实验一　Visual Sutdio 开发环境介绍

一、 实验目的

1. 熟悉 Visual Studio 的集成开发环境。
2. 掌握 C++程序开发的一般流程。
3. 掌握 Visual C++中编辑、调试程序的基本方法。
4. 掌握 C++程序的一般结构，完成一个简单的 C++程序。

二、 实验内容

1. C++开发工具介绍

“工欲善其事，必先利其器”。在开始 C++学习之前，我们首先介绍一下相关的开发工具。

在用 C++进行程序开发时，一般按照图 1.1 的流程进行：首先，用编辑器编写相应的 C++源文件；然后用编译器将源文件编译成可执行的二进制代码。

图 1-1 C++程序开发的流程。

C++对于编辑器并没有特殊的要求，你既可以用最简单的记事本来写，也可以用各种专门的编辑器（如 Notepad++、Vim、Emacs 等）。但是，好的编辑器一般会提供语法高亮、自动缩进乃至代码自动补全的功能，可以大大提高编程效率。

C++是一种高级语言，而我们的计算机通常只理解 0 和 1 组成的机器代码，因此必须通过一个“翻译”将 C++源代码转换成机器代码，这个翻译就是编译器。作为一种流行的编程语言，C++有很多成熟的编译器，比较流行的编译器见表 1.1。此外、还有 C++ Builder、Comeau C++等编译器，但因为使用较少这里就不多介绍。

在这些编译器中，Visual C++是 Windows 平台上应用最广的编译器；GCC 是 Linux 和开源社区应用最多的编译器；Clang 是苹果公司系统开发的主要编译器，随着 Mac、iPhone、iPad 的兴起也逐渐为人们所重视；而英特尔的编译器更注重程序的性能，适合于对性能要求较高的应用。

表 1-1 当前流行的 C++编译器比较。

编译器	厂商	支持平台	集成开发环境(IDE)
Visual C++	微软公司	Windows	Visual Studio、Qt Creator 等
GCC	GNU 计划	Linux、Windows、Mac	QtCreator、Code::Blocks、Eclipse 等
Clang	LLVM 计划	Mac、Linux、Windows	Xcode
Intel C++	英特尔	Windows、Linux、Mac	Visual Studio

我们可以在命令行下面调用编译器，将 C++的源文件编译成相应的二进制可执行文件。比如可以在 Windows 下的 CMD 或者其它平台的 Terminal 下输入下面的命令来调用 GCC 编译器来把 main.cpp 编译成可执行文件 app:

```
g++ -O2 -o app main.cpp
```

这里的“g++”是 C++编译器的名字；“-O2”是编译选项，表示编译时优化的程度；“-o app”表示把编译出来的可执行文件输出到文件“app”中；最后的“main.cpp”表示 C++的源文件。这只是一个单文件的 C++程序，当程序规模变大时通常需要编写复杂的 Makefile 来生成编译的规则。如果再涉及到调试将更为复杂。

人们为了减轻编程的工作量、提高编程效率，通常会使用集成开发环境（integrated development environment，简称 IDE）。IDE 通常包含以下几个部分：

- 源代码编辑器。提供代码缩进、语法高亮、自动补全等功能，方便编写源代码。
- 自动构建工具。自动调用编译、链接器将你的 C++程序编译成二进制可执行程序。
- 调试器。提供可视化的调试手段，方便程序调试。

常见的 C++ IDE 有 Visual Studio、Xcode、Qt Creator、Eclipse CDT、Code::Blocks、Dev-C++等，这些 IDE 一般都提供了丰富的功能，能大大提高你的编程效率。

2. 熟悉 Visual Studio 集成环境

Visual Studio 是 Windows 平台上使用最广的开发环境，除了 C++之外它还支持 VB.NET、C#、F#等其它语言，本节将对其进行简单介绍。

Visual Studio 本身是一个商业软件，但是微软提供了免费的 Express 版本供学习使用。虽然 Profession 版本提供了更多的功能，但对于本课程的学习而言 Express 版本就足够了。本书写作时 Visual Studio 2010 Express 是最新版本，建议大家使用该版本进行开发。

启动后 Visual Studio 的初始界面如图 1-2 所示，你可以查看一下各项菜单以及工具栏，熟悉一下各项功能及配置。

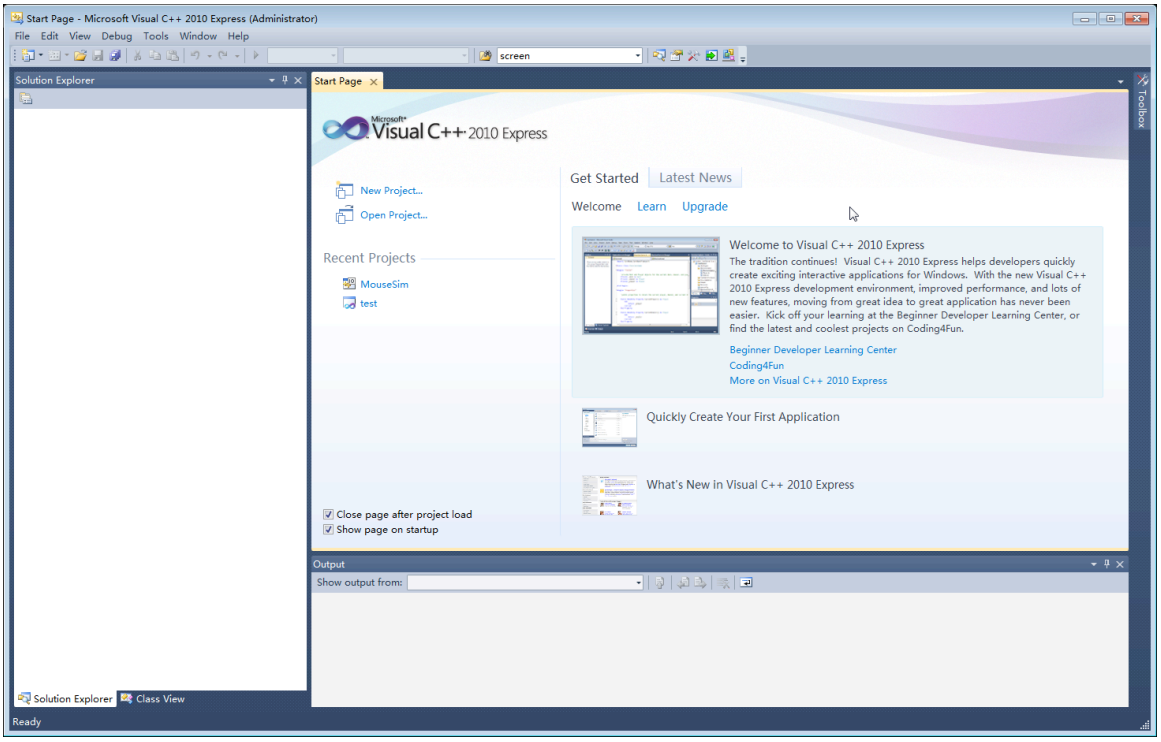


图 1-2 Visual Studio 启动初始界面。

3. 完成一个 Hello World 程序，熟悉 C++开发流程

在计算机领域，人们学习每种编程语言基本上是从“Hello World”开始的，我们现在就利用 Visual C++写一个控制台上打印“hello, world”的程序。

1) 新建一个 C++的工程。打开 Visual C++后，选择菜单【File】→【New】→【Project...】，或者用快捷键“Ctrl+Shift+N”（图 1-3）。
2) 选择工程类型。Visual Studio 支持多种工程，每种工程都有不同的作用，对于我们的 C++控制台程序，选择 Win32 下面的“Win32 Console Application”（图 1-4）。特别注意的是，如果工程类型选择的不对，在编译的时候就有可能出问题。在对话框下面，“Name”是工程的名字，这里我们把它叫做“hello“，而“Location”是工程要保存在什么位置（比如 D:\cpp），填好后按“Ok”。

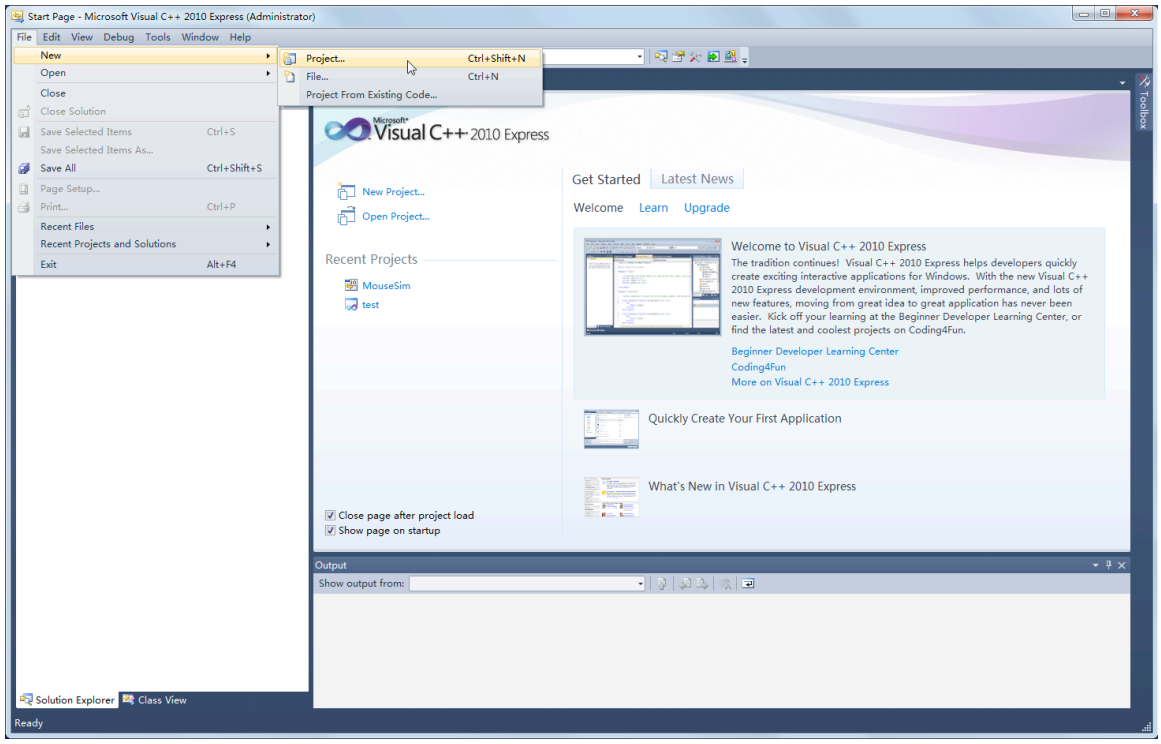


图 1-3 新建工程。

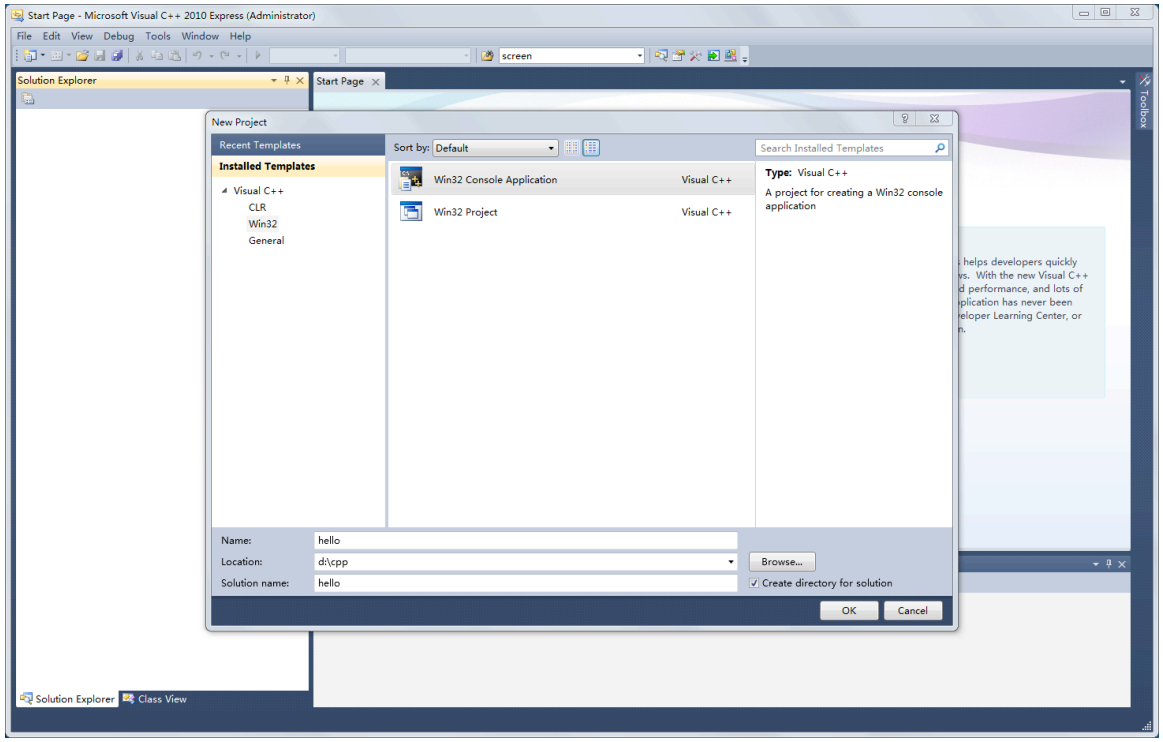
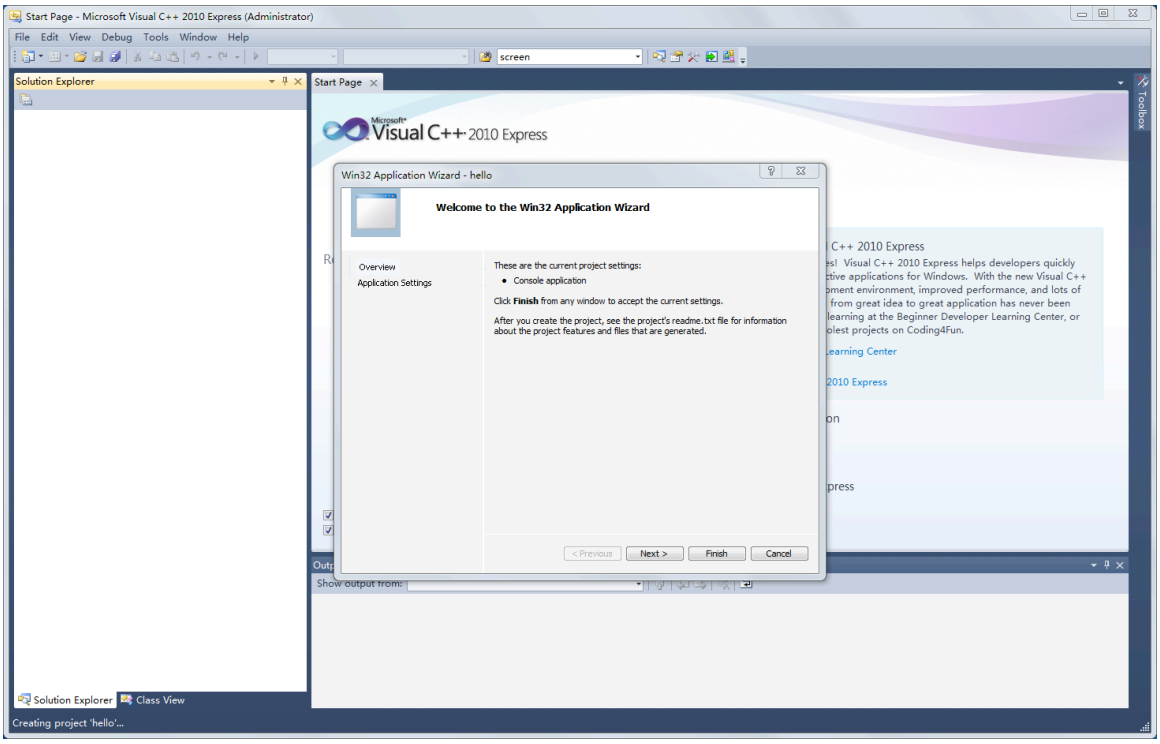


图 1-4 选择工程类型。

3) 工程设置。上一步完成之后，会跳出一个工程设置的对话框，如果无需其它设置直接点“Finish”即可；不过，一般来说我们都需要进行一些额外的设置，这时候点击“Next”。在接下来的对话框中，确保上面选择的是“Console application”，表示我们要创建一个控制台程序；下面的选项中勾上“Empty project”，表示我们要创建一个空的工程（图 1-5）。



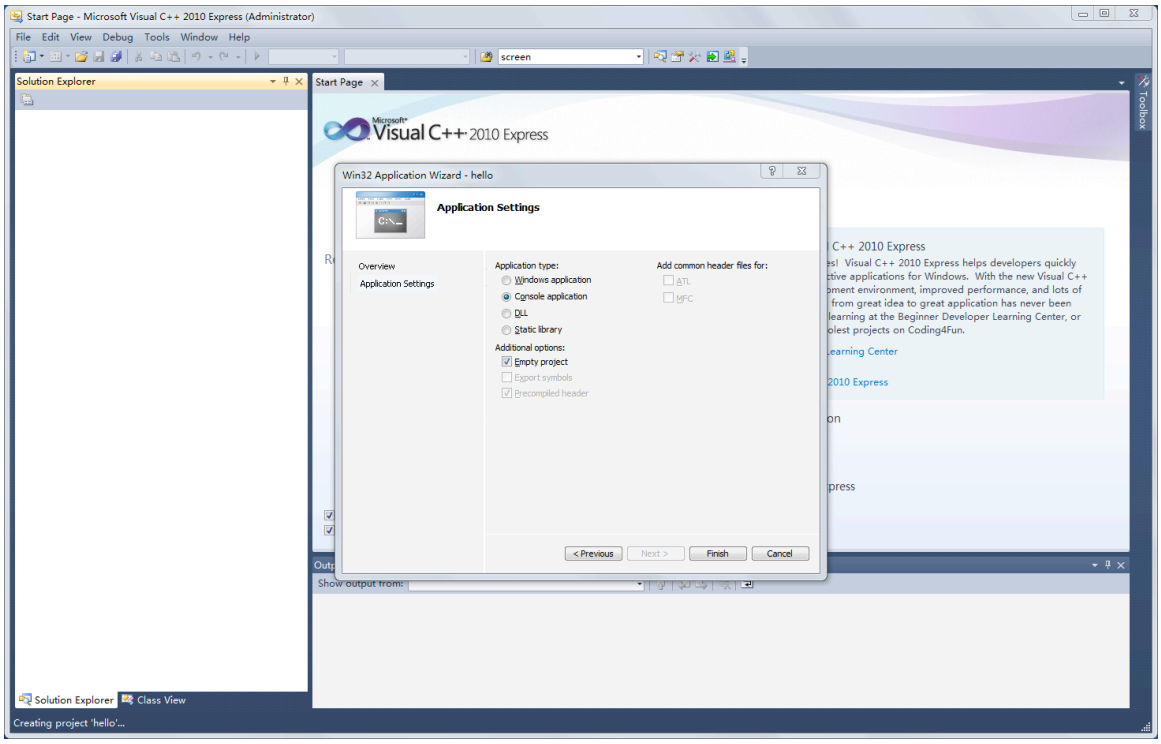


图 1-5 工程设置。

4) 添加文件。上述步骤创建了一个空的工程。要使得工程能够正常运行，必须向工程中添加源文件。右键点击工程名（即 hello），在弹出菜单中选择【Add】→【New item...】；在弹出的对话框中选择“C++ File”，并在下面的“Name”中填写文件名（这里是 main.cpp），在“Location”中填写该文件的存储地址（默认即可）（图 1-6）。需要注意的是，C++的头文件一般和 C 语言相同，都是以“.h”结尾，但是 C++的源文件一般是以“.cpp”结尾的。

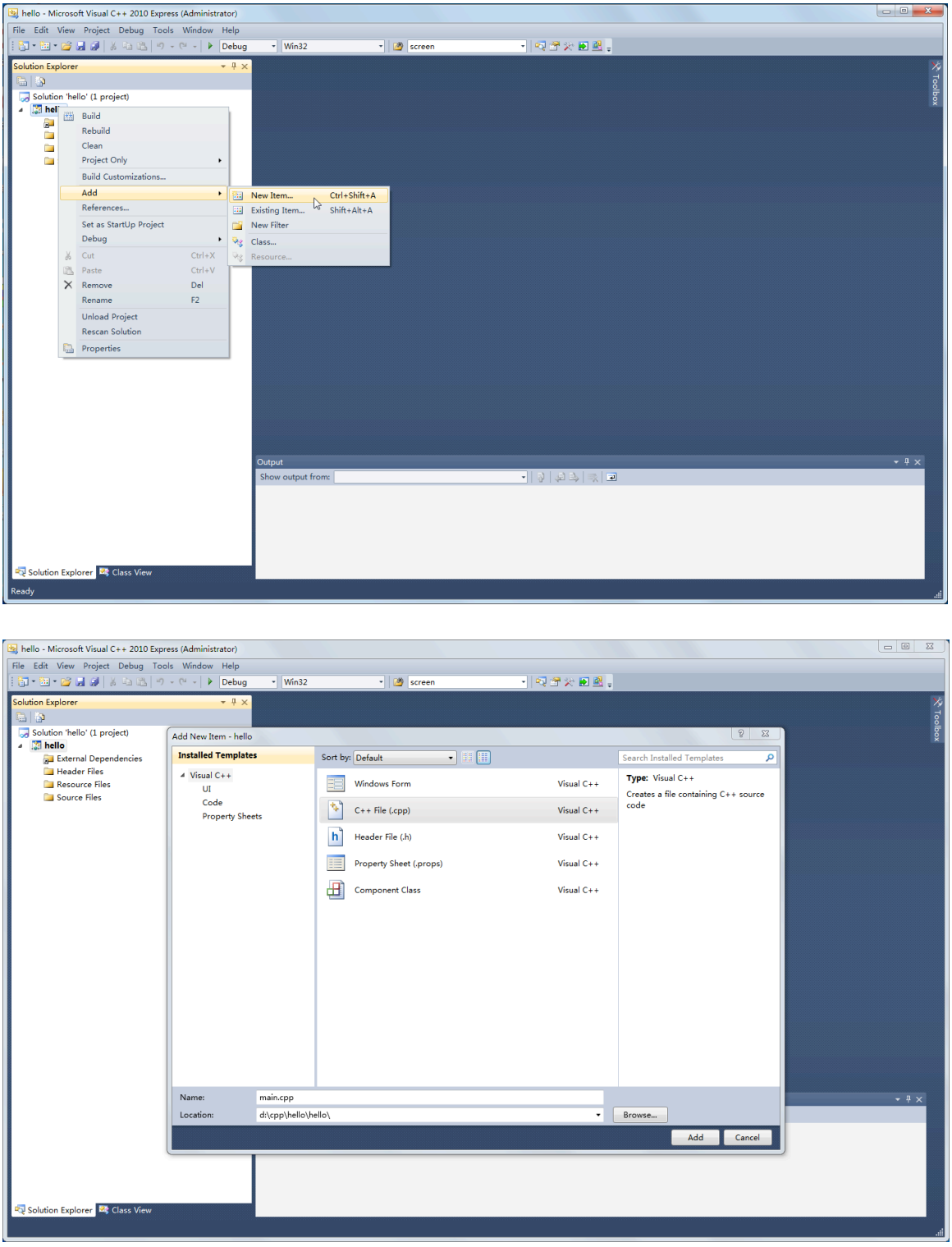


图 1-6 添加源文件。

5) 编写源文件。在新建的文件中输入以下程序，并保存（注意在 Visual C++中未保存部分前面会以黄色显示，已保存部分用绿色显示）：

```
// 文件: helloworld.cpp

#include <iostream>

using namespace std;

int main()
{
    cout << "hello, world" << endl;
    return 0;
}
```

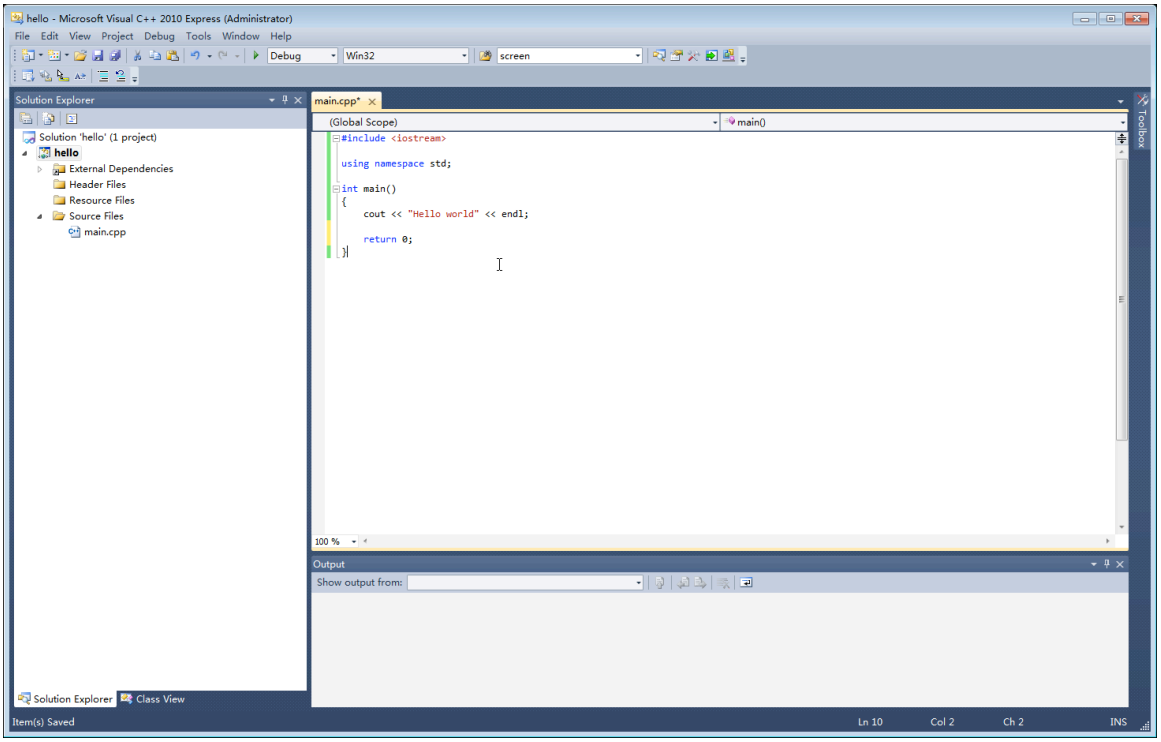


图 1-7 源代码编辑

6) 编译、构建工程。选择菜单【Debug】→【Build Solution】，或者按快捷键“F7”编译工程，生成相应的二进制文件（图 1-8）。

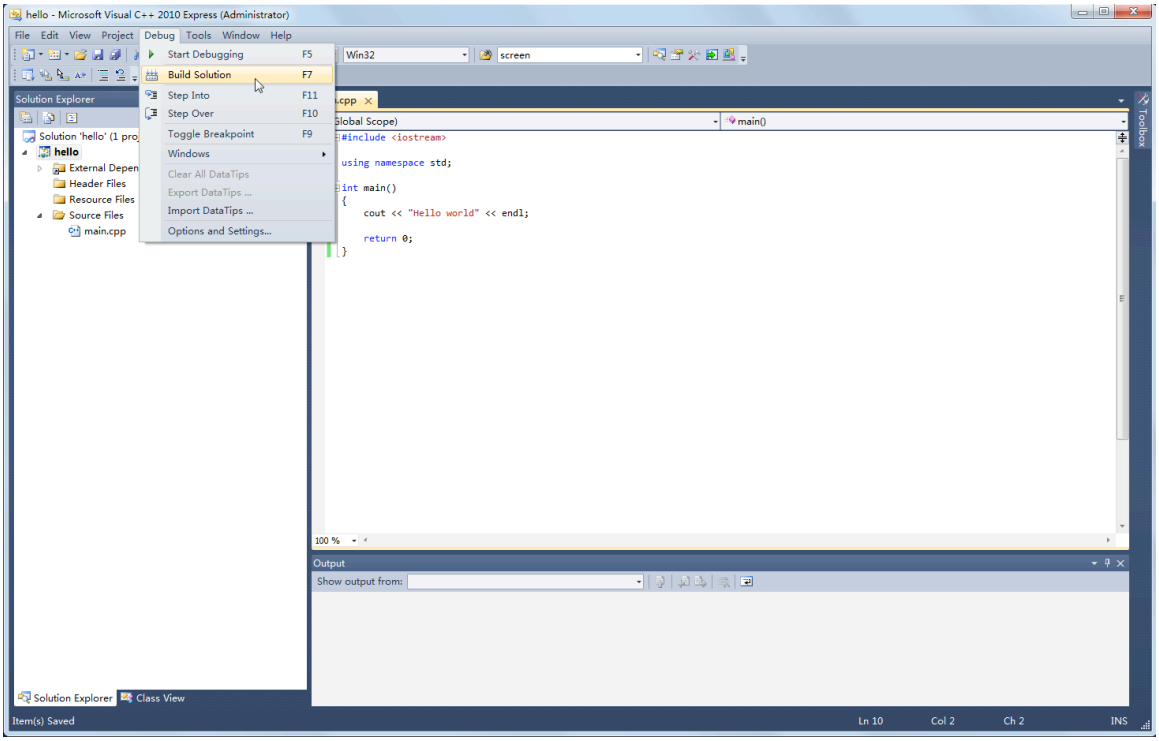


图 1-8 编译工程

7) 运行程序。用快捷键“Ctrl+ F5”可以直接运行程序（图 1-9）。

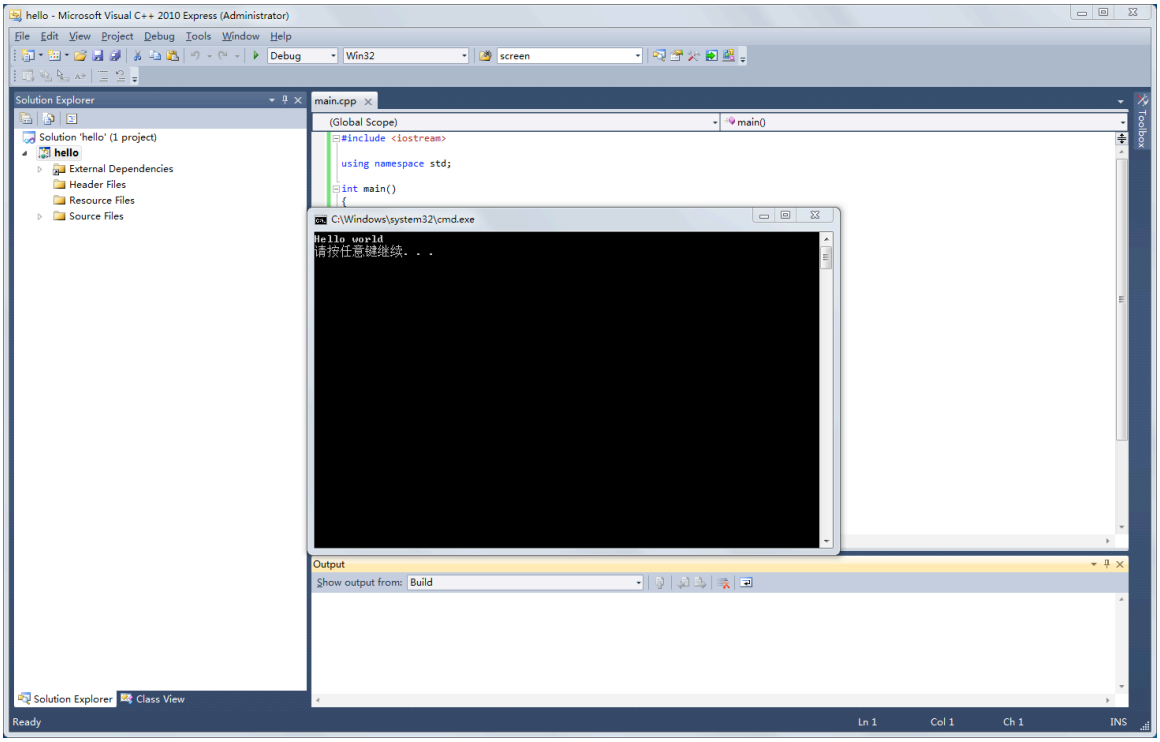


图 1-9 运行程序

8) 程序的相关文件位于工程目录下（见上面第 2 步），可以在该目录下找到程序的工程文件（以.vcxproj 结尾的文件）、源文件（以.h 和.cpp 结尾）以及可执行程序（位于 Debug 或 Release 目录下）（图 1-10）。

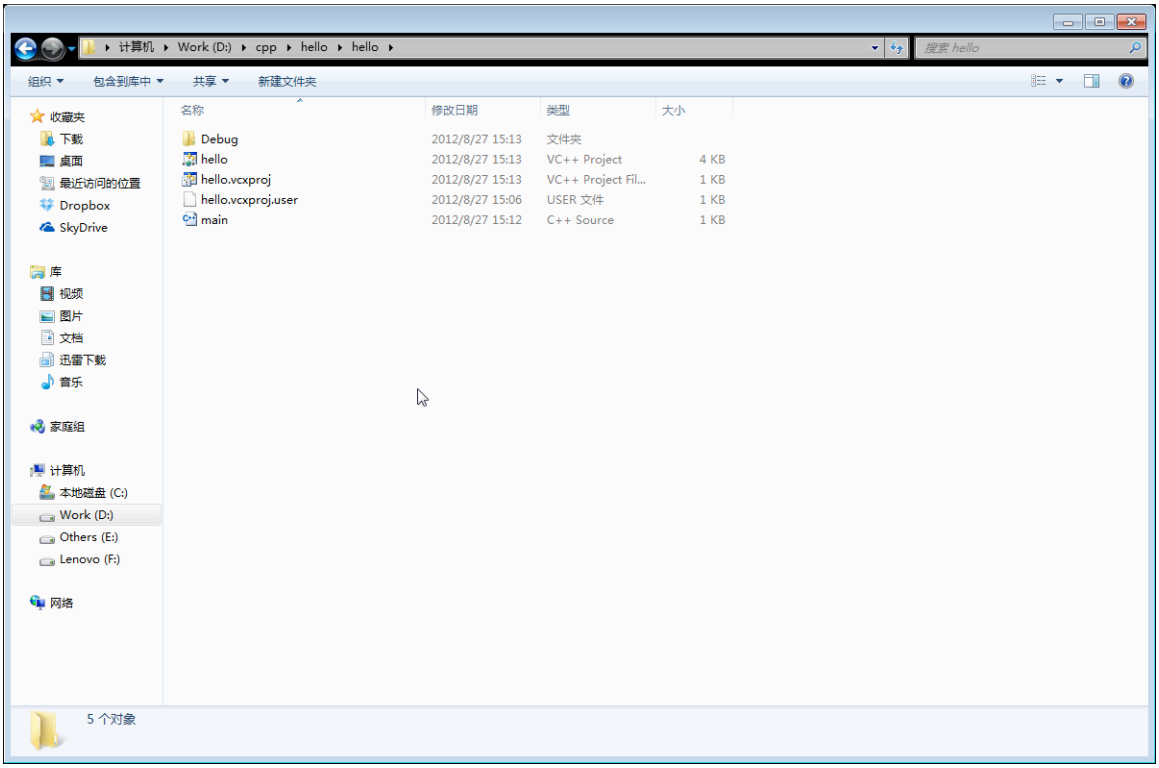


图 1-10 工程目录

到此为止，我们完成了第一个 C++程序，其它程序的开发流程与之类似。在编程过程中，要不断总结、不断分析，才能进步。对于 Visual C++集成环境，我们只讲了其中很小一部分，要充分利用其功能必须进行进一步的学习。此外，Visual C++的菜单上都有其相应的快捷键，熟记这些快捷键也有助于提高你的编程效率。

4. 程序调试

在编程过程中，程序出错是在所难免的，如果错误出在编译阶段，可以根据编译器给出的出错信息来改正错误；然而，如果运行中出错，就需要调试技术了。调试是程序员必须掌握的一门技术，否则就很难把程序中的错误改正。

在进行调试之前，首先要对程序进行仔细分析，推测出程序可能出错的位置以及涉及到的变量，然后对这些变量的进行跟踪查看，看其值的变化是否合理。

调试的最基本也是最重要的方法就是打印输出（用 printf 或者 cout），即在程序运行过程中打印出相关变量，看其值是否正确。如果程序较长，可以采用折半检查的方法，即先检查变量的值在程序中间是否正确：如果中间的值正确，则错误应该在程序后半部分；如果中间的值不正确，则错误应该在前半部分。然后按照类似的方法，对程序的错误部分再进行二分查找。

Visual C++本身也支持调试，它以可视化的方式来监测变量的值，使得调试更加简单。下面我们就以一个程序为例，说明一下 Visual C++中基本的调试方法。

首先我们先新建一个工程，并输入如下计算 1 到 100 求和的代码，并确保你的工程处于调试模式：

```
// 文件: sum.cpp

#include <iostream>

using namespace std;

int main()
{
    int sum = 0;

    for (int i = 1; i <= 100; ++i)
        sum += i;

    cout << sum << endl;

    return 0;
}
```

1) 设置断点。所谓断点，就是程序运行的过程中可以暂停下来的点，比如我们要想在 sum += i 这一行停下来查看 sum 和 i 的值，则先把光标放在这一行，然后选择菜单【Debug】→【Windows】→【Breakpoints】，或者按快捷键“F9”，设置断点之后，在该行前面就会有一个红色圆点（图 1-11）。在该行再次按“F9”就会取消断点。

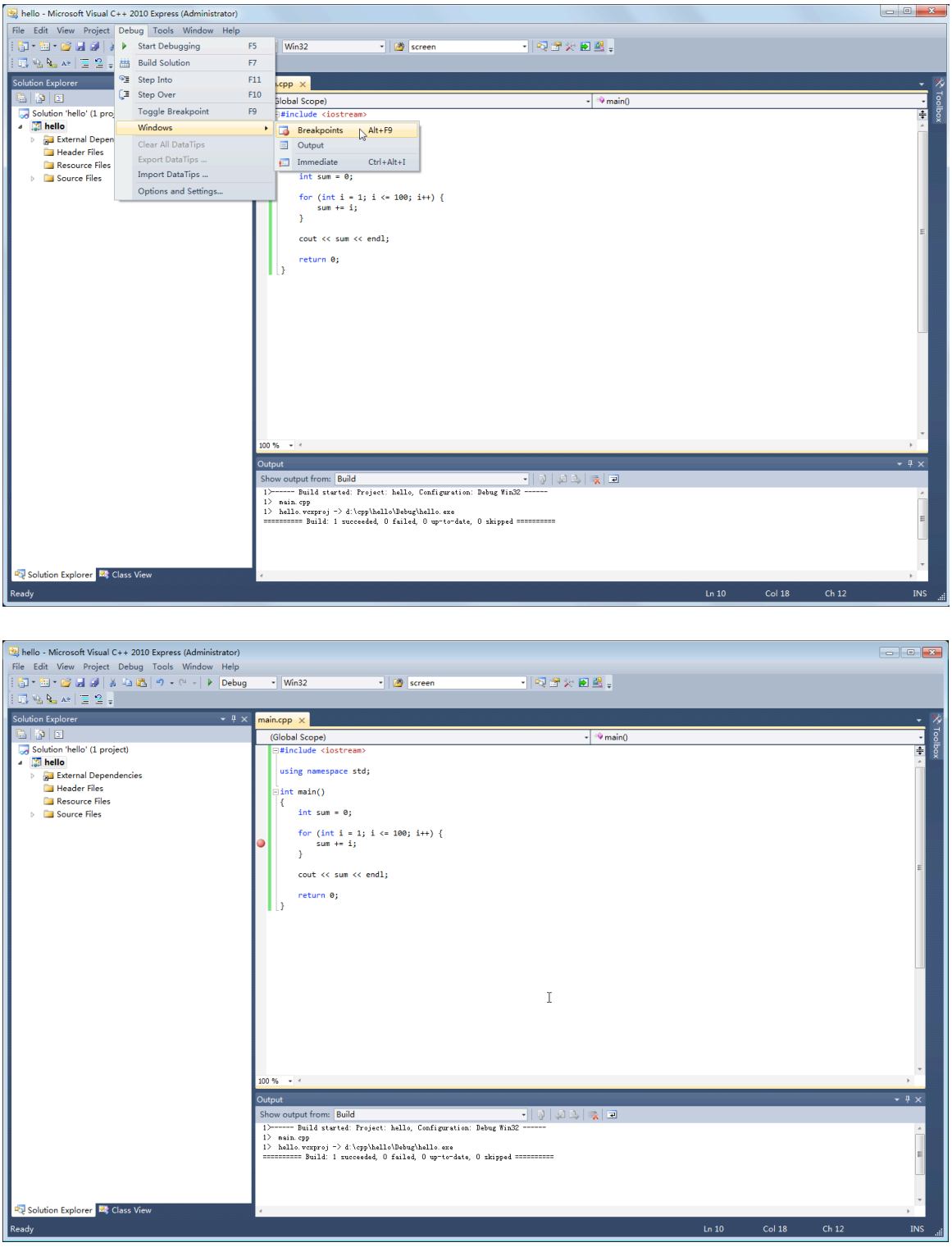
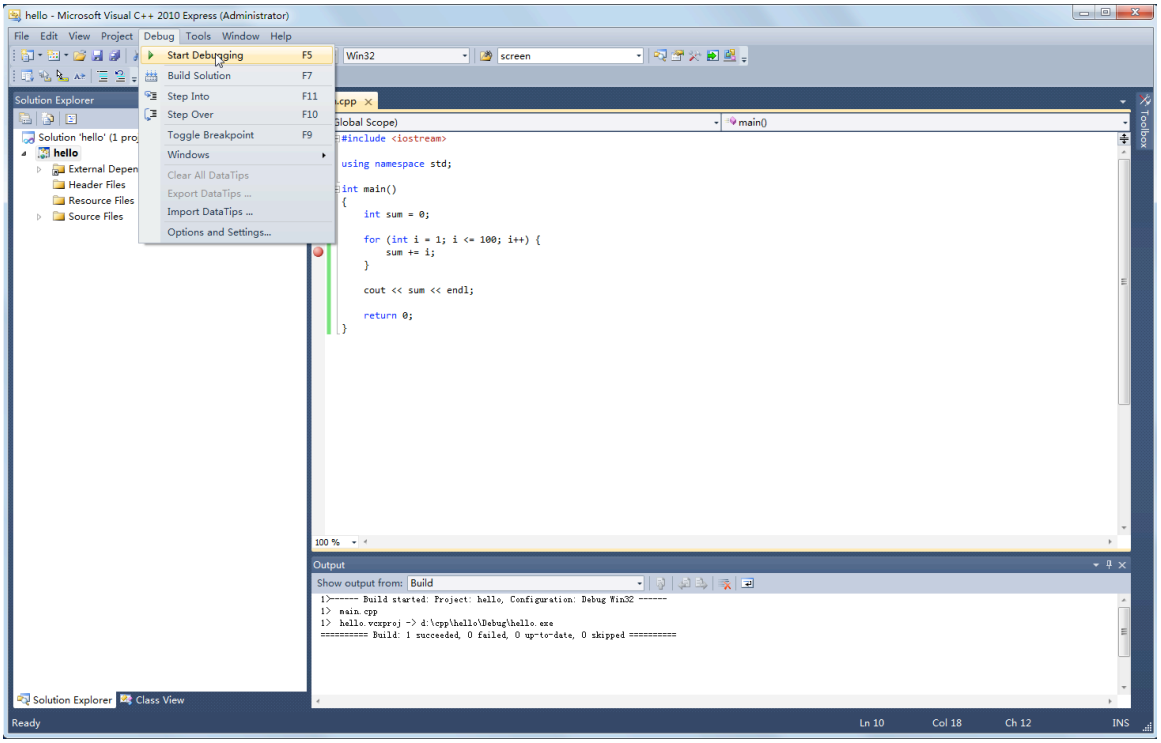


图 1-11 设置断点

2) 开始调试。选择菜单【Debug】→【Start Debugging】，程序就会开始运行，并停在断点处（图 1-12）。此时我们可以开到一个黄色的箭头，它表示下一条要运

行的语句。在 Visual C++的下方有两个窗口，左边是当前变量的值，右边是函数调用的栈，这些窗口的位置都是可以调整的，还可以在 Watch 窗口中添加变量以查看其值的变化。



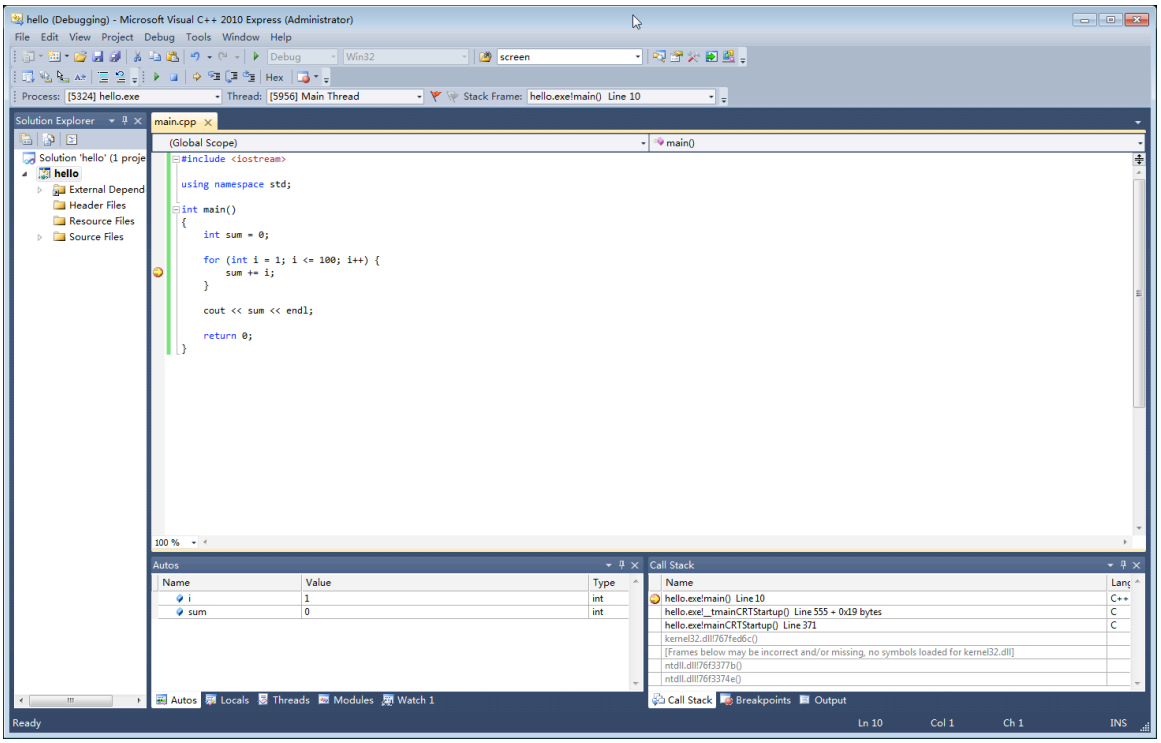
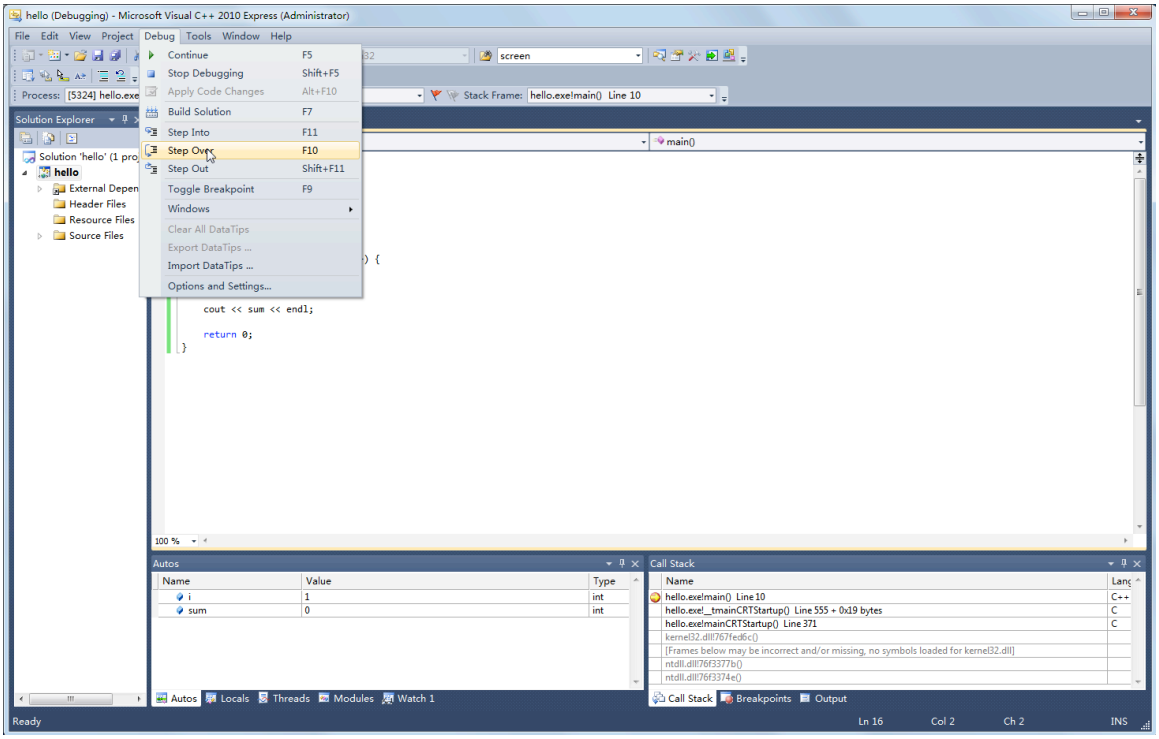


图 1-12 开始调试

3) 单步运行。有时候我们需要一步一步来运行程序以查看变量值的变化，这时候可以选择菜单【Debug】→【Step over】，或者快捷键“F10”，这时候我们就会发现黄色箭头向下移动一步，同时下方左侧窗口的变量值发生了改变（图 1-13）



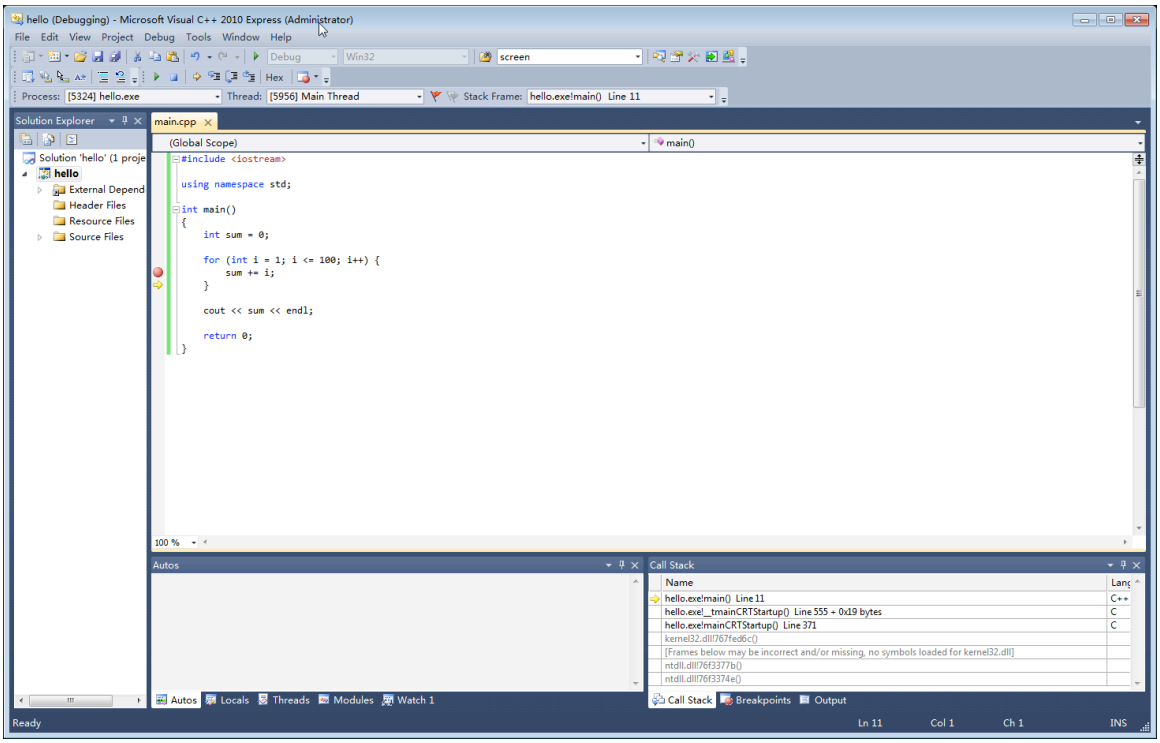


图 1-13 单步运行

4) 结束调试。通过以上步骤发现错误后，可以随时退出调试，选择菜单【Debug】→【Stop debugging】，或者快捷键“Shift + F5”即可退出调试模式（图 1-14）。

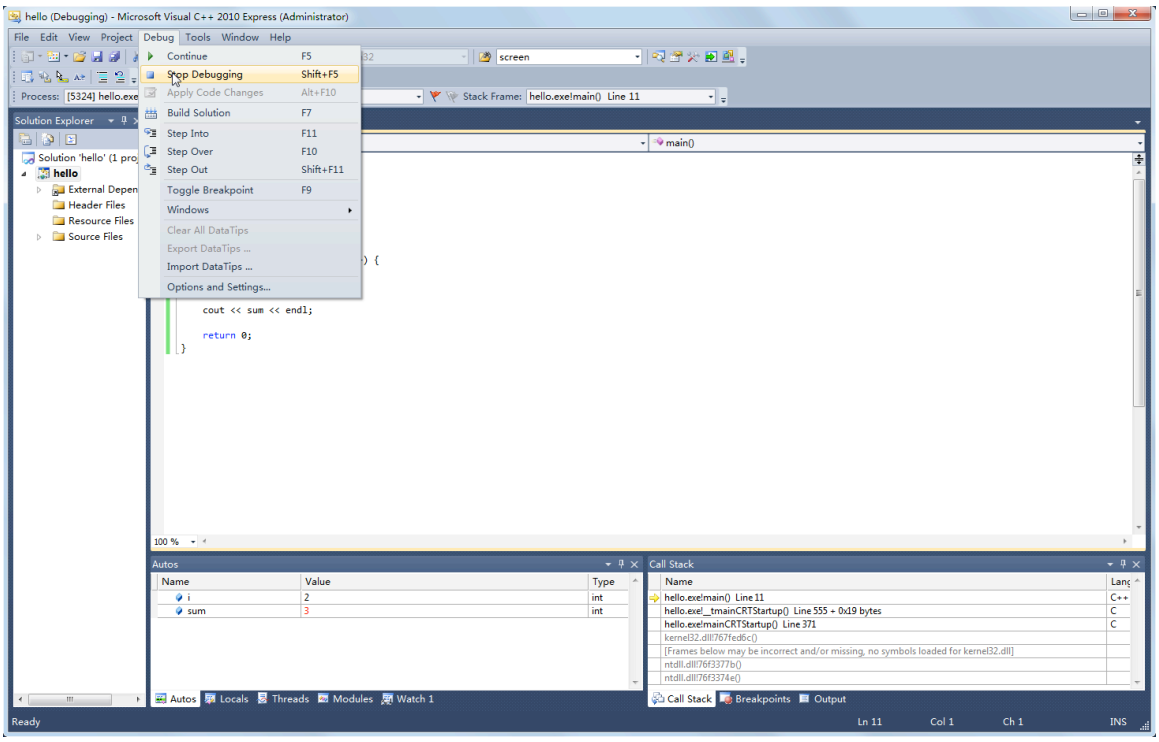


图 1-14 结束调试

三、 上机思考题

1) 在调试时，如何进入一个函数？如何查看一个变量的地址？如何查看一个数组的内容？
2) 用 C++的方式扩展 Hello World 程序，使其能够处理用户输入，比如：

   What's your name?

   Tom

   Hello, Tom!

   其中加下划线的第二句是用户输入。
3) 使用 Visual C++的帮助系统，将光标放置在需要帮助的代码位置，按“F1”。思考一下，当你编程遇到问题时，有哪些解决方案？