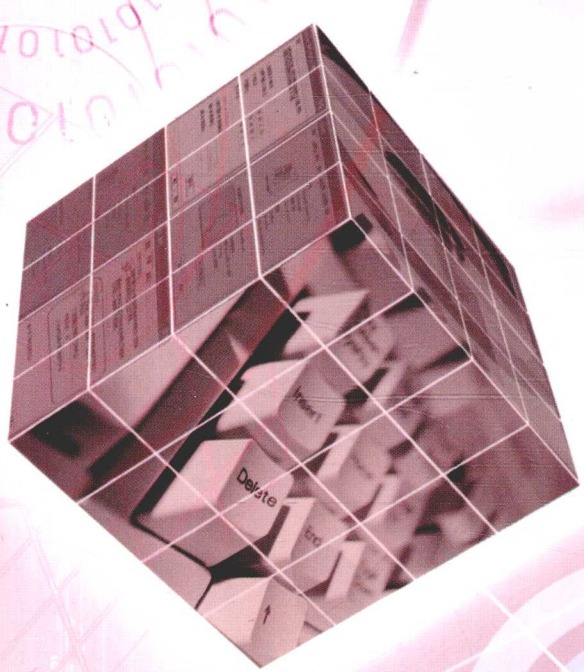
普通高等学校计算机教育规划教材

C/C++程序设计教程

张世民 主 编
刘志超 梁普选 杨秦建 李 颖 副主编



中国铁道出版社
CHINA RAILWAY PUBLISHING HOUSE

普通高等学校计算机教育规划教材

C/C++程序设计教程

张世民　主　编
刘志超　梁普选　杨秦建　李　颖　副主编

中国铁道出版社
CHINA RAILWAY PUBLISHING HOUSE

内 容 简 介

本书主要内容包括预备知识、基本数据类型和表达式、C 程序的流程控制、复杂数据类型、结构化程序设计的应用、函数和预处理、文件和面向对象程序设计。用 Visual C++ 6.0 作为调试和运行程序的环境。本书各章后附有多种类型的习题，可帮助学生从多个角度掌握所学内容。

本书适合作为高等院校“C 程序设计”课程的教材，也可作为参加计算机等级考试和其他自学者的参考用书。

图书在版编目（CIP）数据

C/C++程序设计教程 / 张世民主编. —北京：中国铁道出版社，2009.1
普通高等学校计算机教育规划教材
ISBN 978-7-113-09551-2

Ⅰ.C… Ⅱ.张… Ⅲ.C 语言—程序设计—高等学校—教材 Ⅳ.TP312

中国版本图书馆 CIP 数据核字（2009）第 012807 号

书　　名：C/C++程序设计教程
作　　者：张世民　主编

策划编辑：杨　勇
责任编辑：秦绪好　　**编辑部电话：**（010）63583215
编辑助理：刘彦会　杜　鹃
封面设计：路　瑶　　**封面制作：**白　雪
责任印制：李　佳

出版发行：中国铁道出版社（北京市宣武区右安门西街 8 号　邮政编码：100054）
印　　刷：三河市华丰印刷厂
版　　次：2009 年 2 月第 1 版　2009 年 2 月第 1 次印刷
开　　本：787mm×1092mm　1/16　**印张：**17.75　**字数：**416 千
书　　号：ISBN 978-7-113-09551-2/TP・3125
定　　价：28.00 元

FOREWORD

前 言

高级语言程序设计是高等学校重要的计算机基础课程。这是一门以培养学生程序设计基本方法和技能为目标、以实践能力为重点的特色鲜明的课程。它以编程语言为平台，介绍高级语言程序设计的知识，在实践中逐步掌握程序设计的思想和方法，培养分析问题、解决问题和程序设计的能力。

C 语言是国内外广泛使用的计算机程序设计语言之一，是国内外大学普遍开设的程序设计课程。C++是目前广泛使用的面向对象的程序设计语言。

C/C++语言程序设计是一门实践性很强的课程。学习者必须在深入理解教材知识的基础上，通过大量的上机实践加强程序设计技术的基本技能训练，从而逐步理解和掌握编程的基本思想和调试程序的能力。

我们编写本教材，力求突出以下特点：

① 符合 C 语言程序设计教学要求，按照大一新生第一次接触计算机语言零起点的角度组织编写，删繁就简，针对性强，教学内容循序渐进，符合学习规律。

② 书中示例丰富，习题类型多样、难易适中，覆盖面广，例题和习题突出了编程能力训练的重要性，多角度加深学生对程序设计思想的理解，提高编程能力。

全书共 8 章，主要内容包括预备知识、基本数据类型和表达式、C 程序的流程控制、复杂数据类型、结构化程序设计的应用、函数和预处理、文件和面向对象程序设计。另外，每章均配有精心安排的多种类型的习题，以帮助提高学生学习兴趣和主动性，帮助学生多角度掌握所学内容。

本书的作者长期从事 C/C++程序设计语言课程的教学工作，并利用 C/C++等语言开发过多个软件项目，有着丰富的教学经验和较强的科研能力，对 C/C++程序设计有比较深入的理解和掌握。

本书使用较流行的 Visual C++ 6.0 作为调试和运行程序的环境。本书中所给出的程序示例均在 Visual C++ 6.0 环境下通过调试与运行。

本书由张世民主编。各章编写分工如下：第 1 章由梁普选、杨秦建编写，第 2 章由李颖编写，第 3 章由李媛、李颖编写，第 4 章由王美华编写，第 5 章由张立岩、郑琨编写，第 6 章由张世民、刘志超编写，第 7 章由刘志超编写，第 8 章由梁普选、杨秦建编写。全书由张世民统稿。

本书适合作为高等院校“C 程序设计”课程的教材，也可作为参加计算机等级考试和其他自学者的参考用书。

在编写本书的过程中参考了很多教材，在此向这些作者表示感谢。同时，向对教材提出中肯建议的老师表示谢意和致敬。由于作者水平有限，书中难免会有疏漏和不足之处，敬请读者指正。

编 者

2008 年 11 月

CONTENTS

目录

第 1 章 预备知识

对于理工科大学生而言，除了掌握专业知识外，分析问题、解决问题的能力训练以及严谨踏实的科研作风和思维方法的培养，都是日后工作的基础。学习计算机编程语言是一种十分有益的训练方式，而计算机语言本身又是与计算机进行交互的有力工具。因此，在校期间掌握一门编程语言并逐渐提高程序设计能力是必需的。本章简要介绍了计算机软件基础、C 语言的发展、结构特点和基本格式要求等知识，并针对初学者提出了 C 语言程序设计的学习方法和建议，最后介绍了 Visual C++编程环境的使用及 C 语言程序的开发过程。

1.1 计算机软件基础

一台计算机是由硬件系统和软件系统两大部分构成的。硬件是计算机的物质基础，而软件是计算机的“灵魂”，没有软件的计算机什么也做不了。计算机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历程。

1．机器语言

二进制是计算机语言的基础，也就是说计算机内部存储、加工并处理的信息是由 0 和 1 组成的二进制序列。计算机诞生之初，人们只能用二进制指令去命令计算机工作，即写出一串串由 0 和 1 组成的指令序列交由计算机执行，这种语言称为机器语言。使用机器语言是十分痛苦的，需要程序员记忆大量的机器码，特别是在程序有错误需要修改时，更是如此。由于机器语言十分依赖于计算机硬件结构，每台计算机的指令系统往往各不相同，导致在一台计算机上执行的程序，要想在另一台计算机上执行，必须另外编写程序，造成了重复工作。但由于使用的是针对特定型号计算机的语言，因此运算效率是所有语言中最高的，这就是第一代计算机语言。

2．汇编语言

为了减轻使用机器语言编程的烦琐，人们进行了一种改进：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，例如，使用 ADD 代表加法，MOV 代表数据传递等。这样，人们很容易读懂并理解程序在做什么，纠错和维护都变得比较方便，这种程序设计语言称为汇编语言，即第二代计算机语言。然而，计算机是不认识这些符号的，这就需要一个专门的程序，专门负责将这些符号翻译成二进制数的机器语言。汇编语言同样十分依赖于机器硬件，移植性不好，但效率非常高，针对计算机特定硬件而编制的汇编语言程序，能准确地发挥计算机硬件功能和特长，程序精练而质量高，所以至今仍是一种强有力的软件开发工具。

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接近于数学或人的自然语言，同时又不依赖于计算机硬件的计算机语言。经过不断地努力，从 1954 年出现第一个完全脱离机器硬件的高级语言到现在，共有几百种高级语言出现。有重要意义的有几十种，影响较大使用较广泛的有 Fortran、Algol、Cobol、BASIC、Lisp、Ada、C/C++、Visual Basic、Delphi、Java 等。高级语言的发展也经历了从早期语言到结构化程序设计语言的发展过程。同样，软件的开发也由最初的个体手工作坊式逐渐过渡到现在的流水线式的工业化生产模式。

20 世纪 60 年代中后期，软件越来越多，规模越来越大，而当时软件的生产基本上是人自为战，缺乏科学规范的规划与测试、评估标准，其恶果是大批耗费巨资建立起来的软件系统，由于包含错误而无法使用，甚至带来巨大损失，软件给人的感觉是越来越不可靠，以致几乎没有不出错的软件。这一切，极大地震动了计算机界，史称“软件危机”。人们认识到：大型程序的编制不同于一些小程序，它应该是一项新的技术，应该像处理工程一样处理软件研制的全过程。程序的设计应易于保证正确性，也便于验证正确性。1970 年，第一个结构化程序设计语言 Pascal 出现，标志着结构化程序设计的开始。到了 20 世纪 80 年代末，随着软件规模的扩展，程序员逐渐认识到结构化程序设计模式也有其不足，从而导致软件设计理念发生了革命性变化，出现了面向对象的程序设计模式，这种软件开发模式加速了软件作为产品的商业化发展趋势。可以说，面向对象的编程模式对于软件的发展所起到的作用是无法估量的。

1.2 C 语言发展史

C 语言诞生于 20 世纪 70 年代，是由 UNIX 操作系统的研制者 Dennis Ritchie 和 Ken Thompson 于 1970 年在 B 语言的基础上发展和完善起来的。1972 年，Thompson 等人在小型机 PDP-11 上用 C 语言重写 UNIX 操作系统内核，可以说 C 语言与 UNIX 操作系统同时诞生。

20 世纪 80 年代，C 语言被程序员广泛使用，从而逐渐演化为个人计算机上流行的编程工具。1983 年，美国国家标准委员会（ANSI）对 C 语言进行了标准化，颁布了第一个 C 语言标准草案（1983 ANSI C）。

为了适应大规模软件的生产制作，在 C 语言基础上，贝尔实验室的 Bjarne Stroustrup 博士及其同事开始对其进行了改进和扩充，将“类”引入到了 C 语言中，1983 年构成了最早的 C++语言。为了适应大规模软件的开发，Stroustrup 博士又为 C++引进了多重继承、运算符重载、引用、虚函数等许多特性。美国国家标准协会 ANSI 和国际标准化组织 ISO 一起进行了标准化工作，并于 1998 年正式发布了 C++语言的国际标准 ISO/IEC:1998—14882，从此软件开发进入到一个快速发展的阶段。

20 世纪 90 年代，美国微软公司（Microsoft）为了降低 Windows 应用程序的开发成本，拉动应用软件在软件市场的地位，于 1992 年发布了含有 MFC 2.0 的 Visual C++ 1.0，一个划时代的可视化 C++集成开发环境诞生了。所谓的 MFC 就是一个软件包（framework），即用面向对象的方法对 Win32 API（应用程序接口）进行了封装，提高了 Windows 平台上的程序开发效率。1998 年，Microsoft 公司推出了目前最流行的 Visual C++ 6.0 版本。2002 年，推出了 Visual C++ 7.0，即嵌入在 Visual Studio.NET 框架中的 Visual C++.NET 2002。目前，最新的 VC++版本是 Visual C++.NET 2005——VC 8.0。

随着 Internet 的普及，美国 Sun 公司于 1995 年推出了因特网环境下通用的编程语言——Java 语言。Java 语言吸取了 C++的成功之处，摒弃了 C++的不足。使得 Java 语言逐渐演化成为 Internet 环境下的世界级通用语言。而 Microsoft 公司为了与如日中天的 Sun 公司抗衡，于 2005 年推出了 Visual C# .NET 2005。

尽管软件开发环境的研发脚步一直没有停止，但对于初学者而言，最重要的是打好计算机编程的基础。

1.3 C 语言特征

1. 中级语言

C 语言是一门中级语言，也就是说它有“低级语言”的固有特征：允许自由访问计算机物理地址，能进行位操作，可以对计算机硬件接口直接访问，生成目标代码的质量高，程序执行效率高。它又兼备“高级语言”的固有特征：语句简洁、紧凑，运算符灵活、数据类型丰富，具有结构化的控制语句，可移植性好。

2. 编程环境及使用

为了适应计算机软件市场的需求，计算机语言的编程环境随着操作系统的变化在同步变化。早期的磁盘操作系统下的 C 编程环境为 Borland 公司的 Turbo C 2.0，随着 Windows 操作系统的推出，其编程环境演化为可视化的集成编程环境，其代表作有 Borland 公司的 C++ Build 5.0 和 Microsoft 公司的 Visual C++ 6.0，本书所用的 C 编程环境就是 Visual C++ 6.0。

任何一种编程环境都是一个封闭的西文符号体系，C 语言也不例外。在学习 C 语言程序设计之路上，首先要很快熟悉 C 语言编程环境，可以说熟练掌握 C 语言编程环境是提高学习效率的主要途径之一。从另一个角度来看，程序是在不违反编译器基本“规则”的前提下，把求解实际问题的方法、步骤交给计算机来实现。如果教材中、课堂上讲解的例题或自己编写的程序不能交给计算机实现，又有何意义？因此，学习 C 语言的过程中必须有大量的上机实践作为支撑，从这个角度看，计算机语言程序设计是一门实践性很强的课程。

因此，对于初学者来说，学会上机调试程序是学习计算机编程首要解决的问题。这是因为从课堂或书本获取的计算机编程知识，必须通过亲手编制并上机调试，才会对该程序算法有更为深刻的消化和理解，这是学会编程的必要条件。对于入门者，上机调试程序可以对教材中的实例举一反三，使相关知识融会贯通，迈向程序设计自由之路。对于熟练者，只有上机调试程序才会实现预期的软件设计目标。

3. C 语言格式和结构特点

任何一种计算机语言都具有特定的语法规则和独有的表现形式。因此，程序的书写格式和程序的构成规则是程序语言表现形式的一个重要方面。按照规定和构成规则书写程序，不仅可使程序设计人员和使用程序的人员容易理解，更重要的是，把程序输入到计算机中时，计算机能够理解并正确执行。

（1）C 程序格式

下面编写一段计算圆柱体体积的程序。

【例 1.1】计算圆柱体的体积。

源程序如下：

```
#include <stdio.h>    /*包含头文件 stdio.h，支持程序中的输入/输出语句功能*/
int main()
{
    int radiu,hight;                    /*定义表示圆半径和圆柱体高的两个整型变量*/
    float volum;                        /*定义保存圆柱体体积的实型变量 volum*/
    scanf("%d%d",&radiu,&hight);        /*由键盘输入圆的半径和圆柱体的高*/
    volum=3.14159*radiu*radiu*hight;    /*计算圆柱体体积*/
    printf("volum=%f\n",volum);         /*屏幕输出计算结果*/
    return 0;
}
```

这里暂且不必顾及 C 语言源程序中各个语句的功能，首先把注意力放在其格式特点上。我们看到 C 语言程序格式有以下特点：

① C 语言程序习惯上使用小写英文字母，也可以使用大写字母，但大写字母常常用于符号常量的定义或其他特殊用途。

② C 语言使用分号“;”作为语句之间的分隔符，每一条语句占用一个书写行的位置。

③ C 语言程序中用大括弧对“{}”表示程序结构的层次范围。一个完整的程序模块要用一对大括弧表示该程序模块的范围，如上面程序中的第 3 行和最后一行的大括弧对。

④ 一般情况下每个语句占一行，采用缩进式书写 C 程序。即每个控制结构（一对花括弧）都缩进一个跳格键（Tab）位。

⑤ 空格作为语句中标识符、关键字间的分隔符。为了增强可读性，程序中可适当加些空格和空行。但不能在程序中所使用的关键词（称为保留字）及各种标识符（变量名、函数名）名字中间插入空格。

⑥ 为了便于阅读理解 C 源程序，例 1.1 中使用了注释语句对每条语句做出解释。C 编译器在编译源程序时，对注释语句不予执行。

应该说采用这样格式书写的程序，便于阅读和理解，同时也体现了结构化程序设计特征。

对专业术语的进一步说明：

⑦ 关键词：关键词是被定义在 C 编译器系统内部的一些特定符号，对一条语句的作用做出解释，在程序中起到命令动词的作用。例如，例 1.1 中的 scanf(...)表示实现数据输入库函数的关键词，而 printf(...)是实现数据输出的库函数的关键词。

⑧ 标识符：标识符是由编程者所定义，通常表示程序中的常量或变量的名称。如例 1.1 中表示圆柱体体积的变量 volum 就是用户定义的标识符。

由例 1.1 可知，一段程序都是由系统定义的关键字、用户定义的标识符加之实现算法的表达式所组合而成。

（2）C 程序结构

一个完整的 C 语言程序是由一个或多个具有相对独立功能的程序模块结合而成，这样的程序称为函数。每个函数都是由函数名和大括弧对“{}”包围的若干语句组成，为了更直观地了解 C 语言程序的特点，重新编写例 1.1。

【例 1.2】计算圆柱体的体积。

源程序如下：

```
#include <stdio.h>
float func1(int,int);                    /*函数 func1()的原型声明*/
int main()
{
    int radiu,hight;
    float v;
    scanf("%d%d",&radiu,&hight);
    v=func1(radiu, hight);               /*调用函数 func1(), 得到圆柱体体积计算结果*/
    printf("volum=%f\n",v);
    return 0;
}
float func1(int r,int h)                 /*定义函数 func1(), 实现圆柱体体积的计算*/
{
    return 3.14159*r*r*h;
}
```

说明：主函数 main()是 C 源程序中唯一不可缺的函数，它表示程序运行的入口，无论主函数位于整个源程序的什么位置，都从主函数开始执行。也就是说，一个完整的 C 程序，主函数 main()是必不可少的。

对例 1.2 的分析：第 2 条语句是对函数 func1()进行了原型声明，通知后续程序按原型声明定义并调用函数 func1()。主函数中的第 3 条语句是定义了表示圆半径和圆柱体高的两个整型变量 radiu 和 hight，接着定义一个实型变量 volum，用于保存圆柱体体积的计算结果。由键盘输入圆的半径和圆柱体的高之后，调用函数 func1()，实现了圆柱体体积的计算，最后在屏幕上显示输出圆柱体体积的计算结果。

1.4 C 语言学习方法

由于 C 语言灵活、强大，初学者要在短时间内全面地掌握它是不现实的，因而学习 C 语言编程是一个循序渐进的过程。

1. 预习教材

首先要预习教材，了解教师在课堂中要讲解的知识。在听课过程中，认真做好笔记，特别是教师在课堂中讲解的例题。课后仔细阅读教材中课堂讲解的章节内容，在阅读过程中，重点理解教材中的例题。不要在细枝末节上浪费过多的精力。(如++、--前缀/后缀用于表达式的计算，不常用运算符的运算顺序和类型转换等。)

2. 完成作业

认真完成教师在课堂上布置的作业。认真是指独立完成作业，当然对于初学者借鉴教材或他人所编写的源程序完成作业无可非议，但如果从始至终都在抄袭或模仿他人的程序，那就永远学不会编程。可按照教材或课堂中的实例试着改编程序，逐渐地独立编写程序，完成作业。

3. 上机调试程序

由于高级语言程序设计课程是一门实践性很强的课程，因而对于学生而言，上机调试程序尤为重要。首先要带着完成的作业与课堂笔记去上机，因为教师课堂讲解的例题或教材中的实例可通过上机实践加深理解，融会贯通。更为重要的是，你自己完成的作业可通过上机得到验证。初学者调试程序出错是正常的，关键是如何面对它。千万不能回避调试程序中出现的错误，因为这正是我们学习编程的最好时机。要把计算机当做自己的第二教师，面对屏幕上提示的错误，不仅要记录下错误语句及错误提示，更要记下纠正错误的方法和思路。这样，在下次上机时就不会重蹈覆辙。

4. 课后总结

将上机调试程序的过程与教科书中的相关知识联系在一起，再次阅读教材相关内容，我们会感到豁然开朗，课堂或书本中的疑惑就会迎刃而解，我们所学到的章节知识就会融会贯通。总而言之，在学习编程过程中，不要在细枝末节上浪费过多的精力，但一定要熟练掌握最基本内容。例如，C 语言中的流程控制语句的使用，数组、函数、指针等基本知识及算法应用，为逐步提高程序设计能力打下坚实的基础。

5. 难点处理

C 语言中有些内容的确比较难于理解，如 C 语言中的指针。遇到这样的情况，一是求得教师的辅导答疑和帮助；二是同学之间针对某个问题进行交流探讨，实际上这一点很重要，因为您所遇到的困惑，其他同学可能会给你恰如其分的解释，使你茅塞顿开；三是借阅相关的 C 语言教材，查阅相关的章节，或许其他教材中的内容能对你遇到的问题给予启发、提示，帮助您理解。当然，如果拥有正常上机实验之外的机会，特别是自己拥有计算机，要珍惜并好好利用它，安装上 C/C++ 编译器进行编译、验证。

1.5 程序与算法

算法是思想，程序是表达。算法是求解问题的思路，程序是这个思路的具体实现过程。

一个完整的计算机程序包括数据和操作步骤两方面内容，而操作步骤就是算法。

数据是程序的基本元素，算法是程序的核心，缺一不可。算法来源于生活，例如要做一盘红烧肉。第一步要准备做菜的各种原料，需要什么部位的猪肉及放置葱、姜的量是多少（数据及数据类型）。准备好这些数据之后，第二步就可以按照做红烧肉菜谱（算法），完成一道菜的制作（运行程序），程序运行的结果就是一盘可口美味的红烧肉。根据这个例子，使我们认识到做任何事情时都有一定的步骤，这就是算法。其实，生活中还有许许多多这样的例子，如过马路、买东西等。表示算法的方法有以下三种：

① 用自然语言表示算法。

② 用伪代码表示算法。

③ 用流程图表示算法。

算法的最佳表示方法是流程图，程序设计者常以流程图的方式来描述算法，就是用一些框图来表示各种操作，使算法直观形象，易于理解。美国国家标准化协会 ANSI 规定了一些常用的流

程图符号，已被世界各国程序工作者普遍采用，流程图符号如图 1-1 所示。

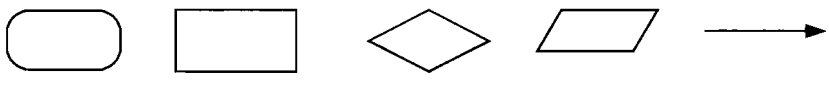


图 1-1 流程图符号

绘制流程图时一定不要忘记加上箭头，因为它反映流程的先后顺序，如果不画箭头就难以判断每一个框的执行顺序。下面就将日常生活中过马路的情形用流程图表示出来，如图 1-2 所示。

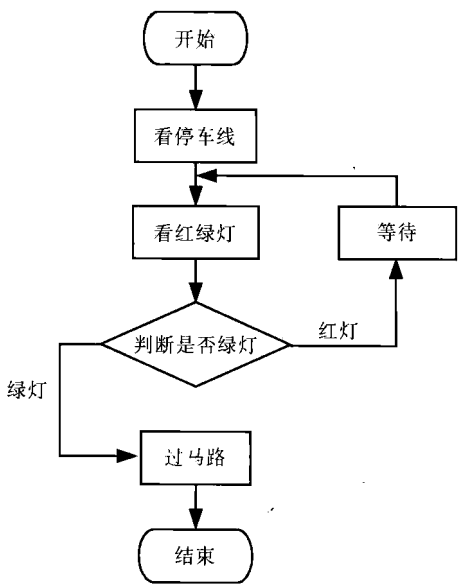


图 1-2 穿越马路的流程图

由此看来，在程序设计初期，可使用流程图形象地表述源程序的算法和执行过程。

1.6 C 程序开发过程

C 语言是一种编译型程序语言，和大多数流行的软件开发环境类似，C 语言程序的开发过程要经历四个基本阶段：编辑→编译→连接→运行。

1．源程序文件的编辑

过去要把自己编制的 C 语言程序输入到计算机，首先要使用系统提供的编辑程序建立 C 语言程序的源文件。建立之后的源文件以文本文件形式存储在计算机的文件系统（硬盘）中，再进行编译、连接。源程序文件名称由用户自定义，C 语言源文件的文件扩展名为.c，例如 test1.c、test2.c，C++语言源文件的文件扩展名.cpp。用于建立 C 源程序文件的编辑器很多，但目前流行的 C/C++集成开发环境，如 Turbo C 3.0、Borland C++ 5.0 及后面介绍的 Visual C++ 6.0 已经把 C/C++源程序文件的编辑、编译、连接和运行集成为一体，从整体上提高了编程的效率。