1 C 语言概述

C 语言是一种通用的计算机编程语言，被广泛应用于操作系统和应用软件的开发。C 语言的设计目标是提供一种能对硬件进行编程、编译方法简单、产生机器代码少的编程语言。

1.1 C 语言简介

C 语言比较接近计算机底层，能够直接操作硬件。学习 C 语言，对于理解计算机有很大的帮助。C 语言程序不但执行效率高，而且应用广泛，可以用来开发桌面软件、硬件驱动、操作系统、单片机等，从微波炉到手机，都有 C 语言的影子。C 语言不但是一种优秀的语言，也是学习其他语言的阶梯。

1.1.1 C 语言的发展史

C 语言之所以命名为 C，是因为 C 语言源自 Ken Thompson 发明的 B 语言，而 B 语言则源自 BCPL 语言。1967 年，剑桥大学的 Martin Richards 对 CPL 语言进行了简化，于是产生了 BCPL（Basic Combined Programming Language）语言。1970 年，美国贝尔实验室的 Ken Thompson 以 BCPL 语言为基础，设计出很简单并且很接近硬件的 B 语言（取 BCPL 的首字母），并且用 B 语言写了第一个 UNIX 操作系统。1972 年，美国贝尔实验室的 Dennis Matin Ritchie 在 B 语言的基础上最终设计出了一种新的语言，他取了 BCPL 的第二个字母作为这种语言的名字，这就是 C 语言。随着 UNIX 的发展，C 语言自身也在不断地完善。直到今天，各种版本的 UNIX 内核和周边工具仍然使用 C 语言作为最主要的开发语言，其中还有不少继承自 Thompson 和 Ritchie 之手的代码。

机器语言和汇编语言都不具有移植性，而 C 语言程序则可以使用在任意架构的处理器上，只要那种架构的处理器具有对应的 C 语言编译器和库，然后将 C 源代码编译、连接成目标二进制文件之后即可运行。

1983 年由美国国家标准局（American National Standards Institute，简称 ANSI）开始制定 C 语言标准，于 1989 年 12 月完成，并在 1990 年春天发布，称之为 ANSI C，有时也被称为 C89

或 C90。

1999 年 1 月，国际标准化组织和国际电工委员会发布了 C 语言的新标准 C99，这是 C 语言的第二个官方标准。2011 年 12 月 8 日，国际标准化组织和国际电工委员会再次发布了 C 语言的新标准 ISO/IEC 9899:2011-Information technology-programming language-C，简称 C11 标准，这是 C 语言的第三个官方标准，也是 C 语言的最新标准。新的标准提高了对 C++的兼容性，并增加了一些新的特性。

1.1.2 C 语言的特点

C 语言之所以能存在和发展，并具有持续的生命力，总是有其优于其他语言的特点。C 语言的主要特点如下：

1. 简洁紧凑、灵活方便

C 语言一共只有 32 个关键字，9 种控制语句，程序书写形式自由，区分大小写。它把高级语言的基本结构和语句与低级语言的实用性结合起来。

2. 运算符丰富

C 语言的运算符包含的范围很广泛，共有 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。

3. 数据类型丰富

C 语言的数据类型有：整型、实型、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的运算，并引入了指针概念，使程序效率更高。

4. 表达方式灵活实用

C 语言提供多种运算符和表达式的方法，对问题的表达可通过多种途径获得，其程序设计更主动、灵活，语法限制也不太严格，程序设计自由度大，如对整型数据与字符型数据可以通用等。

5. 允许直接访问物理地址，对硬件进行操作

由于 C 语言允许直接访问物理地址，可以直接对硬件进行操作，因此它既具有高级语言的功能，又具有低级语言的许多功能。C 语言能直接访问硬件的物理地址，能进行位（bit）运算，兼有高级语言和低级语言的许多优点。它既可用来编写系统软件，又可用来开发应用软件，已成为一种通用程序设计语言。

6. 生成目标代码质量高，程序执行效率高

C 语言描述问题比汇编语言迅速，工作量小、可读性好，易于调试、修改和移植，而代码质量与汇编语言相当。C 语言代码一般只比汇编程序生成的目标代码效率低 10%～20%。

7. 可移植性好

C 语言在不同机器上的 C 编译程序，86%的代码是公共的，所以 C 语言的编译程序便于移植。在一个环境上用 C 语言编写的程序，不改动或稍加改动，就可移植到另一个完全不同的环境中运行。

C 语言具有强大的图形功能，支持多种显示器和驱动器，且计算功能、逻辑判断功能强大。

1.2 C 语言程序结构

我们通过一个简单的例子来说明 C 语言程序的结构。

【例 1-1】

```
#include <stdio.h>
int main()                              /*主函数：求两数之和*/
{
    int a,b,sum;                        /*定义三个变量 a，b，sum*/
    a=2;b=4;                            /*给 a、b 赋值*/
    sum=a+b;                            //求和的结果赋给 sum
    printf("%d +%d =%d",a,b,sum);       //输出 sum 的值
    return 0;
}
```

程序运行结果：

```
2+4=6
```

本程序的作用是求两个整数 a 和 b 之和 sum。其中 main 表示“主函数”，每一个 C 程序都必须有一个 main 函数。C 标准中规定，main 函数的返回值类型应该定义为 int 整型，main 函数的返回值用于说明程序的退出状态，返回 0，则代表程序正常退出；返回其他数字的含义则由系统决定，通常，返回非零代表程序异常退出。函数体由大括号{ }括起来。

程序中/*……*/和//后面的内容表示注释部分，注释只是给人看的，可以加在程序中的任何位置。第三行是变量定义部分，说明 a 和 b 为整型变量。第四行是两个赋值语句，使 a 和 b 的值分别为 2 和 4。第五行使 sum 的值为 a+b。第六行的 printf 是 C 语言中的输出函数（详见第 3 章），双引号内的字符串原样输出，“%d”是输入输出的格式字符串，表示“十进制整数类型”，在执行输出时，此位置上代以一个十进制整数值。printf 函数中括号内最右端 a,b,sum 是要输出的变量，它们的值分别为 2,4,6。

通过以上例子可以看出 C 语言程序的结构：

（1）C 程序是由函数构成的，每一个函数完成相对独立的功能。一个程序可以由多个函数组成，但至少包含一个函数，即 main 函数，它称为主函数，程序总是从 main 函数开始执行，并在 main 函数中结束的。

（2）C 语言程序通常使用英文小写字母书写，只有符号常量或其他特殊用途的符号才使用大写。应该注意的是，C 语言对大小写是区分的，它们代表着不同的字符。

（3）C 语言程序是由一条条语句组成的，每条语句都具有规定的语法格式和特定的功能。上面的程序中，printf()是输出变量数值的函数调用语句；a=2 是赋值语句。

（4）C 语句以分号（;）结束，分号是语句不可缺少的组成部分。

（5）C 语言程序中，一条语句可以占用多行，一行也可以有多条语句。

（6）C 语言程序中使用大括号“{”和“}”来表示程序的结构层次范围。一个完整的程序模块要用一对大括号括起来，以表示该程序模块的范围。应该注意的是，左大括号“{”和右大括号“}”应该成对使用。

（7）为了增加程序的可读性，可以使用适量的空格和空行。但是，变量名、函数名和 C 语言保留字中间不能加入空格。除此之外的空格和空行可以任意设置，C 语言编译系统是不会理会这些空格和空行的。

（8）可以用/*…*/或//对 C 程序中的任何部分做注释。一个好的、有使用价值的源程序都应当加上必要的注释，以增加程序的可读性。

1.3 C 语言程序的执行

C 语言程序的调试步骤包括编辑、编译、连接、运行。本节介绍使用 Visual C++调试 C 语言程序的过程。

1.3.1 C 语言程序调试的基本步骤

1. 编辑

可以用任何一种编辑软件将 C 语言程序输入计算机，并将 C 语言源程序文件（*.c）以纯文本文件形式保存在计算机的磁盘上（不能设置字体、字号等）。

2. 编译

编译过程将编辑好的源程序文件“*.c”，翻译成二进制目标代码文件“*.obj”。编译程序对源程序逐句检查语法错误，发现错误后，不仅会显示错误的位置（行号），还会告知错误的类型信息。我们需要再次回到编辑环境修改源程序的错误，然后，再进行编译，直至排除所有语法和语义错误。

3. 连接

程序编译后产生的目标文件是可重定位的程序模块，不能直接运行。连接将编译生成的各个目标程序模块和系统或第三方提供的库函数“*.lib”连接在一起，生成可以脱离开发环境、直接在操作系统下运行的可执行文件“*.exe”。

4. 运行

如果经过测试，运行可执行文件达到预期设计目的，这个 C 语言程序的开发工作便到此完成了。如果运行出错，这说明程序处理的逻辑存在问题，需要再次回到编辑环境针对程序出现的逻辑错误进一步检查、修改源程序，重复编辑→编译→连接→运行的过程，直到取得预期结果为止。

1.3.2 使用 Visual C++调试 C 语言程序

C 语言是一种历史很长的编程语言，其编译器和开发工具也多种多样，常见的开发工具有下面几个：

Turbo C 2.0——在 DOS 环境下编译运行，小巧灵活，但是不能使用鼠标，不支持复制、粘贴等功能，需要记住几个常用的快捷键。

Win-TC——在 Turbo C 的基础上加了界面，能够使用鼠标，界面简洁美观，适合编一些自己用的小程序。

Dev-C++——是一个 Windows 环境下的 C/C++开发工具，它是一款自由软件，遵守 GPL 许可协议分发源代码。

Visual C++6.0——微软的产品，报错比较准确，是集编译器、链接器、运行、调试等功能于一体的、功能强大的可视化软件开发工具。本书中 C 语言程序的编译运行采用 Visual C++ 6.0。

下面介绍如何利用 Visual C++ 6.0 的开发环境，编辑、编译和运行 C 语言程序。

1. 启动 Visual C++ 6.0 开发环境

从“开始”菜单中选择“程序”→Microsoft Visual Studio 6.0→Microsoft Visual C++ 6.0，

显示 Visual C++ 6.0 开发环境主窗口，如图 1-1 所示。

图 1-1　Visual C++ 6.0 开发环境

2. 建立 C 语言源程序文件

在“文件”菜单中单击“新建”命令，在显示的“新建”对话框中选择“文件”选项卡，如图 1-2 所示。在“文件”选项卡中选择 C++ Source File，在该对话框右边的两个文本框中分别是新建文件的名字与保存位置。注意：文件扩展名要切换到英文输入法后输入“.c”（一定要有.c 作为文件扩展名，否则新建的文件将默认为.cpp 格式，也就是 C++语言的文件格式）。例如：“第一个 C 程序.c”。

图 1-2　建立 C 语言源程序文件

3. 编辑 C 语言程序的内容

在新建文件的源程序编辑窗口（即右边空白处），输入程序。也可以利用“文件”菜单中的“打开”命令打开现有的 C 语言程序。该步骤为编辑过程，编辑完程序后，单击“保存”按钮，如图 1-3 所示。

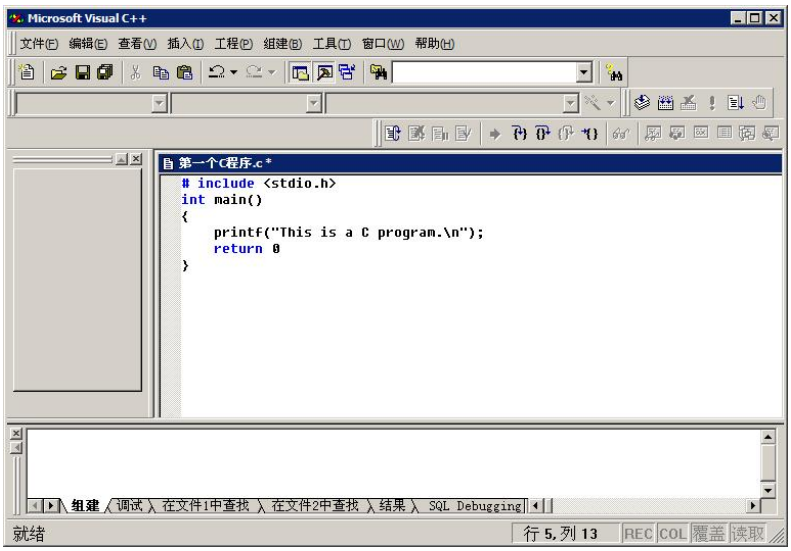

图 1-3　编辑 C 语言程序的内容

4. 建立并运行可执行程序

（1）在“组建”菜单中单击“编译”命令，建立目标文件。

（2）在“组建”菜单中单击“组建”命令，建立可执行程序。

如果源程序编写正确，此时可以成功生成可执行程序；如果程序中有语法错误，则在屏幕下方的信息窗口中会显示错误信息，如图 1-4 所示。

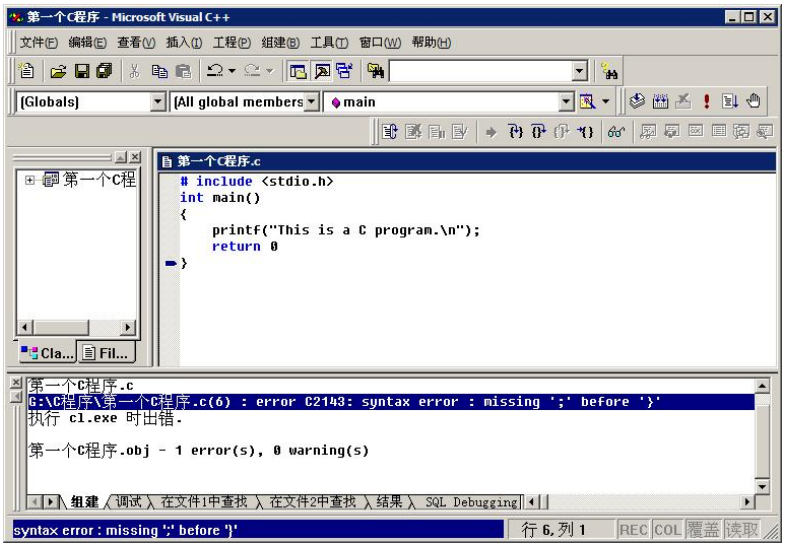

图 1-4　编译 C 语言源程序

根据错误信息提示对源程序文件进行修改之后，再重新对源文件进行编译、构建，即可建立可执行文件。

（3）在“组建”菜单中单击“执行”命令，运行程序，就会弹出一个窗口，显示程序的运行结果，如图 1-5 所示。

图 1-5　C 语言程序的运行结果

运行结果窗口中的“Press any key to continue”是 Visual C++自己输出的，旨在提醒程序已经运行完成，可按下键盘上的任意键返回 Visual C++的程序编写界面。

5. 关闭程序

如果想在不关闭软件的情况下，编辑运行另外一个程序，需要关闭上一次运行的工作空间，方法是选择“文件”菜单中的“关闭工作空间”命令，这样才能再编辑、编译一个新的程序，否则即使新建了一个新的 C 语言源程序文件，新编辑了一个程序，仍然会编译上一次的那个程序，所以一定要关闭工作空间。

另外，创建 C 程序文件时，可以更改程序的存放位置。写好程序以后，也可以依次按 Ctrl+F7、F7、Ctrl+F5 三组组合键编译并运行程序。

1.4　VC++运行 C 程序说明

Visual C++以拥有语法着色、Intellisense（自动完成功能）以及高级排错功能而著称。比如，它允许用户进行远程调试、单步执行等，还允许在调试期间重新编译被修改的代码，而不必重新启动正在调试的程序。Visual C++编译及构建系统以预编译头文件、最小重建功能及累加连接著称，这些特征明显缩短程序编辑、编译及连接花费的时间。

1.4.1　编辑程序

在编辑窗口打开文件、浏览文件、输入、修改、复制、剪切、粘贴、查找、替换、撤销等操作，可以通过菜单完成，也可以通过工具栏按钮完成，这些与 Word 之类的 Windows 编辑器用法完全相同。

使用 Visual C++编辑 C 语言源程序文件时，在其中输入的任何内容（如关键字、用户标识符及各种运算符等），Visual C++系统都会按照 C 语言源程序的格式进行编排、组织。

在编辑过程中，当输入一个 C 语言的关键字时，Visual C++系统自动将其设定为蓝色字体

以示区别。如果输入了一个块结构语句（如 for(i=0;i<10;i++)、while(n<5)），按回车键后，Visual C++系统会把光标定位在该块结构语句起始位置开始的下一行的第五个字符位置上，以表示下面输入的内容是属于该块结构语句的，以体现 C 语言源程序的缩进式书写格式。此时，如果输入一个左花括号“{”并回车，Visual C++系统将把该花括号左移到与上一行块结构语句起始位置对齐的位置上；接着再按下回车键，Visual C++系统会自动采用缩进格式，将当前光标位置定位在此花括号的下一行的第五个字符位置上。如果上一行语句与下一行语句同属于一个程序段（比如同一个复合语句中的语句），Visual C++系统会自动将这两个程序的起始位置对齐排列。更详细的内容请读者自行上机实习，并认真体会其中的输入技巧。

1.4.2 编译、运行程序

程序编辑完成后，可以使用“编译微型条”中的快捷按钮对程序进行编译、连接、运行，如图 1-6 所示。

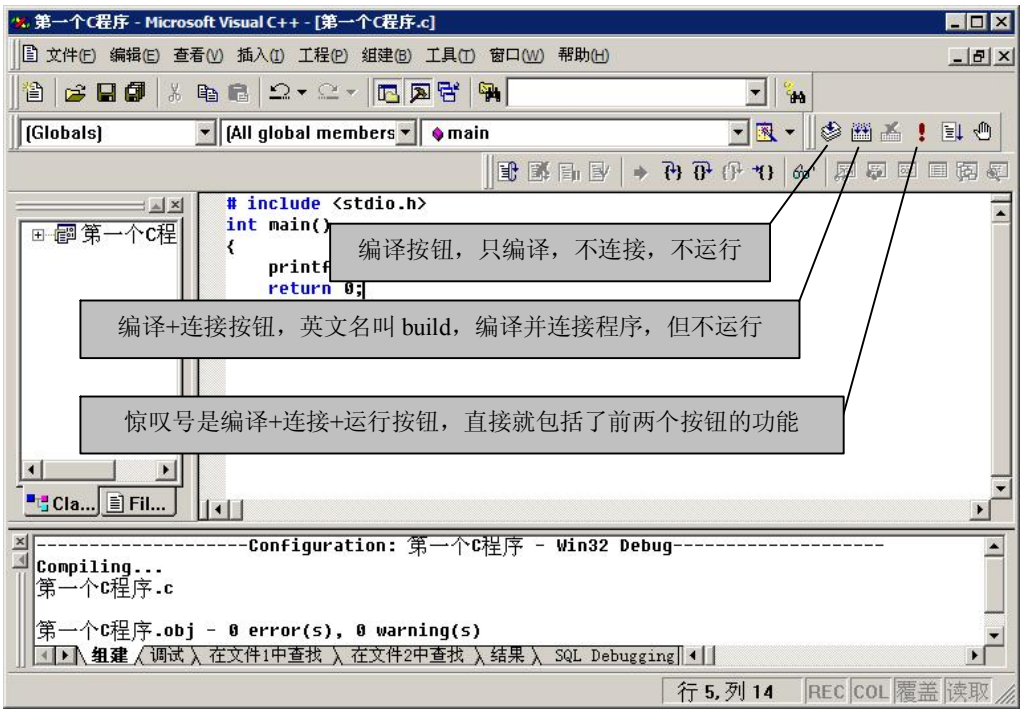

图 1-6 编译微型条

程序在编译、连接时，Visual C++会在最下端的输出窗口动态地输出“编译、连接”过程中的状态报告，如图 1-7 所示。

Visual C++输出的编译信息中，最后的“0 error(s)，0 warning(s)”表示编写的 C 程序经过编译连接后生成可执行程序的过程中没有错误，也没有警告。如果有错误的话，错误会导致程序无法编译通过，程序将无法运行，必须修改错误之后再重新编译程序。而警告是提示程序中有些代码的编写不是非常恰当，虽不会影响程序的编译，但在少数情况下会影响程序运行。比如声明了一个变量，程序中并没有使用，则系统会给出警告。

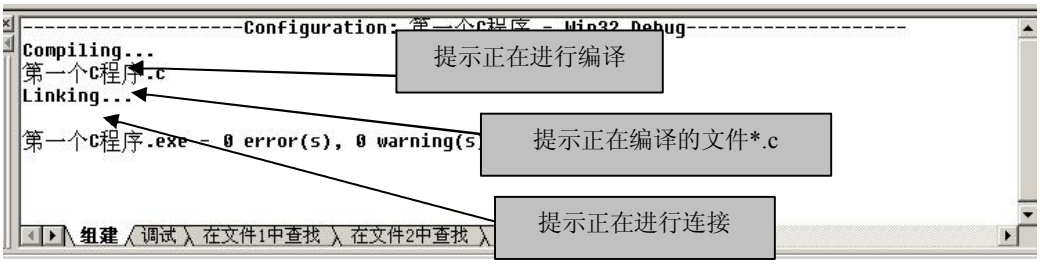

图 1-7　编译连接过程中的状态报告

如果在操作过程中，Visual C++界面上有些工具栏、窗口（如工作区间、信息输出窗口等）无法显示，可以通过“查看”菜单中的相关命令，或者在菜单栏的空白地方单击鼠标右键从快捷菜单中使其显示，如图 1-8 所示。

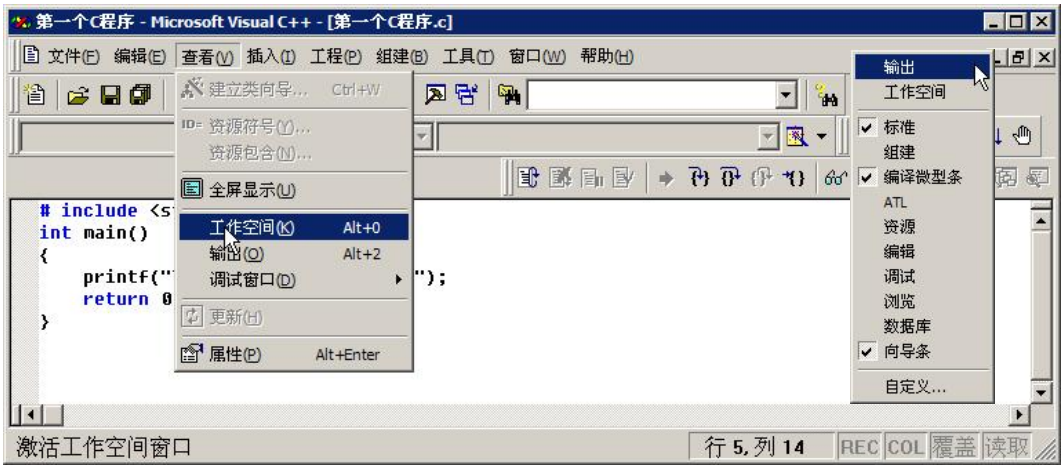

图 1-8　显示 Visual C++界面元素

1.4.3　调试程序

1. 错误类型

初学 C 语言程序设计，往往一看到自己编的程序出现错误就不知所措了，而程序能够顺利运行，并不意味着大功告成，程序中可能还存在某些隐患。要想不犯或少犯错误，就需要了解 C 语言程序设计的错误类型和纠正方法。C 语言程序设计的错误可分为语法错误、连接错误、逻辑错误和运行错误。

语法错误：在编写程序时违反了 C 语言的语法规定。语法不正确、关键字拼错、标点漏写、数据运算类型不匹配、括号不配对等都属于语法错误。进入程序编译阶段，编译系统会给出出错行和相应“错误信息”。可以双击错误提示行，将光标快速定位到出错代码所在的出错行上。根据错误提示修改源程序，排除错误。

连接错误：如果使用了错误的函数调用，比如书写了错误的函数名或不存在的函数名，编译系统在对其进行连接时便会发现这一错误。

逻辑错误：虽然程序不存在上述两种错误，但程序运行结果就是与预期效果不符。逻辑错误往往是因为程序采用的算法有问题，或编写的程序逻辑与算法不完全吻合。逻辑错误比语法

错误更难排除，需要对程序逐步调试，检测循环、分支调用是否正确，变量值是否按照预期产生变化等。

运行错误：程序不存在上述错误，但运行结果时对时错。运行错误往往是由于程序的容错性不高，可能在设计时仅考虑了一部分数据的情况，对于其他数据就不能适用了。例如打开文件时没有检测打开是否成功就开始对文件进行读写，结果程序运行时，如果文件能够顺利打开，程序运行正确，反之则程序运行出错。要避免这种类型的错误，需要对程序反复测试，完备算法，使程序能够适应各种情况的数据。

为了方便用户排除程序中的逻辑错误，Visual C++提供了强大的调试功能。每当用户创建一个新的工程项目时，默认状态就是 Debug（调试）版本。调试版本会执行编译命令_D_DEBUG，将头文件的调试语句 ifdef 分支代码添加到可执行文件中；同时加入的调试信息可以让用户观察变量，单步执行程序。由于调试版本包含了大量信息，所以生成的 Debug 版本可执行文件容量会远远大于 Release（发行）版本。

2. 设置断点

Visual C++可以在程序中设置断点，跟踪程序实际执行流程。设置断点后，可以按 F5 功能键启动 Debug 模式，程序会在断点处停止。我们可以接着单步执行程序，观察各变量的值如何变化，确认程序是否按照设想的方式运行。设置断点的方法是：将光标停在要被暂停的那一行，选择“编译微型条”按钮“Insert/Remove Breakpoint (F9)”添加断点，如图 1-9 所示，断点所在代码行的最左边出现了一个深红色的实心圆点，这表示断点设置成功。

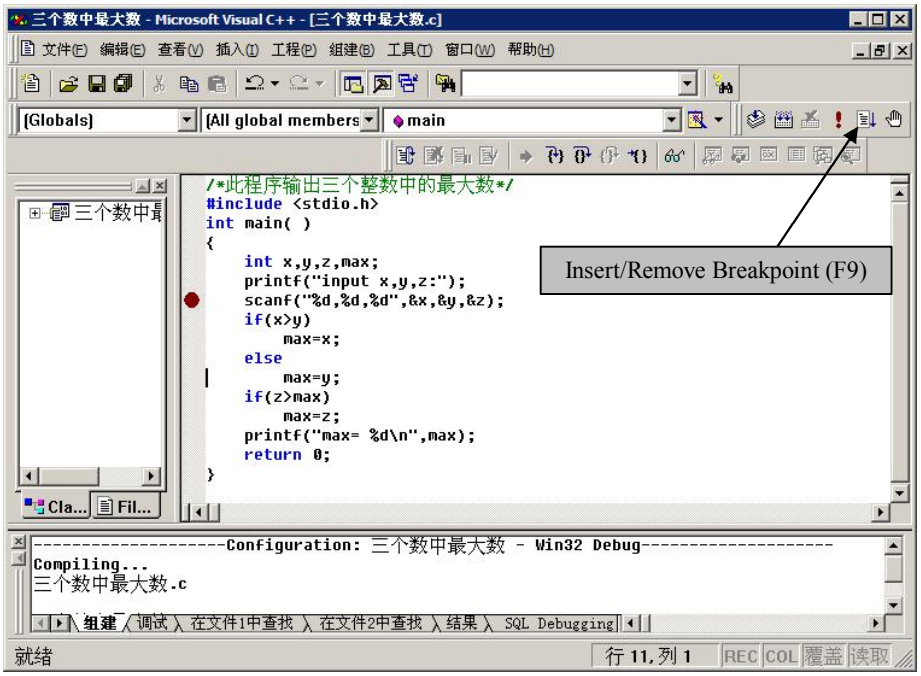

图 1-9　设置断点

如果该行已经设置了断点，那么再次按 F9 功能键会清除该断点。

3. 调试命令

在 Visual C++“组建”菜单下的“开始调试”中单击 Go（F5）命令进入调试状态，“组建”菜单自动变成“调试”菜单，提供以下专用的调试命令：

- Go(F5)：从当前语句开始运行程序，直到程序结束或断点处。
- Step Into(F11)：单步执行下条语句，并跟踪遇到的函数。
- Step Over(F10)：单步执行（跳过所调用的函数）。
- Run to Cursor(Ctrl+F10)：运行程序到光标所在的代码行。
- Step Out(Shift+F11)：执行函数调用外的语句，并终止在函数调用语句处。
- Stop Debugging(Shift+F5)：停止调试，返回正常的编辑状态。

必须在运行程序时用 Go 命令才能启动调试模式。在调试模式下，程序停止在某条语句，该条语句左边就会出现一个黄色的小箭头，可以随时中断程序、单步执行、查看变量、检查调用情况。比如，按 F5 功能键进入调试模式，程序运行到断点处暂停；不断按 F10 功能键，接着一行一行地执行程序，直到程序运行结束。

需要说明的是，如果希望能一句一句地单步调试程序，在编写程序时就必须一行只写一条语句。

4. 查看变量

单步调试程序的过程中，可以在下方的 Variables（变量）子窗口和 Watch（监视）子窗口中动态地察看变量的值，如图 1-10 所示。Variables 子窗口中自动显示当前运行上下文中的各个变量的值，而 Watch 子窗口内只显示在此 Watch 子窗口输入的变量或表达式的值。随着程序的逐步运行，也可以直接用鼠标指向程序中变量查看其值。例如在图 1-10 中，可以清楚地看到，程序已经为整型变量 x、y、max 分配了内存，但它们的初始值是不确定的。

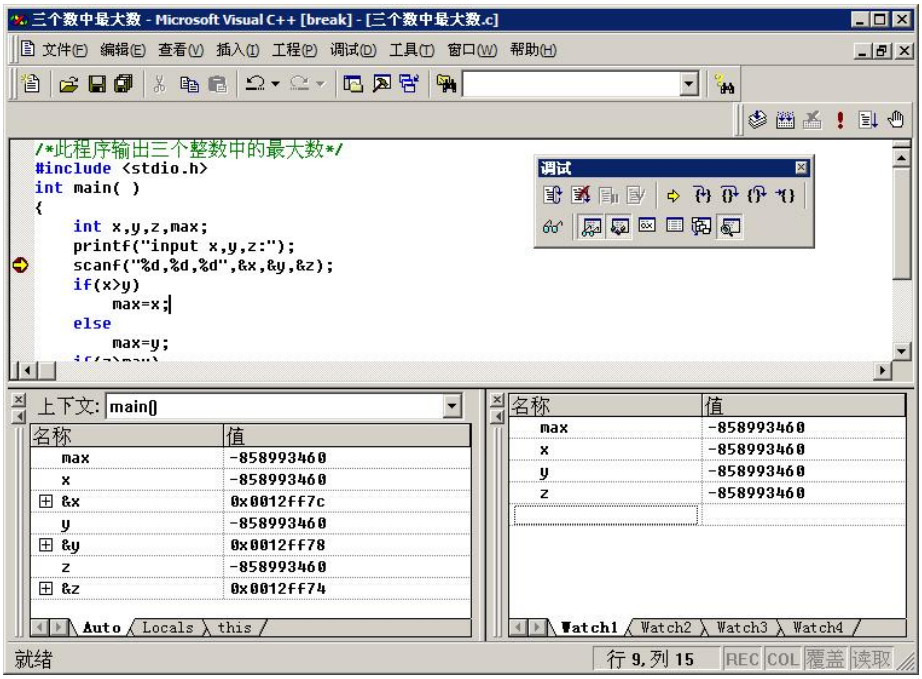

图 1-10 查看变量

Variables 子窗口有 3 个选项卡：Auto、Locals 和 this。

Auto 选项卡：显示出当前语句和上一条语句使用的变量，它还显示使用 Step Over 或 Step Out 命令后函数的返回值。

Locals 选项卡：显示出当前函数使用的局部变量。

如果变量较多，自动显示的 Variables 子窗口难以查看时，还可以在右边的 Watch 子窗口中添加想要监控的变量名。例如，图 1-10 在 Watch1 子窗口中添加了变量“max”。还可以直接将变量拖动到 Watch 子窗口的空白名称框中。添加结束后，该变量的值会被显示出来。并且随着单步调试的进行，会看到变量 max 的值逐渐变化。如果各变量的值按照设想的方式逐渐变化，程序运行结果无误，本次开发就顺利结束了。如果发现各变量值的变化和设想的不一致，说明程序存在逻辑错误，那就需要停止调试，返回编辑窗口，查错并修改程序。

5. 查看内存

数组和指针指向了一段连续的内存中的若干个数据。可以使用 memory 功能显示数组和指针指向的连续内存中的内容。在 Debug 工具栏上单击 memory 按钮，弹出一个对话框，在其中输入数组或指针的地址，就可以显示该地址指向的内存的内容，如图 1-11 所示。

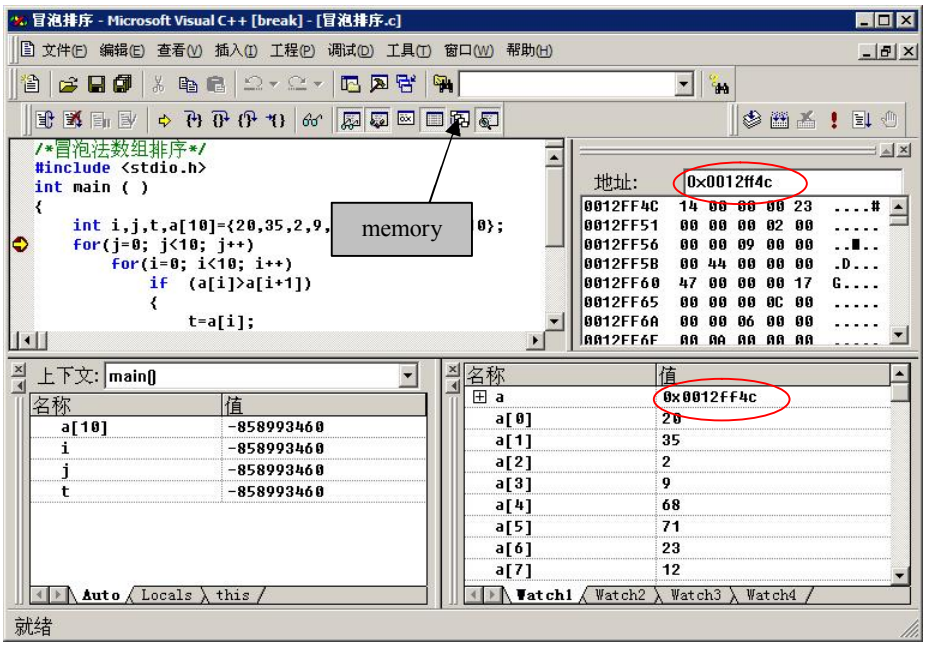

图 1-11 查看内存

习题 1

1. 请上网查阅 C 语言的产生、发展和目前状况。
2. 请上网查找 C 语言的作用（能编写哪些程序）。