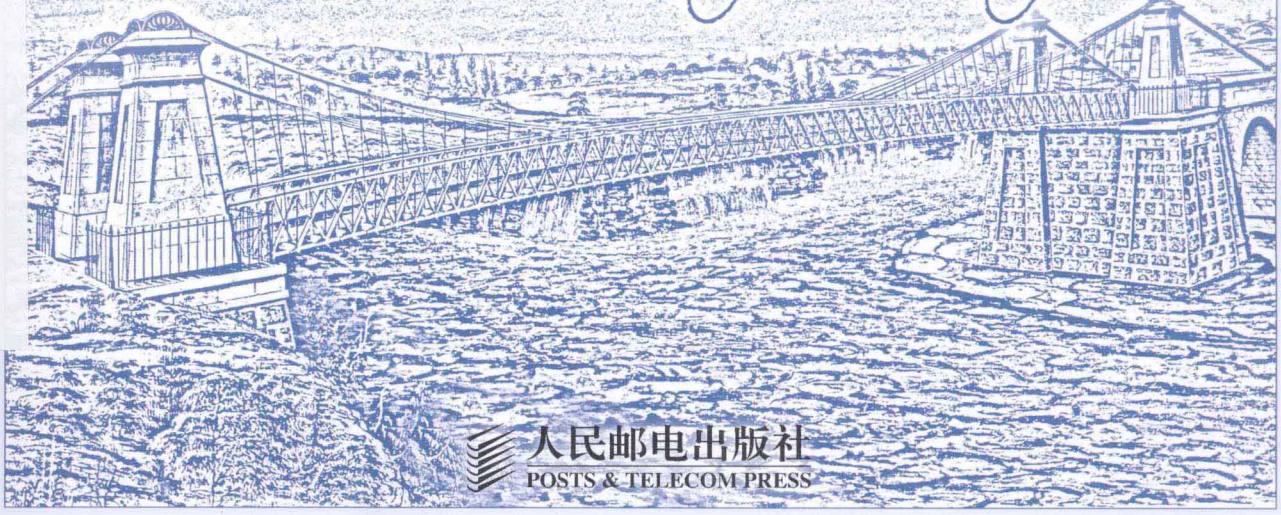
普通高等教育
软件工程 “十二五”规划教材

12th Five-Year Plan Textbooks
of Software Engineering

C语言工程实训教程

王卓 万立中 刘伯成 ◎ 编著

The Practice of C Programming



人民邮电出版社
POSTS & TELECOM PRESS

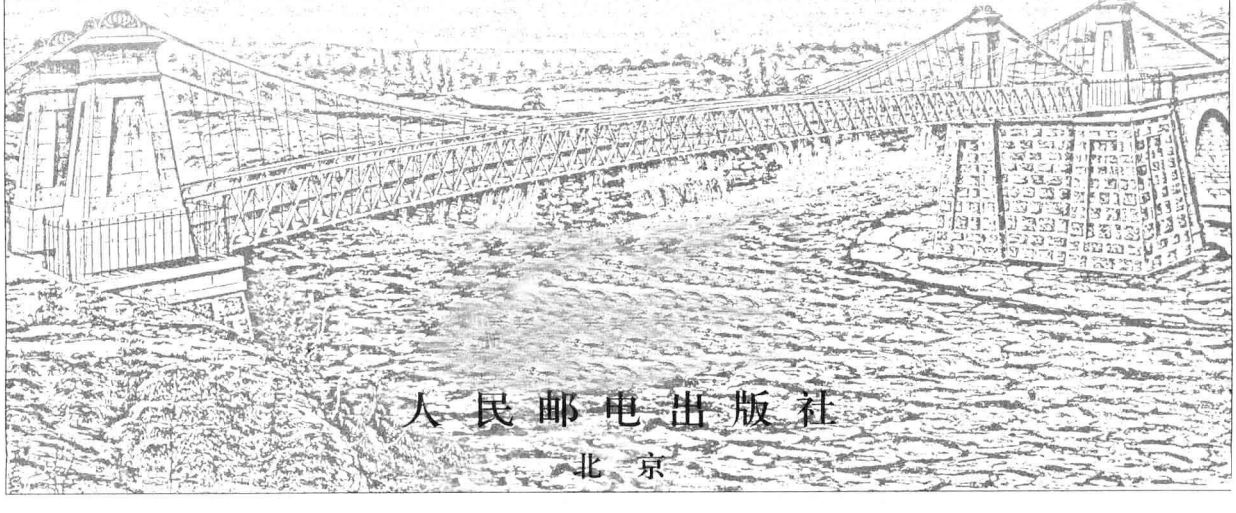
普通高等教育
软件工程 “十二五”规划教材

12th Five-Year Plan Textbooks
of Software Engineering

C语言工程实训教程

王卓 万立中 刘伯成 ◎ 编著

The Practice of C Programming



人民邮电出版社
北京

图书在版编目（CIP）数据

C语言工程实训教程 / 王卓，万立中，刘伯成编著
. -- 北京 : 人民邮电出版社，2013.12
普通高等教育软件工程“十二五”规划教材
ISBN 978-7-115-34236-2

Ⅰ. ①C… Ⅱ. ①王… ②万… ③刘… Ⅲ. ①C语言－程序设计－高等学校－教材 Ⅳ. ①TP312

中国版本图书馆CIP数据核字(2014)第004997号

内 容 提 要

本书是以C语言实际运用为导向，以解决实际问题为编程内容的实训教程。全书共分10章，第1章回顾C语言基础语法知识，第2章到第7章分模块介绍图书馆管理系统的设计与实现，第8章和第9章分别介绍两个拓展性的应用实训——内存分区管理算法和字母贪吃蛇游戏的设计与实现。第10章简要介绍了C语言编程规范相关的内容，以便学习者在实际代码的编写中参考学习。

本书适合作为高等院校、高职高专计算机或相关专业工程实训教学用书或参考用书，也可供在校教师以及相关工程技术人员参考使用。

◆ 编　　著　王　卓　万立中　刘伯成
责任编辑　刘　博
责任印制　彭志环　杨林杰
◆ 人民邮电出版社出版发行　　北京市丰台区成寿寺路11号
邮编　100164　　电子邮件　315@ptpress.com.cn
网址　http://www.ptpress.com.cn
大厂聚鑫印刷有限责任公司印刷
◆ 开本：787×1092　1/16
印张：11.75　　2013年12月第1版
字数：313千字　　2013年12月河北第1次印刷

定价：29.80元

读者服务热线：(010)81055256　印装质量热线：(010)81055316
反盗版热线：(010)81055315

前　言

C 语言既有高级语言的优势又有低级语言的特点，既适于编写系统软件又能方便地用来编写应用软件，是众多计算机语言初学者最早接触的计算机语言之一。C 语言入门容易，但使用灵活，要真正学好它，除了掌握必需的理论知识，如数据类型、流程控制、函数、数组、指针、文件以及结构体和共用体等，最重要的是要学会如何综合运用这些理论知识解决实际问题，毕竟这才是学习计算机语言的最终目的。

目前，C 语言已经成为高校计算机专业一门重要的基础课程，但不少高校只开设一个学期。由于 C 语言本身涉及的概念繁多，在有限的课堂授课时数中，只能顾及广度而无法兼顾深度。最终导致许多学习者在课程结束后，除了解一些理论知识外，仍然不明白 C 语言到底能干什么。

对于刚学完 C 语言基础的学习者而言，迫切需要一本从纵深的视角阐述理论应用，以 C 语言实际运用为导向，以解决实际问题为编程内容的实训教程。为此，一批长期在一线从事教学、具有多年项目开发和编程经验的教师，专门编写了本教程。教程中设计的编程项目是为了夯实 C 语言基础，以解决实际问题为核心，以培养 C 语言知识的综合运用和融会贯通能力为目标。学习者实训本教程的全部内容后，C 语言编程能力会有显著提高，对书中涉及的理论也会有更深入的理解，为今后进一步学习其他计算机语言打下坚实的基础。

本书以一个实际的图书馆管理系统的设计与实现为主线，涉及的知识点也是围绕该系统的详细实现过程而安排的，这样做的目的就是让理论和实践直接关联，让学习者直观地了解理论是如何被运用到实际问题的解决中。为了拓展学习者的视野，书中还安排了内存管理和游戏设计两个拓展性的应用实训。此外，本书还有意识地加入了与编程规范相关的内容，以便学习者在实际代码的编写中参考学习。

全书共分 10 章，以知识点的回顾为起点，以项目实训为贯穿全书的核心，以编程规范作为全书的结束篇。具体内容安排如下：

第 1 章　基础知识回顾
第 2 章　图书馆管理系统分析
第 3 章　读者管理模块的设计与实现
第 4 章　分类目录管理模块的设计与实现
第 5 章　图书编目管理模块的设计与实现
第 6 章　图书流通管理模块的设计与实现
第 7 章　账户管理模块的设计与实现
第 8 章　内存分区管理算法的设计与实现
第 9 章　字母贪吃蛇游戏的设计与实现
第 10 章　C 语言编程规范

本书第 2～8 章由王卓编写，第 9 章由万立中编写，第 1、10 章由刘伯成编写，全书由王卓统稿，参加编写的还有徐健锋、黄旭慧、胡然老师。

本书的编写参考了国内外相关资料，以及借鉴了互联网上众多程序的思想，在此谨向参考资料的作者表示感谢。由于编者学识有限，成书匆匆，书中内容不可能做到面面俱到，难免有疏漏、不足、错误和不当之处，恳请广大读者和同行批评指正，多提宝贵意见，使本书在下一版得以改进和完善，对此我们表示衷心的感谢。

编　者

2013 年 11 月

目　录

第 1 章 基础知识回顾

本书的所有程序都是基于 C 语言实现的，因此本章首先对 C 语言做一个简要的回顾，其中包括 C 语言的基本结构、数据类型、函数等。由于本书不是 C 语言教程，所以，对于 C 语言的系统学习，建议读者先参见其他基础教程。

1.1 概　　述

本节简述了 C 程序的基本结构、高级语言的编译和执行过程，介绍了 Visual C++6.0 的开发环境，以及基本使用方法。

1.1.1 C 程序的基本结构

C 程序由语句、函数、包含文件等部分组成。本小节将以一个简单的 C 程序实例来讲解 C 程序的基本结构问题。

例 1-1 编写一个比较两个整数大小的简单 C 程序。

```
#include <stdio.h>                    /*包含文件。*/
int max(int i , int j)
{
     /*注释：这是一个自定义函数，实现两个整数的大小比较功能。 */
    if(i>=j)
    return(i);                        /*返回较大的数。*/
    else
    return(j);
}
void main()                           /*主函数。*/
{
  int i =2;                           /*定义三个变量，并且赋值。*/
  int j =4;
  int t;
  t=max(2,4);                    /*将较大的数赋值给 t。*/
  printf("the max is %d\n",t); /*输出文本和结果。*/
}
```

由上面的例子可知，C 程序的源代码具有以下特点。

（1）程序一般用小写字母书写。

（2）大多数语句结尾必须要分号作为终止符，表示一个语句结束。同一个语句需要写在一

行上。

（3）每个程序必须有一个主函数，主函数用 main()声明，并且只能有一个主函数。

（4）每个程序中的自定义函数和主函数，需要用一对大括号括起来。

（5）程序需要使用#include< >语句来包含系统文件，这些系统文件完成系统函数的定义。包含自定义函数也可以用#include “”。

（6）一个较完整的程序大致包括下面这些内容。

① 包含文件：一组#include<*.h>语句。

② 用户自定义函数：用户已编写的完成特定功能的模块。

③ 主函数：程序自动执行的程序体。

④ 变量定义：定义变量以存储程序中的数据。

⑤ 数据运算：程序通过运算完成各种逻辑功能。这些运算是由各种语句和函数实现的。

⑥ 注释：注释写在/* */符号之间，不是程序的必需部分，但是可以增强程序的可读性；或者采用//符号，//是行注释。

1.1.2 高级语言的编译和执行

高级语言的编译和执行如图 1-1 所示。高级语言按照执行方式可以分为编译型和解释型两种。

编译型是专门的编译器，针对指定的平台一次性将某种高级语言代码翻译成可被平台硬件执行的机器码（包括机器指令和操作数），并包装成该平台可以识别的可执行程序的格式。这个过程就叫做 compile。编译生成的可执行程序可以脱离开发环境，在特定的平台上独立运行。有些程序编译结束后，还可能需要对其他编译好的目标代码进行连接，即组装成两个以上的目标代码模块生成的最终可执行程序，通过这种方式实现低层次代码的复用。因为编译型语言是一次性编译成机器码，脱离开发环境独立运行，所以运行效率较高，但是由于编译成的是特定平台上机器码，所以可移植性差。编译型语言的典型代表有 C、C++、FORTRAN、Pascal 等。

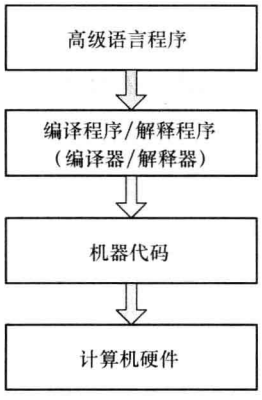


图 1-1 高级语言的编译和执行

解释型语言是专门的解释器对源程序逐行解释成特定平台的机器码并执行的语言。解释型语言通常不会进行整体性的编译和链接处理，解释语言相当于把编译型语言的编译和解释过程混合到了一起同时完成。于是，每次执行解释型语言的程序都要进行一次编译，因此解释型语言的程序运行效率通常较低，而且不能脱离解释器独立运行。但解释型语言跨平台容易，只需要提供特定的平台解释器即可。解释型语言可以方便地进行程序的移植，但是以牺牲程序的执行效率为代价的。解释型语言的典型代表有 Ruby、Python 等。

1.1.3 编译和执行 C 程序

编译和执行 C 程序如图 1-2 所示。简单说 C 语言从代码编译到执行要经历以下过程。

（1）编辑 C 源代码。

（2）编译---->形成目标代码，目标代码是在目标机器上运行的代码。

（3）连接---->将目标代码与 C 函数库相连接，并将源程序所用的库代码与目标代码合并，并形成最终可执行的二进制机器代码（程序）。

（4）执行----->在特定的机器环境下运行 C 程序。

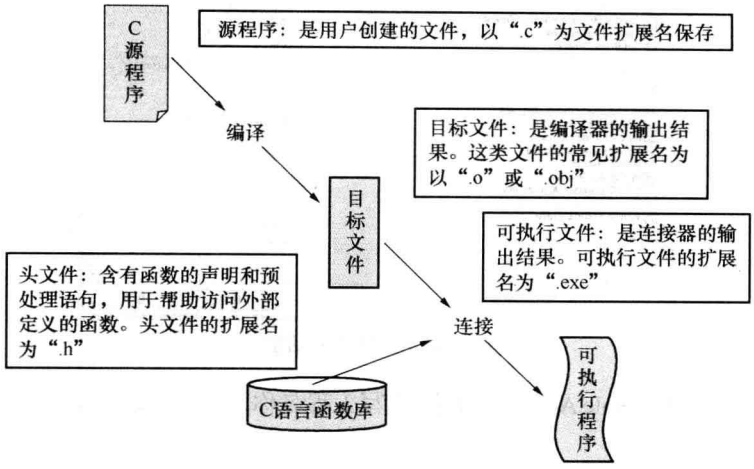


图 1-2　编译和执行 C 程序

1.1.4　Visual C++ 6.0

为了方便程序开发，人们开发了一类称作 IDE 的软件。Visual C++6.0 是目前国内比较流行的一种 C++语言源程序的编译系统，使用该系统也可以编辑和运行 C 语言的源程序。启动 Visual C++6.0，如图 1-3 所示就是 Visual C++6.0 的界面。

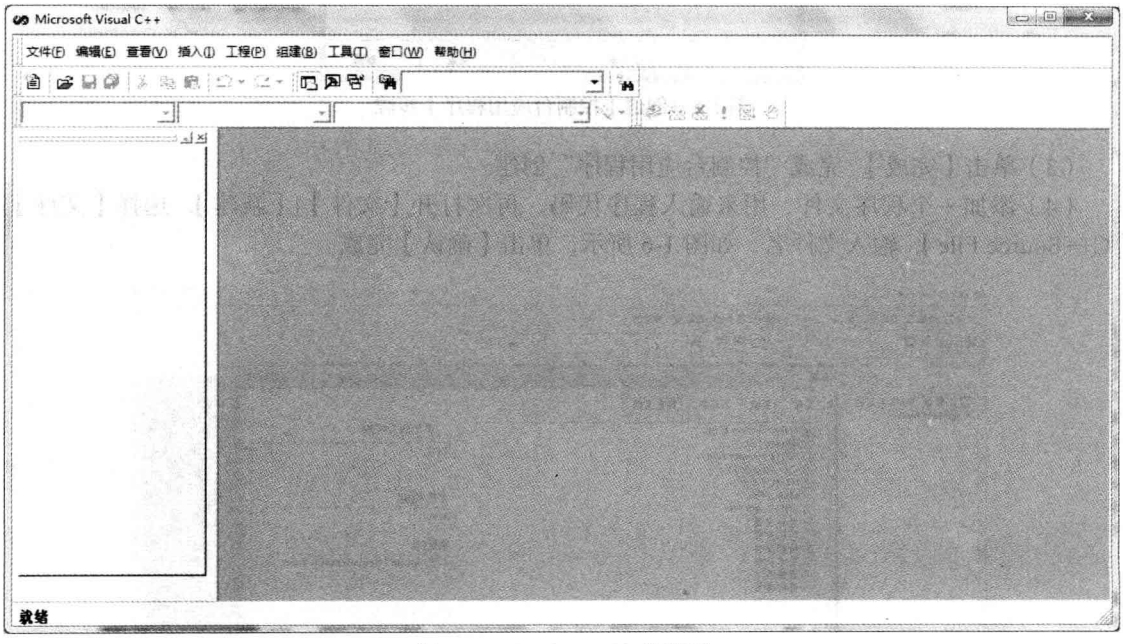


图 1-3　Visual C++6.0 的界面

1．使用 Microsoft Visual C++可以创建控制台应用程序

（1）打开【文件】|【新建】，出现如图 1-4 所示的对话框。

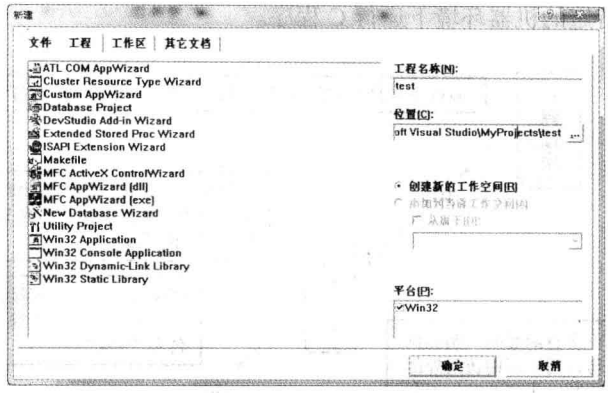


图 1-4 【新建】对话框

（2）选择【Win 32 Console Application】，填写工程名称，单击【确定】进入下一步，如图 1-5 所示。

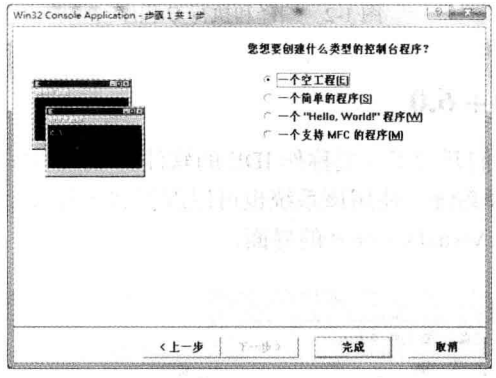


图 1-5 创建【控制台应用程序】步骤

（3）单击【完成】，完成“控制台应用程序”创建。

（4）添加一个程序文件，用来输入程序代码。再次打开【文件】|【新建】，选择【文件】|【C++Source File】，输入文件名，如图 1-6 所示，单击【确认】完成。

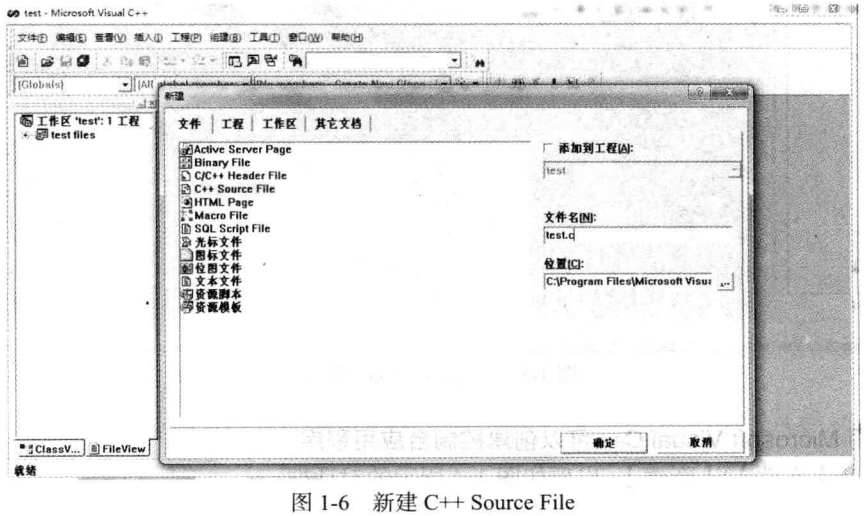


图 1-6 新建 C++ Source File

（5）在右边的编辑框编写程序代码，如图 1-7 所示。

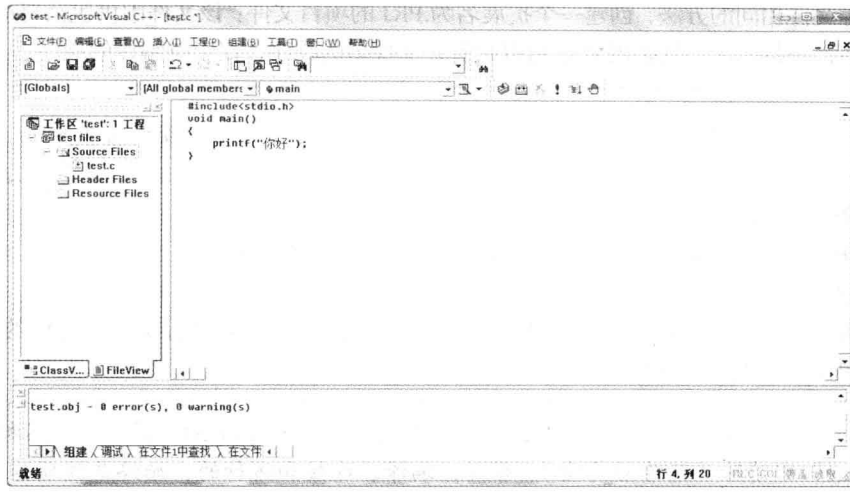


图 1-7　编辑源程序并编译

（6）输入程序代码，单击工具栏上的【compile】按钮，编译源程序，生成.obj 的目标文件，如果没有错误，则按【build】按钮，连接并生成.exe 的可执行文件，如图 1-8 所示。

```
test.c
Linking...

test.exe - 0 error(s), 0 warning(s)
```

图 1-8　生成 exe 文件

（7）按工具栏上的红色“!”，运行程序，如图 1-9 所示。执行的结果将在一个黑色的控制台窗口显示。

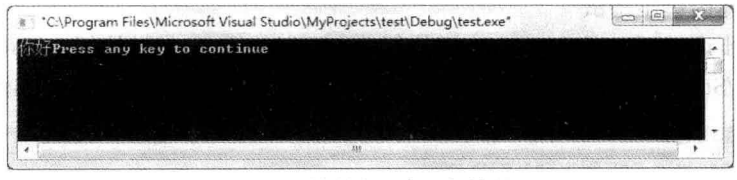


图 1-9　控制台程序运行结果

2. 用 Visual C++创建的 C 程序被存储为一个独立工程

也可以在开始时不建立工程，直接新建一个【C++ Source File】，待编写完程序代码后，编译时会提示创建工程，如图 1-10 所示。其他编译执行过程跟上面相同。

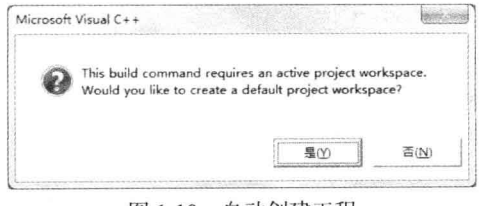


图 1-10　自动创建工程

3. 多个源程序文件的编译和连接

（1）一般过程

编辑各源文件→创建 Project（项目）文件→设置项目名称→编译、连接、运行、查看结果。

（2）创建 Project（项目）文件

用编辑源文件相同的方法，创建一个扩展名为.PRJ 的项目文件：该文件中仅包括将被编译、连接的各源文件名，一行一个，其扩展名.C 可以缺省；文件名的顺序，仅影响编译的顺序，与运行无关。

如果有某个（些）源文件不在当前目录下，则应在文件名前冠以路径。

（3）设置项目名称

打开菜单，选取 Project / Project name，输入项目文件名即可。

（4）编译、连接和运行，查看结果

与单个源文件相同，编译产生的目标文件，以及连接产生的可执行文件，它们的主文件名，均与项目文件的主文件名相同。

当前项目文件调试完毕后，应选取 Project / Clear project，将其项目名称从“Project name”中清除（清除后为空）。否则，编译、连接和运行的，始终是该项目文件！

（5）关于错误跟踪

缺省时，仅跟踪当前一个源程序文件。如果希望自动跟踪项目中的所有源文件，则应将 Options / Environment / Message Tracking 开关置为“All files”：连续按回车键，直至“All files”出现为止。此时，滚动消息窗口中的错误信息时，系统会自动加载相应的源文件到编辑窗口中。也可关闭跟踪（将“Message Tracking”置为“Off”）。此时，只要定位于感兴趣的错误信息上，然后回车，系统也会自动将相应源文件加载到编辑窗口中。

4. C 语言单步调试与跟踪

（1）单步调试的类别

单步调试分为函数内部单步调试和函数调用单步调试两种。

（2）主函数体内单步调试

主函数内单步调试可以通过按 F10 键执行，也可以使用 F11 键执行；还可以通过菜单栏中 Build/Start Debug/Step Into 执行。打开一个 Visual C++工程，启动单步调试，如图 1-11 所示。

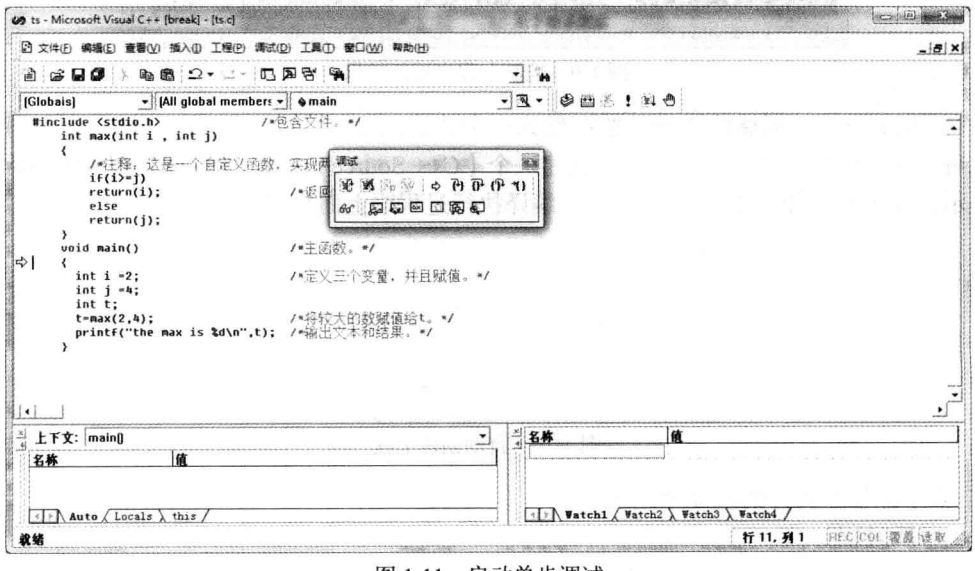


图 1-11　启动单步调试

（3）单步调试

继续按 F10 键，程序将顺序执行下去，当程序执行到代码第 15 行时，将调用被调函数。

（4）被调函数单步调试

此时，程序流程进入函数 int max（int i，int j）的调用。按 F11 键以进入函数 max 的函数体内，如图 1-12 所示。

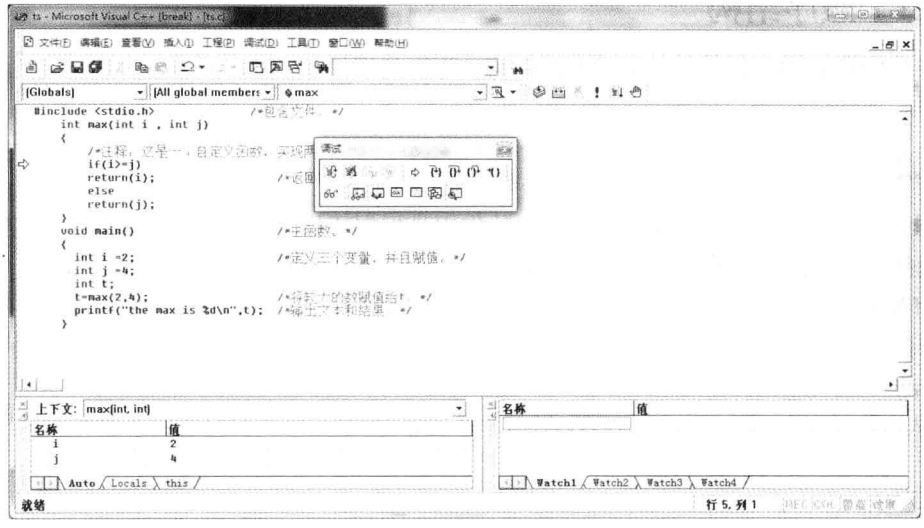


图 1-12　被调函数单步调试

（5）单步调试结束

可以使用 F11 或 F10 键继续进行程序的跟踪，当程序执行到第 17 行时，表示程序运行已结束，此时可以看到命令窗口输出的结果，如图 1-13 所示。

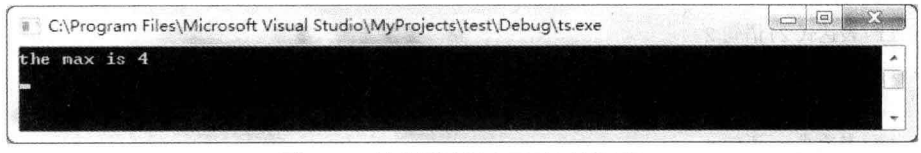


图 1-13　单步调试结束显示运行结果

5. 断点调试与跟踪

断点是程序调试时最常用的手段之一。在 C 语言中，断点可以看成是一个输入信号，在程序调试阶段，它通知调试器，在某个特定点上暂时中断程序执行，通常称为程序的挂起。当程序在某个断点处挂起时，称这种状态为中断模式。中断模式下程序并不会终止，也不会结束程序的执行，仅仅是将程序暂时停住，以备后续调试。当需要继续执行程序时，可以按照一定的操作继续执行程序。

（1）设置位置断点

将光标移动到要设置断点的行，选择要设置断点的位置。通过快捷键 F9 可以设置断点，设置断点之后，在断点行左侧将出现红色圆点，表示断点设置成功。

（2）设置条件断点

条件断点是指在满足某些设定的条件时断点才起作用。可以通过快捷键 Alt+F9 打开 breakpoints 对话框进行条件断点设置。

1.2 程序设计结构

C 语言中常用的程序设计结构有顺序结构、选择结构和循环结构。

1.2.1 顺序结构

顺序结构是 C 语言最简单、最基本的结构，程序从编译预处理到结束全部顺序进行，没有跳转和循环。

1.2.2 选择结构

选择结构在大多数程序中都会存在。它的作用是根据所指定的条件是否满足，决定从给定的操作中选择其一。选择结构主要是通过 if 语句和 switch 语句实现的。

1. if 语句

C 语言提供了以下几种 if 语句。

第一种：

```
if(表达式)
     语句
```

第二种：

```
if(表达式)
     语句 1
else
     语句 2
```

第三种：

```
if(表达式 1)语句 1
else if(表达式 2)语句 2
else if(表达式 3)语句 3
……
else if(表达式 m)语句 m
else 语句 n
```

此外，if 语句还可以嵌套使用，即一个 if...else 语句中又包含一个或者多个 if...else 语句。应当注意的是 if 与 else 的配对关系，else 总是和它上面最近的未配对的 if 配对。

例 1-2 判断考试成绩级别。

```
#include <stdio.h>
void main()
{
     float grade;
     printf("\n 请输入期末考试成绩： ");
     scanf("%f", &grade);
     if(grade>=90)
          printf("\n 优");
     else if ((grade>=80) && (grade<90))
          printf("\n 良");
     else if ((grade>=60) && (grade<80))
```

```
        printf("\n 中");
    else
        printf("\n 差");
    printf("\n");
        }
```

2. switch 语句

switch-case 语句是多路判断语句。

switch 语句计算条件表达式并对照多个常数值进行检查，语法如下。

```
switch (表达式)
{
        case 常量 1:        语句;     break;
        case 常量 2:        语句;     break;
         default:      语句;
}
```

例 1-3　switch 语句判断输入字符是否元音字母。

```
char in_char;
printf("\n 请输入一个小写字母: ");
scanf("%c", &in_char);
switch(in_char)
{
      case 'a': printf("\n 您输入的是元音字母 a\n");
                break;
      case 'e': printf("\n 您输入的是元音字母 e\n");
                break;
      case 'i': printf("\n 您输入的是元音字母 i\n");
                break;
      case 'o': printf("\n 您输入的是元音字母 o\n");
                break;
      case 'u': printf("\n 您输入的是元音字母 u\n");
                break;
      default:  printf("\n 您输入的不是元音字母 \n");
}
```

比较多重 if 和 switch 结构：

（1）多重 if 结构和 switch 结构都可以用来实现多路分支；

（2）多重 if 结构用来实现两路、三路分支比较方便，而 switch 结构实现三路以上分支比较方便；

（3）在使用 switch 结构时，应注意分支条件要求是整型表达式（包括字符型），而且 case 语句后面必须是常量表达式；

（4）有些问题只能使用多重 if 结构来实现，例如要判断一个值是否处在某个区间的情况。

1.2.3　循环结构

在 C 语言中，对于某些要根据条件重复执行的语句，我们可以使用循环语句来实现。循环结构的特点是，在给定条件成立时，重复执行某程序段，直到条件不成立为止。

C 语言中的循环语句主要有 4 种：while 语句、do…while 语句、for 语句和 goto 语句。