

## 9.3 编程环境

### 9.3.1 认识编程环境

编程环境也叫集成开发环境(Integrated Developing Environment, IDE), 是一个综合性的工具软件, 它把程序设计全过程所需的各项功能集合在一起, 为程序设计人员提供完整的服务。编程环境并不是把各种功能简单地拼装在一起, 而是把它们有机地结合起来, 统一在一个图形化操作界面下, 为程序设计人员提供尽可能高效、便利的服务。例如, 程序设计过程中为了排除语法错误, 需要反复执行编译—查错—修改—再编译的循环过程, 编程环境使得各步骤之间能够方便快捷地切换, 输入源程序后用简单的菜单命令或快捷键启动编译, 出现错误后又能立即转到对源程序的修改, 甚至直接把光标定位到出错的位置上。再如, 编程环境的编辑器除了具备一般文本编辑器的基本功能外, 还能根据 C 的语法规则, 自动识别程序文本中的不同成分, 并且用不同的颜色显示不同的成分, 给使用者产生很好的提示效果。

对于编写 C 语言程序, 下面简单地介绍四种常用的集成开发环境。

Visual C++ 6.0, 简称 VC 或者 VC6.0, 是微软推出的一款 C++编译器。Visual C++是一个功能强大的可视化软件开发工具。

Code Blocks 是一款开源的跨平台开发软件。Code Blocks 支持使用 C 以及 C++程序开发。Code Blocks 有着快速的反应速度, 而且体积也不大。

Visual Studio2010 其实是微软开发的一套工具集, 它由各种各样的工具组成, Visual Studio 可以用于生成 Web 应用程序, 也可以生成桌面应用程序, 可以使用 C 语言、C++语言、C#语言或者 Basic 语言进行开发。

Dev-C++是一个可视化集成开发环境, 可以实现 C/C++程序的编辑、预处理、编译、运行和调试。

本章结合具体用例, 介绍以上四种编程环境的使用, 包括如何建立项目工程, 如何编写程序, 如何运行程序, 如何调试程序等。另外, 考虑到使用的编程环境可能是中文汉化的, 也可能是西文的, 因此对于使用的菜单, 同时给出菜单的英文和中文, 方便对照。

本章实验用例为: 输入两个圆的半径 (整型), 计算这两个圆的周长以及它们的周长之比。由于圆的周长等于半径乘以  $2\pi$ , 故两个圆的周长的比值就等于它们的半径的比值。

### 9.3.2 Visual C++6.0 环境

Visual C++6.0 不仅是一个 C++编译器，而且是一个基于 Windows 操作系统的可视化集成开发环境，包括编辑器、调试器等开发工具。下面介绍一下在 Visual C++ 6.0 环境下运行一个 C 语言程序的基本步骤。

(1) 新建工程。启动 Visual C++6.0，执行“File”→“New”→“Project”（即“文件”→“新建”→“项目”）命令，在弹出的“New”对话框中单击“Project”选项卡，建立一个 Win32 控制台工程（即在列表框中选择 Win32 Console Application 选项），在“Project name”（即“工程名称”）文本框中输入工程名称，点击“Location”（即“位置”）文本框右侧的按钮可以改变工程的保存路径，如图 9-9 所示。

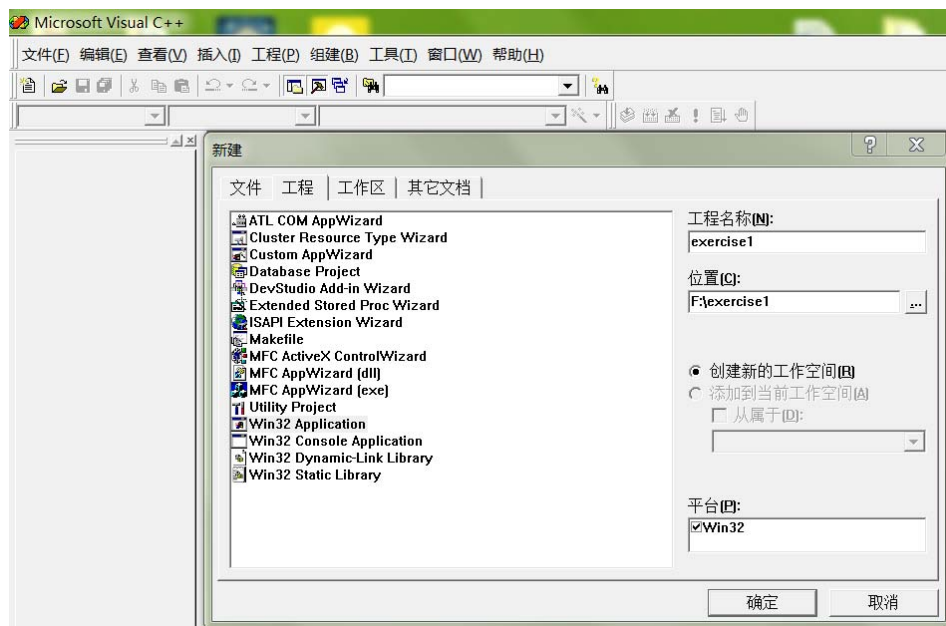


图 9-9 Visual C++6.0 建立工程第一步

(2) 然后点击“确定”按钮，弹出如图 9-10 所示的对话框，选择第一个单选按钮“An empty project”（即“一个空工程”），最后单击“完成”按钮。

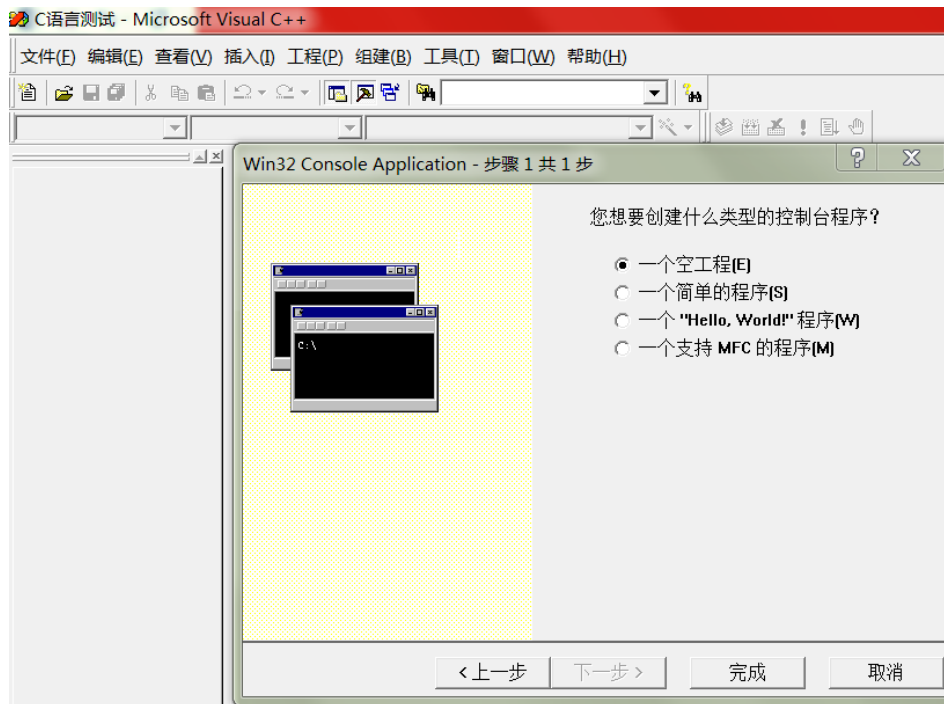


图 9-10 Visual C++6.0 建立工程第二步

(3) 新建文件。执行“File”→“New”（即“文件”→“新建”）命令，在弹出的“New”对话框中单击“Files”选项卡，建立一个 C++源文件(即在列表框中选择 C++ Source File 选项),在“File”(即“文件名”)文本框中输入源文件名称,点击“Location”（即“位置”)文本框右侧的按钮可以改变源文件的保存路径（一般情况下，为源文件与工程路径一致），如图 9-11 所示。

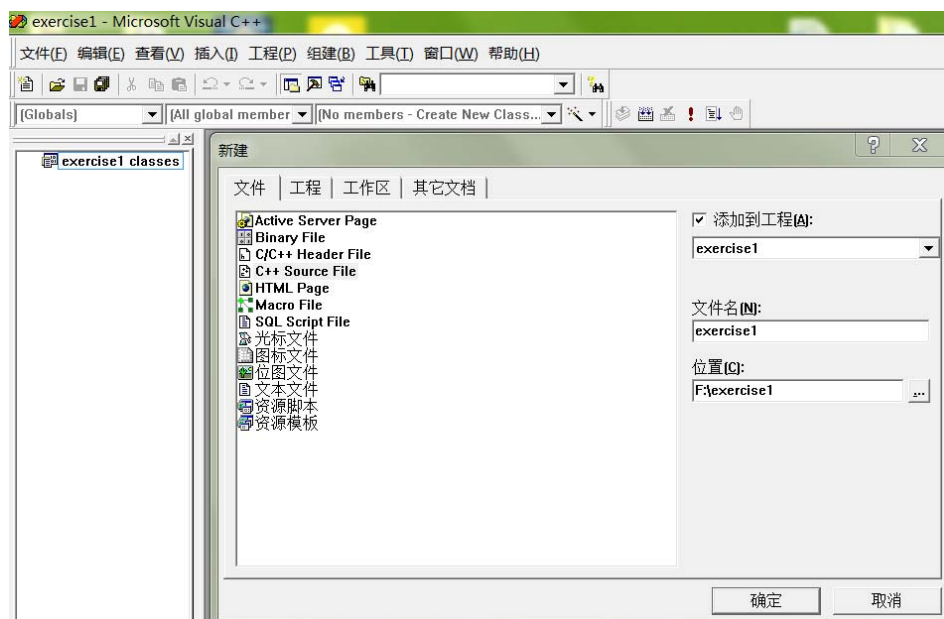


图 9-11 Visual C++6.0 建立源文件

(4) 编辑、保存文件。在编辑窗口中输入源程序，然后执行“File”→“Save”

(即“文件”→“保存”)或按【Ctrl + S】快捷键保存创建的源文件,如图 9-12 所示。

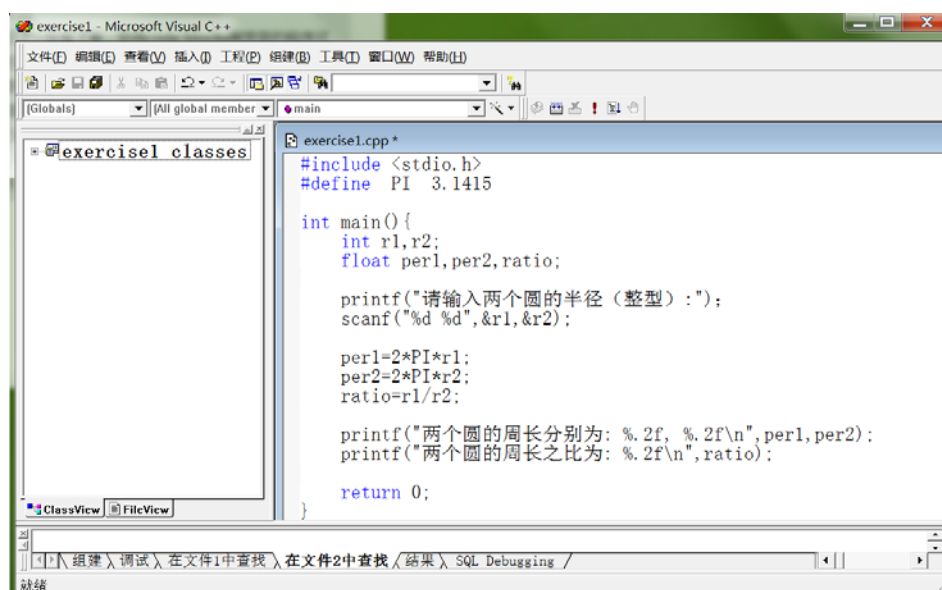


图 9-12 Visual C++6.0 编辑程序

(5)编译文件。执行“Build”→“Build”(即“组建”→“生成”)或按快捷键【F7】编译源文件。如果程序没有语法错误,则生成后缀为 `exe` 的可执行文件。

(6)修改和调试程序。编译完成后在信息窗口显示结果,如果程序有错误,则会在信息窗口提示并告知错误位置。在信息提示窗口点击错误信息,编辑窗口就会出现一个箭头指向程序出错的位置,如图 9-13 所示。信息提示存在语法错误,指示标识符“`scanf`”的前面缺少分号。这个语法错误的原因是,`scanf` 语句的前面语句(即 `printf` 语句)末尾的分号输入成了中文的分号。修改后没有其它的语法错误,编译结果如图 9-14 所示。

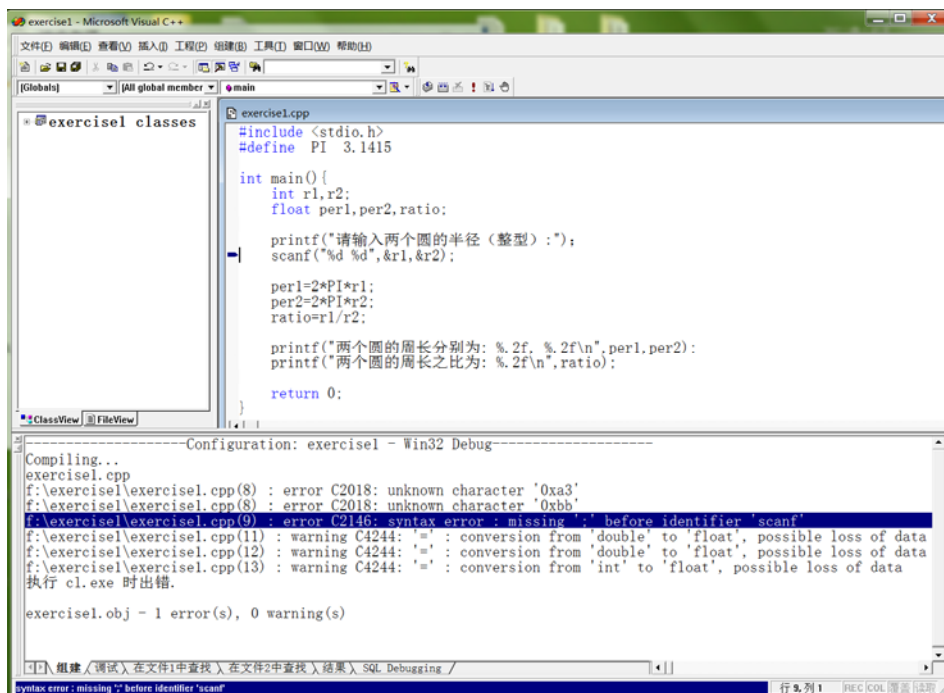


图 9-13 Visual C++6.0 编译产生的错误信息

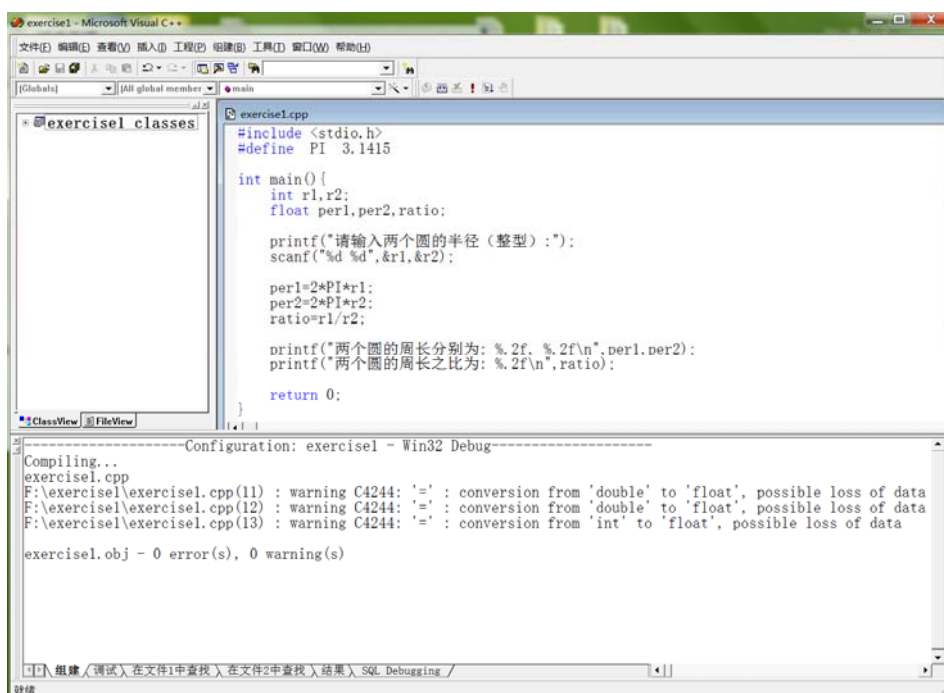


图 9-14 Visual C++6.0 更正错误后编译和连接结果

(7) 运行程序。更正源程序中的代码错误后，编译无错误，然后执行“Build”→“Execute”（即“组建”→“运行”）或按快捷键【Ctrl+F5】运行源文件，输入两个圆的半径，然后按回车键，结果如图 9-15 所示。

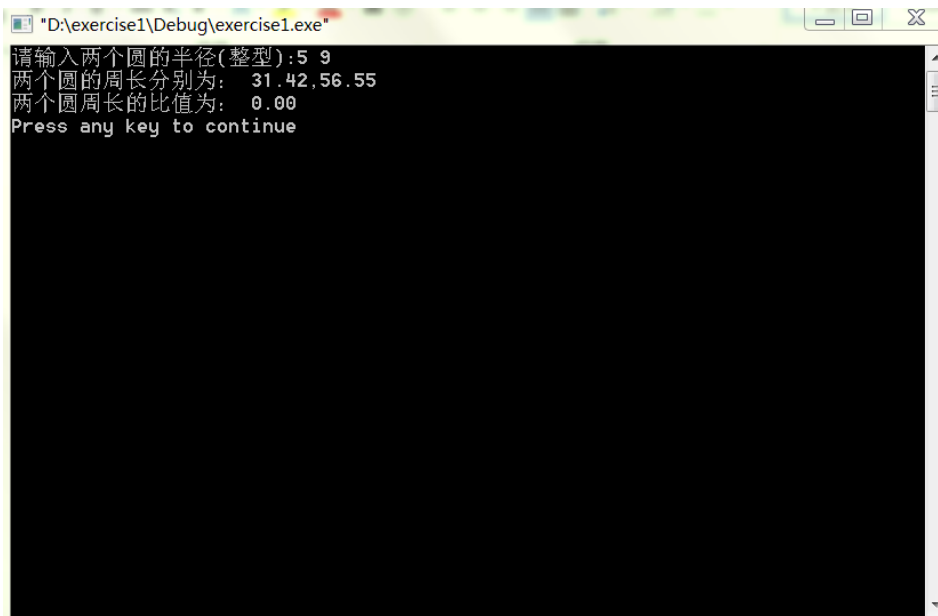


图 9-15 Visual C++6.0 程序运行结果

(8) 调试。编译器可以发现程序的语法错误，但是却不能发现程序的逻辑错误。在这个例子中，当两个圆的半径分别为 5 和 9 时，它们周长的比值却为 0，说明程序中存在逻辑错误，这时我们需要借助程序调试手段，来排除程序错误。调试的基本思想是让程序运行到你认为可能有错误的代码前，然后停下来，在人为的控制下逐条语句的运行，通过在运行过程中查看相关变量的值，来判断错误产生的原因。

1) 设置断点。如果想让程序运行到某一行前停下来，就需要在该行设置断点。具体方法是在代码所在行的行首右键单击，再单击手型按钮（即“Insert/Remove Breakpoint”），该行将被加亮。默认的加亮颜色是红色。如图 9-16 所示。如果想取消不让某行代码成为断点，则再此点击手型按钮即可。

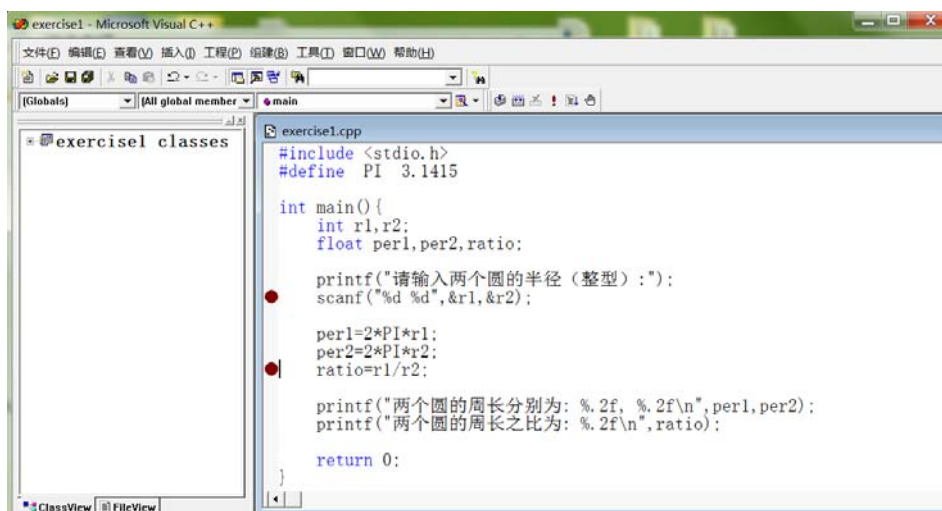


图 9-16 Visual C++6.0 设置断点

2) 调试程序。设置断点后，以调试模式运行程序，即执行“Build”→“Start Debug”→“Go”（即“组建”→“开始调试程序”）或按快捷键【F5】，程序运行到断点处暂停，此时我们可以观察程序运行的情况。Visual C++6.0 会在程序下方出现调试框，并且根据当前执行的程序，自动显示相关的变量名称和值。由于还没有输入两个圆的半径，故变量 r1 和 r2 都是一个随机值。同理，两个周长 per1 和 per2，以及比值 ratio 也是随机值。注意，这些变量的值都没有初始化，是无效的值，因此不能使用。如图 9-17 所示。

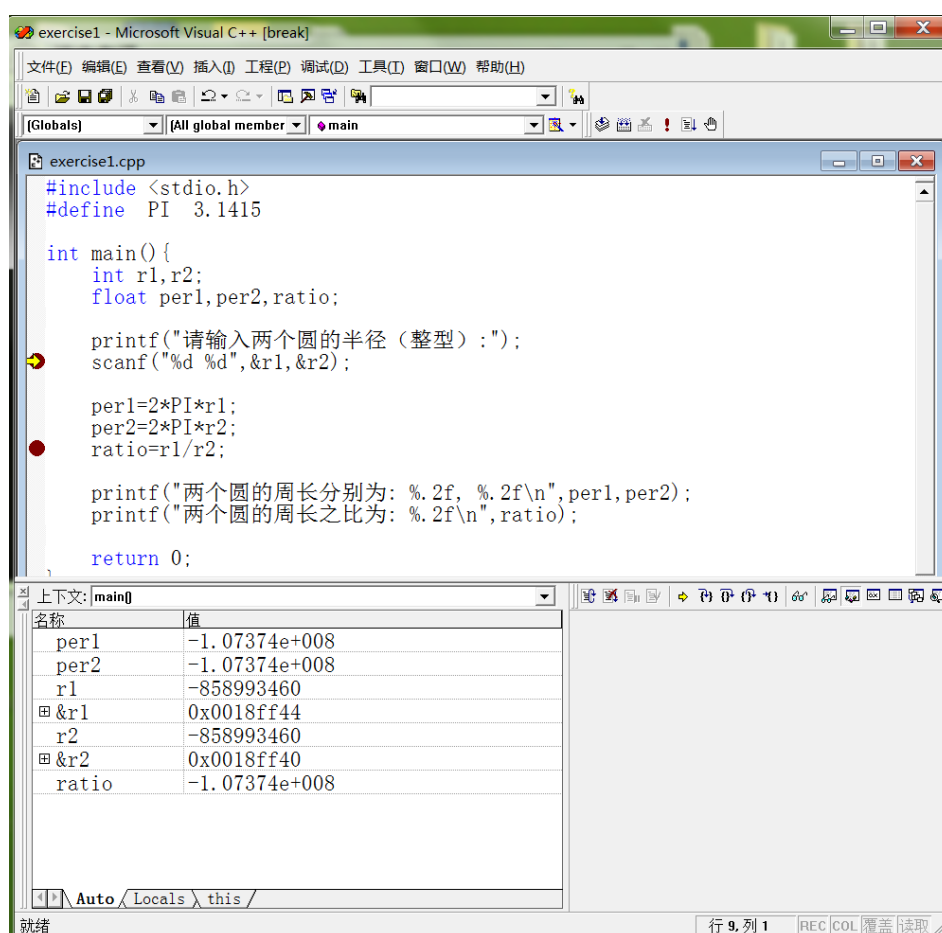


图 9-17 Visual C++6.0 调试程序

3) 单步调试程序。通过单步调试程序的方式观察程序运行的情况。执行“Build”→“Start Debug”→“StepInto”（即“组建”→“开始调试程序”→“单步调试”）或按快捷键【F11】来进行单步调试。本实验程序单步调试 1 次以后，输入两个半径的值，如图 9-18 所示，各变量的值如图 9-19 所示。



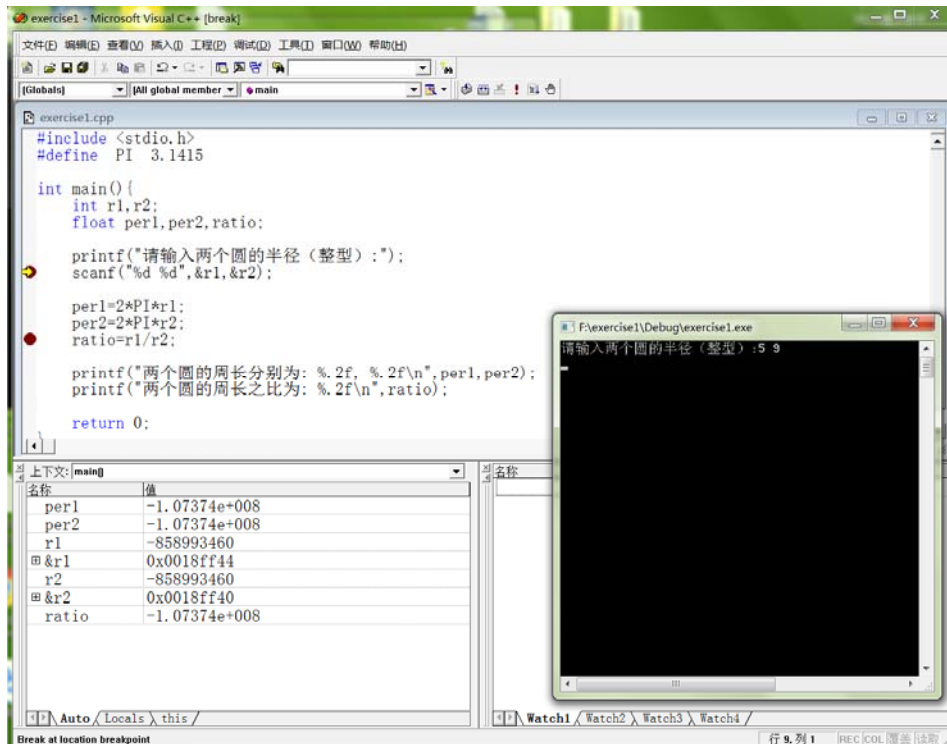


图 9-18 Visual C++6.0 单步调试一次

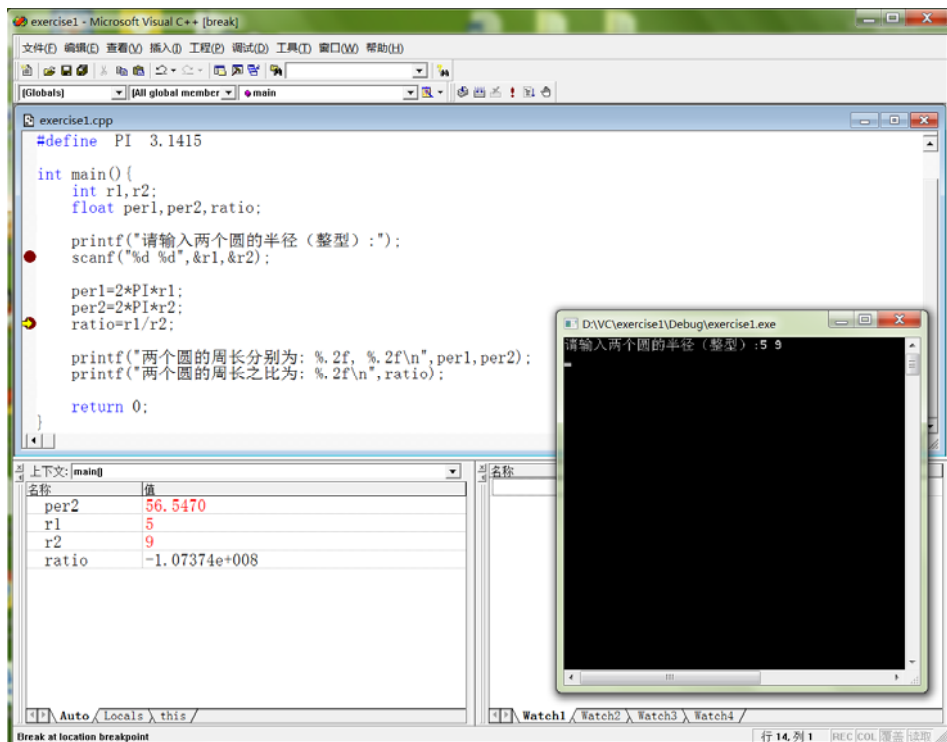


图 9-19 Visual C++6.0 变量的值

4) 查找程序错误。再执行单步调试，查看变量 `r1`、`r2` 和 `ratio` 的值。此时，`r1=5`，`r2=9`，而 `ratio=0`，如图 9-20 所示。通过调试，我们发现变量 `r1`、`r2` 的值是正确的，但是它们的比值计算不正确：比值应该是 0.56，而不应该是 0。出现



此问题的原因是，除法运算符 (/) 在用于两个整型数据时，只保留比值的整数部分，小数部分被忽略。如果想保留小数部分，可以进行类型转换。例如，把变量 r1 的数据类型由整型转为浮点型。改正错误，如图 9-21 所示，然后再运行测试程序，结果正确，如图 9-22 所示，其中“Press any key to continue”表示按任意键退出运行窗口，返回编辑窗口。

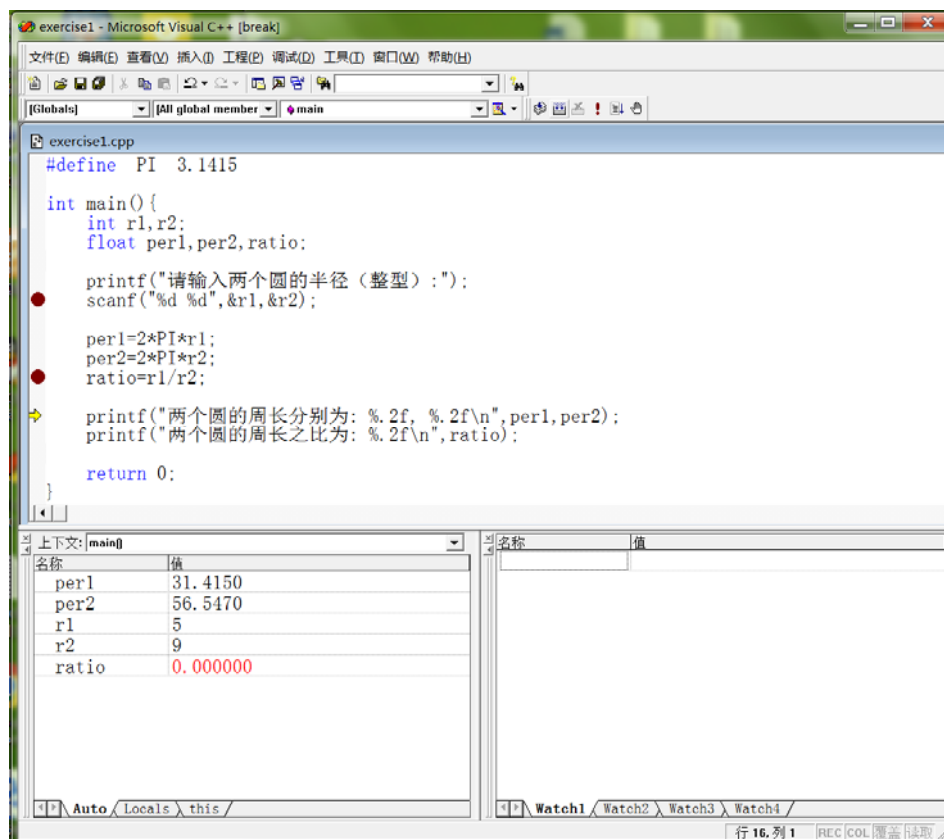


图 9-20 Visual C++6.0 变量 ratio 的值

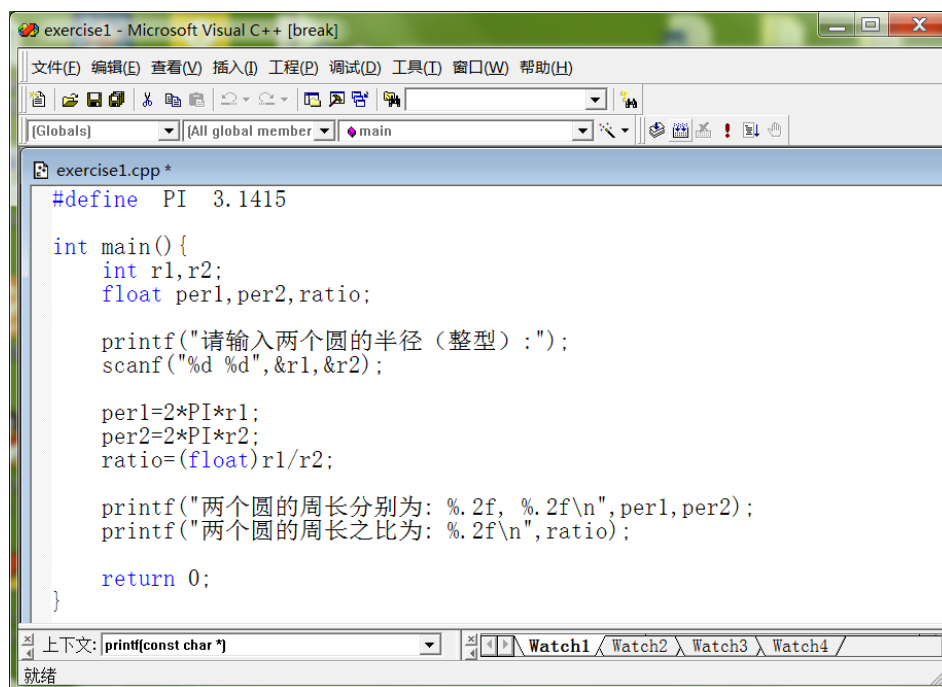


图 9-21 Visual C++6.0 更正程序

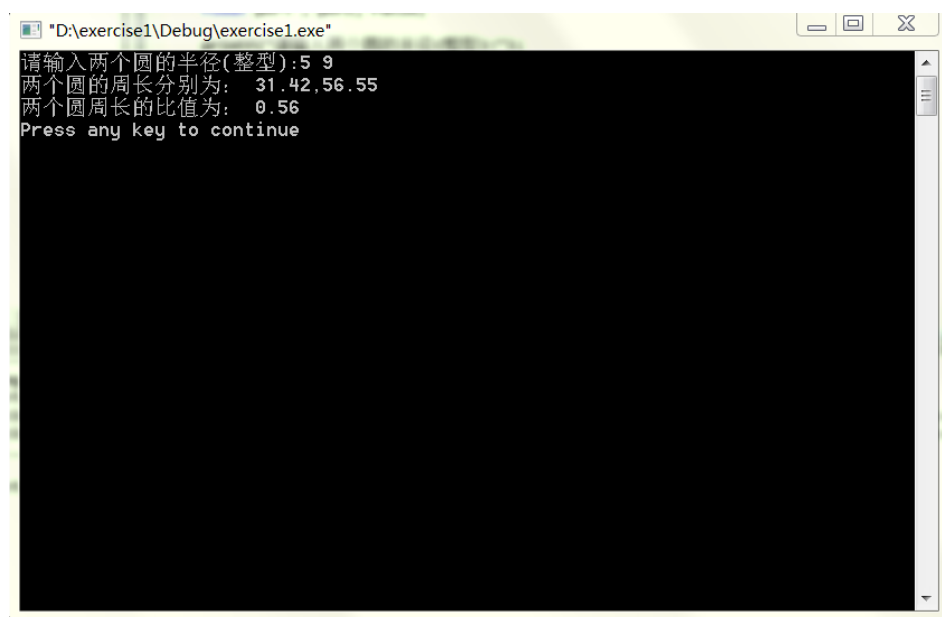


图 9-22 Visual C++6.0 程序运行结果

### 9.3.3 Code Blocks 环境

Code Blocks 是一款开源的跨平台开发软件。Code Blocks 支持使用 C 以及 C++ 程序开发，其体积不大，而且运行速度快。下面介绍一下在 Code Blocks 环境下编写和运行程序的基本步骤。

(1) 新建工程。启动 Code Blocks，执行“File”→“New”→“Project”（即“文

件”→“新建”→“项目”) 命令, 在弹出的对话框中选择 Console Application 选项 (即“控制台应用程序”), 如图 9-23 所示。

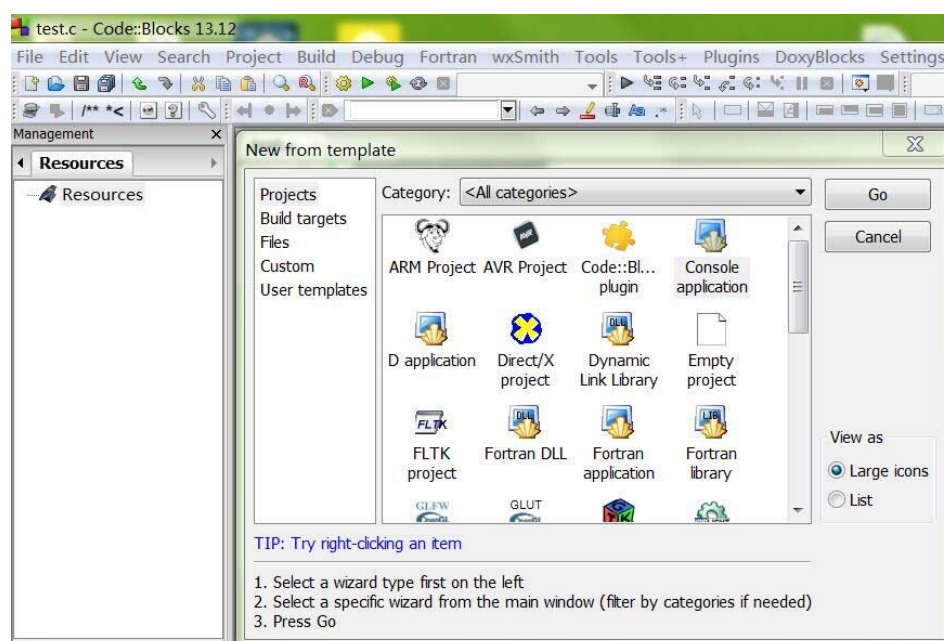


图 9-23 Code Blocks 新建项目

(2) 然后点击确定 “GO” 按钮, 在弹出的对话框中直接点击“Next”进行下一步, 然后选择建立一个 C 项目, 点击“Next”进行下一步, 在弹出的对话框中确定项目的位置及文件名, 如图 9-24 所示。

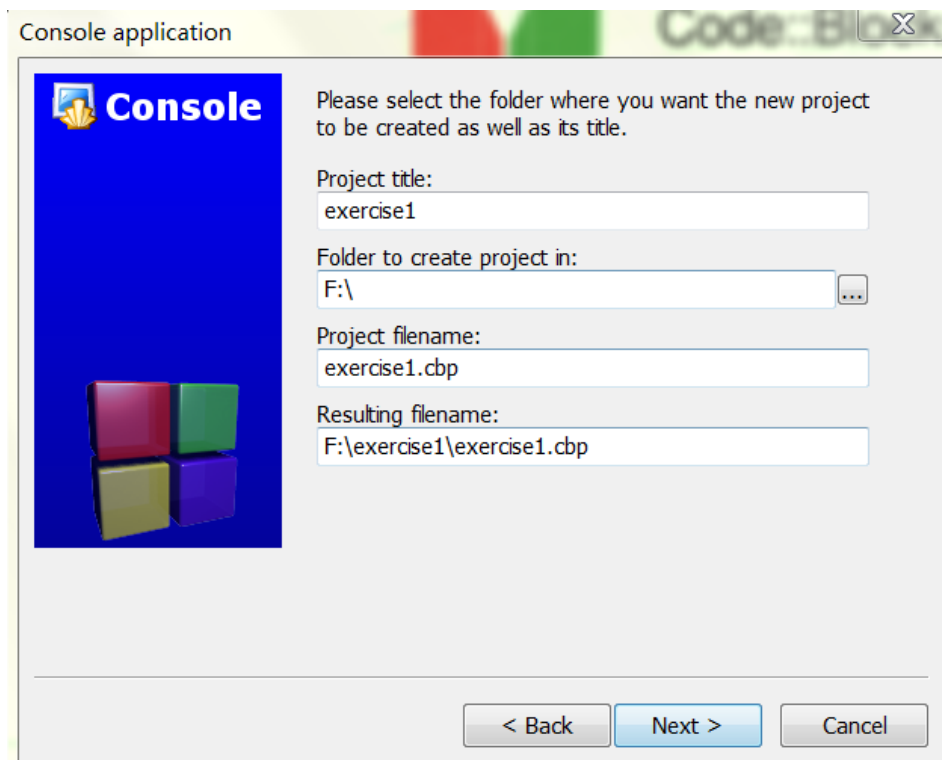


图 9-24 Code Blocks 确定项目名称及位置

(3) 在弹出的对话框中最后选择编译器和编译生成位置，如果你安装的是自带 Mingw 的 Code Blocks，就选默认 GNU GCC 编译器；如果你装了 Turbo C 或者 VisualC++ 等第三方编译器，还可以选择其它的对应的编译器选项。这里选择默认的编译器，如图 9-25 所示。

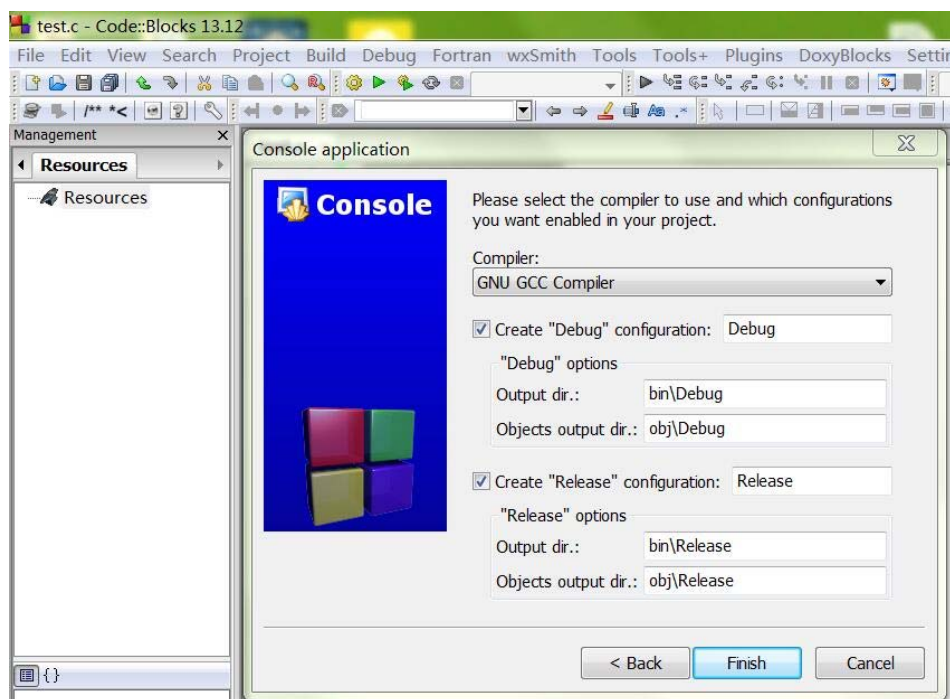


图 9-25 Code Blocks 选择编译器

(4) 新建、编辑源文件。现在很多高版本的 Code Blocks 已经不需要再手动创建文件，在创建控制台项目后，可以在左侧项目管理窗口中点开 Resources 文件夹，里面有 main.c 或 main.cpp 文件。你可以直接编辑这个文件，写好代码后跳到编译运行这一步。如果没有这个文件的话，可以按以下步骤新建文件。执行“File”→“New”→“File”（即“文件”→“新建”→“文件”）命令，在弹出的对话框中选择“C/C++ Source”，然后点击确定“GO”按钮，如图 9-26 所示。

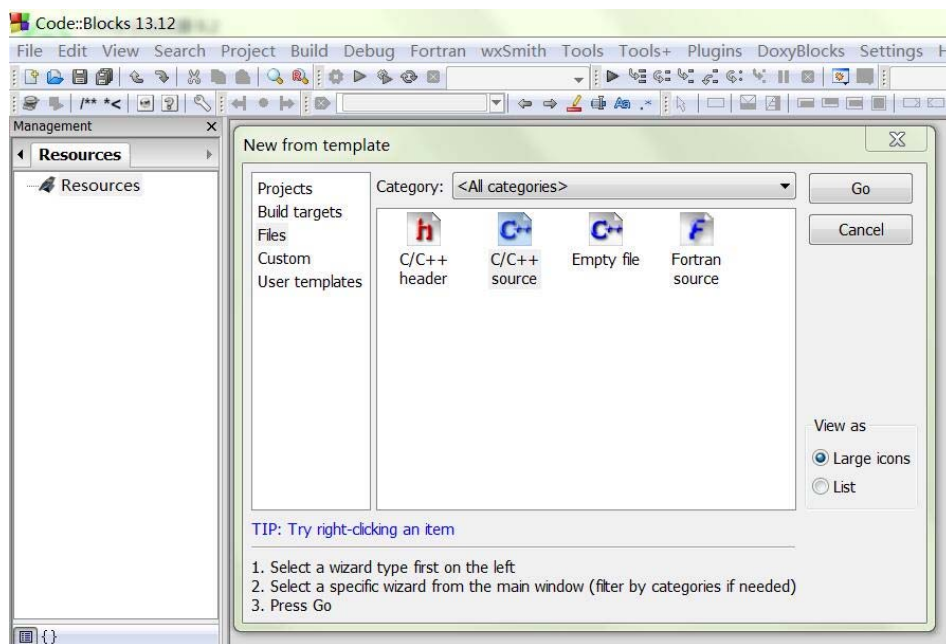


图 9-26 Code Blocks 新建源文件

(5) 选择建立一个 C 项目，在弹出的对话框中确定文件的位置及文件名，如图 9-27 所示。在新建源文件时需要选择“Add file to active project”(即“添加到活动项目”)，并选中下面的 Debug 和 Release。

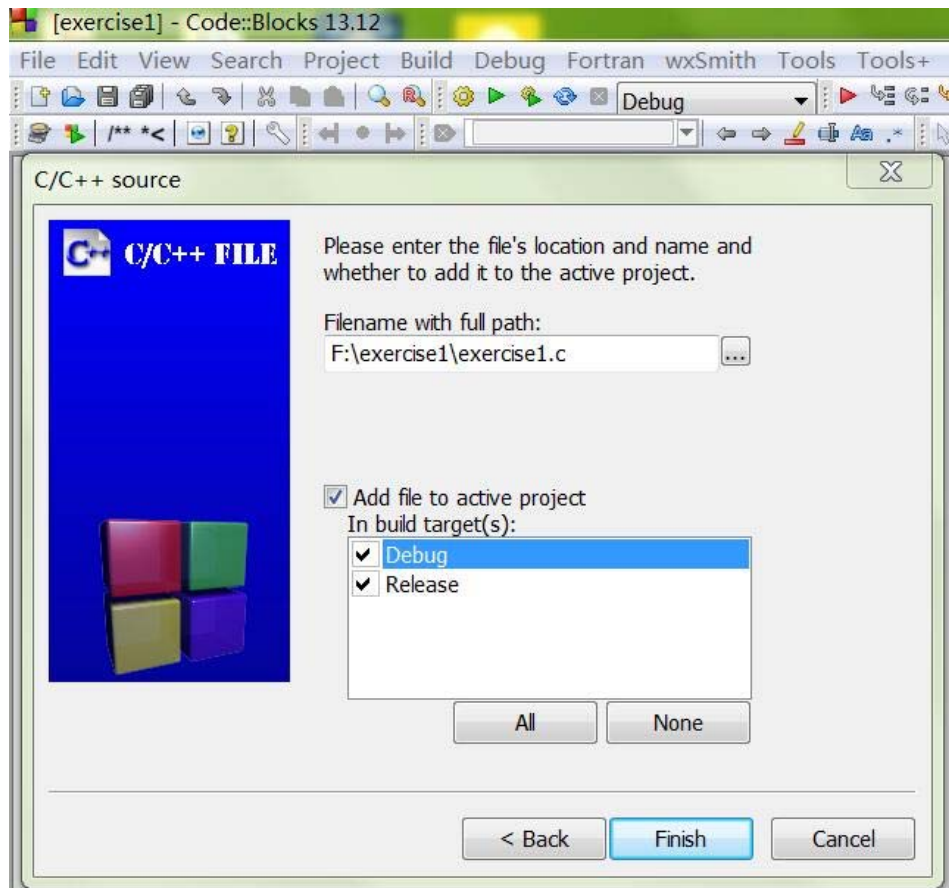


图 9-27 Code Blocks 确定源文件位置

(6) 编辑、保存文件。在编辑窗口中输入源程序后，执行“File” → “Save”（即“文件”→“保存”）或按【Ctrl + S】快捷键保存创建的源文件，如图 9-28 所示。

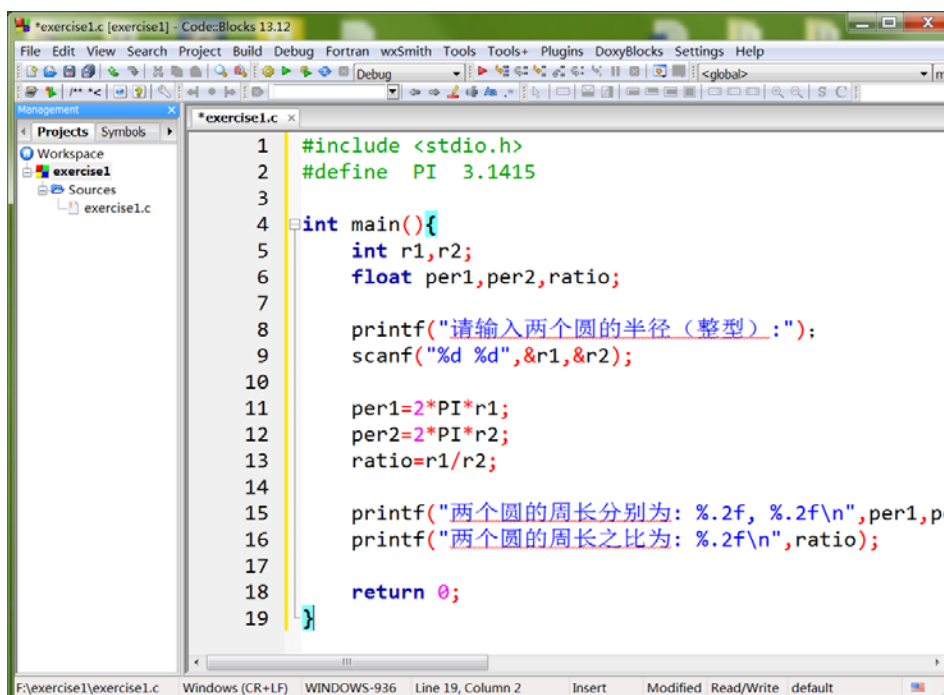


图 9-28 Code Blocks 编辑程序

(7) 编译文件。执行“Build”→“Build & Run”（即“组建”→“生成+运行”）或按快捷键【F9】，可以一次性完成程序的预处理、编译及运行。这里先执行“Build”→“Build”（即“组建”→“生成”）或按快捷键【Ctrl+F9】完成对程序的预处理、编译。编译完成后在信息窗口显示结果，如图 9-29 所示。

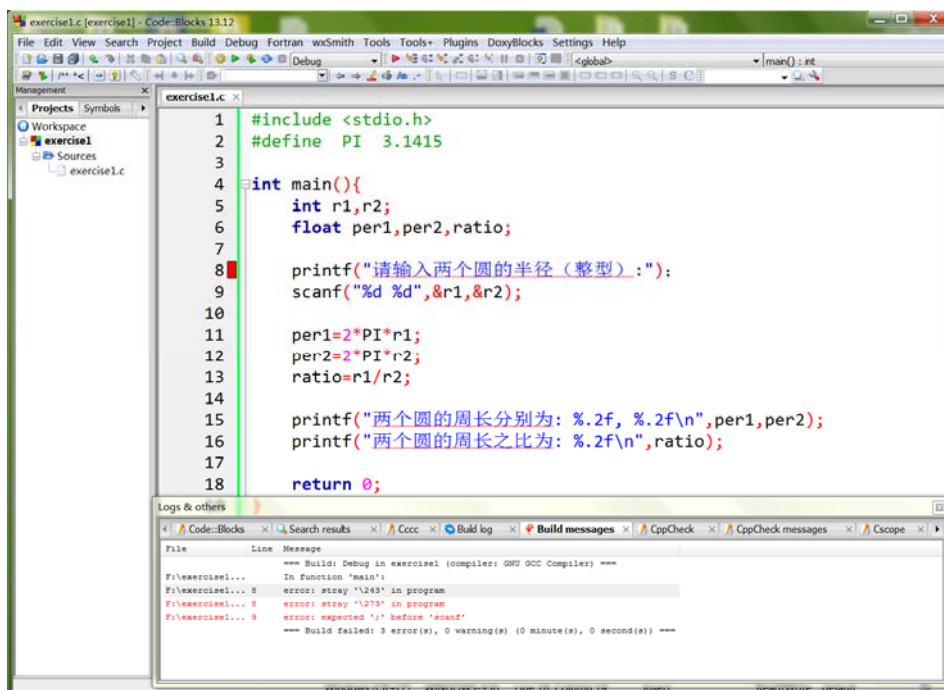


图 9-29 Code Blocks 编译程序



(8) 修改和调试程序。如果程序存在语法错误，在信息窗口“Logs & others”标签页中将会显示错误信息，并在源程序相应的错误行的前面有红色标志，如图 9-29 所示。第八行代码的分号错误地输入成了中文，改正错误后重新编译，结果如图 9-30 所示。

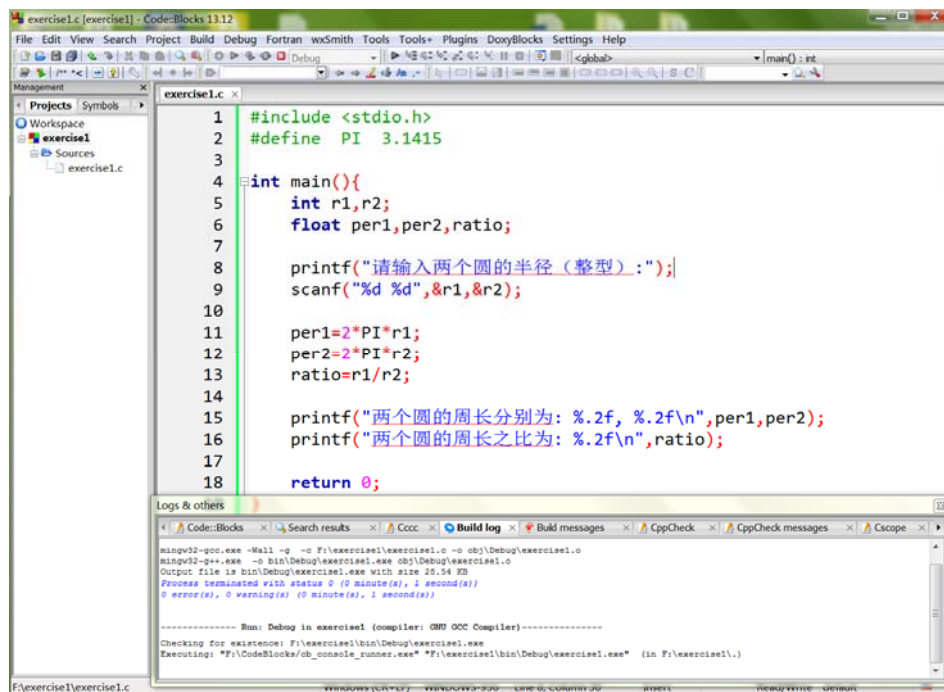


图 9-30 Code Blocks 更正错误后编译结果

(9) 运行程序。更正源程序中的代码错误后，编译无错误，然后执行“Build”→“Run”（即“组建”→“运行”）或按【Ctrl + F10】快捷键运行源文件，自动弹出运行窗口，如图 9-31 所示。

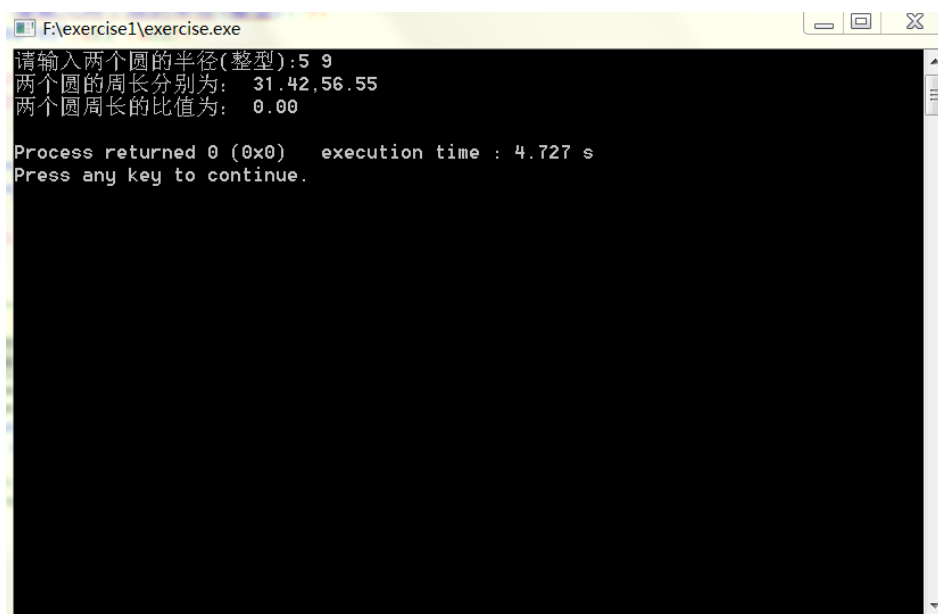


图 9-31 Code Blocks 程序运行结果

(10) 调试。

1) 设置断点。如果想让程序运行到某一行前能暂停下来，就需要将该行设成断点。具体方法是在代码所在行行首左键单击，该行将被加亮。默认有加亮颜色是红色。如图 9-32 所示。如果想取消不让某行代码成为断点，则在代码行首位置再点击即可。

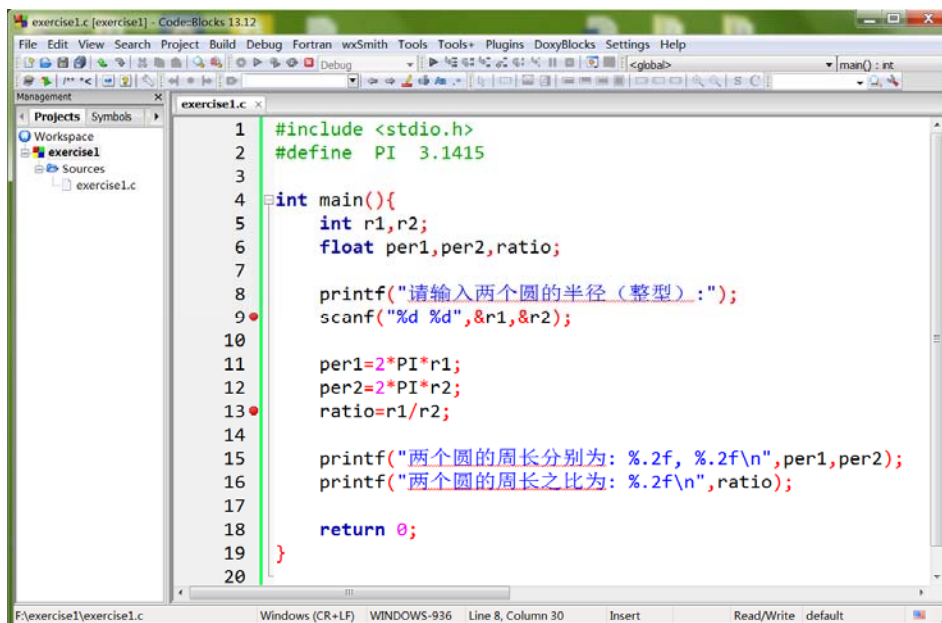


图 9-32 Code Blocks 设置断点

2) 调试运行程序。设置断点后，以调试模式运行程序，即执行“Debug” → “Start/Continue”（即“调试”→“开始/继续”）或按快捷键【F8】，并且运行到断点处暂停下了，此时我们可以观察程序运行的情况。如图 9-33 所示。

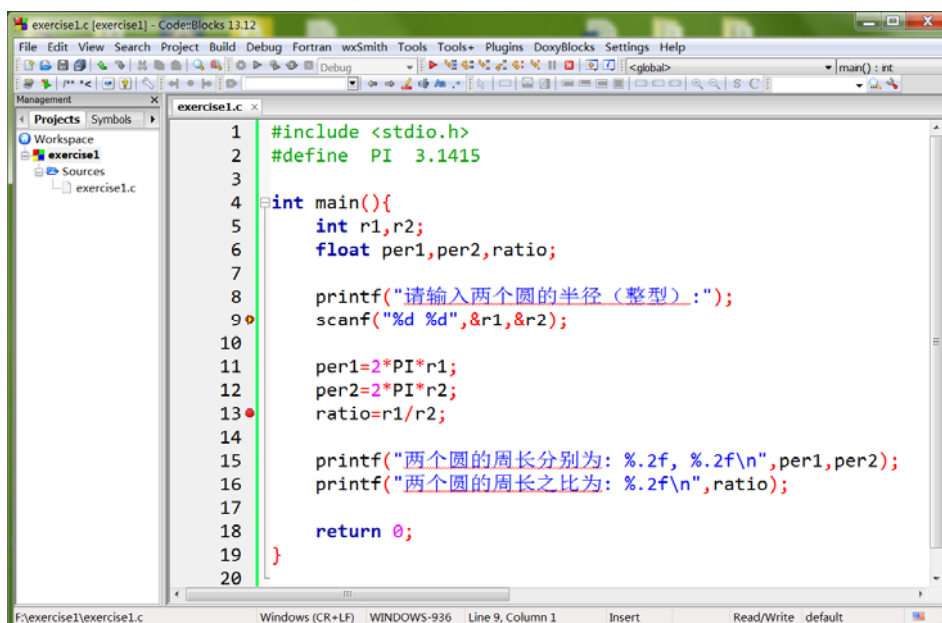


图 9-33 Code Blocks 调试程序

3) 设置 watch 窗口。在调试程序的时候，可能需要看程序运行过程中的变量的值，以检测程序对变量的处理是否正确。执行“Debug”→“debugging windows”→“watches”（即“调试”→“调试窗口”→“变量”），会出现一个 watches 窗口，可以查看程序执行过程中每个变量数据值的变化，如图 9-34 所示。这里对于变量的查看和 9.2.1 节所描述的一样。

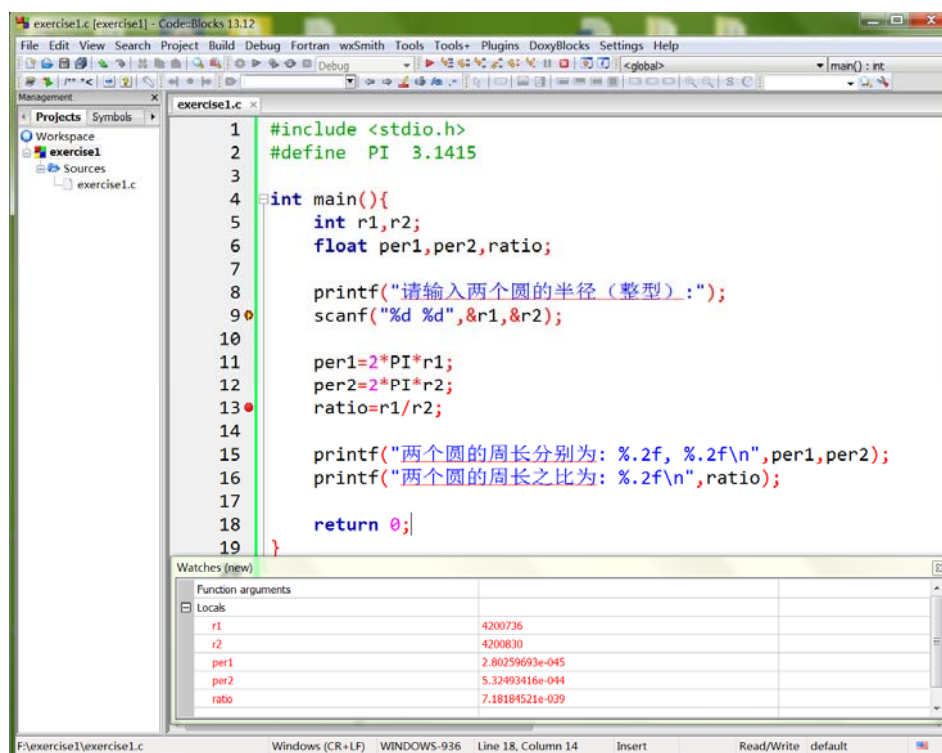


图 9-34 Code Blocks 设置 watches 窗口

4) 单步调试程序。执行“Debug”→“Step into”（即“调试”→“单步调试”）或按快捷键【Shift+F7】来进行单步调试，在弹出的运行框中输入两个圆的半径，然后点击回车键，各变量值的结果如图 9-35 所示。再执行单步调试，查看变量 ratio 的值，如图 9-36 所示。

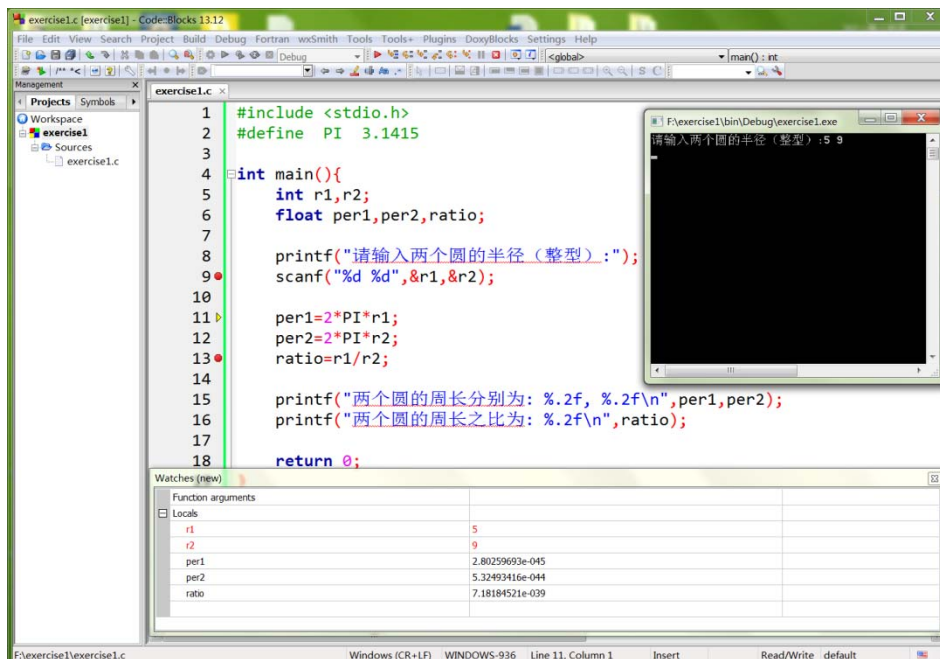


图 9-35 Code Blocks 查看各变量的值

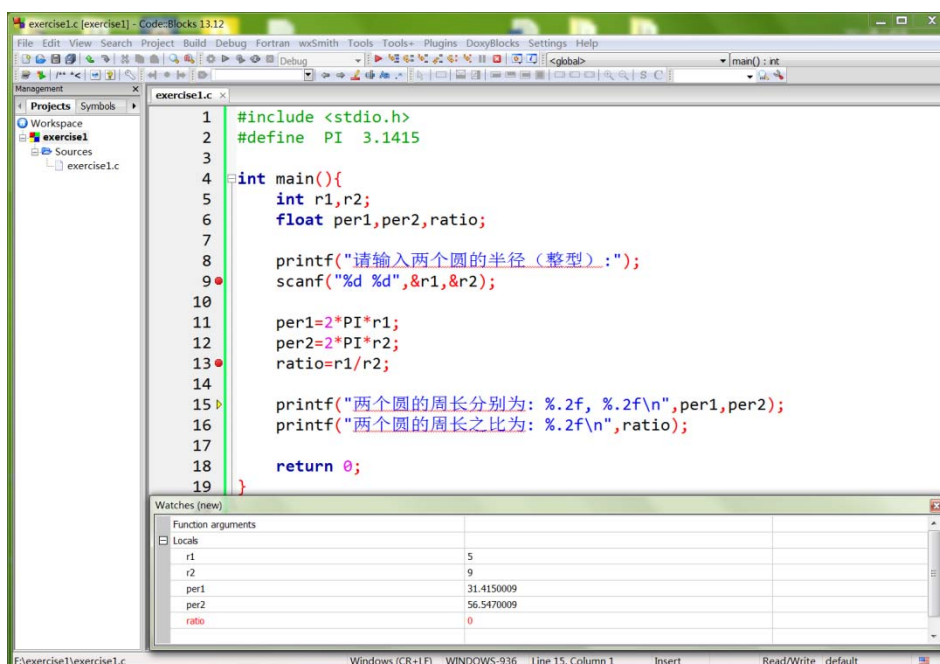
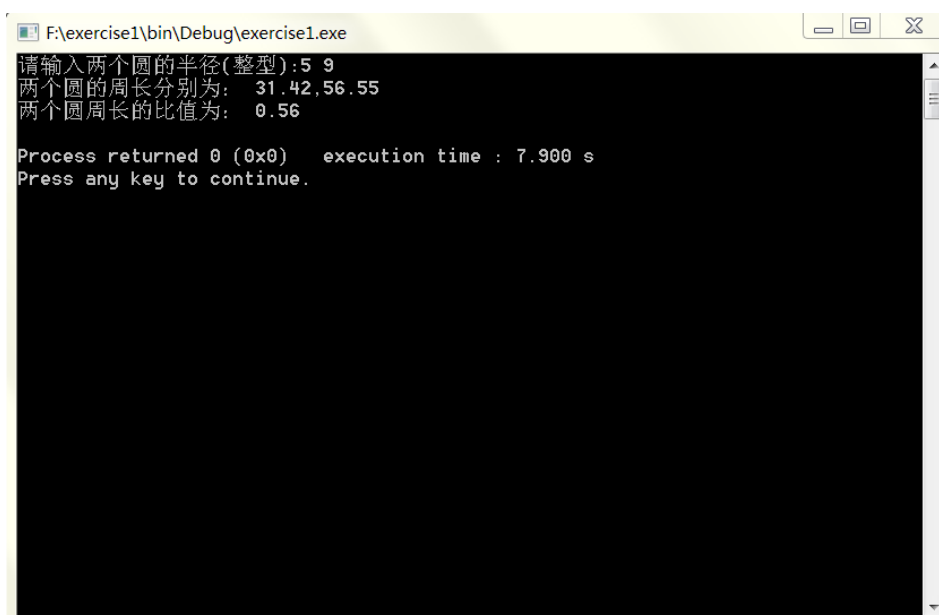


图 9-36 Code Blocks 查看 ratio 的值

5) 查找程序错误。再执行单步调试。查找错误的过程和 9.2.1 节所描述的一样。修改程序后，结果正确，如图 9-37 所示。



```
F:\exercise1\bin\Debug\exercise1.exe
请输入两个圆的半径(整型):5 9
两个圆的周长分别为: 31.42,56.55
两个圆周长的比值为: 0.56

Process returned 0 (0x0)   execution time : 7.900 s
Press any key to continue.
```

图 9-37 Code Blocks 程序运行结果

### 9.3.4 Visual Studio 2010 环境

Visual Studio 是微软开发的一套工具集，它由各种各样的工具组成，Visual Studio 可以用于生成 Web 应用程序，也可以生成桌面应用程序，在 Visual Studio 下面，除了 VC，还有 Visual C#等组件工具，使用这些工具你可以使用 C 语言、C++语言、C#语言等进行开发。下面介绍一下在 Visual Studio 2010 环境下编写和运行程序的基本步骤。

(1) 新建工程。启动 Visual Studio 2010，执行“File”→“New”→“Project”（即“文件”→“新建”→“项目”）选项，在弹出的对话框中选择 Visual C++选项，建立一个 Win32 控制台工程（即在列表框中选择 Win32 Console Application 选项），在“Name”（即“名称”）文本框中输入工程名称，点击“Location”（即“位置”）文本框右侧的按钮可以改变工程的保存路径，如图 9-38 所示。



图 9-38 Visual Studio 2010 新建工程第一步

(2) 创建工程之后，弹出新的对话框，点击“Application Settings”进行应用程序设置，如图 9-39 所示。

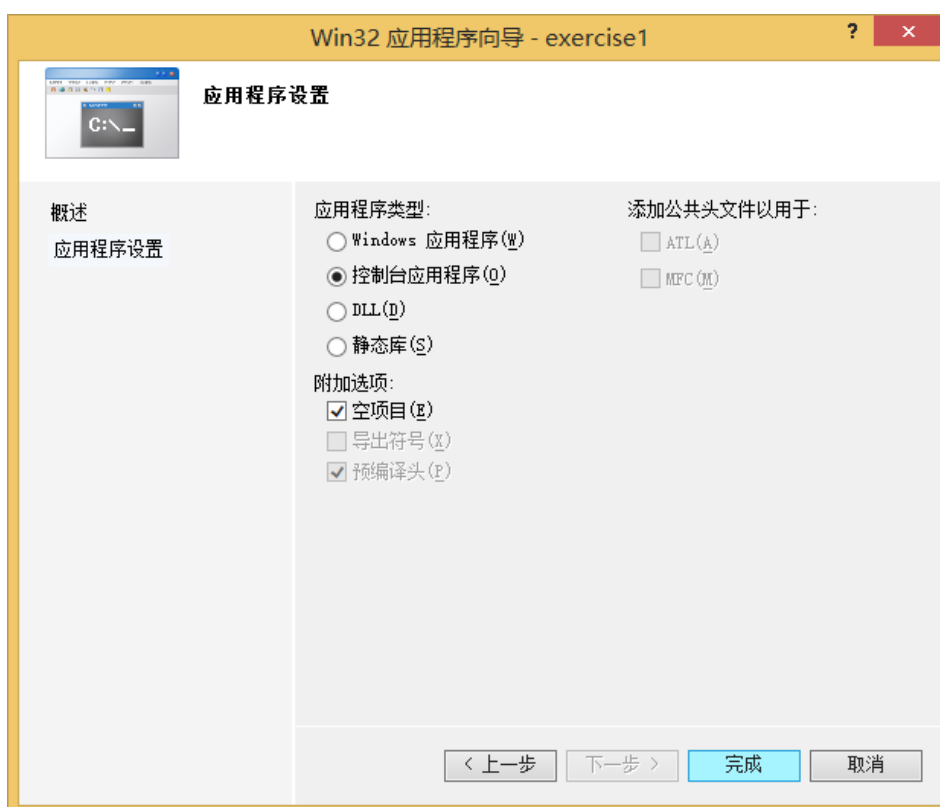


图 9-39 Visual Studio 2010 新建工程第二步

(3) 新建 C 文件。右击新建的工程下的源文件，新建一个 C 源程序，在弹出的对话框中选择 Visual C++ 选项，创建包含 C++ 源代码的文件，在“Name”文本

框中输入源程序名称，点击“Location”文本框右侧的按钮可以改变源程序的保存路径（一般情况下，源文件与工程路径一致），如图 9-40 所示。

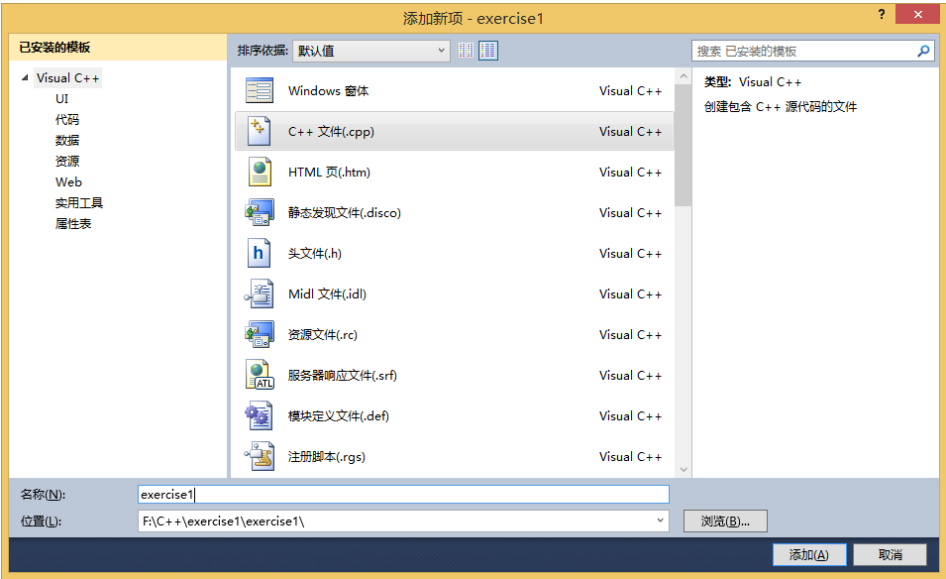


图 9-40 Visual Studio 2010 确定源文件名称及位置

（4）编辑、保存文件。创建源文件之后就可以进行源程序的编辑，然后执行“File”→“Save”（即“文件”→“保存”），保存创建的源文件，如图 9-41 所示。

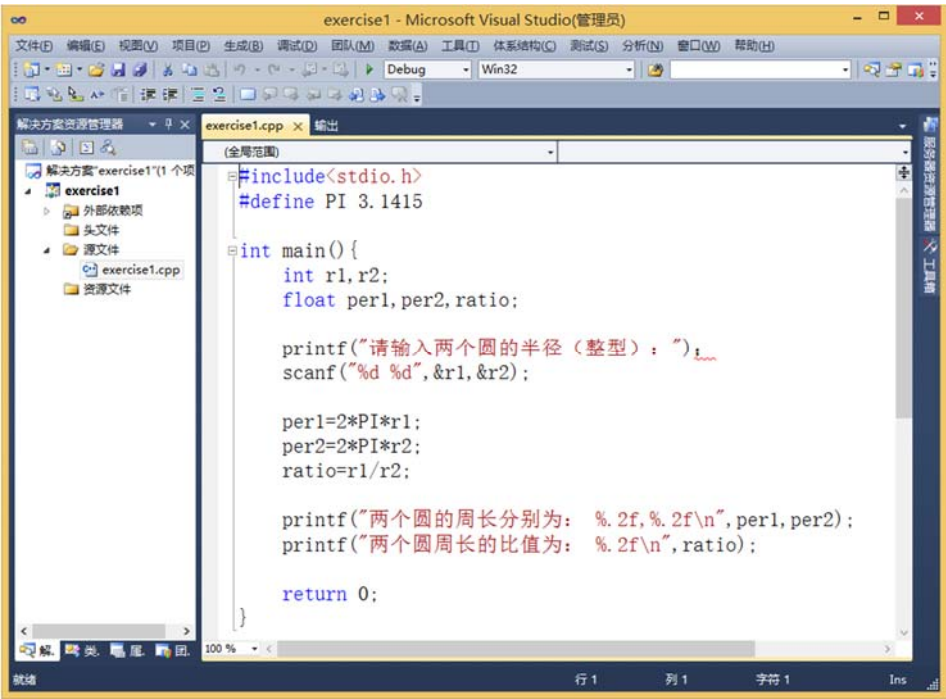
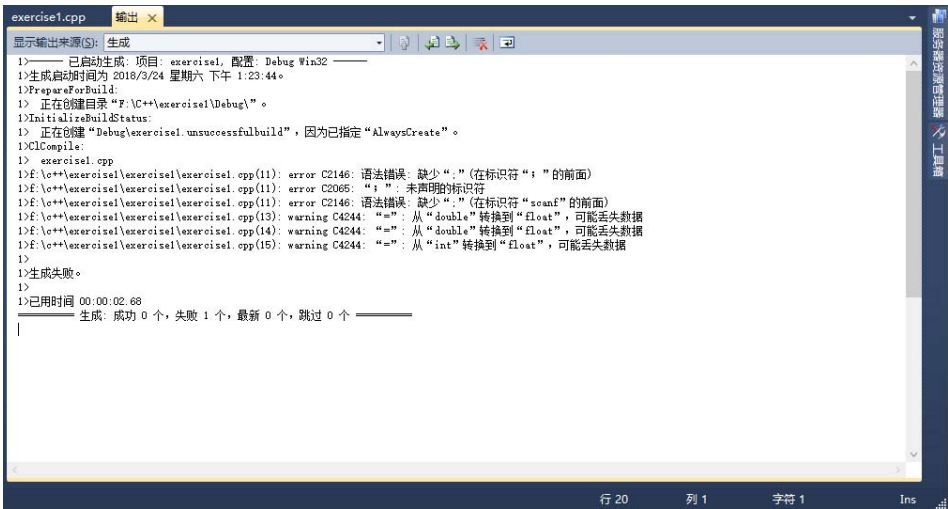


图 9-41 Visual Studio 2010 编辑程序

（5）编译文件。执行“Build”→“Build Solution”（即“生成”→“生成解决方



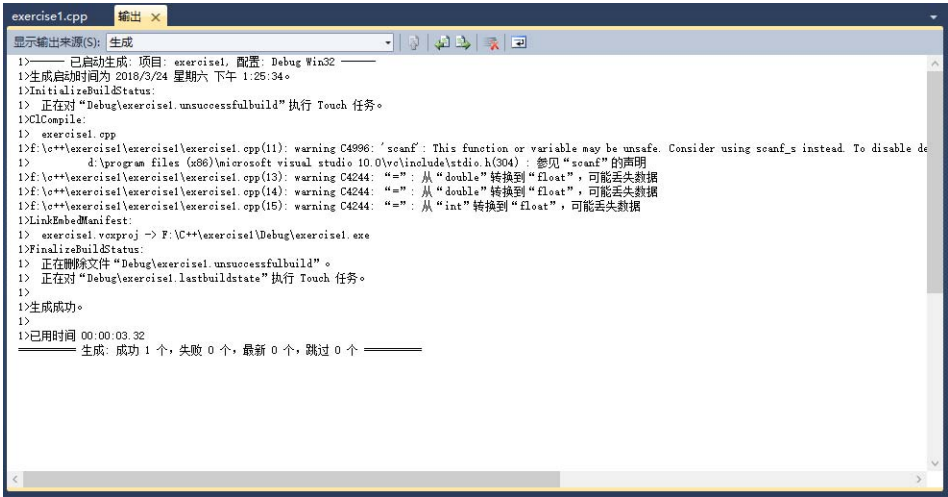
案”)或按【F7】快捷键。如果程序中存在语法等错误,则编译过程失败,编译器将会在程序下方显示错误信息,如图 9-42 所示。



```
exercisel.cpp 输出
显示输出来源(S): 生成
1>----- 已启动生成: 项目: exercisel, 配置: Debug Win32 -----
1>生成启动时间为 2018/3/24 星期六 下午 1:23:44。
1>PrepareForBuild:
1> 正在创建目录 "F:\C++\exercisel\Debug\"。
1>InitializeBuildStatus:
1> 正在创建 "Debug\exercisel.unsuccessfulbuild", 因为已指定 "AlwaysCreate"。
1>CLCompile:
1> exercisel.cpp
1>F:\c++\exercisel\exercisel\exercisel.cpp(11): error C2146: 语法错误: 缺少 ";" (在标识符 ";" 的前面)
1>F:\c++\exercisel\exercisel\exercisel.cpp(11): error C2065: "i": 未声明的标识符
1>F:\c++\exercisel\exercisel\exercisel.cpp(11): error C2146: 语法错误: 缺少 ";" (在标识符 "scanf" 的前面)
1>F:\c++\exercisel\exercisel\exercisel.cpp(13): warning C4244: "==": 从 "double" 转换到 "float", 可能丢失数据
1>F:\c++\exercisel\exercisel\exercisel.cpp(14): warning C4244: "==": 从 "double" 转换到 "float", 可能丢失数据
1>F:\c++\exercisel\exercisel\exercisel.cpp(15): warning C4244: "==": 从 "int" 转换到 "float", 可能丢失数据
1>
1>生成失败。
1>
1>已用时间 00:00:02.68
===== 生成: 成功 0 个, 失败 1 个, 最新 0 个, 跳过 0 个 =====
|
```

图 9-42 Visual Studio 2010 编译程序

(6) 修改和调试程序。在信息提示窗口点击错误信息, 编辑窗口就会出现一个箭头指向程序出错的位置。信息提示存在语法错误, 错误原因是 printf 语句后面的分号错误地输入成了中文的分号, 改正错误后重新编译, 结果如图 9-43 所示。



```
exercisel.cpp 输出
显示输出来源(S): 生成
1>----- 已启动生成: 项目: exercisel, 配置: Debug Win32 -----
1>生成启动时间为 2018/3/24 星期六 下午 1:25:34。
1>InitializeBuildStatus:
1> 正在对 "Debug\exercisel.unsuccessfulbuild" 执行 Touch 任务。
1>CLCompile:
1> exercisel.cpp
1>F:\c++\exercisel\exercisel\exercisel.cpp(11): warning C4896: 'scanf': This function or variable may be unsafe. Consider using scanf_s instead. To disable de
1> d:\program files (x86)\microsoft visual studio 10.0\vc\include\stdio.h(304): 参见 "scanf" 的声明
1>F:\c++\exercisel\exercisel\exercisel.cpp(13): warning C4244: "==": 从 "double" 转换到 "float", 可能丢失数据
1>F:\c++\exercisel\exercisel\exercisel.cpp(14): warning C4244: "==": 从 "double" 转换到 "float", 可能丢失数据
1>F:\c++\exercisel\exercisel\exercisel.cpp(15): warning C4244: "==": 从 "int" 转换到 "float", 可能丢失数据
1>LinkEmbedManifest:
1> exercisel.vcxproj -> F:\C++\exercisel\Debug\exercisel.exe
1>FinalizeBuildStatus:
1> 正在删除文件 "Debug\exercisel.unsuccessfulbuild"。
1> 正在对 "Debug\exercisel.lastbuildstate" 执行 Touch 任务。
1>
1>生成成功。
1>
1>已用时间 00:00:03.32
===== 生成: 成功 1 个, 失败 0 个, 最新 0 个, 跳过 0 个 =====
```

图 9-43 Visual Studio 2010 改正后程序生成结果

(7) 运行程序。执行“Debug”→“Start Without Debugging”(即“调试”→“开始执行”)或按【Ctrl+F5】快捷键, 自动弹出运行窗口, 如图 9-44 所示。

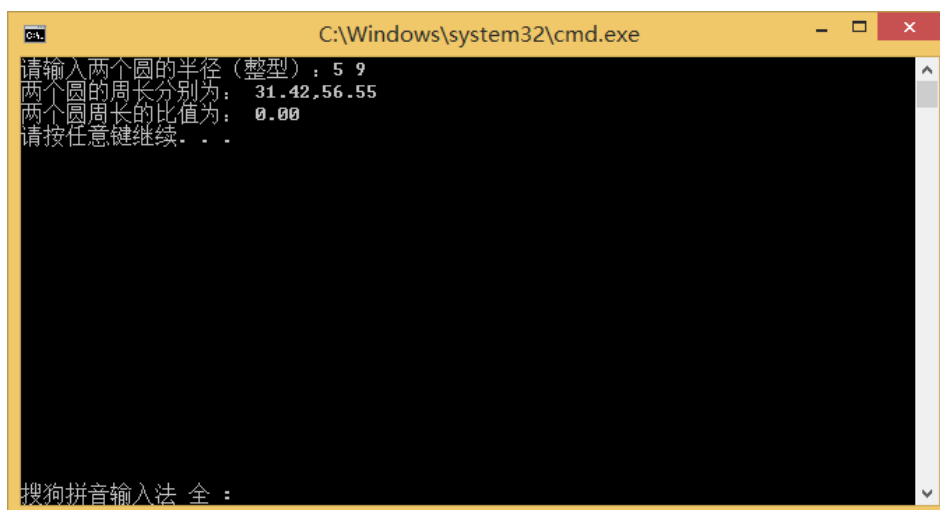


图 9-44 Visual Studio 2010 程序运行结果

#### (8) 调试。

1) 设置断点。点击代码所在行的左边边框或通过鼠标右击，执行“Breakpoint”→“Insert Breakpoint”（即“断点”→“插入断点”）按快捷键【F9】，该行前面出现一个红色的圆点，如图 9-45 所示。

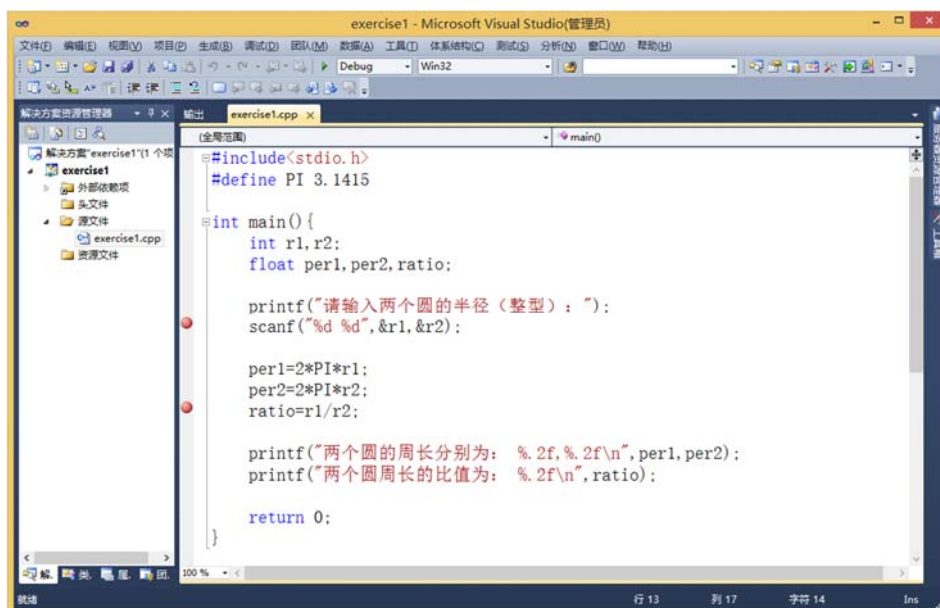


图 9-45 Visual Studio 2010 设置断点

2) 设置 watch 窗口。右击变量并选择“Add to Watch”，则变量的名称和值会出现在程序下方调试框中，如图 9-46 所示。这里对于变量的查看和 9.2.1 节所描述的一样。

局部变量		
名称	值	类型
r2	-858993460	int
per2	-1.0737418e+008	float
r1	-858993460	int
ratio	-1.0737418e+008	float
per1	-1.0737418e+008	float

图 9-46 Visual Studio 2010 设置查看变量

3) 单步调试程序。执行“Debug”→“Start Debugging”（即“调试”→“开始调试”）或按快捷键【F5】，开始调试模式。调试程序开始并运行到断点处暂停下了，此时执行“Debug”→“Step Over”（即“调试”→“单步调试”）或按快捷键【F10】来进行单步调试。执行一次单步调试，输入两个半径的值，按回车键，各变量的值如图 9-47 所示。再执行单步调试，查看变量 ratio 的值，如图 9-48 所示。

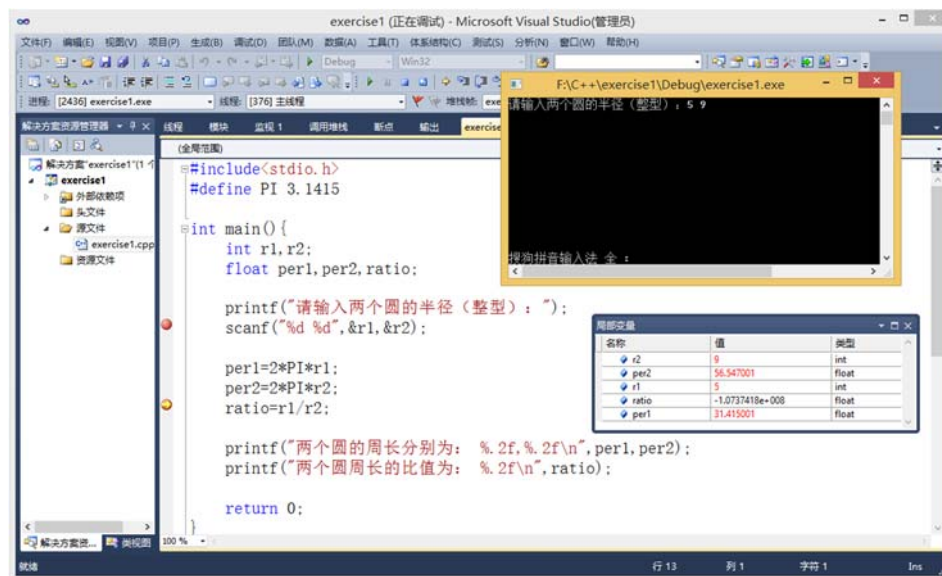


图 9-47 Visual Studio 2010 单步调试一次后变量的值

局部变量		
名称	值	类型
r2	9	int
per2	56.547001	float
r1	5	int
ratio	0.00000000	float
per1	31.415001	float

图 9-48 Visual Studio 2010 查看 ratio 的值

6) 查找程序错误。再执行单步调试。查找错误的过程和 9.2.1 节所描述的一样。修改程序后，结果正确，如图 9-49 所示。

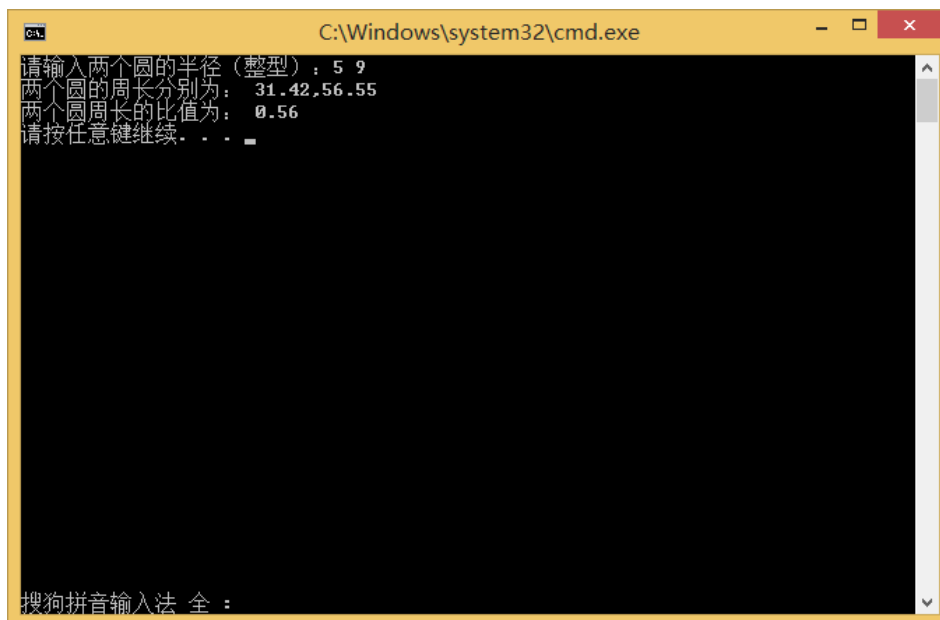


图 9-49 Visual Studio 2010 程序运行结果

### 9.3.5 Dev-C++ 6.0 环境

Dev-C++是一个可视化集成开发环境，可以用此软件实现 C/C++程序的编辑、预处理、编译、运行和调试。下面介绍一下在 Dev-C++环境下编写和运行程序的基本步骤。

(1) 新建工程。启动 Dev-C++，执行“File” → “New” → “Project”（即“文件”→“新建”→“项目”）命令，在弹出的“New”对话框中单击“Basic”选项卡，在列表框中选择 A Console Application 选项，并选择“C 项目”，在“Project name”（即“名称”）文本框中输入工程名称，如图 9-50 所示。

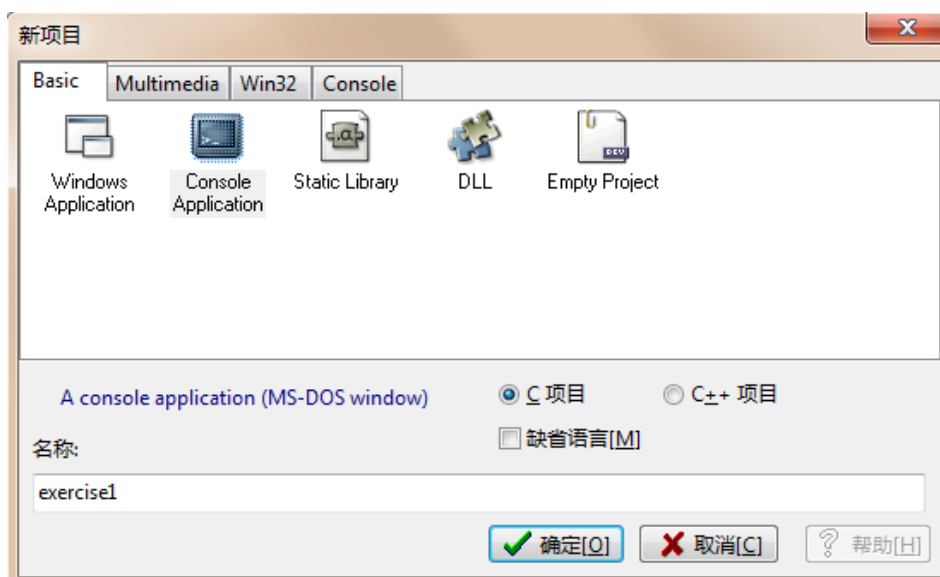


图 9-50 Dev-C++ 6.0 新建项目第一步

(2) 然后点击“确定”按钮，弹出如图 9-51 所示的对话框，点击“Save as”（即“保存在”）文本框右侧的按钮可以改变项目的保存路径，最后单击“保存”按钮。

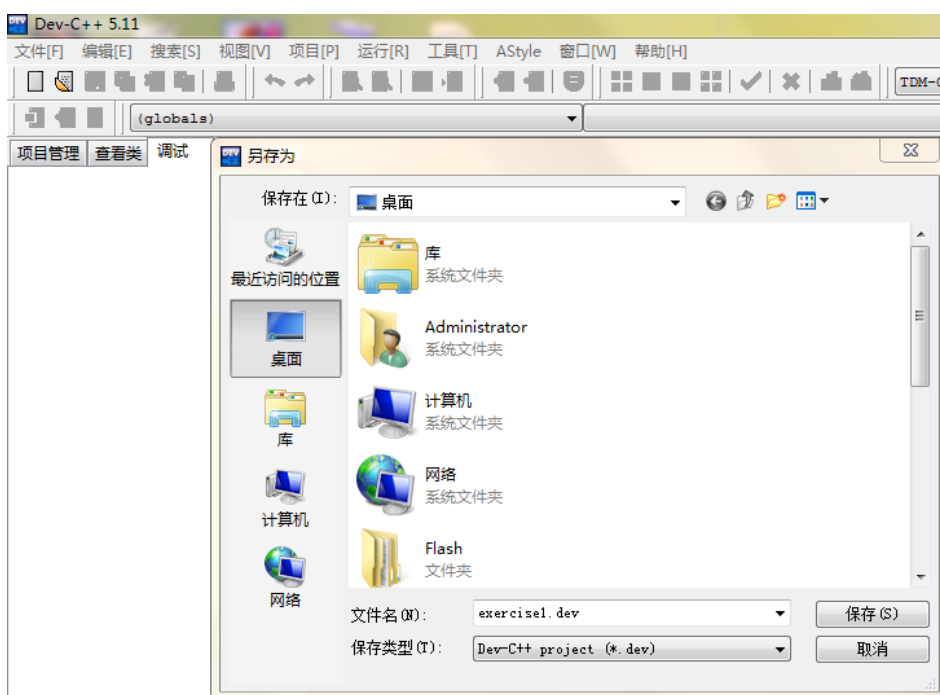


图 9-51 Dev-C++ 6.0 新建项目第二步

(3) 新建、编辑源文件。执行“File”→“New”→“Source File”（即“文件”→“新建”→“源文件”）命令，在弹出的白色区域，可以输入程序，如图 9-52 所示。

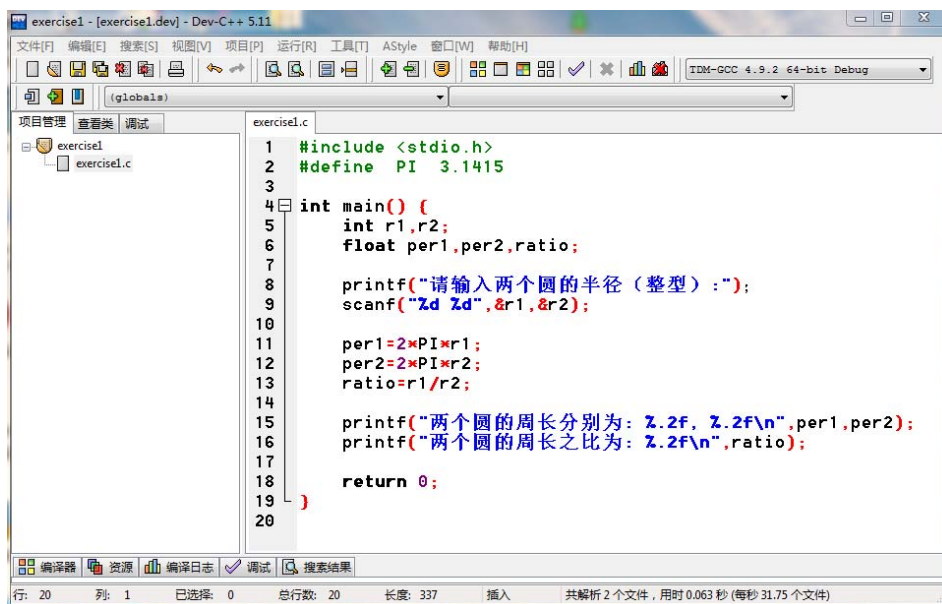


图 9-52 Dev-C++ 6.0 编辑程序

(4) 保存文件。执行“File”→“Save”（即“文件”→“保存”）或按【Ctrl+S】快捷键保存创建的源文件（一般情况下，源文件与工程路径一致）。在“File”（即“文件名”）文本框中选择输入源文件名称，保存类型选择“C++ source files (\*.C)”，意思是保存一个 C 文件，如图 9-53 所示。

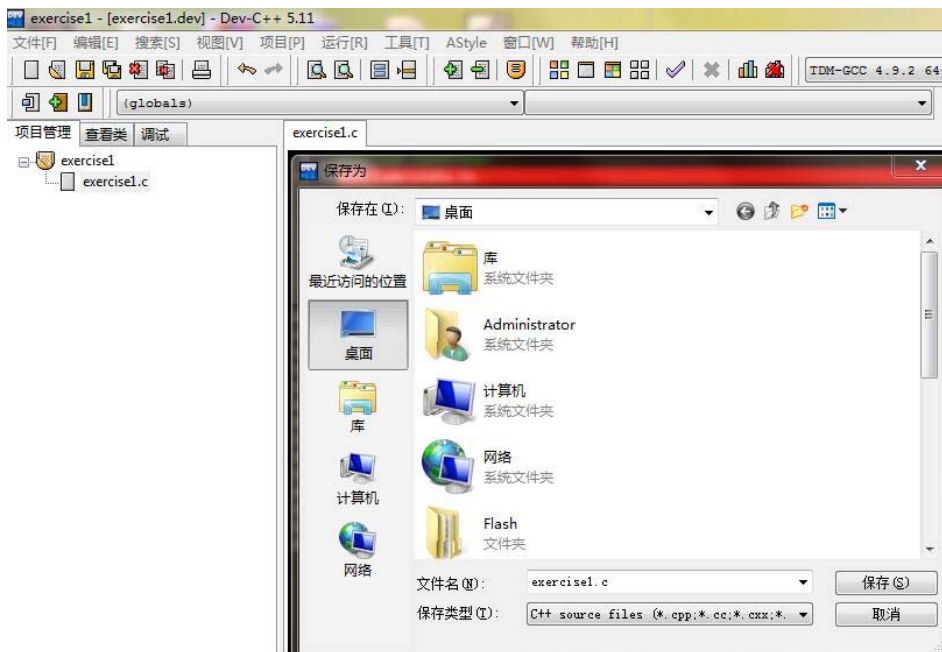


图 9-53 Dev-C++ 6.0 保存源文件

(5) 编译文件。从主菜单选“Execute”→“Compile & Run”（即“运行”→“编译+运行”）或按【F9】快捷键，可以一次性完成程序的预处理、编译及运行过程。

这里先执行“Execute”→“Compile”（即“运行”→“编译”）或按【Ctrl+F9】快捷键完成对源文件的预处理、编译。编译文件完成后在信息窗口显示结果，如图 9-54 所示。

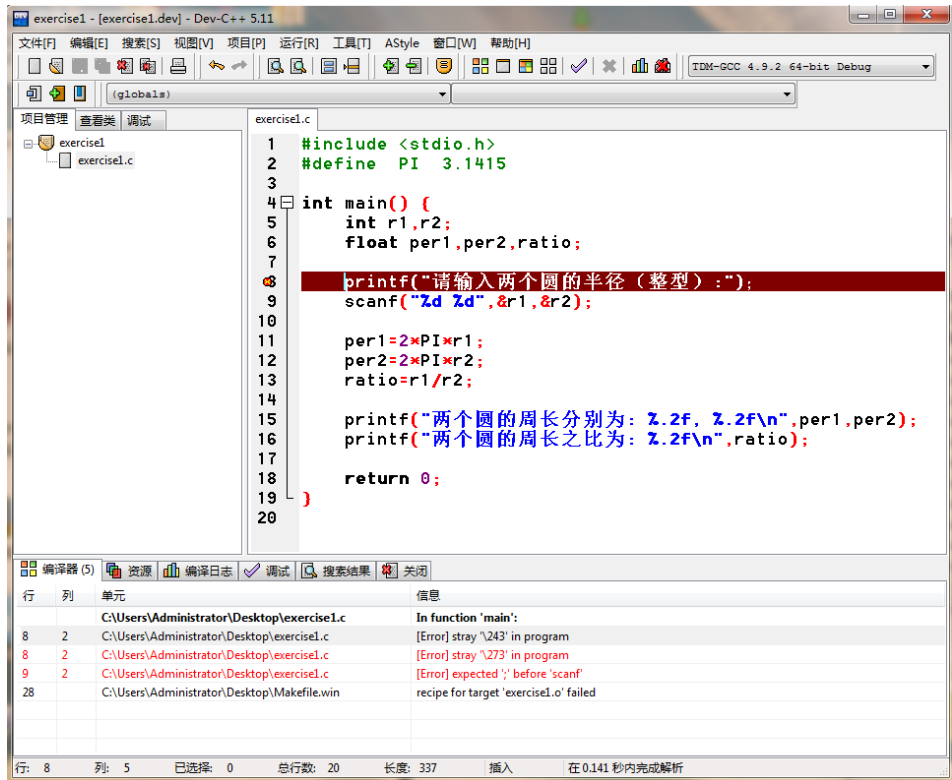


图 9-54 Dev-C++ 6.0 编译源文件

（6）修改和调试程序。如果程序中存在语法等错误，则编译过程失败，编译器将会在屏幕右下角的“Compile Log”标签页中显示错误信息，并且将源程序相应的错误行的底色标为红色，如图 9-54 所示。信息提示存在语法错误，错误原因是 printf 语句后面的分号错误地输入成了中文的分号，改正错误后重新编译，结果如图 9-55 所示。



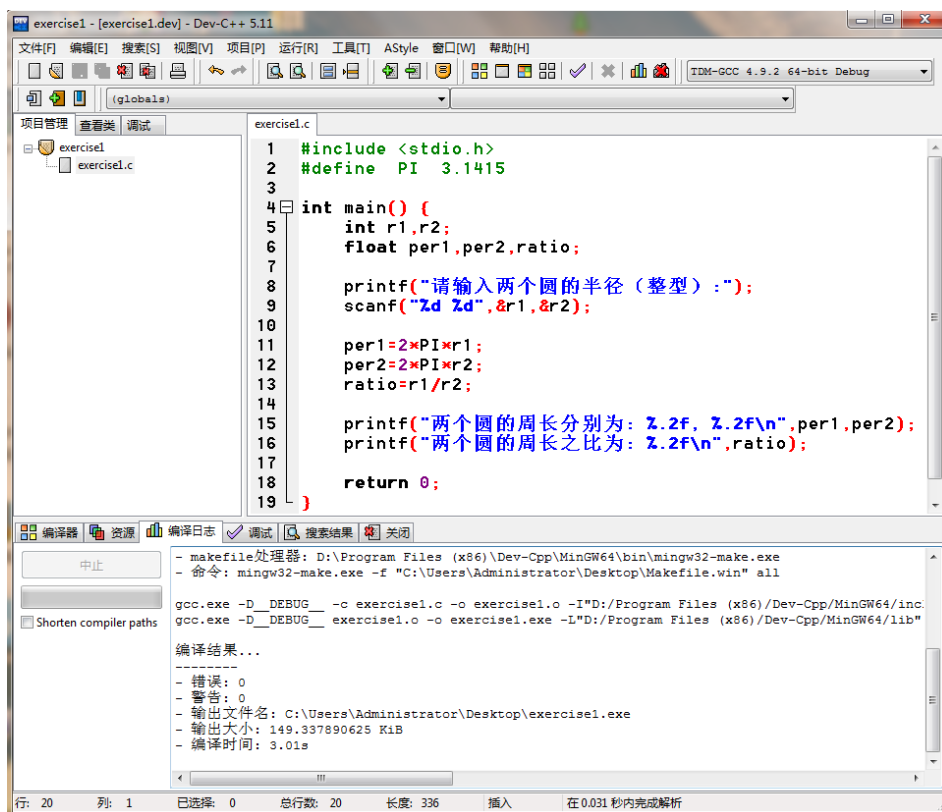


图 9-55 Dev-C++ 6.0 重新编译源文件

(7)运行程序。更正源程序中的代码错误后，编译无错误，然后执行“Execute”→“Run”（即“运行”→“运行”）或按【Ctrl+F10】快捷键，运行源文件。输入两个圆的半径，然后按回车键，如图 9-56 所示。

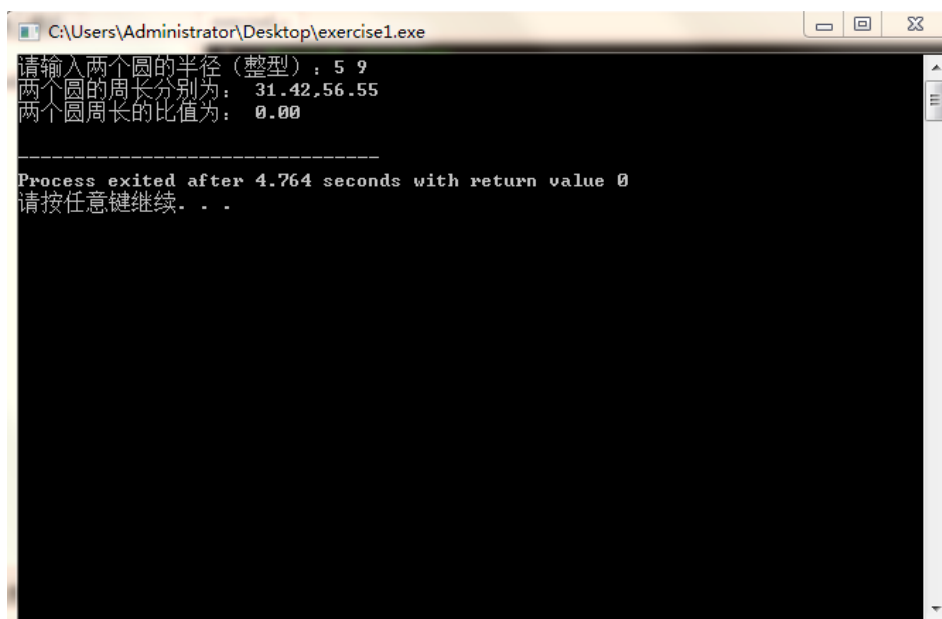


图 9-56 Dev-C++ 6.0 程序运行结果

## (8) 调试

1) 设置断点。在代码所在行行首左键单击，该行将被加亮。默认的加亮颜色是红色。如图 9-57 所示。

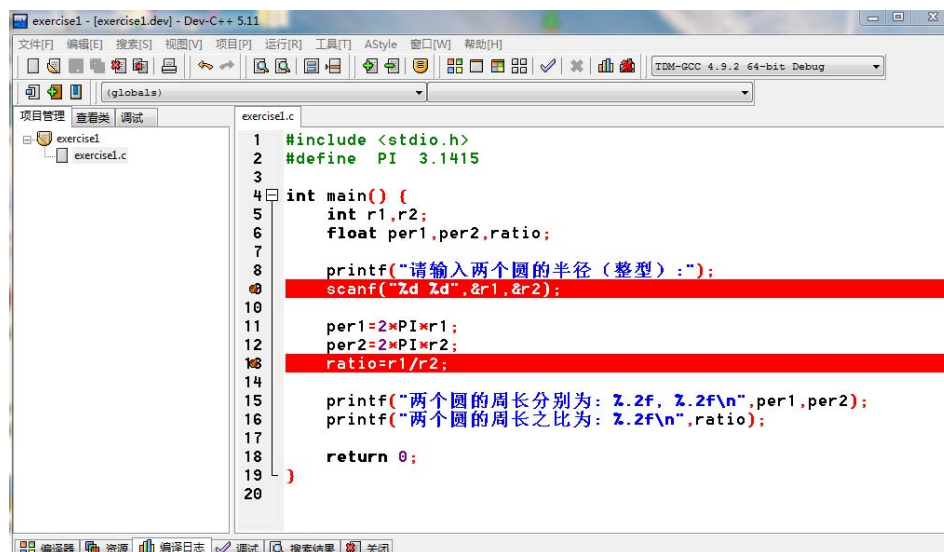


图 9-57 Dev-C++ 6.0 设置断点

2) 调试运行程序。设置断点后，执行“Debug”→“Debug”（即“运行”→“调试”）或按快捷键【F5】，开始调试程序，程序运行到断点处暂停下了，此时我们可以观察程序运行的情况。并且程序下方会出现调试框，如图 9-58 所示。

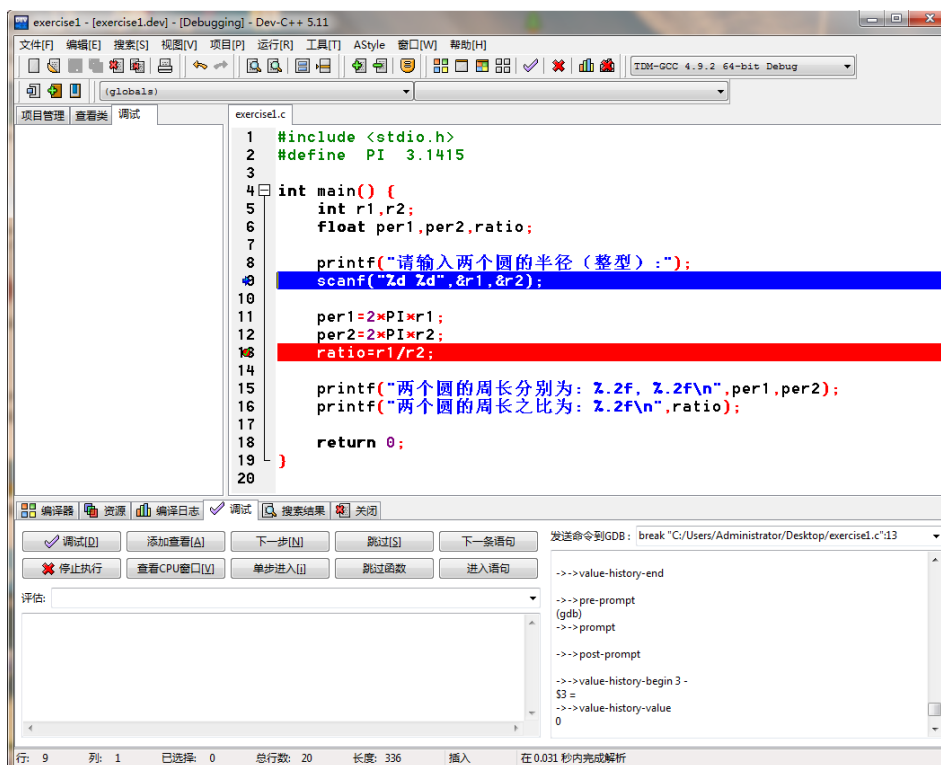


图 9-58 Dev-C++ 6.0 调试运行程序

3) 设置 watch 窗口。通过调试菜单下的添加变量窗口来增加变量，新增的变量将会显示在最左边 Explore 的 Debug 页中，如图 9-59 所示。

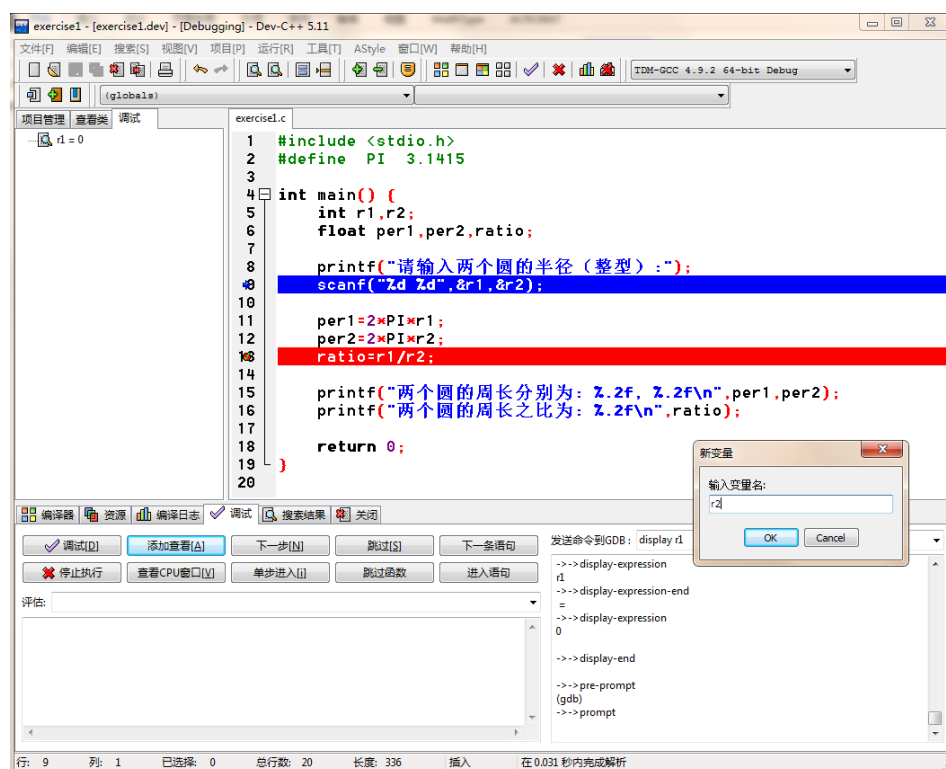


图 9-59 Dev-C++ 6.0 添加查看变量

4) 单步调试程序。通过调试框，执行“下一步”，输入两个半径的值，再执行“下一步”，各变量的值如图 9-60 所示。

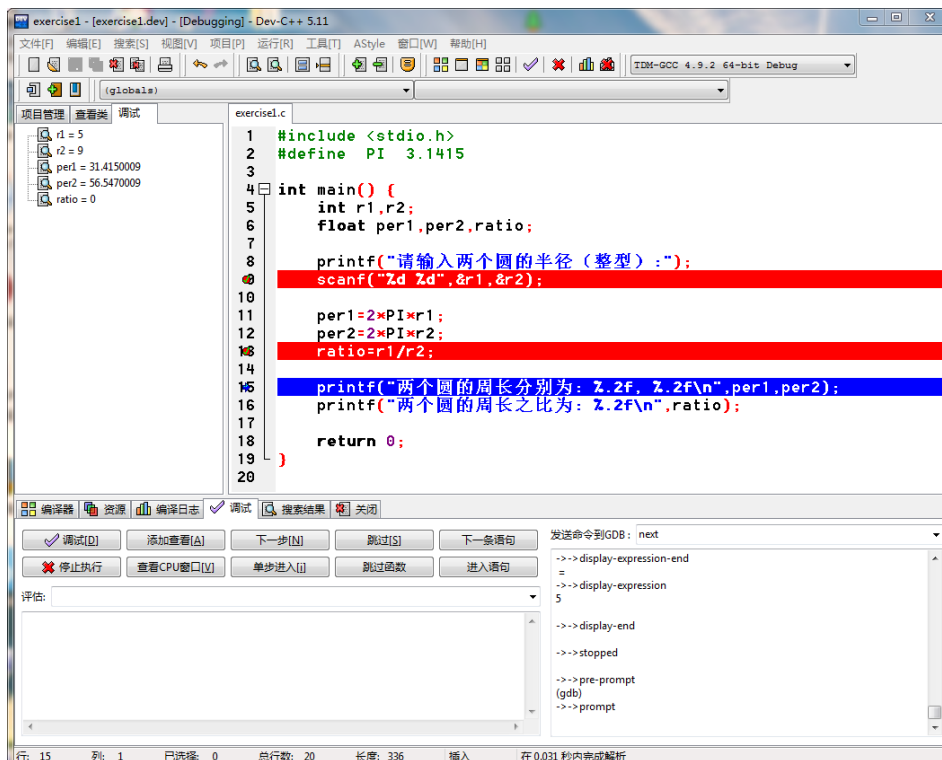


图 9-60 Dev-C++ 6.0 查看各变量的值

5) 查找程序错误。再执行单步调试，查找错误的过程和 9.2.1 节所描述的一样。修改程序后，结果正确，如图 9-61 所示。

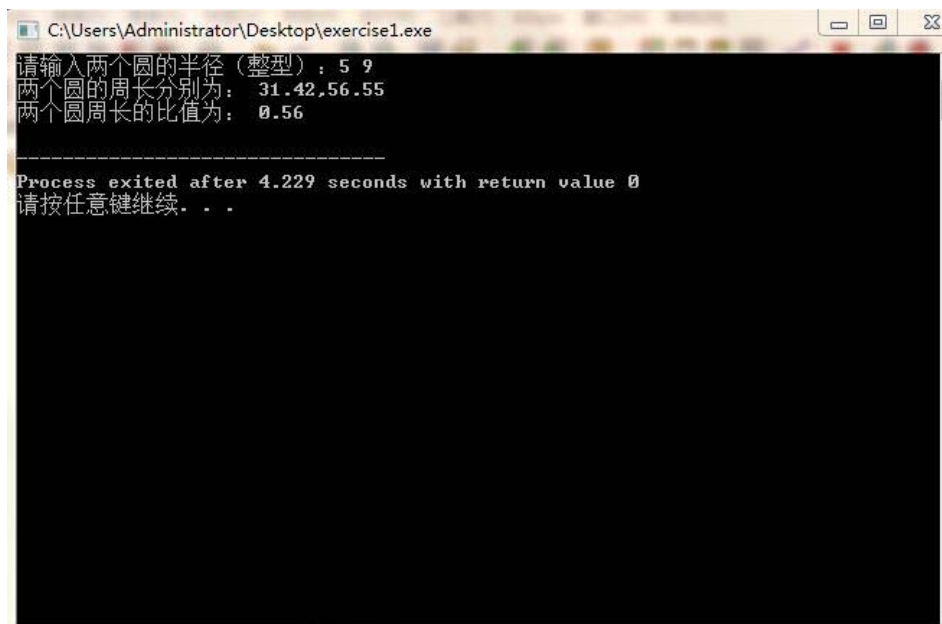


图 9-61 Dev-C++ 6.0 程序运行结果