高等学校计算机应用规划教材

C 语言程序设计
习题指导与上机实践
(第 2 版)

刘韶涛　潘秀霞　应　晖　编著

清华大学出版社

北　京

内 容 简 介

本书是《C 语言程序设计(第 2 版)》(ISBN 978-7-302-54458-6)的配套习题指导与上机实践。与第 1 版相比，本书对各章的基本概念、基本理论、典型应用和重点与难点等内容的描述做了全面的修订和完善，补充了大量的例题及其分析。针对主教材内容的修改变化，本书做了相应内容的修改和补充。同时，也对各章之后的习题和上机实践题做了部分修改完善，给出了全部的参考答案或解答提示。

经过精心分析、筛选、归类和整理，本书重新对原有的全部考试模拟试卷与自测试卷以及对应的解析与参考答案等都做了修改和完善。对每道题的题目表述力求更为规范，考试内容更为科学，分析更为透彻。模拟试卷和自测试卷的考试知识内容和考试难度也更贴近考试实际。在附录中也增加了全国计算机等级考试二级 C 语言的模拟试题及参考答案。希望借助这些努力，能帮助读者顺利通过全国或者福建省计算机应用水平等级二级 C 语言的考试，也更希望能提高读者对 C 程序设计基本概念、基本结构、基本应用的理解和把握。

本书既适合于 C 语言程序设计的初学者使用，也适合于具有一定 C 语言学习基础，想进一步提高 C 语言编程能力的读者使用，尤其是那些准备参加计算机应用水平等级考试(二级 C 语言)的读者，相信本书一定能起到事半功倍的效果。

图书在版编目(CIP)数据

C 语言程序设计习题指导与上机实践 / 刘韶涛，潘秀霞，应晖 编著. —2 版. —北京：清华大学出版社，2020.1
高等学校计算机应用规划教材
ISBN 978-7-302-54360-2

Ⅰ. ①C… Ⅱ. ①刘… ②潘… ③应… Ⅲ. ①C 语言—程序设计—高等学校—教学参考资料 Ⅳ. ①TP312.8

中国版本图书馆 CIP 数据核字(2019)第 263971 号

责任编辑：王　定
装帧设计：孔祥峰
责任校对：成凤进
责任印制：沈　露

出版发行：清华大学出版社
网　　址：http://www.tup.com.cn，http://www.wqbook.com
地　　址：北京清华大学学研大厦 A 座　　**邮　　编**：100084
社 总 机：010-62770175　　**邮　　购**：010-62786544
投稿与读者服务：010-62776969, c-service@tup.tsinghua.edu.cn
质 量 反 馈：010-62772015, zhiliang@tup.tsinghua.edu.cn
印 装 者：清华大学印刷厂
经　　销：全国新华书店
开　　本：203mm×260mm　　**印　　张**：19.5　　**字　　数**：474 千字
版　　次：2015 年 1 月第 1 版　　2020 年 1 月第 2 版　　**印　　次**：2020 年 1 月第 1 次印刷
定　　价：58.00 元

产品编号：079604-01

前　言

目前，国内外C语言程序设计的相关教材较多，但大多数教材着重C语言基本语法规则和基本概念的阐述，学生学完之后并不能真正掌握和灵活使用C语言来解决一些实际应用问题。有很多教材，是为了让学生学完后参加全国或者省计算机等级考试而编写的，其内容完全是为了应对考试，再加上目前计算机等级考试中的C语言程序设计的考试内容、考试方法等都存在很多不完善的地方，考试内容和考试成绩并不能全面反映考生真正运用C语言进行程序设计来解决实际问题的能力和水平。因此，编写高质量的C程序设计教材和对应的学习指导与上机实践教程，培养学生思考问题、分析问题和解决问题的能力和水平，提高其计算机的应用能力水平，对学生来说既是非常必要的，也是非常重要的。

本套教材在第1版的基础上，根据初学者的特点，由浅入深，循序渐进，旨在帮助读者掌握C语言程序设计的基本方法，理解和领会C语言的特点和本质，提高运用C语言解决实际问题的综合能力。同时进一步丰富了一些典型的应用案例和模拟试题分析等，其基本概念和规则的表述更为科学、文字更为精炼通顺、数据更为准确有据。在内容上，增加了C在数据结构等领域中的具体应用和实际的开发范例等。案例程序都给出完整的注释、运行结果和分析说明，所有习题和上机实践题也都增加了参考答案或分析和解答提示等，以利于读者自我解题时进一步参考和对比。对考试模拟试卷的解析，力求更为详尽、准确和重点突出，让读者明白解决问题的思路和方法，起到举一反三的作用，更为方便读者的自我练习、检查和理解、掌握，尽量做到一题多解，着重对读者分析和思考能力的培养和训练。

根据近年来实际教学过程中，读者使用初版教材遇到的各种问题和反馈意见，我们组织编著教师和授课教师总结讨论、分析提炼，经过细心筛选、整理，重新编著了《C 语言程序设计(第2版)》和《C语言程序设计习题指导与上机实践(第2版)》，纠正、修改和进一步完善了初版教材的内容，增加了学科发展和知识更新相关的新章节。另外，我们本着“不求全面，但求实用”的理念，尽量让读者都能寻找到各个知识点的方向和途径，指引出什么问题应该从哪些地方寻找答案。不仅仅局限于C语言程序设计知识的描述，也把与程序设计相关的其他知识加以阐述，特别介绍C语言在其他交叉学科和相关领域中的新应用，让读者对C程序设计在整个学科体系、不同的软件开发环境和工程实践背景等中都有一个较清楚的了解和认识。

对于本书的编写，我们所追求的目标是：

(1) 既能作为一本学习C语言程序设计的学习教材，又能作为一本C语言程序设计的实验指导教材，还可作为一本探讨C程序设计学习和实践的艺术书籍。

(2) 突出C语言的应用重点和难点，但不拘于具体语法细节的学习指导，而更注重C语言的应用实践环节的上机实训。紧密联系教学实践，在教材中力求反映出学生学习相关知识

的各类疑难问题，从学习的角度对相关知识加以阐述和提炼。

(3) 引导读者养成良好的程序设计风格和程序设计思路，让读者能够理解解决问题的方法，以达到触类旁通的效果。应用举例讲究经典实用而且丰富有趣，注重前后章节例题的连贯、一致和逐步深入。

(4) 主要面向初、中级读者群，又能兼顾高级读者的一些需求。内容在深浅选择上，既适合大多数初学者，又能满足少数高级读者深入学习的需求。先讲解基本知识，再探讨深层次的若干问题，以引起高级读者的兴趣。尽量把教学实践中学生学习中的问题反映到教材编写中，并加以解决，所以不但适用于学生，也对教师有一定的作用。

(5) 进一步完善初版教材中的学习指导内容和上机实践题目的设计，突出重点，加强应用，力求表述更为科学、阐述更加准确。所有例题、习题和上机操作题，都经过调试、运行和分析，以更好地方便读者进行自我测试、自我检查和自我提高。

(6) 增加大量的例题、实验上机题和考试模拟试题等，在附录中增加了全国计算机等级二级 C 语言考试的模拟试题和参考答案等，对各个考点进行详尽的分析、研究和探讨，旨在帮助学生通过学习和练习，真正理解和掌握 C 程序设计的基本概念、基本理论知识和基本实践能力。设置各种不同层次、等级的题目，以适用于不同的读者对象。

(7) 增加 C 语言在其他工程实践项目中的应用，让读者进一步理解 C 语言在各个工程领域中的应用实践情况，激发他们运用 C 语言解决专业问题的兴趣，切实提高他们应用 C 程序设计，解决工程实际问题的能力和水平。

全书由刘韶涛进行全面的修订、补充和完善。此外，华侨大学计算机科学与技术学院的缑锦、田晖、王靖、范慧琳、余坚等同志，对教材和习题指导教材的再版给予了全程的指导和关心，并给出了很多建设性的意见和建议。华侨大学教务处和教材科，也对教材的编写和立项等工作给予了大力支持。在此，对他们表示衷心的感谢！

由于时间仓促，加上编者水平有限，书中难免存在不足之处，恳请广大读者批评指正，联系邮箱是 shaotaol@hqu.edu.cn。

作者寄语

编　者

2019 年 11 月 22 日

目　　录

第1章　程序设计基础

1.1　学习指导


程序概述

本章主要讲解程序设计的基础知识，首先介绍了计算机系统的组成，详细阐述了计算机的硬件系统和软件系统以及它们之间的关系等；然后介绍了数据在计算机内存中的存储和程序设计语言的基础知识，以及用高级语言编写程序的过程和步骤等；接着阐述了程序设计的基础——算法和数据结构的基础知识；最后，介绍了结构化程序设计的基本概念等。下面围绕本章的几个重要知识点对这些基本概念和基本知识做进一步的分析和讨论，以帮助初学者能对这些基础知识有较好的理解和掌握。

1.1.1　计算机中数据的表示

计算机最主要的功能是处理数值、文字、声音、图形和图像等信息。在计算机内部，各种信息都必须经过数字化编码后才能被传送、存储和处理。因此，理解和掌握信息编码的概念与处理技术是至关重要的。所谓编码，就是采用少量的基本符号，选用一定的组合规则，以表示大量复杂多样的信息。基本符号的种类和这些符号的组合规则是一切信息编码的两大要素。例如，用 10 个阿拉伯数码表示数字，用 26 个英文字母表示英文词汇等，都是编码的典型例子。

1. 进位记数制及其转换

在采用进位记数的数字系统中，如果只用 r 个基本符号表示数值，则称其为 r 进制(radix-r number system)，r 称为该数制的基数(radix)。对于不同的进制，它们的共同特点如下：

(1) 每一种数制都有固定的符号集。例如十进制数制的基本符号有 10 个(0, 1, 2, …, 9)。二进制数制的基本符号有 0 和 1 两个。

(2) 每一种数制都用位置表示法。即处于不同位置的数符所代表的值不同，与它所在位置的权值有关。

对任何一种进位记数制表示的数都可以写成按权展开的多项式之和，任意一个 r 进制数 N 可表示为：

$$N_r = \sum_{i=m-1}^{k} D_i \times r^i$$

其中，D_i 为该数制采用的基本数符，r^i 是权，r 是基数。

例如，十进制数 12345.67 可表示为：

$$12345.67=1\times10^4+2\times10^3+3\times10^2+4\times10^1+5\times10^0+6\times10^{-1}+7\times10^{-2}$$

表 1-1 所示是计算机中常用的进位数制的表示。

表 1-1　计算机中常用的进位数制的表示

进位制	二进制	八进制	十进制	十六进制
规则	逢二进一	逢八进一	逢十进一	逢十六进一
基数	r=2	r=8	r=10	r=16
数符	0, 1	0, 1, 2, …, 7	0, 1, 2, …, 9	0, 1, 2, … , 9, A, B, …, F
权	2^i	8^i	10^i	16^i
表示符	B	O	D	H

(1) 二进制数转换成十进制数的方法：将二进制数的每一位乘以它的权，然后相加，即可求得对应的十进制数值。

例如，把二进制数 100110.101 转换成相应的十进制数。

$(100110.101)_2=1\times2^5+0\times2^4+0\times2^3+1\times2^2+1\times2^1+0\times2^0+1\times2^{-1}+0\times2^{-2}+1\times2^{-3}=38.625$

(2) 十进制转换成二进制的方法：

① 将整数部分和小数部分分别转换，然后合并。十进制整数转换为二进制整数的方法是“除以 2 取余”；十进制小数转换为二进制小数的方法是“乘以 2 取整”。

② 把一个十进制数写成按二进制数权的大小展开的多项式，按权值从高到低依次取各项的系数就可得到相应的二进制数。

例如，把十进制数 175.71875 转换为相应的二进制数：

$$(175.71875)_{10}=2^7+2^5+2^3+2^2+2^1+2^0+2^{-1}+2^{-3}+2^{-4}+2^{-5}=(10101111.10111)_2$$

(3) 十进制数转换为八进制数的方法：对于十进制整数采用“除以 8 取余”的方法转换为八进制整数；对于十进制小数则采用“乘以 8 取整”的方法转换为八进制小数。

(4) 二进制数转换成八进制的方法：从小数点起，把二进制数每 3 位分成一组，然后写出每一组的等值八进制数，顺序排列起来就得到所要求的八进制数。

(5) 八进制转换成二进制的方法：同理，将一位八进制数用 3 位二进制数表示，就可以直接将八进制数转换成二进制数。

二进制、八进制和十六进制数之间的对应关系如表 1-2 所示。

表 1-2　二进制、八进制和十六进制之间的对应关系

二进制	八进制	二进制	十六进制
000	0	0000	0
001	1	0001	1
010	2	0010	2
011	3	0011	3
100	4	0100	4
101	5	0101	5
110	6	0110	6
111	7	0111	7
		1000	8
		1001	9
		1010	A
		1011	B
		1100	C
		1101	D
		1110	E
		1111	F

例如，把二进制数 10101111.10111 转换为相应的八进制数：

$$(10\ 101\ 111.101\ 11)_2=(257.56)_8$$

(6) 十进制数转换为十六进制数的方法：十进制整数部分“除以十六取余”，十进制数的小数部分“乘以十六取整”，进行转换。

(7) 二进制数转换成十六进制的方法：从小数点起，把二进制数每 4 位分成一组，然后写出每一组的等值十六进制数，顺序排列起来就得到所要求的十六进制数。

例如，把二进制数 10101111.10111 转换为相应的十六进制数：

$$(1010\ 1111.1011\ 1)_2=(AF.B8)_{16}$$

2. 二进制运算规则

加法：二进制加法的进位规则是“逢二进一”。

0+0=0　　1+0=1　　0+1=1　　1+1=0(有进位)

减法：在二进制减法的借位规则是“借一当二”。

0−0=0　　1−0=1　　1−1=0　　0−1=1(有借位)

乘法：二进制乘法规则是

0×0=0　　1×0=0　　0×1=0　　1×1=1

除法：二进制除法是乘法的逆运算，其运算方法与十进制除法是一样的。

3. 机器数和码制

各种数据在计算机中表示的形式称为机器数，其特点是采用二进制计数制，数的符号用 0 和 1 表示，小数点则隐含表示不占位置。机器数对应的实际数值称为数的真值。

机器数有无符号数和带符号数之分。无符号数表示正数，在机器数中没有符号位。对于无符号数，若约定小数点的位置在机器数的最低位之后，则是纯整数；若约定小数点的位置在机器数的最高位之前，则是纯小数。对于带符号数，机器数的最高位是表示正、负的符号位，其余位则表示数值。若约定小数点的位置在机器数的最低数值位之后，则是纯整数；若约定小数点的位置在机器数的最高数值位之前(符号位之后)，则是纯小数。

为了便于运算，带符号的机器数可采用原码、反码和补码等不同的编码方法，机器数的这些编码方法称为码制。

1) 原码表示法

数值 X 的原码记为$[X]_{原}$。设机器字长为 n(即采用 n 个二进制位表示数据)，则最高位是符号位，0 表示正号，1 表示负号；其余 n−1 位表示数值的绝对值。数值 0 的原码表示有两种形式：$[+0]_{原}$=00000000，$[-0]_{原}$=10000000 (设 n=8)。

例如，若机器字长 n 等于 8，则：

$[+1]_{原}$=0000 0001　　$[-1]_{原}$=1000 0001

$[+127]_{原}$=0111 1111　　$[-127]_{原}$=1111 1111

$[+45]_{原}$=0010 1101　　$[-45]_{原}$=1010 1101

$[+0.5]_{原}$=0◇100 0000　　$[-0.5]_{原}$=1◇100 0000(其中◇是小数点的位置)

2) 反码表示法

数值 X 的反码记作$[X]_{反}$。设机器字长为 n，则最高位是符号位，0 表示正号，1 表示负号，正数的反码与原码相同，负数的反码则是其绝对值按位求反。数值 0 的反码表示有两种形式：$[+0]_{反}$=00000000，$[-0]_{反}$=11111111 (设 n=8)。

例如，若机器字长 n 等于 8，则：

[+1] $_{反}$=0000 0001　　[−1] $_{反}$=1111 1110

[+127] $_{反}$=0111 1111　　[−127] $_{反}$=1000 0000

[+45] $_{反}$=0010 1101　　[−45] $_{反}$=1101 0010

[+0.5] $_{反}$=0◇100 0000　　[−0.5] $_{反}$=1◇011 1111(其中◇是小数点的位置)

3) 补码表示法

数值 X 的补码记作[X]$_{补}$。设机器字长为 n，则最高位是符号位，0 表示正号，1 表示负号，正数的补码与原码相同，负数的补码则是其反码的末尾加 1。在补码表示中，0 的补码是唯一的：[+0] $_{补}$=00000000，[-0] $_{补}$=00000000 (设 n=8)。

例如，若机器字长 n 等于 8，则：

[+1] $_{补}$=0000 0001　　[−1] $_{补}$=11111 1111

[+127] $_{补}$=0111 1111　　[−127] $_{补}$=1000 0001

[+45] $_{补}$=0010 1101　　[−45] $_{补}$=1101 0011

[+0.5] $_{补}$=0◇100 0000　　[−0.5] $_{补}$=1◇100 0000(其中◇是小数点的位置)

4. 定点数和浮点数

1) 定点数

所谓定点数，就是小数点的位置固定不变的数。小数点的位置通常有两种约定方式：定点整数(纯整数，小数点在最低有效数值位之后)和定点小数(纯小数，小数点在最高有效数值位之前)。

设机器字长为 n，各种码制表示下的带符号数的范围如表 1-3 所示。

表 1-3　各种码制表示下的带符号数的范围

码制	定点整数	定点小数
原码	$-(2^{n-1}-1)\sim+(2^{n-1}-1)$	$-(1-2^{-(n-1)})\sim+(1-2^{-(n-1)})$
反码	$-(2^{n-1}-1)\sim+(2^{n-1}-1)$	$-(1-2^{-(n-1)})\sim+(1-2^{-(n-1)})$
补码	$-2^{n-1}\sim+(2^{n-1}-1)$	$-1\sim+(1-2^{-(n-1)})$

2) 浮点数

当机器字长为 n 时，定点数的补码可表示 2^{n-1} 个数，而其原码和反码只能表示 $2^{n-1}-1$ 个数(0 的表示占用了两个编码)，定点数所能表示的数值范围比较小，运算中很容易因结果超出范围而溢出。引入浮点数，浮点数是小数点位置不固定的数，它能表示更大范围的数。

与十进制数类似，一个二进制数也可以写成多种表示形式。例如，二进制数 1011.10101 可以写成 0.1011 10101×2^4、0.01011 10101×2^5、0.001011 10101×2^6等。由此可知，一个二进制数 N 可以表示为更一般的形式：

$$N=2^E\times M$$

其中，E 称为阶码，M 称为尾数。用阶码和尾数表示的数称为浮点数，这种表示数的方法称为浮点表示法。

在浮点表示法中，阶码通常为带符号的纯整数，尾数为带符号的纯小数。浮点数的表示格式如下：

阶符	阶码	数符	尾数

很明显，一个数的浮点表示不是唯一的。当小数点的位置改变时，阶码也随着相应改变，因此可以用多种浮点形式表示同一个数。

浮点数所能表示的数值范围主要由阶码决定，所表示数值的精度则由尾数决定。为了充分利用尾数来标识更多的有效数字，通常采用规格化浮点数。规格化就是将尾数的绝对值限定在区间[0.5, 1]。当尾数用补码表示时，需要注意：

(1) 若尾数 M≥0，则其规格化的尾数形式为 M=0.1××…×，其中×可为 0，也可为 1，即将尾数 M 的范围限定在区间[0.5, 1]。

(2) 若尾数 M<0，则其规格化的尾数形式为 M=1.0××…×，其中×可为 0，也可为 1，即将尾数 M 的范围限定在区间[-1, -0.5]。

5. 十进制数与字符的编码表示

数值、文字和英文字母等都认为是字符，任何字符进入计算机时，都必须换成二进制表示形式，称为字符编码。

用 4 位二进制代码表示 1 位十进制数，称为二-十进制编码，简称 BCD 编码。因为 2^4=16，而十进制数只有 0～9 这 10 个不同的数符，故有多种 BCD 编码。根据 4 位代码中每一位是否有确定的权来划分，可分为有权码和无权码两类。

应用最多的有权码是 8421 码，即 4 个二进制位的权从高到低分别为 8、4、2 和 1。8421BCD 码与十进制数的对应关系如表 1-4 所示。

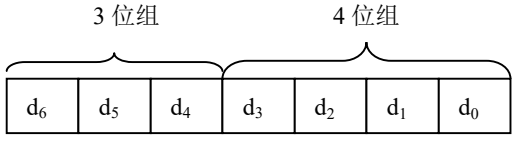
表 1-4　8421BCD 码与十进制数的对应关系

十进制数	8421BCD 码
0	0000
1	0001
2	0010
3	0011
4	0100
5	0101
6	0110
7	0111
8	1000
9	1001

6. ASCII 码

ASCII 码(American Standard Code for Information Interchange)是美国标准信息交换码的简称，已被国际标准化组织 ISO 采纳，成为一种国际通用的信息交换用标准代码。

ASCII 码采用 7 个二进制位对字符进行编码：低 4 位组 $d_3d_2d_1d_0$ 用作行编码，高 3 位组 $d_6d_5d_4$ 用作列编码。其格式为：



根据 ASCII 码的构成格式，可以方便地从对应的码表中查出每一个字符的编码。参见 ASCII 码表。

7. 汉字编码

汉字处理包括汉字的编码输入、汉字的存储和汉字的输出等环节。也就是说，在计算机中处理汉字，必须先将汉字代码化，即对汉字进行编码。汉字种类繁多，编码比西文拼音文字困难，而且在一个汉字处理系统中，输入、内部处理、存储和输出对汉字的要求不尽相同，所以采用的编码也不尽相同。汉字信息处理系统在处理汉字和词语时，关键的问题是要进行一系列的汉字代码转换。

1) 输入码

中文的字数繁多，字形复杂，字音多变，常用汉字就有 7000 个左右。在计算机系统中使用汉字，首先遇到的问题就是如何把汉字输入到计算机内。为了能直接使用西文标准键盘进行输入，必须为汉字设计相应的编码方法。汉字编码方法主要分为 3 类：数字编码、拼音码和字形码。

(1) 数字编码。数字编码是用数字串代表一个汉字的输入，常用的是国际区位码。国际区位码将国家标准局公布的 6763 个两级汉字分成 94 个区，每个区 94 位，实际上就是把汉字表示成二维数组，区码和位码各两位十进制数字，因此，输入一个汉字需要按键 4 次。例如，“中”字位于第 54 区 48 位，区位码为 5448。数字编码输入的优点是无重码，而且输入码和内部编码的转换比较方便，但是每个编码都是等长的数字串，代码难以记忆。

(2) 拼音码。拼音码是以含英语读音为基础的输入方法。由于汉字同音字太多，输入重码率很高，因此，按拼音输入后还必须进行同音字选择，影响输入的速度。

(3) 字形编码。字形编码是以汉字的现状确定的编码。汉字总数虽多，但都是由一笔一划组成，全部汉字的部件和笔画是有限的。因此，把汉字的笔画部件用字母或数字进行编码，按笔画书写的顺序依次输入，就能表示一个汉字，五笔字型、表形码等都是这种编码法。五笔字型编码是目前最有影响的汉字编码方法。

2) 内部码

汉字内部码(简称汉字内码)是汉字在设备或信息处理系统内部最基本的表达形式，是在设备和信息处理系统内部存储、处理、传输汉字用的代码。汉字数量多，用一个字节无法区分，采用国家标准局 GB 2312—1980 中规定的汉字国标码，两个字节存放一个汉字的内码，每个字节的最高位置“1”，作为汉字机内码。由于两个字节各用 7 位，因此可表示 16384 个可区别的机内码。例如，汉字“大”的国标码为 3473H，两个字节的高位置“1”，得到的机内码为 B4F3H。

3) 字形码

汉字字形码是表示汉字字形的字模数据，通常用点阵、矢量函数等方式表示。用点阵表示汉字时，汉字字形码指的就是这个汉字字形点阵的代码。字形码也称字模码，是用点阵表示的汉字字形码，它是汉字的输出方式，根据输出汉字的要求不同，点阵的多少也不同。简易型汉字为 16×16 点阵，高精度型汉字为 24×24 点阵、32×32 点阵、48×48 点阵等。字模点阵的信息量很大，每个汉字就不能用于机内存储，字库中存储了每个汉字的点阵代码，当显示输出时才检索字库，输出字模点阵得到字形。

汉字的矢量表示法是将汉字看作由笔画组成的图形，提取每个笔画的坐标值，这些坐标值可以决定每一笔画的位置，将每一个汉字的所有坐标值信息组合起来就是该汉字字形的矢

量信息。同样，将各个汉字的矢量信息集中在一起就构成了汉字库。当需要汉字输出时，利用汉字字形检索程序根据汉字内码从字模库中找到相应的字形码。

1.1.2　算法和数据结构的基本概念


算法和数据结构概述

在程序设计过程中，首先要对解决的问题进行分析和建模，理解所要解决问题中的各个对象和它们之间的关系；然后考虑在计算机内部该如何表示这些对象和这些对象之间的关系；最后，考虑采用什么样的办法来得到问题的解。显然，程序设计的关键在于分析问题域中的对象和关系、在计算机内部如何存储表示出这些对象和关系，以及在此基础上的解决问题的策略。

数据结构就是建立问题数学模型的关键技术。算法就是在数学模型基础上寻求解决问题的策略。而程序就是为计算机处理问题而编制的一组指令集。显然，程序设计的关键就在于数据结构和算法。这就是发明了多种影响深远的程序设计语言(Pascal 之父)，并提出结构化程序设计这一革命性概念(结构化程序设计的首创者)而获得 1984 年图灵奖的瑞士计算机科学家 Niklaus Wirth(尼克劳斯 • 沃思)提出的著名的公式：

Data Structures + Algorithm = Programs(数据结构+算法 = 程序)

他也以此公式作为其一本专著的书名。

关于算法的概念和算法的表示方法请读者参看配套教材《C 语言程序设计(第 2 版)》，在此不再阐述。

在程序设计中，经常会遇到两类问题：一类是数值性问题，如求和、求方程的根等，这些问题相对来说，比较简单，不需要建立复杂的数学模型。还有一类是非数值性的问题，如交叉路口的交通管制问题、最短路径问题和排序问题等。这些问题往往比较复杂，主要是问题域中的对象及其关系比较复杂。因此，对于这些问题的一般求解方法是：

(1) 建立问题的数学模型(如线性模型、树状模型、网状模型等)。

(2) 按照数学模型设计解决问题的算法。

(3) 根据算法编写程序，运行程序得到问题的解答。

数据结构就是一门讨论“描述现实世界实体的数学模型(非数值计算)及其上的操作在计算机中如何表示和实现”的学科。它与算法一样，都是进行复杂程序设计的基础。

数据结构包含 3 个层面的含义：

(1) 问题所涉及的数据对象，以及数据对象内部各个数据元素之间的特定关系——数据的逻辑结构(逻辑关系)。

(2) 全体数据元素以及数据元素之间的特定关系在计算机内部的表达——数据的存储结构(物理关系)。

(3) 为解决问题而对数据施加的一组操作——数据的运算集合(操作/运算)。

例如，要编写程序，实现按学号从低到高，随机输入若干个同班学生的信息(包括：学号，姓名，年龄和成绩总分)，按照成绩总分从高到低的顺序，重新列出学生的信息。

分析：首先要弄清问题域的对象(若干个学生)以及他们之间的关系(同班)。再考虑如何在计算机中表示出这些对象及其关系。可以将若干个学生的信息存储在一个一般高级语言都提供的一维数组中，这个数组就是一种数据结构。它表达了组成数组的各个元素(数组元素)之间的逻辑关系和物理关系(详见第 7 章)。最后可以在此基础上设计算法来进行排序。编写出的 C 语言程序如下(可能无法理解具体语言的细节，但可以理解解题的思路和步骤)：

```
#include <stdio.h>
#define N 10            /* 学生数 */
struct Student{         /* 存储每个学生信息的数据结构 */
    char Num[20];       /* 学号 */
    char Name[10];      /* 姓名 */
    int age;            /* 年龄 */
    float SumScore;     /* 总成绩 */
};
void main(void){
    int i,j;
    struct Student s[N],temp;  /* s[N]存储各个学生信息的数据结构 */
    printf("Input students information:\n");
    for(i=1;i<=N;i++)  /* 输入各个学生信息 */
    scanf("%s%s%d%f",s[i-1].Num,s[i-1].Name,&s[i-1].age,&s[i-1].SumScore );
    /* 排序算法实现 */
    for(i=1;i<N;i++)
      for(j=0;j<N-i;j++)
           if(s[j].SumScore<s[j+1].SumScore){
             temp=s[j];
             s[j]=s[j+1];
             s[j+1]=temp;
           }
   printf("\n\nInformation after sort:\n"); /* 输出排序后的各个学生信息 */
   for(i=1;i<=N;i++)
   printf("%s,%s,%d,%f\n",s[i-1].Num,s[i-1].Name,s[i-1].age,s[i-1].SumScore);
}
```

程序运行结果如图 1.1 所示。

```
1003 xu 18 600
1004 lin 17 655
1005 ouyang 18 666
1006 sun 18 655
1007 chen 18 635
1008 zhao 17 666
1009 li 18 500
1010 zhonghua 17 688


Information after sort:
1010,zhonghua,17,688.000000
1005,ouyang,18,666.000000
1008,zhao,17,666.000000
1004,lin,17,655.000000
1006,sun,18,655.000000
1007,chen,18,635.000000
1003,xu,18,600.000000
1002,zhang,17,598.000000
1001,liu,18,568.000000
1009,li,18,500.000000
```

图 1.1　程序运行结果

1.1.3　结构化程序设计的基本概念

结构化程序设计

结构化的程序设计强调程序设计风格和程序结构的规范化，提倡清晰的结构。结构化程序设计方法的基本思路是把一个复杂问题的求解过程分阶段进行，每个阶段处理的问题都控制在人们容易理解和处理的范围内。

具体来说，采用如下方法以保证得到结构化的程序：

(1) 自顶向下。

(2) 逐步细化。

(3) 模块化设计。

(4) 结构化编码。

在程序设计过程中，应当按自顶向下逐步细化的方法，将一个大的程序分解成足够小的部分。尤其是当程序比较复杂时，更有必要这样做。在拿到一个程序模块以后，根据程序模块的功能将它划分为若干个子模块，如果嫌这些子模块太大，还可以划分成更小的模块。

划分模块时应注意模块的独立性，即一个模块完成一项功能，耦合性越小越好。模块化设计的思路实际上是一种“分而治之”的思想，把一个大的任务分成若干个小的子任务，每一个子任务就相对简单了。

例如，绘制流程图，求 sum=1+2+3+…+100 的值。

分析：求解这个问题要定义两个变量 sum 和 i，分别存放结果和以及整数 1, 2, …, 100。先用自然语言描述这组数和算法：

第一步：将 sum 置 0，存储单元 i 置 1，即 sum=0, i=1。

第二步：执行循环结构，如果 n≤100, 执行循环体一次，直到 n＞100，退出循环。在循环体中，

(1) sum 累加上 i, 即 sum=sum+i。

(2) i 的值更改为 n+1, 即 n=n+1。

第三步：输出 sum 中的值。

流程图如图 1.2 所示。

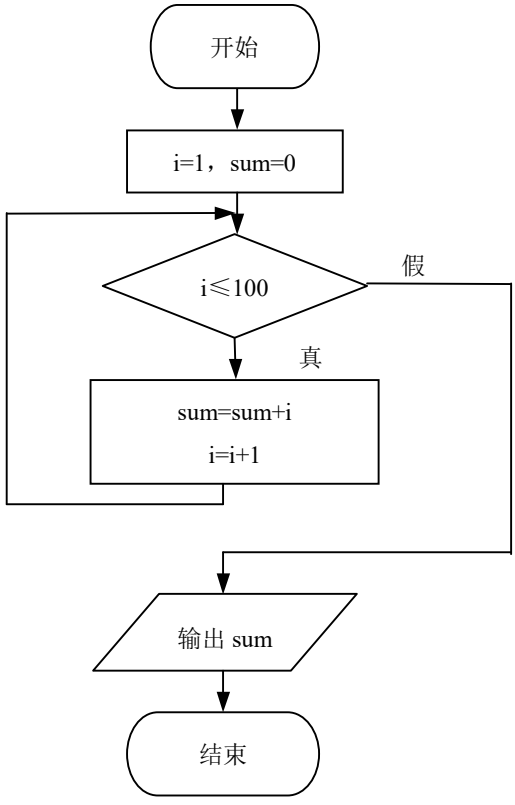


图 1.2　流程图

1.2　学习与思考

本章是程序设计的基础知识讲解，主要为今后学习 C 语言程序设计打下良好的基础。由于没有上机实践练习，这里通过几道习题进行练习和思考。

1. 将$(01110101)_2$、$(113)_{10}$、$(75)_8$、$(C4)_{16}$按从小到大的顺序排列。

【提示】

将各种进制的数都转换为对应的十进制数，可以得到它们从小到大的排列顺序。

$(01110101)_2=(117)_{10}$, $(75)_8=(61)_{10}$、$(C4)_{16}=(196)_{10}$。

2. 将$(3251)_{10}$转换成其他数制。

【提示】

采用“除以 2，8，16，取余数”法，可以将十进制数转换为对应的 2，8，16 进制数。

3. 写出$(-3251)_{10}$的原码、反码和补码(设字长 n=16)。

【提示】方法可参见教材或学习指导。

4. 程序与程序设计语言的含义分别是什么？

【提示】略。

5. 怎么理解高级语言的“高级”特性？

【提示】略。

6. 用流程图表示算法，求两个正整数 m、n(m>n)的最大公约数。

【提示】

可以用如图 1.3 所示的“辗转先除法”得到两个整数的最大公约数。

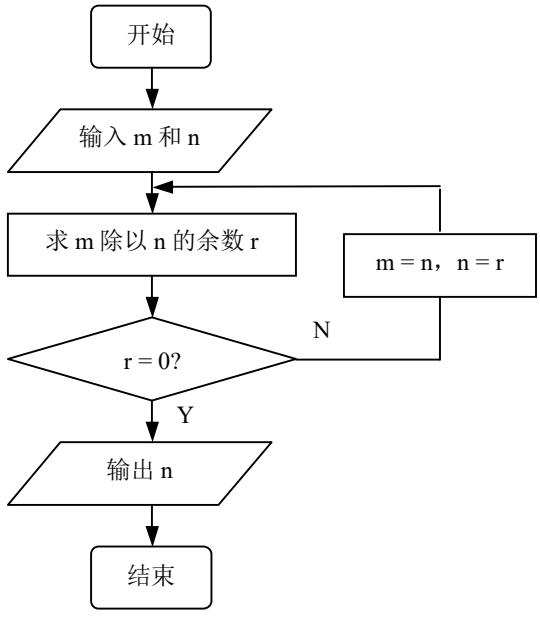


图 1.3　“辗转先除法”示意图

7. 用伪代码流程图表示算法，将 100～200 之间的素数打印出来。

【提示】

```
Begin
  i=100
  repeat do
  {
    k=2,flag=1
   repeat do
   {
        if i%k==0 then
           flag=0, break
```

```
        endif
        else  k=k+1
     } until k>(int)sqrt(i)
    if(flag==1)  then
          output  i
    endif
    i=i+1
  }until i>200
End
```

8. 什么是数据结构？它具有什么样的含义？

【提示】略。

9. 如何理解结构化程序设计的思想？

【提示】略。

第2章　C语言与C程序概述

2.1　学习指导

2.1.1　C 语言简介


C 语言概述

C 语言是国际上广泛流行的计算机高级编程语言，适合作为系统描述语言，既可以用来编写系统软件，也可以用来编写应用软件。

C 语言诞生于 20 世纪 70 年代初，1983 年，美国国家标准协会(ANSI)根据 C 语言问世以来的各种版本，对 C 的发展和扩充指定了新的标准，称为 ANSI C。1987 年，ANSI 又公布了新的标准：87ANSI C。1990 年，国际标准化组织(ISO)接受了 87ANSI C 为 ISO C 的标准(ISO 9899—1990)。目前流行的各种版本都是以它为基础的。

一种语言之所以能存在和发展，并具有生命力，总是有其不同于(或优于)其他语言的特点。C 语言的主要特点如下：

(1) 语言简洁、紧凑，使用方便、灵活。
(2) 运算符丰富。
(3) 数据结构丰富，具有现代化语言的各种数据结构。
(4) 具有结构化的控制语句，用函数作为程序的模块单位，便于实现程序的模块化。
(5) 语法限制不太严格，程序设计自由度大。
(6) 能进行位操作，能实现汇编语言的大部分功能，可以直接对硬件进行操作。
(7) 生成目标代码质量高，程序执行效率高。
(8) 程序可移植性好，基本上不做修改就能用于各种型号的计算机和操作系统。

2.1.2　简单的 C 程序介绍

下面介绍几个简单的 C 程序。

```
#include <stdio.h>
void main(void){
  printf("This is the first C program.\n");
}
```

程序运行情况如图 2.1 所示。

```
This is the first C program.
Press any key to continue
```

图 2.1　程序运行结果 1

其中，#include <stdio.h>是一个编译预处理命令，表示要嵌入头文件 stdio.h。stdio.h 是已经写好的一个头文件，在这个文件中已经定义好了常用的一些有关输入和输出的函数，在需要时可直接使用这些函数。

main()表示“主函数”。每一个 C 程序都必须有且仅有一个 main()函数，函数体由大括号{}括起来。本例中主函数内只有一个输出语句，这个输出语句通过调用 printf()函数实现字符串的输出。双引号内的字符串按原样输出，"\n"是换行符，即输出 This is the first C program. 后按 Enter 键换行，语句的最后有一个分号。

```
#include <stdio.h>
int max(int x,int y){       /* 定义函数 max，其值为整型，形式参数为整型 */
     int z;      /* 函数 max 中的声明部分，定义本函数中要用到的变量 z 为整型 */
     if(x>y) z=x;
     else z=y;
   return z;                /* 将 c 的值返回，通过 max 带回调用处 */
}

void main(void){            /* 主函数 */
       int a,b,c;           /* 声明部分，定义变量 */
    printf("Please enter a and b:");   /* 输入变量值的提示信息 */
      scanf("%d%d",&a,&b); /* 输入变量的值 */

      c=max(a,b);           /* 调用 max()函数，将得到的值赋给 z */
    printf("max=%d\n",c);  /* 输出 z 的值 */
}
```

程序运行情况如图 2.2 所示。

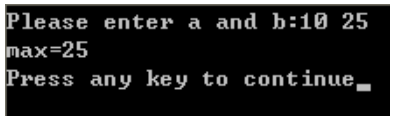


图 2.2　程序运行结果 2

本程序包括两个函数：主函数 main()和被调用的函数 max()。max()的作用是将 x, y 中的较大值赋给 z，return 语句再将 z 的值返回主调函数 main()。返回值是通过函数名 max()带回 main()函数的调用处。程序中 scanf()函数的作用是输入 a 和 b 的值。&a 和&b 中的“&”的含义是“取地址”，此函数的作用是将两个数值分别放到变量所对应的地址单元中。这种形式与其他语言有所不同。

函数 main()中的 c=max(a,b); 语句通过调用 max()函数，得到 a 和 b 的最大值，再赋给变量 c。在调用 max()函数时，将实际参数 a 和 b 的值分别传送给 max()函数中的形式参数 x 和 y。经过运算后得到一个最大值给变量 z，然后输出 z 的值。printf()函数中双引号内的 max=%d 在输出时，%d 将由 z 的值取代，max=原样输出。

从上面的两个例子可以看出以下几点：

(1) C 程序是由函数组成的。一个 C 源程序至少包含一个 main()函数，也可以包含一个 main()函数和若干个其他函数。因此，函数是 C 程序的基本单位。被调用的函数可以是系统提供的库函数，也可以是用户根据需要自己编写设计的函数。C 的函数相当于其他语言中的

子程序，用函数来实现特定的功能。程序中的全部工作是由各个函数分别完成的。编写 C 程序就是编写一个个函数。C 的函数库十分丰富，ANSI C 建议的标准库函数中包括 100 多个函数，Turbo C 和 MS C 4.0 中提供了 300 多个库函数。C 语言的这个特点使得程序的模块化很容易实现。

(2) 一个函数由函数的首部和函数体两部分组成。函数的首部包括函数名、函数类型、函数参数名、参数类型；函数体是函数首部下面大括号{}里面的部分。如果一个函数内有多个大括号，则最外层的一对{}为函数体的范围。

(3) 一个 C 程序总是从 main()函数开始执行的，而不论 main()函数在整个程序中的位置如何(可以放在程序的最前、最后或是中间的任何一个位置)。

(4) C 程序书写格式自由，一行内可以写几个语句，一个语句可以分写在多行上。C 程序没有行号，也不像 FORTRAN 或 COBOL 那样严格规定书写格式。

(5) 每个语句和数据定义的最后必须有一个分号。分号是 C 语句的重要组成部分，即使是程序中最后一个语句也应包含分号。

(6) C 语言本身没有输入/输出语句，输入和输出语句是由库函数来完成的。C 语言对输入/输出实行“函数化”。

2.2　C 程序的开发环境及其使用

本部分介绍 C 程序的开发过程和操作环境，重点是在目前应用较广泛的 Turbo C 和 VC++ 编译系统下，如何进行 C 源程序的输入、编辑、编译、链接、运行、调试等过程。最后介绍常见的上机错误和纠正办法。

2.2.1　C 程序开发过程

C 程序要在某种操作系统和编译系统支持下开发。应用 C 语言编写的程序，称为“C 源程序”。计算机不能直接识别 C 源程序。为了使计算机能够执行 C 程序指令，首先需用一种“编译程序”软件将 C 源程序翻译成二进制形式的“目标程序”，然后通过“链接程序”将目标程序和系统库函数链接成完整的“可执行程序”，最后运行可执行程序。在 C 程序开发的各个阶段都可能发生错误，因此需重复上述过程，排除错误，直到程序运行能达到预期的结果为止。C 程序开发过程如图 2.3 所示。

在 C 程序开发过程中，可能发生如下几类典型的错误。

(1) 语法错误，在编译和链接阶段出现。语法错误易纠正。编译时编译器做语法检查，给出错误定位和错误报告信息。错误信息是十分重要的，最有参考价值。

(2) 语义错误，在运行阶段出现。这类错误不能精确定位，只给出错误的信息，纠正错误较难。一般采用跟踪技术检测和定位。

(3) 逻辑错误，在程序运行后出现。这类错误易被忽视，最难纠正，需用测试技术解决。

排除程序错误，提高程序调试能力，是学习 C 程序设计的重要内容之一。

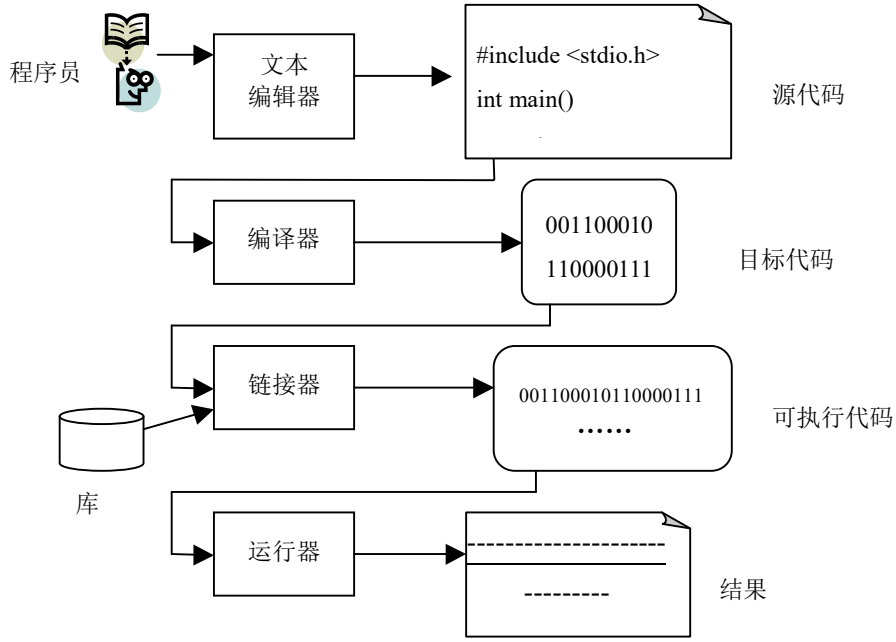


图 2.3　C 程序开发过程

2.2.2　VC++环境下运行 C 程序

C 程序的开发环境很多，常用的有 Turbo C/C++、Visual C++、Visual Studio、Dev C++、GCC 等。大多数的开放环境使用方法相似，这里以 Visual C++为例加以说明，需要使用其他平台时，可以参考相关的说明书。

C++语言是在 C 语言的基础上发展而来的，增加了面向对象的编程，成为当今最流行的一种程序设计语言。Visual C++(简称 VC++)是微软公司开发的面向 Windows 编程的 C++语言工具，不仅支持 C++语言的编程，也兼容 C 语言的编程。Visual C++被广泛地用于各种编程，使用面很广，这里简要介绍如何在 Visual C++下运行简单 C 语言程序。

1. 启动 VC++

(1) 双击桌面上的 Visual C++图标，启动 VC++。

(2) 选择“开始”→“程序”→Visual C++命令，启动 VC++。如图 2.4 所示为 VC++窗口。

2. 新建/打开 C 程序文件

如果要新建一个 C 程序，则选择“文件”→“新建”命令，弹出如图 2.5 所示的对话框，选择 C++ Source File 选项，单击“确定”按钮，即可在编辑窗口中输入程序。

如果要打开已经存在的 C 程序，则选择“文件”→“打开”命令，在弹出的对话框中选择要打开的程序文件即可。

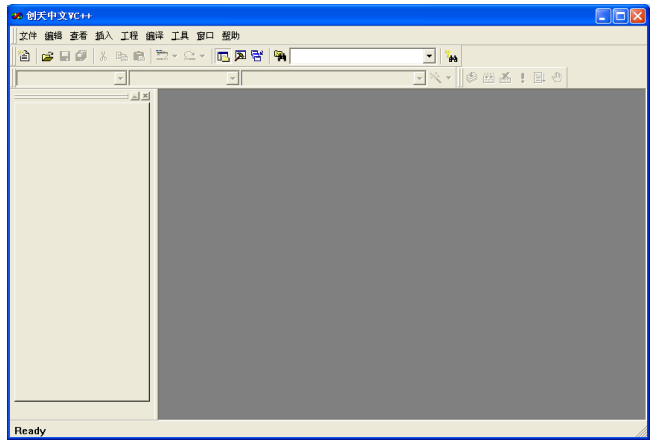


图 2.4　VC++窗口

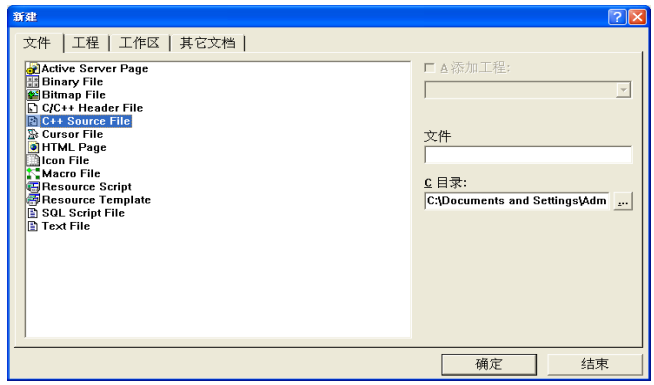


图 2.5　在 VC++中建立 C 源程序文件

3. 程序保存(文件/保存)

启动 VC++以后，实际上就可以直接在编辑窗口中输入程序，但此时要注意。当保存文件时，系统将默认以“.cpp”扩展名保存，所以此时，需要将程序保存为指定扩展名为“.c”。

4. 执行程序

按 F7 键，或者选择“编译/构件”命令。在编译链接过程中，VC++将保存此程序，并生成一个同名的工作区。保存文件时一定要注意扩展名指定为“.c”。

如果执行程序没有错误，弹出的信息窗口将显示：0 error(s) 0 warning(s)。

有时会有几个警告信息(warning)，但不影响程序执行。

假如有致命的错误(error)，可双击某行出错信息，程序窗口中会指示对应的出错位置，根据信息窗口的提示分别予以纠正。纠正之后，选择“编译/执行”命令(或按 Ctrl+F5 组合键)执行程序。

成功运行之后，VC++将自动弹出数据输入/输出窗口，按任意键将关闭该窗口。

此外，对于编译、链接、执行操作，VC++还提供了一组如图 2.6 所示的工具按钮方便操作。

图 2.6　程序运行的工具按钮

5. 关闭程序工作区

当一个程序编译链接后，VC++自动产生相应的工作区，以完成程序的运行和调试。若想执行第二个程序，必须关闭前一个程序的工作区，然后通过新的编译链接，产生第二个程序的工作区，否则运行的将一直是前一个程序。选择“文件”→“关闭工作区”命令，然后在弹出的对话框中单击“否”按钮。如果单击“是”按钮将同时关闭源程序窗口。

6. 命令行参数处理

VC++是一个基于窗口操作的 C++系统，没有提供命令行参数功能。要实现此功能需要在 Windows 的“MS-DOS 方式”窗口中以命令方式实现。具体步骤如下：

(1) 将源程序正确编译链接，生成可执行程序(例如 prog.exe)，查看它所在的路径(在 C 源程序同层的 Debug 文件夹下)。

(2) 单击任务栏中的“开始”按钮，选择“运行”命令，输入 command，然后按 Enter 键，进入 MS-DOS 窗口。

(3) 在 MS-DOS 窗口中输入“文件名　参数 1　参数 2……”，带参数运行程序。

7. 程序调试

VC++是一个完全基于 Windows 的系统，调试过程中可以方便地使用鼠标。在程序的调试过程中，常会遇到以下几种情况。

1) 程序执行到中途暂停以便观察阶段性结果

(1) 操作方法一：使程序执行到光标所在的那一行暂停。

- 将光标移动并定位到需要暂停的所在行上。
- 选择“编译”→“开始调试”→Run to cursor 命令，或者按 Ctrl+F10 组合键。
- 操作之后，程序将执行到光标所在行暂停。如果此时把光标移动到后面的某个位置，再进行相同的操作，程序将从刚才的暂停点继续执行到新的光标位置，第二次暂停。

(2) 操作方法二：在需暂停的行上设置断点。

- 将光标移动并定位到需设置断点的行上。
- 单击“编译微型条”中最右面的按钮或按 F9 键。

这时，被设置了断点的行前面会有一个红色圆点标志。不管是通过光标位置还是断点设置，其所在的程序行必须是程序执行的必经之路，即不应该是分支结构中的语句，因为该语句在程序执行中受条件的限制，有可能因条件的不满足而不被执行，这时程序将一直执行到结束或下一个断点为止。

2) 设置需观察的结果变量

将程序执行到指定的位置暂停的目的是为了查看有关的中间结果。如图 2.7 中，左下角窗口中系统自动显示了有关变量的值。如果还想增加观察变量，可在窗口右下角的 Name 文本框中输入相应的变量名。

3) 单步执行

当程序执行到某个位置时发现结果已经不正确了，说明在此之前肯定有错误存在。如果能够确定一小段程序可能有错，先按上面步骤暂停在该小段程序的头一行，再输入若干个查看变量，然后单步执行，即一次执行一行语句，逐行检查下来，看看到底哪一行出现了错误，从而确定错误的语句并予以纠正。

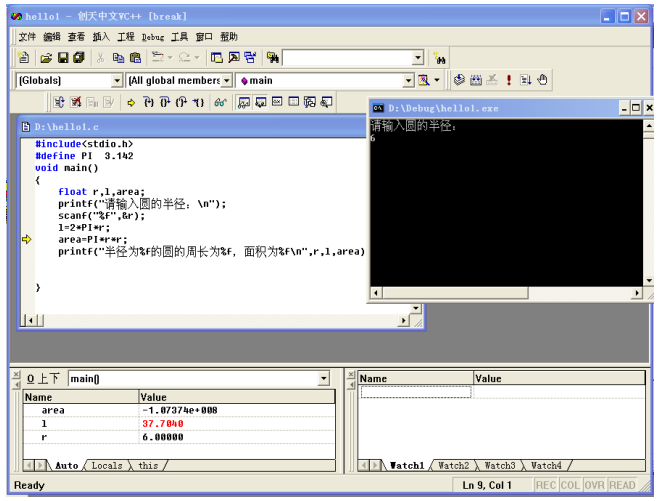


图 2.7　VC++中观察变量值状况

单步执行时单击“调试条”→Step Over 按钮或按 F10 键。若遇到自定义函数调用，想进入函数进行单步执行，可单击 Step Into 按钮或按 F11 键。若想结束函数的单步执行，可单击 Step Out 按钮或按 Shift+F11 组合键。对不是函数调用的语句来说，F11 键与 F10 键的作用相同，但一般不要对系统函数按 F11 键。

4) 断点的使用

使用断点也可以使程序暂停。一旦设置了断点，不管是否还需要调试程序，每次执行程序都会在断点上暂停，因此调试结束后应取消所定义的断点。操作方法如下：

将光标定位到所在行，单击“编译微型条”中最右面的按钮或按 F9 键。

如果有多个断点想全部取消，可执行“编辑”→“断点”命令，屏幕上会出现 Breakpoints 窗口，下方列出了所有断点，单击 Remove All 按钮将取消所有断点。断点通常用于调试较长的程序，可以避免使用 Run to Cursor 命令(运行程序到光标处暂停)或按 Ctrl+F10 组合键时，经常要把光标定位到不同的地方。而对于长度为上百行的程序，要寻找某位置并不太方便。

如果一个程序设置了多个断点，按一次 Ctrl+F5 组合键会暂停在第一个断点，再按一次 Ctrl+F5 组合键会继续执行到第二个断点暂停……这样依次执行下去。

5) 停止调试

使用 Debug→Stop Debugging 命令或按 Shift+F5 组合键，可以结束调试，回到正常的运行状态。

以上操作只是 VC++最基本的操作(编辑、编译、链接和运行简单的 C 程序)，适合初学 C 语言的读者，如果需要更复杂的其他操作，请参考有关 Visual C++手册。

2.3　上机实践：简单 C 程序的编辑、编译、链接和运行

2.3.1　实验目的

(1) 理解和掌握在 VC++等环境下编辑、编译、链接和运行简单 C 程序的方法和过程。

(2) 通过编辑、编译、链接和运行简单的 C 程序，掌握 C 语言源程序的结构特点，了解 C 语言中常量和变量的简单使用方法(输入、输出和简单计算)。

(3) 了解简单的 C 程序调试方法。

(4) 了解 VC++等环境下简单 C 程序的编辑、编译、链接、运行过程。

2.3.2　实验学时

1 学时。

2.3.3　实验内容和步骤

1. 尝试使用不同的方法启动 VC++

(1) 可以通过桌面创建的 VC++程序快捷启动 VC++。

(2) 也可以找到 VC++的安装目录，直接运行 VC++应用程序。例如，C:\Program Files\Microsoft Visual Studio\Common\MSDev98\Bin\MSDEV.EXE。

2. 熟悉使用 VC++集成开发环境

熟悉使用 VC++集成开发环境编辑、编译、链接和运行简单 C 程序的过程，了解 VC++集成环境的其他使用方法。

例如，现在需要设计一个简单的 C 程序，该程序输入 3 个任意整数，输出它们的最大值。

(1) 使用上面的方法启动 VC++，选择“文件”→“新建”命令，打开“新建”对话框，选择“文件”选项卡，选择下面的 C++ Source File，在右边的文件名中输入文件名(如 myfirst.C)，选择保存位置(如桌面)，单击“确定”按钮，如图 2.8 所示。

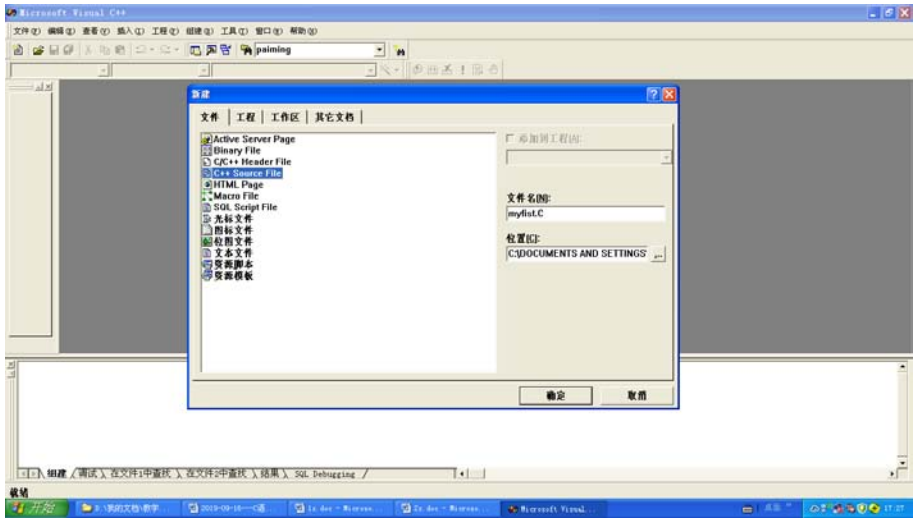


图 2.8　新建 C 程序文件

(2) 进入如下编辑状态，如图 2.9 所示。

(3) 光标停留在编辑区中，等待输入源程序。输入程序如图 2.10 所示。

(4) 输入程序完毕，单击编译快捷工具按钮，创建默认的工作区，如图 2.11 所示。

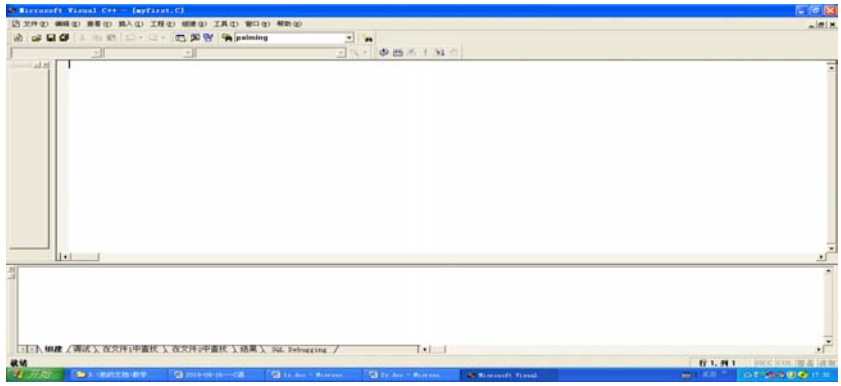


图 2.9　进入编辑状态

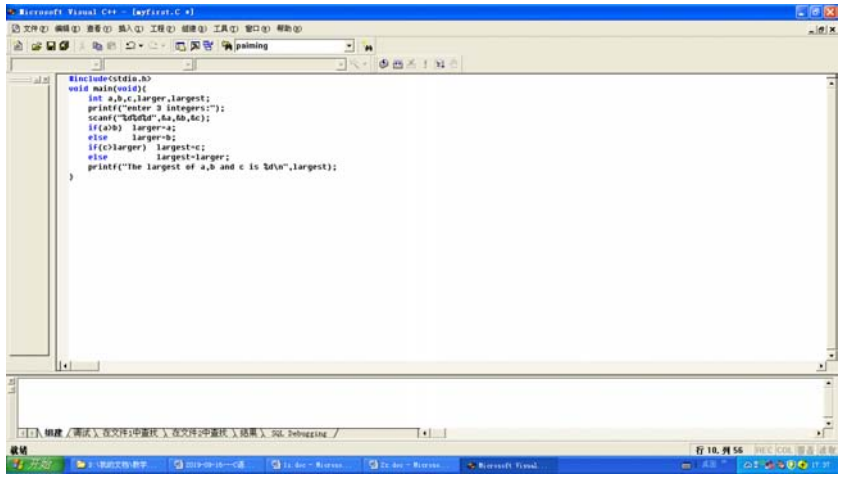


图 2.10　输入源程序文件 myfirst.C

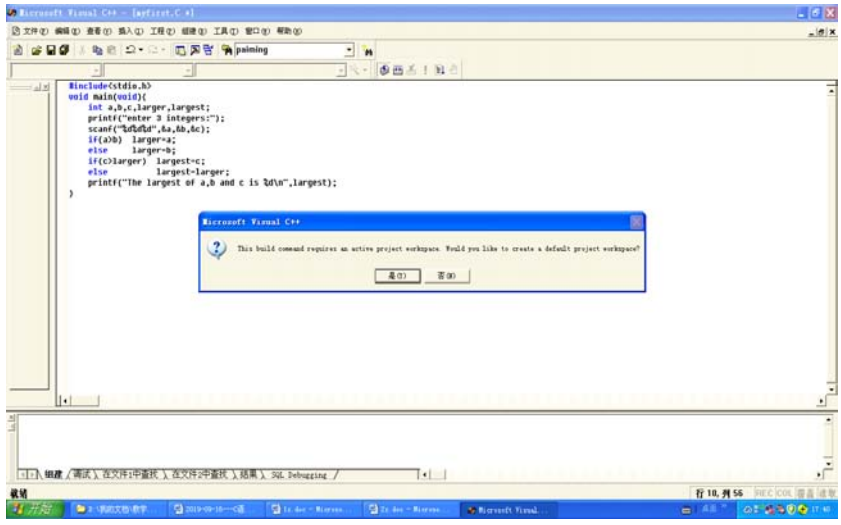


图 2.11　创建默认工作区

(5) 编译之后，下面的信息窗口显示编辑的结果如图 2.12 所示。

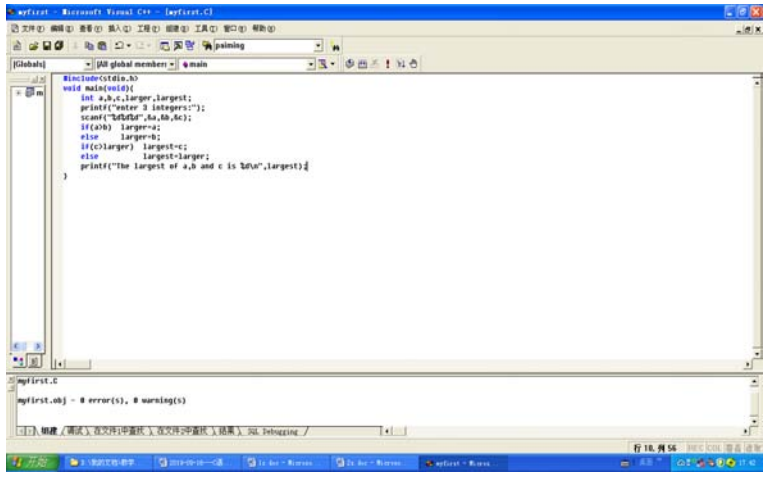


图 2.12　编译结果

(6) 编译结果显示，没有错误，得到目标文件 myfirst.obj，接着就可以进行链接，单击 Build 工具按钮，如图 2.13 所示。

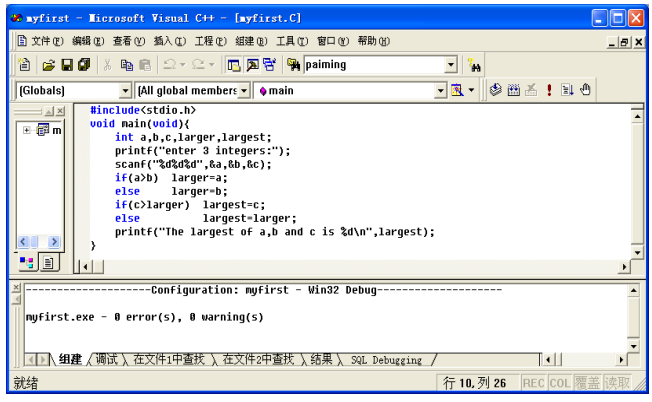


图 2.13　链接生成执行程序时的窗口信息

(7) 经过链接之后，得到可执行的程序文件 myfirst.exe。此时可以通过单击工具按钮 BuildExecute 执行程序，得到程序的运行结果，如图 2.14 所示。

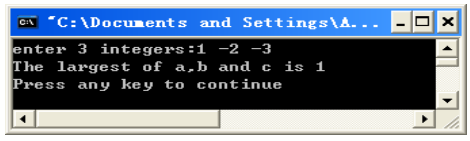


图 2.14　程序运行结果

3.了解 Dev C++平台下 C 程序的编辑、编译、链接和运行过程

Dev C++平台的使用方法类似于 VC++平台，由于这个软件网上容易获取，而且体积较小，在此简单讲述一下它的使用。

(1) 建立 C 源程序。类似 VC++，通过 File→New→SourceFile 创建源程序文件，如图 2.15 所示。

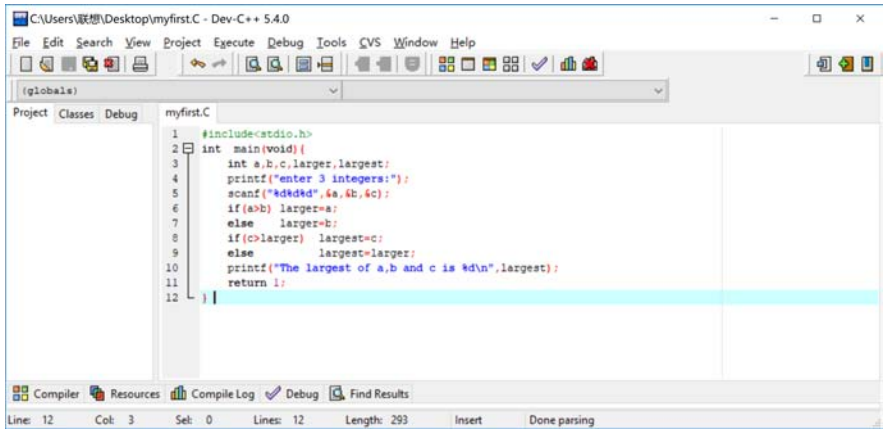


图 2.15　使用 Dev C++创建 C 源程序文件 myfirst.C

(2) 编译 myfirst.C。通过菜单 Execute→Compile 或者直接单击 Compile 工具按钮实现对源程序文件的编译，如图 2.16 所示。

图 2.16　编译源程序文件

编译过程如果有错，则给出错误信息。假设上面的源程序文件中，将语句 printf("enter 3 integers:");故意写错为 printf("enter 3 integers:);，编译时就会给出错误信息，如图 2.17 所示。

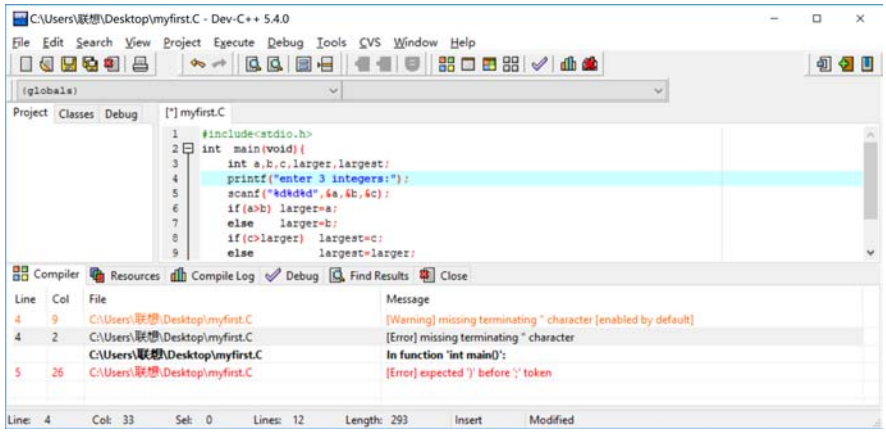


图 2.17　编译错误信息

双节错误信息行，可以点亮错误对应的语句。根据错误提示，修改错误，直到编译通过为止。

(3) 运行程序。类似于 VC++，通过菜单 Execute→Run 或者工具按钮 Run 都可以执行程序。得到运行结果，如图 2.18 所示。

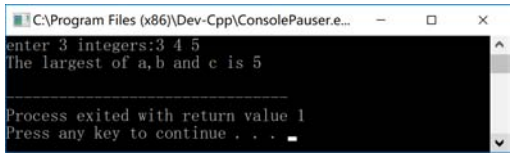


图 2.18　运行结果

当然，我们也可以在直接运行执行程序。比如，可以在命令行方式下，执行程序 myfirst.exe，如图 2.19 所示。

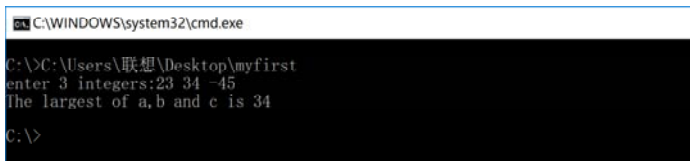


图 2.19　执行程序结果

4. 练习

(1) 尝试将上面实验中的源程序实现的功能改为：输入 4 个整数，输出其中的最小值。分别在 VC++和 DEV C++平台上运行，尽量根据错误信息提示，进行代码的调试修改。

(2) 分别在 Dev C++和 VC++ 6.0 平台上编辑、编译、链接和运行以下程序，体会不同平台上运行 C 程序的特点。

```
#include <stdio.h>
int max(int x,int y){
        int z;
        if(x>y) z=x;
        else z=y;
        return z;
}

void main(void){
      int a,b,c,maximum;
      double average;

   printf("Please enter a,b and c:");
      scanf("%d%d%d",&a,&b,&c);

      maximum=max(max(a,b),c);
      printf("maximum=%d\n",maximum);

      average=(a+b+c)/3.0;
      printf("average=%f\n",average);
}
```

(3) 将上面程序中的 average=(a+b+c)/3.0;语句改为 average=1/3*(a+b+c);，验证结果是否正确？再将上面程序中的语句 printf("average=%f\n",average);改为 printf("average= %d\n", average);，结果还正确吗？再去掉上面程序中的#include <stdio.h>，程序还能正常运行吗？

(4) (选做)自行练习指导书上讲述的其他简单程序调试方法。