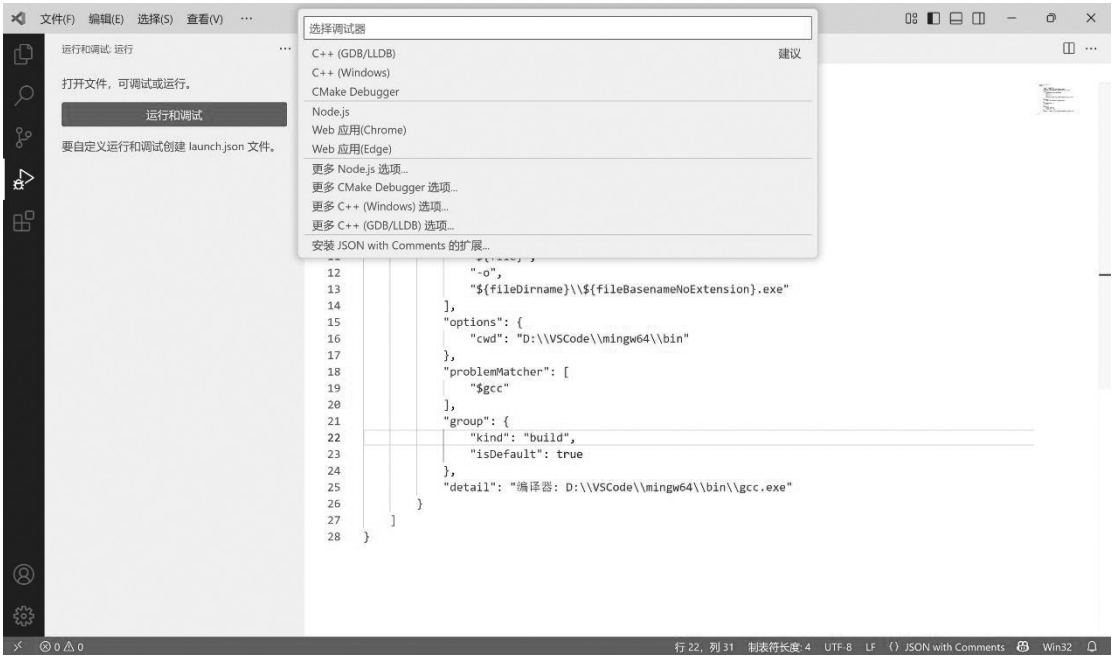

图 1.56　创建 launch.json 文件

（2）此时自动生成了 launch.json 文件，在弹出的列表中选择【C/C++：（gdb）启动】选项，如图 1.57 所示。

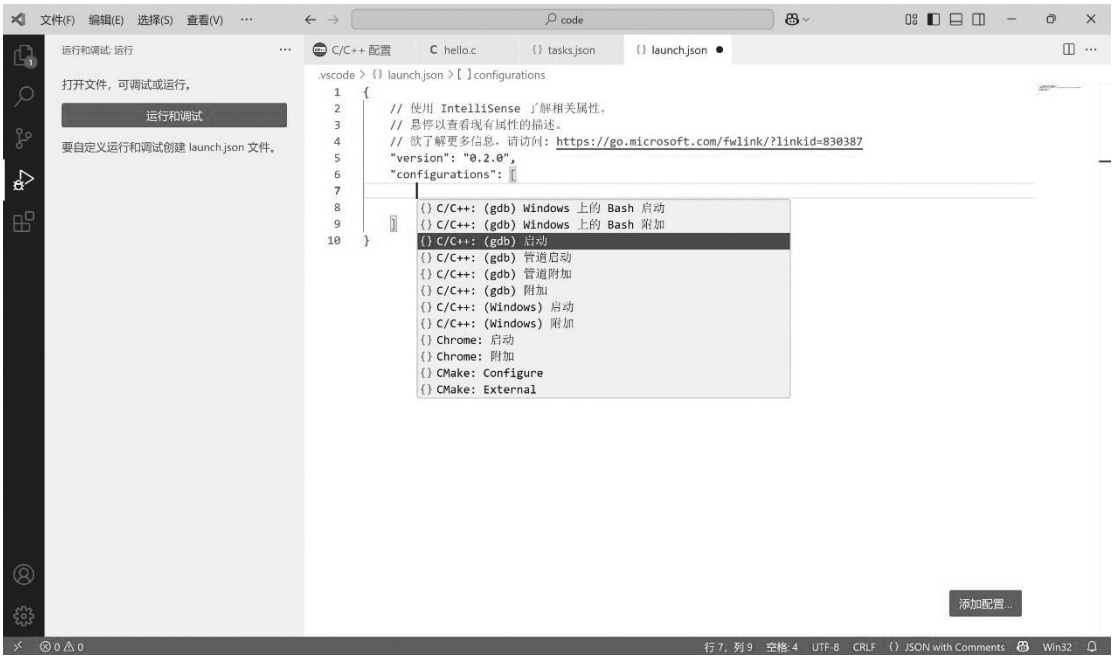

图 1.57　选择【C/C++：（gdb）启动】选项

（3）自动生成了相关的调试配置信息，如图 1.58 所示。

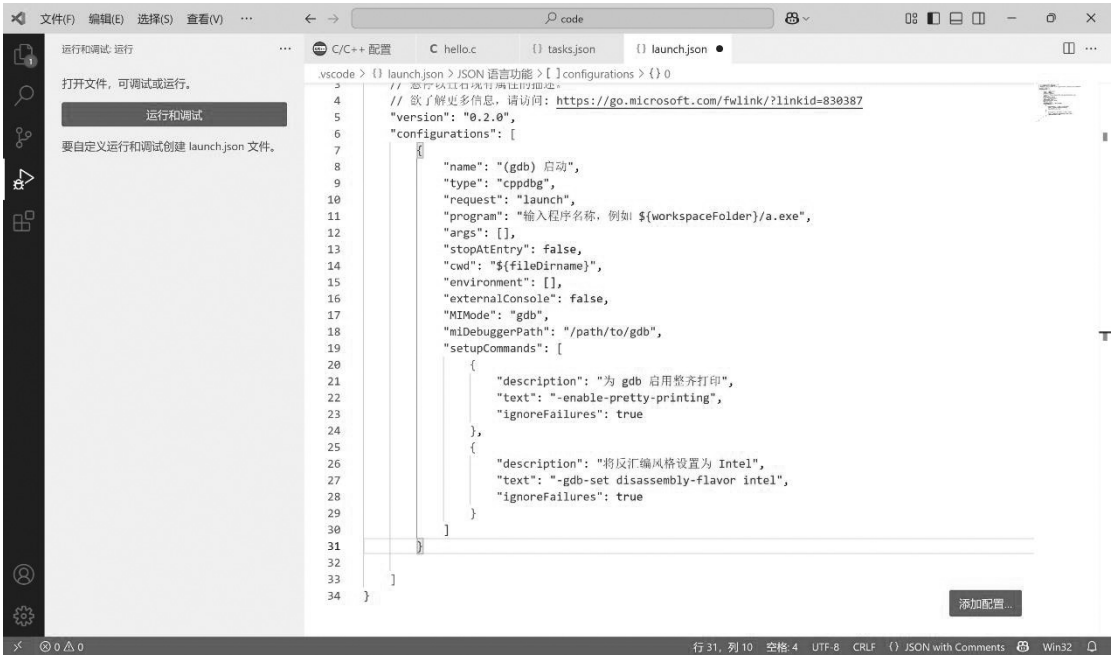

图 1.58　自动生成的“launch.json”文件

（4）修改调试配置文件，对 launch.json 文件进行以下修改：

1）将 program 对应的值改为【${workspaceFolder}/${fileBasenameNoExtension}.exe】，这是为了告诉调试器，调试时应该运行的程序文件位于当前工作区目录下，并且文件名与当前编辑的源代码文件名相同（但扩展名是 .exe）。这样，无论你在工作区中打开哪个源代码文件进行调试，VS Code 都能自动找到对应的可执行文件，而无需手动指定路径。

2）将 miDebuggerPath 对应的值改为【D:/VSCode/mingw64/bin/gdb.exe】，这是为了指定调试器（GDB）的路径。GDB 是 MinGW 提供的调试工具，VS Code 需要知道它的具体位置才能在调试时调用它。通过设置这个路径，确保 VSCode 能够正确启动 GDB 来调试你的程序。

3）在倒数第二个右半边中括号后添加一个英文状态下的逗号【,】，并另起一行添加以下内容：【"preLaunchTask": "C/C++: gcc.exe 生成活动文件"】，这是为了在启动调试之前自动执行一个任务，即使用 GCC 编译器编译当前活动的源代码文件。这个任务的名称（"preLaunchTask"对应的值）需要与 tasks.json 文件中定义的任务名称（"label"对应的值）一致。这样，当你单击调试按钮时，VS Code 会先自动编译源代码，生成可执行文件，然后再启动调试器运行这个文件。如果这个任务名称不匹配，VS Code 将无法找到正确的编译任务，从而导致调试失败。

完成修改后，按 Ctrl+S 保存文件。

“launch.json”文件完整代码如下：

```
{
    // 使用 IntelliSense 了解相关属性。
    // 悬停以查看现有属性的描述。
    // 欲了解更多信息,请访问: https://go.microsoft.com/fwlink/?linkid=830387
    "version": "0.2.0",
    "configurations": [
        {
            "name": "(gdb) 启动",
            "type": "cppdbg",
            "request": "launch",
            "program": "${workspaceFolder}/${fileBasenameNoExtension}.exe",
            "args": [],
            "stopAtEntry": false,
            "cwd": "${fileDirname}",
            "environment": [],
            "externalConsole": false,
            "MIMode": "gdb",
            "miDebuggerPath": "D:/VSCode/mingw64/bin/gdb.exe",
            "setupCommands": [
                {
                    "description": "为 gdb 启用整齐打印",
                    "text": "-enable-pretty-printing",
                    "ignoreFailures": true
                },
                {
                    "description": "将反汇编风格设置为 Intel",
                    "text": "-gdb-set disassembly-flavor intel",
                    "ignoreFailures": true
                }
            ],
            "preLaunchTask": "C/C++: gcc.exe 生成活动文件"
        }
    ]
}
```

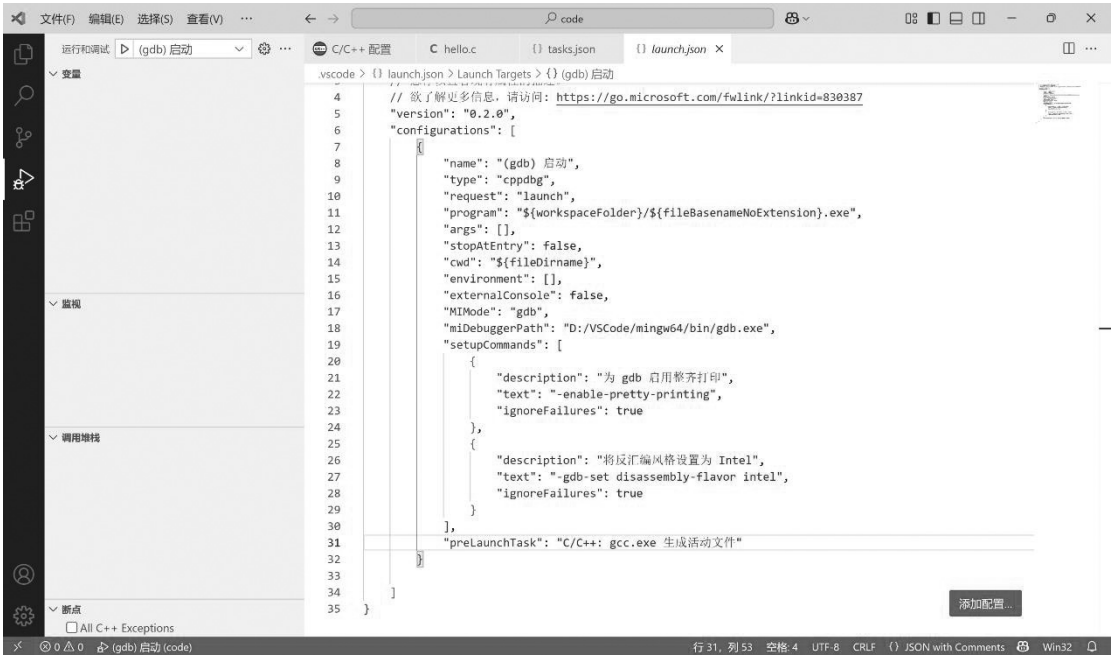

图 1.59　修改后的“launch.json”文件

6. 编译运行代码

（1）选中 hello.c 文件，如图 1.60 所示。

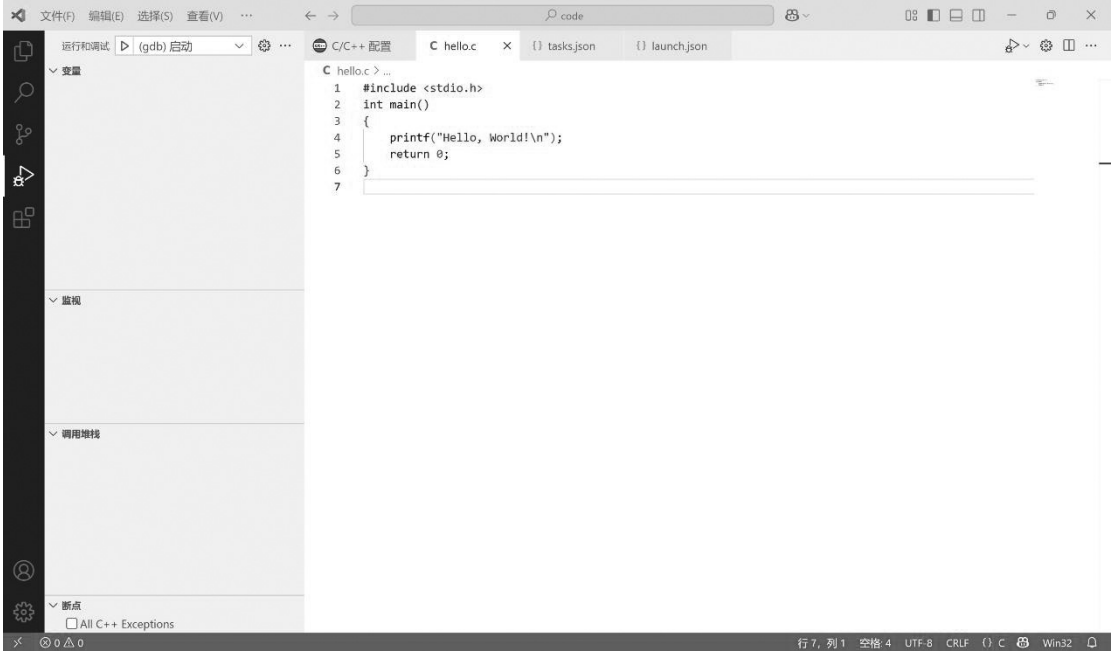

图 1.60　选中 hello.c 文件

（2）单击顶部导航栏最后一个 ··· ，然后单击【运行】【以非调试模式运行】命令，如图 1.61 所示。

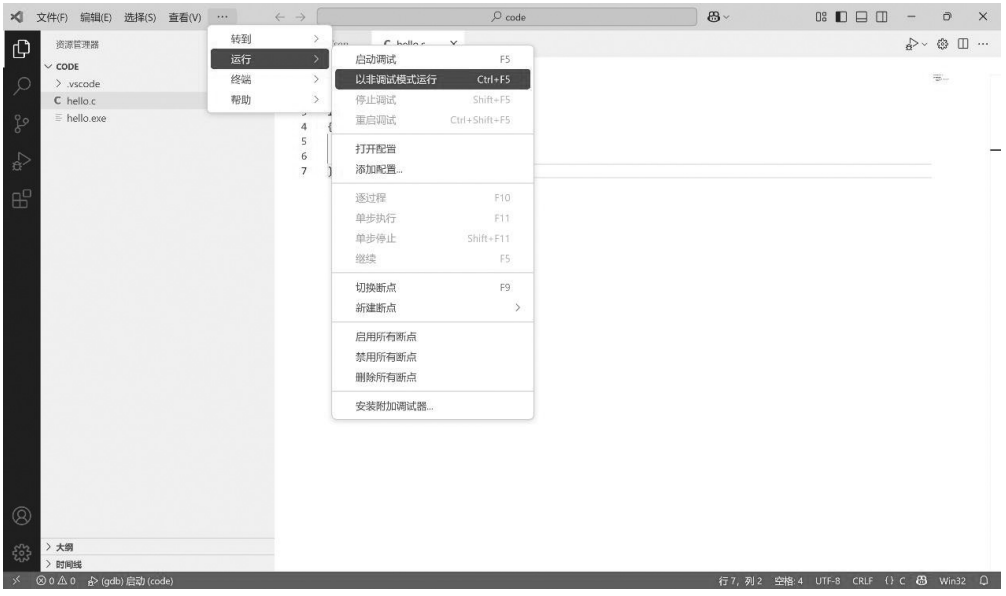

图 1.61　以非调试模式运行 hello.c 文件

（3）选中【终端】选项卡，可以看到程序的执行结果已成功输出，如图 1.62 所示。如果出现了异常报错，请确认 tasks.json 和 launch.json 中的内容是否正确。

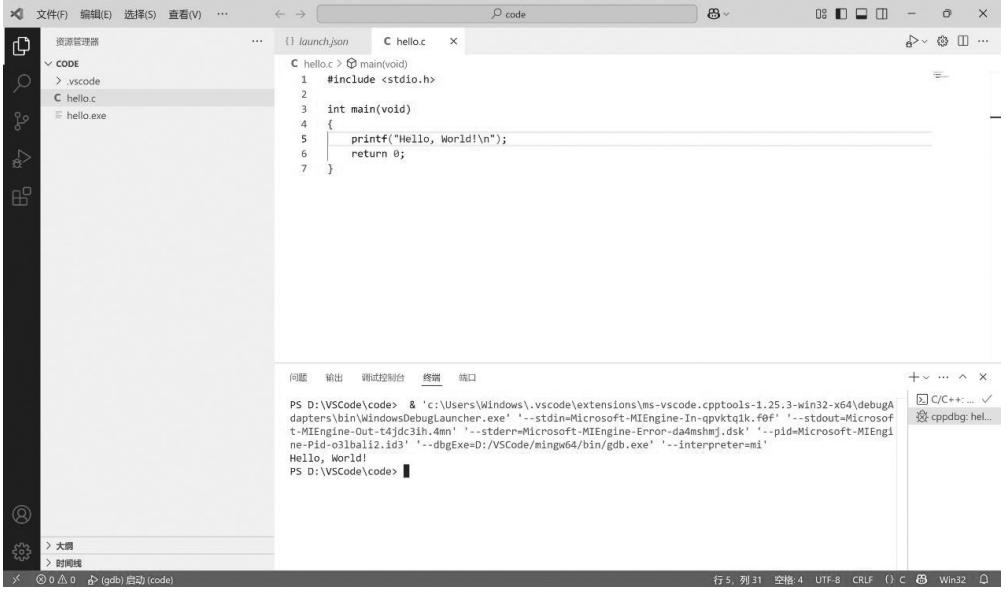

图 1.62　“hello.c”文件的运行结果

四、深入训练

安装 Visual Studio Code，在 VS Code 中配置 C 语言开发环境。在 VS Code 集成开发环境中输入【例 1.1】的程序代码并调试运行。

【例 1.1】求两个整数的和并在屏幕上显示结果。

拓展阅读

程序与人生
——不积跬步，无以至千里

“Hello C!”运行了自己的第一个 C 语言程序，就像在程序世界里说出了自己的第一句话一样。其实，计算机和计算机程序语言都是我们人类利用自己的智慧设计和开发出来的，它体现着人类惯有的逻辑思维，更蕴含着人类几千年的文化积累。

中国古语云：“千里之行，始于足下”“不积跬步，无以至千里；不积小流，无以成江海”。意思是说：走一千里路也要从第一步开始，不积累每一步，永远达不到千里的地方。在今天，我们的千里之行由汽车、飞机、高铁等各类高科技的交通工具来实现，这句话仿佛已距离我们很遥远。但是，我们的每一个成功，又无不包含着其中的深意。一件事情的成功，绝不是偶然，必有一个开始，而这一个又一个的开始，则犹如我们千里之行的“跬步”。

“Hello C!”是我们学习计算机语言的一小步，一个输出语句看似简单，却可以为今后的学习做出积累。在今后的学习中，我们肯定会遇到类似环境安装失败、程序调试出错、功能无法实现等困难，但“千里之行，始于足下”，只有我们主动去迈动自己的步伐，才会到达那遥远的彼岸；学习中遇到的困难，只有你主动去解决，才会发现其中的乐趣所在。如果无论难易，我们都不忘初心、坚持不懈，将每一个知识点都熟练掌握、灵活运用，逐渐积累后我们一定会方得始终、学有所成。

李彦宏、马化腾、李开复，这些中国 IT 界的风云人物，他们的成功也不是一蹴而就，万般造诣，皆是千锤百炼，厚积薄发。作为计算机专业的大学生，我们做事一定要脚踏实地，一步一个脚印，不畏艰难，不怕曲折，只要坚韧不拔地朝着自己设立的目标干下去，最终一定会学有所获、学有所成。

项目实训

一、实训目的

1. 掌握 C 程序的基本结构。
2. 熟悉 Visual Studio Code（简称 VS Code）集成开发环境。
3. 能够在 VS Code 中配置 C 语言开发环境。
4. 能熟练使用 VS Code 集成开发环境调试 C 语言程序。

二、实训任务

1. 下载安装 VS Code，并配置 C 语言开发环境，学习该软件的使用。

2. 在 VS Code 集成开发环境中输入【例 1.2】并调试运行。

【例 1.2】从键盘上输入两个整数，比较两数，将大的数输出。

3. 编写一个 C 语言程序，要求显示如下结果。

＊＊＊$$$＊＊＊###＊＊＊@@@ ＊＊＊###＊＊＊$$$＊＊＊

This is a C program.

4. 已知长方形的长和宽，编写程序求这个长方形的面积并输出。

注意：自己先分析程序的运行结果之后再运行该程序，比较自己的判断与屏幕上的结果是否一致，如果有差异，再想想错误出现在什么地方。这种做法可以逐步训练自己理解程序和分析程序的能力。

项目练习

扫码玩闯关游戏

1. 填空题

（1）C 语言规定，一个程序必须有一个主函数，其函数名为_____。

（2）一般而言，一个 C 语言程序的执行是从_____ 开始，到_____ 结束。

（3）一个 C 语言程序是由____ 组成的。

（4）每个 C 语句必须以____ 号结束。

（5）C 语言规定，源程序的扩展名是_____，可执行文件的扩展名是_____ 。

2. 判断题（判断下列叙述的正确性，正确的请打“√”，错误的请打“×”）

（1）C 语言的源程序是由函数组成的。（　　）

（2）C 语言的任何一个源程序中必须有一个主函数。（　　）

（3）VS Code 不可以开发 C 语言程序。（　　）

（4）VS Code 需要安装插件才可以运行 C 语言程序。（　　）

3. 程序阅读题

```
(1)#include <stdio.h>
int main()
{
    printf("I love China! \n");
    printf("We are students. \n");
    return 0;
}
```

程序的运行结果为_____。

```
(2)#include <stdio.h>
int main()
{
    int a;
    a = 5;
    printf("%d\n", a + 1);
    return 0;
}
```

程序的运行结果为_____。

4. 编程题

已知立方体的长、宽、高分别是 10 厘米、20 厘米、15 厘米，编写程序求立方体的体积。

项目二　基本数据类型、运算符和表达式

在项目一中已经知道，使用 C 语言编写程序，必须在程序中做好两件事情：一是数据的描述；二是数据的操作，即数据的加工与处理。前者通过数据定义语句来实现，后者通过若干程序语句，包括用各种运算符构成的表达式来实现。本项目主要介绍 C 语言的基本数据类型（除枚举类型外），其他数据类型在后续项目中再详细介绍。另外，本项目还将详细介绍变量的存储属性的说明方法、运算符以及表达式的构成方法。

学习目标

1. 了解基本数据类型及其常量的表示方法。
2. 掌握变量的定义及初始化方法。
3. 掌握运算符和表达式的概念。
4. 理解自动类型转换和强制类型转换。
5. 能够将一般的数学算式转化为 C 语言表达式。

任务一　求圆的面积和周长

知识要点：基本数据类型、常量说明与变量定义。

一、任务分析

已知圆的半径，求圆的面积和周长。

在计算圆的面积与周长时，要用到圆周率。众所周知，圆周率 π 的值是固定不变的，也就是说 π 是一个常量，而圆的半径是可以不断变化的，是一个变量。

二、必备知识与理论

1. 数据类型概述

所谓数据类型，是按被定义变量的性质、表示形式、占据存储空间的多少、构造特点来划分的。

C 语言的数据类型分为简单（基本）数据类型和复杂（构造）数据类型两种。简单数据类型是由系统自动规定数据存储空间的大小，它包括整数类型（整型）、实数类型（实型）、字符类型（字符型）和枚举类型。而复杂数据类型是由用户按照一定规则来决定数据占用空间的大小，包括数组类型、结构体类型、共用体类型。

C 语言的数据类型如图 2.1 所示。

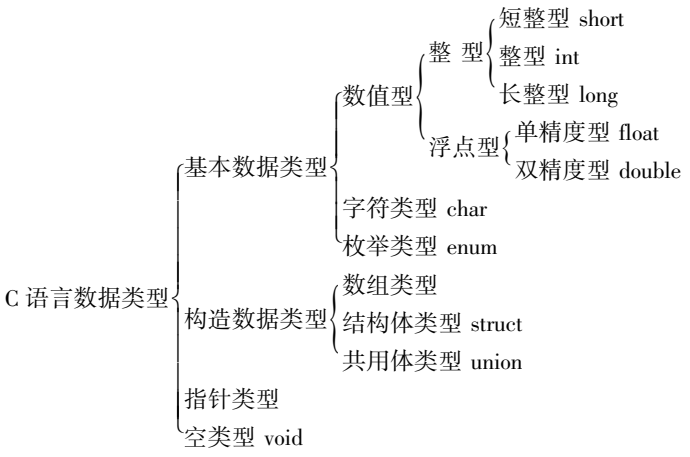

图 2.1　C 语言的数据类型

（1）基本数据类型。基本数据类型最主要的特点是，其值不可以再分解为其他类型。

（2）构造数据类型。构造数据类型是根据已定义的一个或多个数据类型用构造的方法来定义的。在 C 语言中，构造数据类型有数组类型、结构体类型、共用体（联合）类型三种。

（3）指针类型。指针类型是一种既特殊又具有重要作用的数据类型。其值用来表示某个变量在内存储器中的地址。虽然指针变量的取值类似于整型量，但这是两个类型完全不同的量，因此不能混为一谈。

（4）空类型。在调用函数值时，通常应向调用者返回一个函数值。这个函数值是具有一定的数据类型的，应在函数定义及函数说明中给予说明。但也有一类函数调用后并不需要向调用者返回函数值，这种函数可以定义为空类型。其类型说明符为 void。

这里只介绍 C 语言的基本数据类型说明，其他类型在以后项目中陆续介绍。

2. 常量

在程序执行过程中其值始终不变的量称为常量。它们可与数据类型结合起来分类。例如，可以分为整型常量、实型常量、字符常量等。在程序中，常量可以不经说明直接使用。

（1）整型常量。在 C 语言中，整型常量可以用三种形式来表示：

1）八进制整型常量。八进制整型常量必须以 0 开头，即以 0 作为八进制整型常量的前缀，数码取值为 0～7。八进制整型常量通常是无符号数。例如：015（十进制为 13）、0101（十进制为 65）。

2）十进制整型常量。十进制整型常量没有前缀，其数码为 0～9。例如：237、−568。

3）十六进制整型常量。十六进制整型常量的前缀为 0X 或 0x，其数码取值为 0～9，A～F 或 a～f。例如：0X123（十进制为 291）、0XFFFF（十进制为 65535）。

在程序中是根据前缀来区分各种进制数的。因此在书写整型常量时要保证前缀正确，以免造成结果出错。

4）整型常量的后缀。除了基本型的整型常量外，还有长整型常量和无符号整型常量。长整型常量是用后缀“L”或“l”来表示的。例如：158L（十进制为 158）。长整型常量 158L 和基本整型常量 158 在数值上并无区别。

无符号整型常量也可用后缀表示，其后缀为“U”或“u”。例如：358u、0x38Au、235Lu均为无符号整型常量。

（2）实型常量。实型常量也称为实数或浮点数。例如：-1.89、1.23456e5为实型常量。在C语言中，实型常量只采用十进制。它有两种表示形式，即小数形式和指数形式。

小数形式由数码0~9和小数点组成。例如：3.1415926、0.0、5.0、-267.8230等均为合法的实型常量。

注意：实型常量必须有小数点。

指数形式由十进制数加阶码标志“e”或“E”及阶码（只能为整数，可以带符号）组成。其一般形式为：

aEn或aen（a为十进制数，n为十进制整数），其值为$a\times10^n$。例如：2.1E5（等于2.1×10^5）。

注意：字母e（或E）的前后必须有数字，且e（或E）后面的指数必须为整数。

（3）字符常量。字符常量是用单引号括起来的一个字符。例如：‘a’‘A’‘?’‘=’都是合法的字符常量。

转义字符。C语言还允许用一种特殊的字符常量，即以反斜线“\”开头，后跟一个或几个字符。由于转义字符具有特定的含义，不同于字符原有的意义，故称“转义”字符。

例如，在前面各例中出现的printf()函数的格式串中用到的“\n”就是一个转义字符，其意义是“回车换行”。

转义字符主要用来表示那些用一般字符不便于表示的控制代码。常用的转义字符及其含义见表2.1。

表2.1　常用转义字符及其含义

转义字符	转义字符的含义	转义字符	转义字符的含义
\n	回车换行	\\	反斜线符（\）
\t	横向跳到下一制表位置	\'	单引号符
\v	竖向跳格	\"	双引号符
\b	退格	\a	鸣铃
\r	回车	\ddd	1~3位八进制数所代表的字符
\f	走纸换页	\xhh	1~2位十六进制数所代表的字符

表2.1中，\ddd和\xhh可以表示任何可输出的字母字符、专用字符、图形字符和控制字符。ddd和xhh分别为八进制和十六进制的ASCII代码。例如：'\101'表示ASCII值为65的字符'A','\012'表示“换行”等。

（4）字符串常量。字符串常量是由一对双引号引括起来的字符序列。例如："HINA","C program.","$ 12.5"等都是合法的字符串常量。

初学者容易将字符常量与字符串常量混淆。'a'是字符常量,"a"是字符串常量。那么两者有什么区别呢？C语言规定，在每一个字符串结尾自动加一个字符串结束标志'\0'，以便系统据此判断字符串是否结束。'\0'是一个ASCII码值为0的字符，也就是空操作字符，

即它不引起任何控制动作，也不是一个可显示的字符。例如，字符串"a"在内存中的实际存放形式为：

a	\0

注意：'\0'是系统自动加上的。因此,"a"实际包含了两个字符:'a'和'\0'，故不能把"a"赋给一个字符变量。

（5）符号常量。用一个标识符来代表常量，即给某个常量取个有意义的名字，称为符号常量。符号常量必须先定义再使用。

定义形式：**#define 标识符 常量**

例如：#define PI 3.1415926

其中#define 是一条预处理命令（预处理命令都以“#”开头），称为宏定义命令，其功能是把该标识符定义为其后的常量值。一经定义，以后在程序中所有出现该标识符的地方均替换为该常量值。

为了区别程序中的符号常量名与变量名，习惯上符号常量的标识符用大写字母，变量标识符用小写字母，以示区别。

3. 变量

在程序执行过程中其值可变的量称为变量。

一个变量必须有一个名字，变量名在程序运行时不会改变，而变量值可以发生变化。

变量名是一种标识符，必须遵守标识符的命名规则。

变量必须“先定义后使用”，定义时指明数据类型，在编译时为其分配相应的存储单元。格式为：

类型标识符 变量名

如：int a,b;

　　float x,y;

变量的数据类型是由其值决定的，可分为整型变量、实型变量、字符变量等。下面要具体讲到不同数据类型的变量。


扫码看视频

注意：C 语言规定：变量都必须先说明后使用。只有这样，编译时才能为其分配相应的存储单元，也才能以此来检查变量所进行的运算是否合法。定义变量时还要尽量做到“见名知意”。

初学变量时要特别注意区分变量名、变量值、变量地址，如图 2.2 所示。

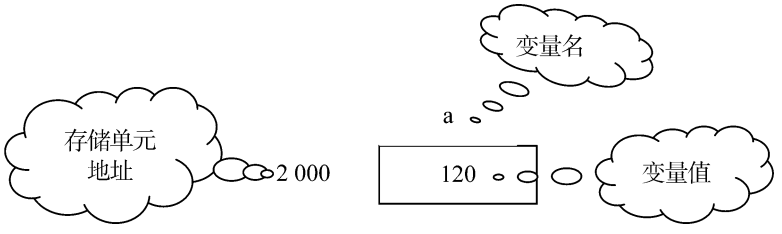

图 2.2　变量名、变量值、变量地址的关系

（1）整型变量。整型变量可分为基本型、短整型、长整型和无符号型 4 种。

1）基本型。类型说明符为 int，在内存中占 4 个字节，其取值为基本整常数。

2）短整型。类型说明符为 short int 或 short，在内存中占 2 个字节，其取值为短整常数。

3）长整型。类型说明符为 long int 或 long，在内存中占 4 个字节，其取值为长整常数。

4）无符号型。类型说明符为 unsigned，无符号型又可与上述 3 种类型匹配而构成无符号基本整型、无符号短整型、无符号长整型，见表 2.2。

表 2.2　整型变量的字节数及表示范围

数据类型	类型声明符	数的范围	分配字节数
整型	int	$-2147483648\sim2147483647$，即$-2^{31}\sim2^{31}-1$	4
无符号整型	unsigned［int］	$0\sim4294967295$，即$0\sim2^{32}-1$	4
短整型	short［int］	$-32768\sim32767$，即$-2^{15}\sim2^{15}-1$	2
无符号短整型	unsigned short	$0\sim65535$，即$0\sim2^{16}-1$	2
长整型	long［int］	$-2147483648\sim2147483647$，即$-2^{31}\sim2^{31}-1$	4
无符号长整型	unsigned long	$0\sim4294967295$，即$0\sim2^{32}-1$	4

各种无符号类型量所占的内存空间字节数与相应的有符号类型量相同。但由于省去了符号位，故不能表示负数，但可存放的数的范围比一般整型变量中数的范围扩大一倍。


扫码看视频

整型变量说明的格式为：**类型标识符 变量名 1[,变量名 2,…];**

例如：int a,b,c;　　/* a,b,c 为整型变量 */

　　long x,y;　　/* x,y 为长整型变量 */

（2）实型变量。实型变量分为两类：单精度型和双精度型，其类型说明符分别为 float（单精度说明符）和 double（双精度说明符）。单精度型占 4 个字节（32 位）内存空间，其数值范围为 3.4E-38～3.4E+38，只能提供 7 位有效数字。双精度型占 8 个字节（64 位）内存空间，其数值范围为 1.7E-308～1.7E+308，可提供 16 位有效数字。

其说明格式为：**类型标识符 变量名 1[,变量名 2,…];**

例如：float x,y,z;　/* x,y,z 为单精度实型量 */

　　double a,b,c;/* a,b,c 为双精度实型量 */

注意：实型常量不分单、双精度。一个实型常量可以赋给一个 float 或 double 型变量，根据变量的类型截取实型常量中相应的有效数字。

【例 2.1】float 和 double 的应用。

```
#include <stdio.h>
int main()
{
  float a;
  double b;
  a = 3333.33333355;
  b = 3333.33333355;
  printf("a=%f\nb=%f\nb=%.8f\n", a, b, b);
  return 0;
}
```

程序运行结果如图 2.3 所示。

```
a=3333.333252
b=3333.333334
b=3333.33333355
```

图 2.3　【例 2.1】程序的运行结果

（3）字符型变量。字符型变量用来存放字符常量，即单个字符，不能存放字符串。

字符型变量的类型说明符是 char。字符变量类型说明的格式和书写规则都与整型变量相同。

其说明格式为：**类型标识符 变量名 1[,变量名 2,…];**

例如：char c1,c2;　/* c1,c2 被说明为字符型变量 */

系统给每个字符变量分配一个字节的内存空间，因此只能存放一个字符。字符值是以 ASCII 码的形式存放在变量的内存单元之中的。如果对字符变量 c1，c2 赋予'A'和'B'值，即 c1='A'，c2='B'。由于字符 A 的十进制 ASCII 码是 65，字符 B 的十进制 ASCII 码是 66。实际上在变量 c1，c2 的两个单元内存放的是 65 和 66 的二进制代码，如图 2.4 所示。

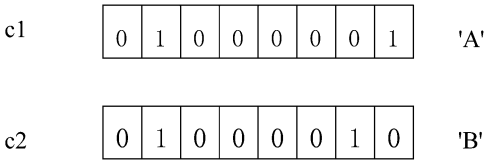

图 2.4　字符'A'、'B'在内存中的存放形式

所以字符型也可以当成是整型量。C 语言允许对整型变量赋以字符值，也允许对字符变量赋以整型值。在输出时，允许把字符变量按整型量输出，也允许把整型量按字符量输出。由于整型量占四个字节内存，字符量占一个字节内存，当整型量按字符型量处理时，只有低 8 位字节参与处理。

【例 2.2】整型量与字符型量的混合使用。

```
#include <stdio.h>
int main()
{
   char c1, c2, c3, c4;
   c1 = 65; c2 = 66;
   c3 = 'A'; c4 = 'B';
   printf("%c,%c,%d,%d\n", c1, c2, c3, c4);
   return 0;
}
```

程序运行结果如图 2.5 所示。

```
A,B,65,66
```

图 2.5　【例 2.2】程序的运行结果

4. 不同类型数据的混合运算

整型、实型和字符型数据间可以混合运算。在进行混合运算时，不同类型的数据要转换成同一类型。转换的方法有两种：一是自动转换，二是强制转换。

（1）类型的自动转换。自动转换发生在不同类型的数据混合运算时，由编译系统自动完成。图 2.6 表示了类型自动转换的规则。

图 2.6 中，横向向左的箭头表示必定发生的转换，如字符型数据必先转换成无符号整型，单精度型必先转换成双精度型等。纵向箭头表示当运算对象为不同类型时转换的方向。注意箭头方向只表示数据类型级别的高低，由低向高转换。不要理解为整型先转换成无符号整型，再转换成长整型，再转换成无符号长整型，再转换成双精度型。如果一个整型数据与一个双精度型数据运算，是直接将整型转换成双精度型。

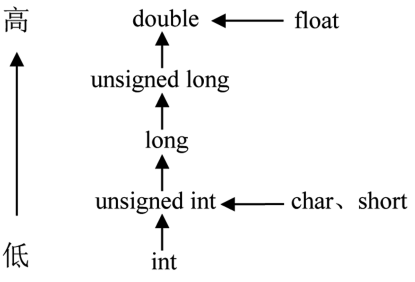

图 2.6　类型转换方向

（2）类型的强制转换。强制类型转换是通过类型转换运算来实现的。强制类型转换的格式为：

(类型说明符)表达式

其功能是把表达式的运算结果强制转换成类型说明符所表示的类型。

例如：　(float)x　　/*把 x 转换为实型 float */

(int)(x+y)/*把 x 与 y 的和转换为整型 */

在使用强制类型转换时应注意以下问题：

（1）类型说明符和表达式都必须加括号（单个变量可以不加括号），如把(int)(x+y)写成(int)x+y,则只将 x 转换成 int 型，然后再与 y 相加。

（2）无论是强制转换或是自动转换，都只是为了本次运算的需要而对变量的数据长度进行的临时性转换，原来变量的类型并未改变。

【例 2.3】强制类型转换的应用。

```
#include <stdio.h>
int main()
{
    float f = 5.75;
    printf("(int)f=%d,f=%f\n", (int)f, f);
    return 0;
}
```

程序运行结果如图 2.7 所示。

```
(int)f=5,f=5.750000
```

图 2.7　【例 2.3】程序的运行结果

本例表明，f 虽然强制转换为 int 型，但只在运算中起作用，这种转换是临时的，而 f 本身的类型并没改变。

三、任务实施


扫码看视频

本任务是在已知圆半径的情况下，计算圆的面积和周长，利用求面积和周长的数学公式即可完成计算。在计算中用到圆周率 π，我们知道圆周率 π 是一个固定值，所以圆周率说明为常量，半径定义为变量。

```
#include <stdio.h>
#define PI 3.14                              /* 定义常量 PI 代表圆周率 */
int main()
{
   int r;                                    /* 定义整型变量 r 表示圆半径 */
   float area, girth;                        /* 定义实型变量 area、girth 表示圆面积和周长 */
   r = 10;                                   /* 给变量 r 赋值 10 */
   area = PI * r * r;                        /* 计算圆面积 */
   girth = 2 * PI * r;                       /* 计算圆周长 */
   printf("area=%f,girth=%f\n", area, girth); /* 输出计算结果 */
   return 0;
}
```

程序运行结果如图 2.8 所示。

```
area=314.000000,girth=62.799999
```

图 2.8　求圆的面积和周长的运行结果

四、深入训练

1. 某物体的质量 m 为 5kg，编写一个 C 语言程序，求它的重力 G。已知重力 $G=mg$

（其中 $g=9.8\text{N/kg}$）。

提示：g 说明为常量，m 说明为变量。

2. 输入小写字母，输出对应的大写字母。

提示：小写字母 a 的 ASCII 值为 97，大写字母 A 的 ASCII 值为 65。

任务二　计算表达式的值

知识要点：算术运算符与算术表达式、赋值运算符与赋值表达式、关系运算符与关系表达式、逻辑运算符与逻辑表达式。

一、任务分析

计算并输出 x 的值：$x=\dfrac{-b+5a^2}{2a}$

（1）如何将数学表达式转换为合法的 C 语言表达式？

（2）确定 a，b 和 x 的数据类型。

二、必备知识与理论

C 语言中把除了控制语句和输入输出以外的几乎所有的基本操作都作为运算符处理。其运算符和表达式数量之多，在高级语言中是少见的。正是丰富的运算符和表达式使 C 语言功能十分完善，这也是 C 语言的主要特点之一。

C 语言中，运算符的优先级共分为 15 级。1 级最高，15 级最低（见附录Ⅲ）。C 语言的运算符不仅具有不同的优先级，还有不同的结合性。在表达式中，优先级较高的先于优先级较低的进行运算，而当一个运算量两侧的运算符优先级相同时，则按运算符的结合性所规定的结合方向自左向右或自右向左进行运算。这种结合性是其他高级语言的运算符所没有的，因此也增加了 C 语言的复杂性。

C 语言的运算符可分为 10 类，见表 2.3。

表 2.3　C 语言的运算符

运算符种类	运算符
算术运算符	+ - * / % ++ --
关系运算符	> >= < <= == !=
逻辑运算符	! && ||
位操作运算符	<< >> & | ^ ~
赋值运算符	= += -= *= /= %= 等
条件运算符	? :

续表

运算符种类	运算符
逗号运算符	,
指针运算符	* &
求字节数运算符	sizeof
其他运算符	( ) [ ] 等

本任务只介绍最常用的算术运算符、赋值运算符、关系运算符、逻辑运算符、逗号运算符及其相应的表达式。其他运算符将在后续任务中介绍。

1. 算术运算符与算术表达式

算术运算符包括基本算术运算符和自增、自减运算符，其中基本算术运算符常简称为算术运算符。

（1）基本算术运算符。用于各类数值运算，包括加（+）、减（-）、乘（*）、除（/）、求余（%，或称模运算），共5种，具有左结合性。

双目运算符是有两个运算量参与运算的运算符。如a+b，4-8，c/5等都是有两个量参加运算。

双目运算符中的加（+）、减（-）、乘（*）运算与普通的算术运算中的加法、减法、乘法相同，这里不再解释。

除法运算符“/”是双目运算符。当参与运算量均为整型时，结果也为整型，舍去小数，如5/2的值为2，而不是2.5；如果运算量中有一个是实型，则结果为双精度实型，如5.0/2的值为2.5。

求余运算符（模运算符）“%”是双目运算符，要求参与运算的量必须为整型。求余运算的结果等于两数相除后的余数，一般情况下，所得余数与被除数符号相同。例如，5%2=1，10%5=0，8%-5=3。

（2）自增（++）、自减（--）运算符。

自增运算符（++）的功能是使变量的值自增1，自减运算符（--）的功能是使变量的值自减1。它们均为单目运算，都具有右结合性。自增、自减运算符只能用于变量，而不能用于常量或表达式，如6++或(a+b)++都是不合法的。自增、自减运算符可有以下几种形式：

```
++i   /* i值自增1后再参与其他运算 */
--i   /* i值自减1后再参与其他运算 */
i++   /* i参与运算后再将值自增1 */
i--   /* i参与运算后再将值自减1 */
```

对于一个变量i实行前置运算（++i）和后置运算（i++），其运算结果是一样的，即都使变量i的值加1（i=i+1）。但++i和i++的不同之处在于，++i是先执行i=i+1后，再使用i的值；而i++是先使用i的值后，再执行i=i+1。

【例 2.4】自增、自减运算符的应用。

```
#include <stdio.h>int main( )
{
    int i, m, n, j, k;
    i = 10;
    m = i++; n = ++i; j = i--; k = --i;
    printf("%d,%d,%d,%d\n", m, n, j, k);
    return 0;
}
```

程序运行结果如图 2.9 所示。

```
10,12,12,10
```

图 2.9　【例 2.4】程序的运行结果

（3）算术表达式。

用算术运算符和圆括号将操作数（即常量、变量和函数）组合起来的符合 C 语言语法规则的式子，称为 C 算术表达式。例如：

a * b/c-1.5+'a',sin(x)+sin(y),(x+r) * 8-(a+b)/7,(++i)-(j++)+(k--)

单个的常量、变量、函数可以看作是表达式的特例。

C 语言算术表达式的书写形式与数学中表达式的书写形式是有区别的，在使用时要注意以下几点：

- C 语言表达式的乘号不能省略。如：b^2-4ac，应写成 b * b-4 * a * c。
- 只能使用系统允许的标识符。如：πr^2，应写成 3.1415926 * r * r。
- C 语言表达式中的内容必须书写在同一行，不允许有分子分母形式，必要时利用圆括号保证运算的顺序。

如：$\frac{a+b}{c+d}$ 相应的 C 语言表达式为(a+b)/(c+d)。

- C 语言表达式不允许使用方括号和花括号，只能使用圆括号帮助限定运算顺序。可以使用多层圆括号，但左、右括号必须配对，运算时从内层圆括号开始，由内向外依次计算表达式的值。

2. 赋值运算符与赋值表达式

赋值运算符用于赋值运算，分为简单赋值(=)、复合算术运算赋值（+=,-=, *=, /=,%=)和复合位运算赋值(&=,|=,^=, >>=, <<=)3 类。

（1）简单赋值运算。

简单赋值的一般格式为：**变量名=表达式**

含义是将赋值运算符右边表达式的值存放到以左边变量名为标识的存储单元中。

例如：语句 i=3 中的赋值运算符“=”的功能是将整型常量 3 赋给整型变量 i，这样 i 的值就是 3。

说明：

● 赋值运算符左边只能是变量，右边的表达式可以是单一的常量、变量、表达式和函数调用语句。

● 赋值运算符“=”不同于数学中使用的等号，它没有相等的含义。例如：x=x+1 的含义是取出变量 x 中的值加 1 后，再存入变量 x 中。

● 一个赋值表达式中可以有多个赋值运算符，其运算顺序是从右向左结合。例如：x=y=z=1，相当于 x=(y=(z=1))。

● 进行赋值运算时，当赋值运算符两边的数据类型不同时，将由系统自动进行类型转换。转换原则是：赋值运算符右边的数据类型转换成左边的变量类型。转换规则见表 2.4。

表 2.4　赋值运算中数据类型的转换规则

运算符左边的类型	运算符右边的类型	转换说明
float	int	将整型数据转换成实型数据后再赋值
int	float	将实型数据的小数部分截去后再赋值
int、long int	short	值不变
short int	int、long int	右侧的值不能超过左侧数据类型的取值范围，否则将导致意外的结果
unsigned	signed	按原样赋值。但如果数据范围超过相应整型的取值范围，将导致意外的结果
signed	unsigned	

（2）复合赋值运算符。

C 语言规定，可以在赋值运算符“=”之前加上其他运算符，以构成复合赋值运算符。其一般格式为：**变量　双目运算符=表达式**

等价于：**变量=变量 双目运算符 表达式**

扫码看视频

例如：n+=1　　/* 等价于 n=n+1 */

x*=y+1　/* 等价于 x=x*(y+1)*/

C 语言规定，双目运算符可以与赋值运算符一起组合成复合赋值运算符。共有 10 种复合赋值运算符，即+=、-=、*=、/=、%=、<<=、>>=、&=、^=、|=。其中后 5 种是有关位运算的，复合赋值运算符的优先级与赋值运算符的优先级相同，且结合方向也一致。

（3）赋值表达式。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”。一般格式为：**变量=表达式**

例如：a=5，赋值表达式的值是 5（变量 a 的值也是 5）。

C 语言中，凡是表达式可以出现的地方均可出现赋值表达式。

赋值表达式也可以包含复合的赋值运算符。如：a+=a-=a*a，假设 a 初值为 6，求解如下：

①先进行 a-=a*a 的运算，相当于 a=a-a*a，结果为-30。

②再进行 a+=-30 的运算，相当于 a=a+(-30)，结果为-60。

按照 C 语言规定，任何表达式在其末尾加上分号就构成语句。

（4）变量赋初值。在程序中常常需要对一些变量赋初值，以便使用变量。C 语言允许在定义变量的同时为其赋初值。例如：

```
int a=1; float x=3.2; char c='A';
```

赋初值时可以只对说明的一部分变量赋初值，也可以将几个变量赋予同一个初值。例如：

```
int a,b,c=1;/*对变量 c 初始化，值为 1 */
float x,y,z;
x=y=z=2.0;     /* 对变量 a，b，c 赋初值 2.0   */
```

3. 逗号运算符与逗号表达式

C 语言中逗号“,”也是一种运算符，称为逗号运算符。其功能是把两个或多个表达式连接起来组成一个表达式，称为逗号表达式。其一般形式为：

表达式 1，表达式 2，…，表达式 n

求值过程为：先求出表达式 1 的值，再求出表达式 2 的值，……依次求出各个表达式的值，并以表达式 n 的值作为整个逗号表达式的值。

逗号运算符是所有运算符中级别最低的，且具有从左至右的结合性。

例如：a=3*4,a*5,a+10,求解过程为：先计算 3*4，将 12 赋给 a，然后计算 a*5，值为 60，最后计算 a+10，值为 12+10=22，所以整个表达式的值为 22，变量 a 的值为 12。

4. 关系运算符和关系表达式

（1）关系运算符。关系运算符用于比较运算。包括大于（>）、小于（<）、大于等于（>=）、小于等于（<=）、等于（==）和不等于（!=）6 种。

关系运算符都是双目运算符，其结合性均为左结合性。在 6 个关系运算符中，前面 4 个的优先级相同（>，<，>=，<=），后两种（==,!=）的优先级相同，并且前面 4 个的优先级高于后面两个。

用关系运算符比较的数据有：整型、实型和字符型，字符串则不能用关系运算符做比较。比较整型或实型数据时，按照数值大小进行比较；比较字符型数据时，按照字符的 ASCII 值进行比较。

关系运算符的优先级低于算术运算符，高于赋值运算符。

（2）关系表达式。用关系运算符将两个要比较的对象连接起来的式子称为关系表达式，其格式为：

表达式　关系运算符　表达式

上面的表达式可以是算术表达式、关系表达式、逻辑表达式、赋值表达式、字符表达式。

例如：a+b>c+d、x>3/2、'a'+1<'c'、j==k+1，都是合法的关系表达式。

C 语言中，当判断关系表达式的值时，若关系表达式成立，则值为真，返回 1；否则，表达式不成立，则为假，返回 0。

例如：3>2 的值为 1，10>(2+10)的值为 0。

【例 2.5】关系表达式运算结果演示。

```
#include <stdio.h>
int main()
{
   printf("55>44:%d\n", 55 > 44);
   printf("z<A:%d\n", 'z' < 'A');
   printf("11<=7:%d\n", 11 <= 7);
   return 0;
}
```

程序运行结果如图 2.10 所示。

```
55>44:1
z<A:0
11<=7:0
```

图 2.10　【例 2.5】程序的运行结果

5. 逻辑运算符和逻辑表达式

（1）逻辑运算符。逻辑运算符用于逻辑运算。包括与（&&）、或（||）、非（!）3 种运算符。与（&&）和或（||）运算符均为双目运算符，具有左结合性。非（!）运算符为单目运算符，具有右结合性。

逻辑运算符优先级从高到低的排列是：非（!）—与（&&）—或（||）。

表 2.5 为逻辑运算的真值表，表示当操作数 a 和 b 的值为不同组合时，各种逻辑运算所得到的值。

表 2.5　逻辑运算的真值表

a	b	a&&b	a || b	!a
0	0	0	0	1
0	1	0	1	1
1	0	0	1	0
1	1	1	1	0

（2）逻辑表达式。用逻辑运算符将运算对象连接起来的有意义的式子称为逻辑表达式，其格式：

表达式　逻辑运算符　表达式

若逻辑表达式成立为真，则返回 1；否则，返回 0。

例如：5>0 || 5>8，由于 5>0 为真，就不再与 5>8 进行或运算了，结果也就为真，返回 1。

!1&&0，先求非运算结果为 0，就不再与 0 进行与运算了，返回值为 0。

注意：对上例中表达式!1&&0，先求!1 和先求 1&&0 将会得出不同的结果。

在用 && 对两个表达式进行计算时，如果第一个表达式的值为“假”，则后面的表达式就可以不用理会，结果肯定为“假”，所以 C 语言规定此时的第二个表达式将不再参与计算。同样的道理，用 || 对两个表达式进行计算时，若第一个表达式的值为“真”，则计算结果与第二个表达式的结果也没有关系，计算结果肯定为“真”。

【例 2.6】逻辑表达式的应用。

```
#include <stdio.h>
int main()
{
    int a = 14, b = 15, x;
    char c = 'A';
    x = a && b && c < 'B';
    printf("x=%d\n", x);
    return 0;
}
```

程序运行结果如图 2.11 所示。

```
x=1
```

图 2.11　【例 2.6】程序的运行结果

6. 运算符的优先级与结合性

C 语言规定了运算符的优先级和结合性。在表达式求解时，先按运算符的优先级别高低次序执行。

例如：a-b*c 等价于 a-(b*c)，运算符“*”的优先级高于运算符“-”。

如果一个运算对象两侧的运算符优先级别相同，则按规定的结合方向处理。左结合性（自左向右结合）是指运算对象先与左边的运算符结合，右结合性（自右向左结合）是指运算对象先与右边的运算符结合。对于复杂的表达式，为了清晰起见，可加圆括号“()”强制规定运算顺序。

三、任务实施


扫码看视频

现在来完成本任务：计算数学表达式的值。

（1）先将数学表达式

$$x=\frac{-b+5a^2}{2a}$$

转换成 C 语言表达式为：x=(-b+5*a*a)/(2*a)。

（2）为便于计算，将 a，b 定义为整型，x 定义为实型。为了得到正确的结果，可进行强制类型转换。

```
#include <stdio.h>
int main()
{
    int a, b;
    float x;
    scanf("%d%d", &a, &b);              /* 通过键盘给 a,b 赋值,&a 表示变量 a 的地址 */
    x = (float)(-b + 5 * a * a) / (2 * a); /* 将右边整型数据转换为实型 */
    printf("x=%f\n", x);
    return 0;

}
```

程序运行结果如图 2.12 所示。

```
2 3
x=4.250000
```

图 2.12　计算表达式的值的运行结果

四、深入训练

1. 编写一个 C 语言程序，输入变量 x，y，z 的值，根据以下算式求 n 的值。

$$n=x^2+\frac{yz}{2}$$

提示：将 x，y，z 定义为整型变量，n 为实型变量。

2. 判断某年是否是闰年需满足下列条件之一：

（1）能被 4 整除但不能被 100 整除。

（2）能被 4 整除又能被 400 整除。

写出判断某年 year 是否是闰年的表达式。

任务三　求三角形的面积

知识要点：格式输出函数 printf()。

一、任务分析

已知三角形三边 a，b，c 的值，求三角形的面积。要求输出 a，b，c 及面积 area 的值，输出结果保留两位小数。

二、必备知识与理论

我们知道，人与人之间是通过语言在外界空气介质的传输下进行交流的。同样，人、外部设备和计算机之间也有一定的交流方式，这种交流方式是靠输入和输出来完成的。

1. 数据输入/输出的概念

所谓输入/输出，是指用计算机的输入设备（键盘、磁盘、光盘和扫描仪等）向计算机输入数据，称为“输入”；从计算机向外部设备（显示器、磁盘、打印机等）输出数据，称为“输出”。

在程序的运行过程中，往往需要输入一些数据（语言内容），而程序运算所得到的计算结果（数据）又需要输出给用户。因此，输入/输出操作是程序设计语言中的重要内容。

C 语言未提供专门的输入/输出语句，所有的输入/输出操作都是通过对标准库函数的调用来实现的（如 printf() 函数和 scanf() 函数）。C 语言提供的函数以库的形式存放在系统中，它们不是 C 语言文本中的组成部分。因此在使用 C 语言库函数时，需要使用预编译命令#include 将相关的头文件 . h 包含到用户源文件中。

使用形式：**#include <头文件>　或 #include “头文件”**

说明：①用尖括号括起表示先在系统目录查找所包含的文件，一般在要包含系统头文件时使用；用双引号引起表示先在当前程序所在的目录查找所包含的文件，如果没有，再在对应系统目录中查找对应的文件。一般在要包含自己写的文件时使用。②标准输入/输出头文件是：stdio. h，它是 standard input & output 的缩写，“h”是 head 的缩写，它包含了与标准 I/O 库有关的变量定义和宏定义。由于 printf() 和 scanf() 函数使用比较频繁，因此有些系统允许在使用这两个函数时不需要包含头文件（即可以不加#include）。

常用的输入/输出函数有：printf() 函数（格式输出函数）、scanf() 函数（格式输入函数）和 putchar() 函数（字符输出函数）、getchar() 函数（字符输入函数）等。

2. 格式输出函数 printf()

printf() 函数称为格式输出函数。其功能是按用户指定的格式，把指定的数据输出到显示器屏幕上。在前面的例题中已多次用到这个函数。

（1）printf() 函数的一般格式：

printf(“格式控制字符串”,输出项列表);

例如：printf("r=%d\tarea=%f\n",r,area);

“格式控制字符串”是用双引号括起来的字符串，也称“转换控制字符串”。它包括以下 3 类字符：

1）普通字符：是一些说明字符，这些字符按原样显示在屏幕上，主要起提示作用。如上面 printf() 函数中的双引号里面的“r=”和“area=”。

2）转义字符：是不可打印的字符，控制产生特殊的输出效果。上例中的“\t”为水平制表符，作用是跳到下一个水平制表位；“\n”为回车换行符，输出自动换到新的一行。

3）格式字符：由“%”引导的格式字符串，用于指定输出格式。上例中的“%d”“%f”，它们的作用是把输出的数据转换为指定的格式输出。格式指示符是由“%”字符开头的。

printf() 函数语句的输出项列表是需要输出的一些数据，可以是常量、变量、表达式，其类型、个数必须与格式控制说明中格式字符的类型、个数一致。当有多个输出项时，各项之间用逗号分隔。

（2）格式字符串的一般格式为：

［标志］［输出最小宽度］［. 精度］［长度］类型

其中方括号“［］”中的项为可选项。

各项的意义介绍如下：

- 类型：用来表示输出数据的类型，其格式符的意义见表 2.6。
- 标志：标志字符为+、-、#、空格，共 4 种，其意义见表 2.7。
- 输出最小宽度：用十进制数来表示输出的最少位数。若实际位数多于定义的宽度，则按实际位数输出，若实际位数少于定义的宽度，则补以空格或 0。
- 精度：精度格式符以“.”开头，后跟十进制数。本项的意义是：如果输出数字，则表示小数的位数；如果输出字符串，则表示输出字符的个数；若实际位数大于所定义的精度数，则截去超过的部分。
- 长度：长度格式符为 h、l 两种，h 表示按短整型量输出，l 表示按长整型量输出，可加在格式符 d、o、x、u 前面。见表 2.8。

表 2.6　格式类型符及其含义（假设 x=3.1415926）

格式符	含义	举例	输出结果
d	按十进制输出带符号整数(正号省略)	printf("%d",'A')	65
o	按八进制输出无符号整数(不输出前缀 0)	printf("%o",'A')	101
x、X	按十六进制输出无符号整数(不输出前缀 0x)	printf("%x",'A')	41
u	按十进制输出无符号整数	printf("%u",'A')	65
f	按小数形式输出单、双精度实数	printf("%f", x)	3.141593
e、E	按指数形式输出单、双精度实数	printf("%e", x)	3.141593e+000
g、G	按 e 或 f 格式中较短的一种输出单、双精度实数	printf("%g",x)	3.141593
c	按字符型输出	printf("%c",'A')	A
s	按字符串输出	printf("%s","abc")	abc

表 2.7　标志及其含义

格式符	含义	举例	输出结果
-	结果左对齐,右边补空格	printf("%-4d",'A')	65
+	输出符号(正号或负号)	printf("%+d",'A')	+65
空格	输出值为正时冠以空格,为负时冠以负号	printf("%d",'A')	65
#	对 c,s,d,u 类无影响;对 o 类,在输出时加前缀 o;对 x 类,在输出时加前缀 0x;对 e,g,f 类,当结果有小数时才给出小数点	printf("%#o",65) printf("%#x",65)	0101 0x41

表 2.8　宽度长度修饰符（假设 x=3.1415926）

修饰符	含义	举例	输出结果
m	以宽度 m 输出整型数,不足 m 时,左补空格	printf("%4d",'A')	65
0m	以宽度 m 输出整型数,不足 m 时,左补 0	printf("%04d",'A')	0065
m.n	以宽度 m 输出实型小数,小数位数为 n 位	printf("%4.2f",x)	3.14
lf	以双精度型格式输出	printf("%lf",x)	3.141593
hd	以短整型格式输出	printf("%hd",'A')	65
ld	以长整型格式输出	printf("%ld",'A')	65
hu	以无符号短整型格式输出	printf("%hu",'A')	65

【例 2.7】printf()函数示例。

```
#include <stdio.h>
main()
{   int a=15;
    float b=123.1234567;
    double d=12345678.1234567;
    char c='p';
    printf("a=%d,%6d,%+6d,%-6d,%o,%x\n",a,a,a,a,a,a);
    printf("b=%f,%lf,%5.4f,%-10.4f,%e\n",b,b,b,b,b);
    printf("d=%f, %8.4lf,%e,%g\n",d,d,d,d);
    printf("c=%c,%8c\n",c,c);
    printf("%s,%-6.2s,%6.2s\n","china","china","china");
}
```

程序运行结果如图 2.13 所示。

```
a=15,    15,   +15,15    ,17,f
b=123.123459,123.1235,123.1235  ,1.231235e+02
d=12345678.123457, 12345678.1235,1.234568e+07
c=p,       p
china,ch    ,    ch
```

图 2.13　【例 2.7】程序的运行结果

3. 字符输出函数 putchar()

putchar()函数的功能是将一个字符输出到显示器上显示。putchar()函数也是一个标准的输入/输出库函数，它的原型也在 stdio.h 头文件中被定义。因此使用时也需要使用预编译处理命令#include。

putchar()函数的一般格式：putchar(c);

即把变量 c 的值输出到显示器上。这里 c 可以是字符型常量或变量，也可以是一个转

义字符。

【例 2.8】putchar()函数应用举例。

```
#include <stdio.h>
int main()
{
    char a, b, c;
    a = 'B';
    b = 'O';
    c = 89;
    putchar(a); putchar('\n');
    putchar(b); putchar('\n');
    putchar(c); putchar('\n');
    return 0;
}
```

程序运行结果如图 2.14 所示。

```
B
O
Y
```

图 2.14　【例 2.8】程序的运行结果

注意：putchar()函数只能用于单个字符的输出，且一次只能输出一个字符。

三、任务实施

扫码看视频

下面来计算三角形的面积。已知三角形三边 a，b，c 的值，要求输出 a，b，c 及面积 area 的值，输出结果保留两位小数。

任务分析：（1）已知三角形三边，求三角形面积，利用海伦公式实现。

面积= $\sqrt{s(s-a)(s-b)(s-c)}$，其中：$s=\frac{1}{2}(a+b+c)$。

（2）将该数学公式转换成 C 语言表达式：s=(a+b+c)/2,area=sqrt(s*(s-a)*(s-b)*(s-c))。

（3）sqrt()为标准数学函数，包含在头文件“math.h”中。

```
#include <stdio.h>
#include <math.h>
int main()
{
    float a, b, c, s, area;
    a = 3; b = 4; c = 5;
    s = (a + b + c) / 2;
```

```
    area = sqrt(s * (s - a) * (s - b) * (s - c));
    printf("a=%5.2f,b=%5.2f,c=%5.2f\narea=%5.2f\n", a, b, c, area);
    return 0;
}
```

程序运行结果如图 2.15 所示。

```
a= 3.00,b= 4.00,c= 5.00
area= 6.00
```

图 2.15　求三角形的面积的运行结果

四、深入训练

1. 写出下列程序运行后的结果。

```
#include <stdio.h>
int main()
{
    int x=97,y=98,z=99;
    printf("x=%d\ny=%d\nz=%d\n",x,y,z);
    printf("x=%c\ty=%c\tz=%d\n",x,y,z);
    return 0;
}
```

2. 已知正方形的边长为 3.6，编程计算它的面积（结果保留两位小数）。

任务四　求长方体的体积

知识要点：格式输入函数 scanf()。

一、任务分析

输入长方体的长、宽、高，输出它的体积。

知道长方体的长、宽、高，可以计算出长方体的体积。使用赋值语句进行赋值，若想改变长、宽、高的值，就必须修改源程序。通过键盘赋值，可以在程序运行时赋任意的值而不必修改源程序。

二、必备知识与理论

在 C 语言程序中，给计算机提供数据，可以用赋值语句，也可以用输入函数。本任务讨论格式输入函数 scanf() 和字符输入函数 getchar()。

1. 格式输入函数 scanf()

scanf() 函数是一个库函数，它与 printf() 函数相同，函数原型也在头文件 stdio.h 中。

（1）scanf()函数的一般格式：

scanf("控制字符串"，输入项地址列表)；

功能：从键盘上输入数据，该输入数据按指定的输入格式被赋给相应的输入项。

说明：

●控制字符串：规定数据的输入格式，作用与 printf()函数相同，但不能显示非格式字符串，也就是不能显示提示字符。

●输入项地址列表：由一个或多个变量地址组成，各变量地址之间用逗号“,”分隔。地址是由地址运算符“&”后跟变量名组成的。

例如：&a，&b 分别表示变量 a 和变量 b 的地址。这个地址就是编译系统在内存中给变量 a、b 分配的地址。在 C 语言中，使用了地址这个概念，这是与其他语言不同的。应该把变量的值和变量的地址这两个不同的概念区分开来。变量的地址是 C 编译系统分配的，用户不必关心具体的地址是多少。

scanf()函数语句在运行时，会停下来，等待用户从键盘上输入数据，然后按格式控制的要求对数据进行转换后送到相应的变量地址。

【例 2.9】scanf()函数使用举例一。

```
#include <stdio.h>
int main()
{
    int a, b, c;
    printf("input a,b,c:\n");     /* 显示提示信息 */
    scanf("%d%d%d", &a, &b, &c);/* 通过键盘给变量 a、b、c 赋值 */
    printf("a=%d,b=%d,c=%d\n", a, b, c);
    return 0;
}
```

程序运行结果如图 2.16 所示。

```
input a,b,c:
5 6 7
a=5,b=6,c=7
```

图 2.16　【例 2.9】程序的运行结果

程序说明：本例中，由于 scanf()函数本身不能显示提示串，故先用 printf()函数在屏幕输出提示，请用户输入 a，b，c 的值。执行 scanf()函数等待用户输入数据，用户输入数据后按 Enter 键，继续执行下面的输出语句，输出结果。

在 scanf()函数的格式串中，由于没有非格式字符在"%d%d%d"之间作为输入时的间隔，因此在输入时要用一个以上的空格或 Enter 键作为两个输入数据之间的间隔。

（2）格式控制字符串。格式控制字符串规定输入项中的变量以何种类型的数据格式被输入，形式是：

%［<修饰符>］<格式字符>

修饰符是可选的，修饰符如下：

• 字段宽度：按指定宽度输入数据。例如 scanf("%3d",&a);输入 123456，按宽度 3 输入一个整数 123 赋给变量 a，其余部分被截去。

• 长度修正符 l 和 h：可与 d、o、x 一起使用，l 表示输入数据为长整型，h 表示输入数据为短整型。例如 scanf("%ld%hd",&x,&i);x 按长整型读入，i 按短整型读入。

输入格式字符及其意义见表 2.9。

表 2.9　输入格式字符及其意义

格式字符	意 义
d、i	输入有符号的十进制整数
u	输入无符号的十进制整数
o	输入无符号的八进制整数
X、x	输入无符号的十六进制整数
f	输入实数，可以用小数或指数形式输入
e	输入一个指数形式的浮点数，可与 f 互换
c	输入一个字符
s	输入一个字符串。将字符串送到一个字符数组中，在输入时以非空白字符开始，以第一个空白字符结束。字符串以串结束标志'\0'作为其最后一个字符

使用 scanf()函数应注意以下几点：

• scanf()函数中没有精度控制，如 scanf("%5.2f",&a)是非法的。

• scanf()函数中要求给出变量的地址，如给出变量名则会出错。例如 scanf("%d",a)是非法的，应改为 scanf("%d",&a)才正确。

• 在输入多个数值时，若格式控制串中没有非格式字符作为输入数据之间的间隔，则可用空格、制表符或回车符作为分隔符。C 语言编译器在遇到空格、制表符、回车符或非法数据（如对"%d"输入"12A"时，A 即为非法数据）时，即认为数据输入结束。

• 在输入字符数据时，若格式控制串中无非格式字符，则认为所有输入的字符均为有效字符。例如 scanf("%c%c",&a,&ch)；输入为：d　e↙，则把“d”赋给 a，将空格赋给 ch。

【例 2.10】scanf()函数使用举例二。

```
#include <stdio.h>
int main()
{
    int a, b;
    char c1, c2;
    float d;
    printf("input a,b,d,c1 c2:\n");
    scanf("%d,%d,%f %c %c", &a, &b, &d, &c1, &c2);
    printf("a=%d,b=%d,d=%f,c1=%c,c2=%c\n", a, b, d, c1, c2);
    return 0;
}
```

程序运行结果如图 2.17 所示。

```
input a,b,d,c1 c2:
10,20,1.234 x y
a=10,b=20,d=1.234000,c1=x,c2=y
```

图 2.17　【例 2.10】程序的运行结果

2. 字符输入函数 getchar()

getchar()函数的功能是从键盘输入一个字符。该函数没有参数。getchar()函数也是一个标准的输入/输出库函数，它的原型也在 stdio.h 头文件中被定义。因此，使用时也需要使用预编译处理命令#include。

getchar()函数的一般格式：**c=getchar();**

执行调用时，变量 c 将得到用户从键盘输入的一个字符值，这里 c 可以是字符型或整型变量。

注意：getchar()函数只能接收单个字符，输入数字也按字符处理。输入多于一个字符时，只接收第一个字符。

【例 2.11】getchar()函数应用举例。输入一个大写字母，输出相应的小写字母。代码如下：

```
#include <stdio.h>
int main()
{
    char c1, c2;
    c1 = getchar();
    c2 = c1 + 32;
    putchar(c2);
    putchar('\n');
    return 0;
}
```

程序运行结果如图 2.18 所示。

```
F
f
```

图 2.18　【例 2.11】程序的运行结果

三、任务实施

扫码看视频

通过键盘输入任意 3 个数作为长方体的长、宽和高，计算长方体的体积。

本任务主要练习输入、输出函数的使用。程序代码如下：

```
#include <stdio.h>
int main()
{
    float x, y, h, v;
    printf("输入长方体的长、宽、高:");/* 显示提示信息 */
    scanf("%f,%f,%f", &x, &y, &h);      /* 输入长、宽、高的值 */
    v = x * y * h;                      /* 计算长方体的体积 */
    printf("长方体体积为:%.2f\n", v);  /* 结果保留两位小数 */
    return 0;
}
```

程序运行结果如图 2.19 所示。

```
输入长方体的长、宽、高：5.64,4.18,3
长方体体积为：70.73
```

图 2.19　求长方体的体积的运行结果

四、深入训练

1. 执行程序时输入 1□2↙（□代表空格），则下列程序运行后的结果是：

```
#include <stdio.h>
int main()
{
    int c,i;
    scanf("%c",&c);
    scanf("%d",&i);
    printf("%c,%d,%6.4s\n",c,i,"3456789");
    return 0;
}
```

2. 输入任意两个整数，编程求它们的商和余数。商为实数，结果保留两位小数。

提示：（1）要求商为实数，需要进行类型转换。

（2）余数的数据类型为整型。

拓展阅读

程序与人生
——以青春之我，厚积薄发

“科学计算”是计算机应用的一个重要领域，而我们当前学习的数据类型和运算符表达式正是进行“科学计算”的必要基础。若想成为一名优秀的程序设计员，就必须刻苦学习计算机基础知识和编程技术，通过自主创新设计不同的计算机应用程序。

在新时代，与“科学计算”密切相关的计算机应用领域的重大突破出现“井喷”态势。譬如，“终有一日，你能重整旗鼓，光耀中华”，人类历史上最大的射电望远镜FAST——天眼，让中国成为天文探测的领航人。全球最大的海上钻井平台“蓝鲸2号”，被称为“钢铁制造的移动国土”玛旁雍错上迁徙的羚羊、“复兴号”的顺利启程、C919的漂亮起飞、珠港澳大桥沉管合龙、国产航母的沉稳下水、“慧眼”遨游太空、光量子计算机的成功亮相等具有里程碑意义的科技成果，如潮水般冲击着我们，无数个用生命奋斗在一线的人们，在时间的齿轮上筑造一个又一个的中国奇迹。正所谓“百花齐放，百家争鸣”，强大的祖国以难能可贵的中国精神滚滚前进，以势不可挡的中国力量迈进大国之列，以匠心独运的中国智慧创造众多奇迹，引领崭新的强国时代，这些让每一个人中国人的内心汹涌澎湃，强烈的爱国情怀和民族自豪感油然而生，不约而同地传递着全国人民的心声：有一种骄傲，叫我们是中国人！

习近平总书记说，幸福都是奋斗出来的，奋斗本身就是一种幸福。我们要培养新时代大学生的奋斗精神，使他们做到理想坚定，信念执着，不怕困难，勇于开拓，顽强拼搏，永不气馁。为实现中华民族伟大复兴的中国梦而奋斗，是我们人生难得的际遇。我们每个人都应该珍惜这个伟大时代，做新时代的奋斗者。在奋斗中释放青春激情、追逐青春理想，以青春之我、奋斗之我，为民族复兴铺路架桥，为祖国建设添砖加瓦。

项目实训

一、实训目的

1. 掌握C语言基本数据类型的常量表示、变量的定义和使用。
2. 学会使用C语言的有关算术运算符以及包含这些运算符的表达式。
3. 能够将数学算式转换为C语言表达式。
4. 熟练掌握标准输入/输出函数进行常见数据类型的数据输入/输出方法，并能正确使用各种格式转换符。
5. 进一步熟悉C程序的结构特点，学习简单程序的编写方法。

二、实训任务

1. 运行与分析程序。

(1)
```
#include <stdio.h>
int main( )
{
    int n1 = 2, n2 = 4;
    float n3, n4;
    n3 = n1 / n2;
    n4 = (float)n1 / n2;
    printf("n1/n2=%f\n", n3);
    printf("(float)n1/n2=%f \n", n4);
    return 0;
}
```

```
(2)#include <stdio.h>
int main()
{
    int a = 7, b = 4;
    float x = 2.5, y;
    y = x + a % 3 * (int)(x + b) % 2 / 4;
    printf("y=%f\n", y);
    return 0;
}
```

```
(3)#include <stdio.h>
int main()
{
    char c1, c2;
    c1 ='A';
    c2 = c1 + 6;
    printf("%d,%c\n", c1, c2);
    return 0;
}
```

注意：先分析程序的运行结果，再运行该程序，比较自己的判断与屏幕上的结果是否一致，如果有差异，再想想错误出现在什么地方。这种做法可以帮助自己理解掌握所学理论，提高分析程序的能力。

2. 编写程序，求下列表达式的值，并分析输出结果。

（1）y=3.4*x-1/2（x、y 为实型变量）。

（2）y=x+a%3*(int)(x+y)%2/4（x、y 为实型变量，a 为整型变量）。

3. 编写程序，通过键盘输入一个字符，输出与之对应的 ASCII 码值。

项目练习

扫码玩闯关游戏

1. 填空题

（1）转义字符“\n”的功能是_____，转义字符“\r”的功能是_____。

（2）运算符“%”两侧运算对象的数据类型必须都是_____，运算符“++”和“--”的运算对象只能是_____。

（3）表达式 8/4*(int)2.5/(int)(1.25*(3.7+2.3))值的数据类型为_____。

（4）表达式(3+10)/2 的值为_____。

（5）设 int x=2,y=1;表达式(!x||y--)的值是_____。

2. 选择题

（1）下列 4 组选项中，均不是 C 语言关键字的选项是（　　）。

A. define　iF　type　　B. getc　char　printf

C. include　case　scanf　　D. while　go　pow

（2）下列 4 组选项中，均是合法转义字符的选项是（　　）。

A. '\' '\017' '\"'　　B. '\"' '\\"\n'

C. '\f' '\018' '\xab'　　D. '\\0' '\101' '\xlf'

（3）已知字母 b 的 ASCII 码值为 98，如 ch 为字符型变量，则表达式 ch='b'+'5'-'2'的值为（　　）。

A. e　　B. d　　C. 102　　D. 100

（4）以下表达式值为 3 的是（　　）。

A. 16-3%10　　B. 2+3/2　　C. 14/3-2　　D. (2+6)/(12-9)

（5）以下叙述不正确的是（　　）。

A. 在 C 程序中，逗号运算符的优先级最低

B. 在 C 程序中，MAX 和 max 是两个不同的变量

C. 若 a 和 b 类型相同，在计算了表达式 a=b 后，b 中的值将放入 a 中，而 b 中的值不变

D. 当从键盘输入数据时，对于整型变量只能输入整型数，对于实型变量只能输入实型数

（6）定义变量：int x；float y；则以下正确的是（　　）。

A. scanf("%f%f", &x, &y)　　B. scanf("%f%d", &x, &y)

C. scanf("%d%f", &x, &y)　　D. scanf("%5.2f%2d", &x, &y)

（7）putchar()函数可以向终端输出一个（　　）。

A. 字符或字符变量的值　　B. 字符串

C. 实型变量　　D. 整型变量的值

（8）以下能正确定义整型变量 a、b 和 c，并赋初值 5 的语句是（　　）。

A. int a=b=c=5;　　B. int a,b,c=5;

C. int a=5,b=5,c=5;　　D. a=b=c=5;

（9）以下叙述正确的是（　　）。

A. 赋值语句中的“=”是表示左边变量等于右边表达式

B. 赋值语句中左边的变量值不一定等于右边表达式的值

C. 赋值语句是由赋值表达式加上分号构成的

D. x+=y;不是赋值语句

3. 分析程序，写出运行结果

(1)
```
#include <stdio.h>
int main()
{
    int x = 010, y = 10, z = 0x10;
    printf("%d,%d,%d\n", x, y, z);
    return 0;
}
```

程序的运行结果为_____。

(2)
```
#include <stdio.h>
int main()
{
    int a = 2, b = 3;
    float x = 3.9, y = 2.3;
    float r;
    r = (float)(a + b) / .2 + (int)x % (int)y;
    printf("%f\n", r);
    return 0;
}
```

程序的运行结果为_____。

(3)
```
#include <stdio.h>
int main()
{
    int x = 12;
    printf("%d,%o,%x,%u\n", x, x, x, x);
    return 0;
}
```

程序的运行结果为_____。

(4)
```
#include <stdio.h>
int main()
{
        printf("%f,%4.2f\n", 3.14, 3.14159);
        return 0;
    }
```

程序的运行结果为_____。

(5)
```
#include <stdio.h>
    int main()
    {
        char x = 'a', y = 'b';
        printf("%d\\%c\n", x, y);
        printf("x=\'%3d\',y=\'%-3d\'\n", x, y);
        return 0;
    }
```

程序的运行结果为_____。

4. 编程题

(1) 已知年利率为3.2%，存款总额为5万元，求一年后的本息合计并输出。

(2) 输入球的半径，输出它的体积。(球的体积计算公式：$V = \frac{4}{3}\pi r^3$)