

Visual C++ 2010 Tutorial

Fall, 2011

Table of Contents

	Page No
Introduction	2
Single file program demo	3
• How to launch Visual C++ 2010	3
• Start a new project	4
• How to add a new source file to a project	6
• How to compile and execute a program	8
• How to Print the Source Code and Output	9
Multiple-File Demo	10
Appendix: demo program source code	15

Program Development with Microsoft Visual C++ 2010

Introduction

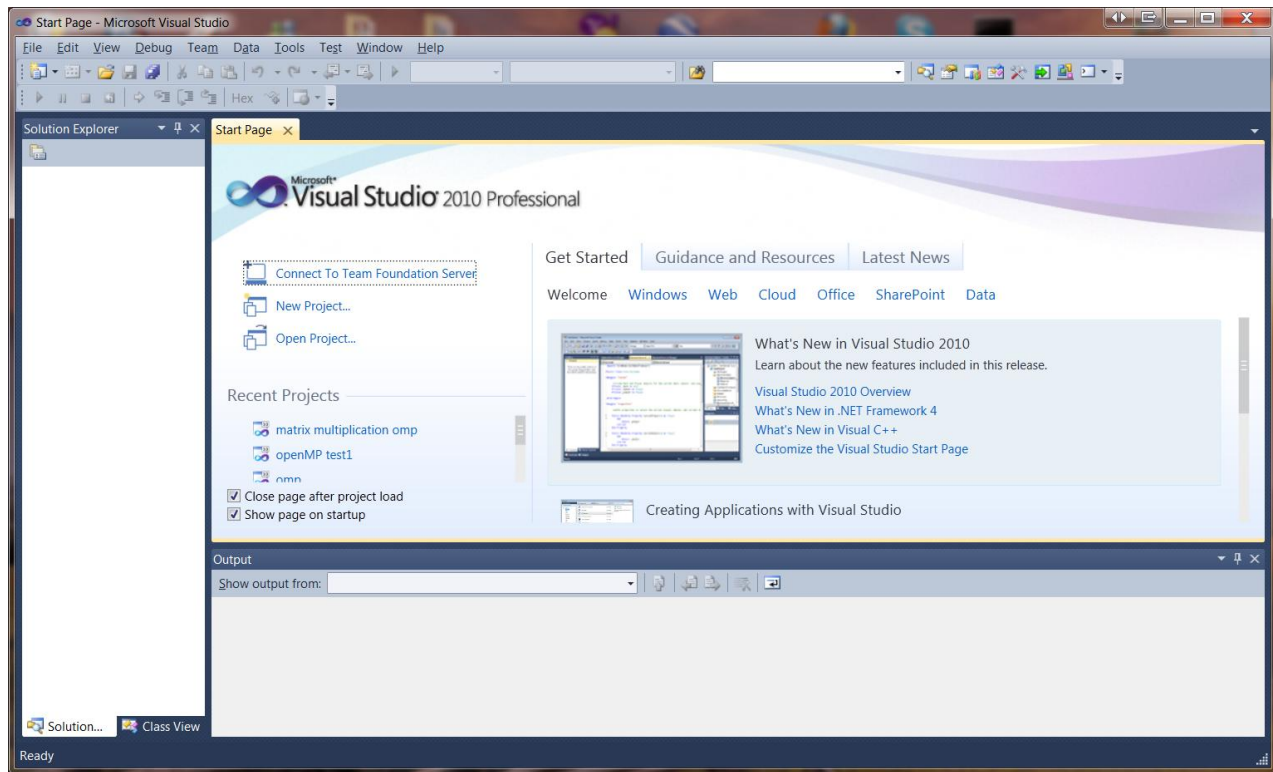
This tutorial is written to help those who are new to Visual C++. It introduces the **Integrated Development Environment (IDE)** of Microsoft Visual C++ 2010 and shows how to enter, edit, save, retrieve, compile, link, and run a C++ program in such an environment. It also shows a simple way to print the source code and the program output.

Two simple C++ projects, one single-file and the other multi-file are used as demo programs so that students can focus on Visual C++ 2010 IDE instead of getting distracted by language features. For the convenience of the reader, source code for both programs is listed in the Appendix.

If you have any comments and/or suggestions that might help improve the effectiveness of this tutorial, please forward them to either Dr. E. Hu at hue@wpunj.edu or Marvin Kiss at kissm@wpunj.edu.

Single-file Project: The *Hello World* Program

Step 1: Launch the MS Visual C/C++ 2010 software from task bar. The main window of Visual Studio 2010 should be similar to the below display:

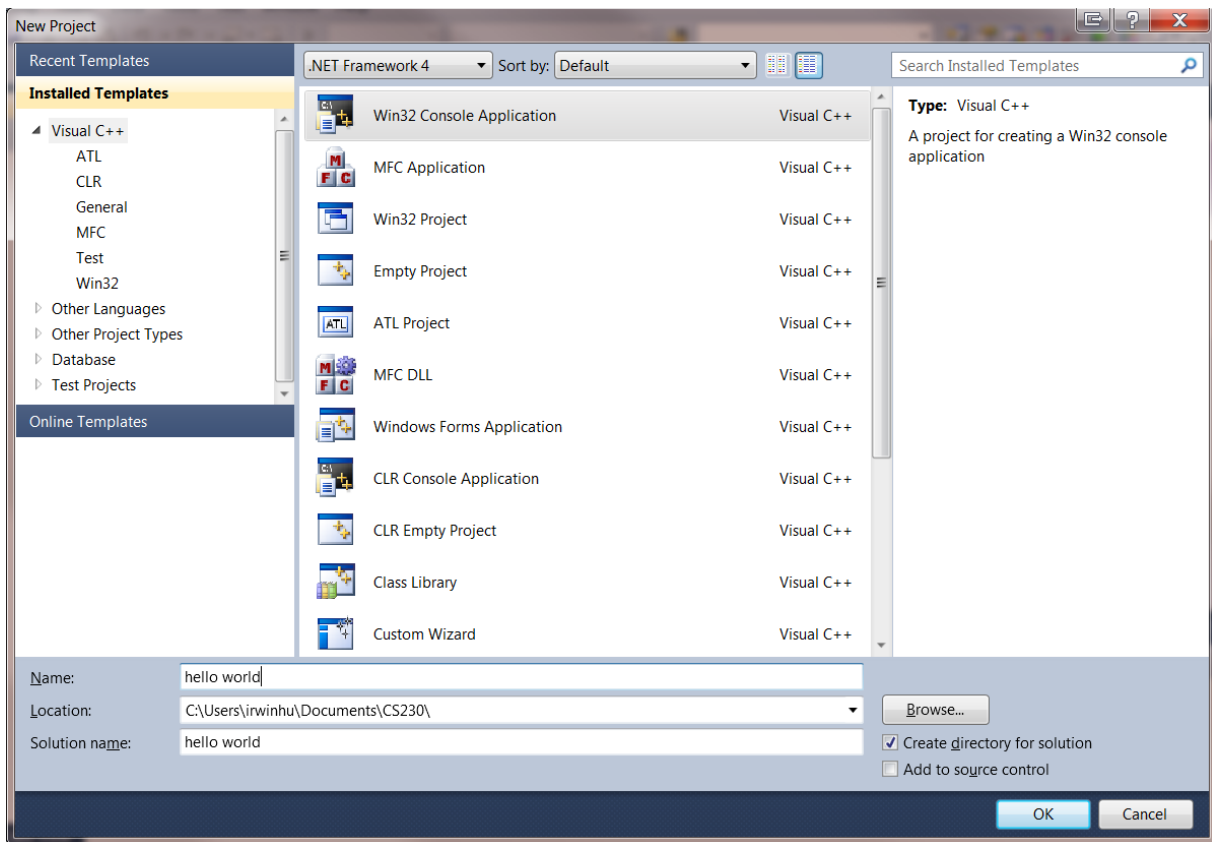


Note:

1. Hereafter, all system defined terms including menu items such as **File** will appear in bold and all entries made by programmers such as a filename are italicized.
2. If the Solution Explorer window on the left is not shown, click **View** in the menu bar and then click **Solution Explorer** to display it.

Step 2: In the menu bar, click **File** → **New** → **Projects...** to display the **New Project** dialog box shown below:

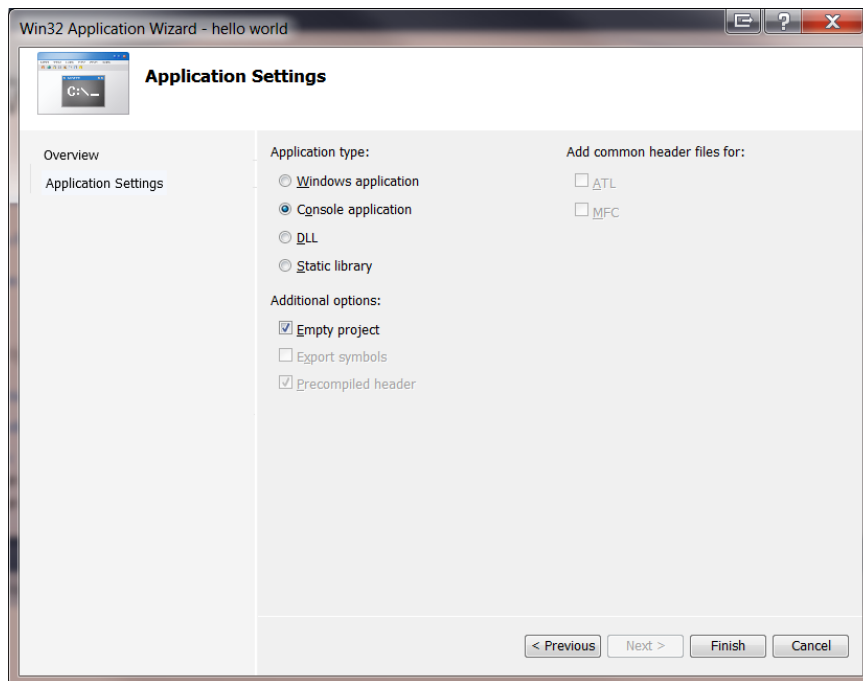
In the **New** dialog box shown below, select by clicking **Visual C++** in the **Installed Template** pane and **Win32 Console Application** in the middle pane. You then enter a name of your choice for the project (e.g., *hello world* as shown) in the Name box, select the location or folder in which you'd like to store project files (e.g., *C:\Users\irwinhu\Documents\CS230* as shown) by clicking the **Browse...** Note that there is no need to enter a name in the Solution name box; the system fills the project name in it by default.



Click on the **OK** button to display the **Win32 Application Wizard – *hello world*** window shown below:

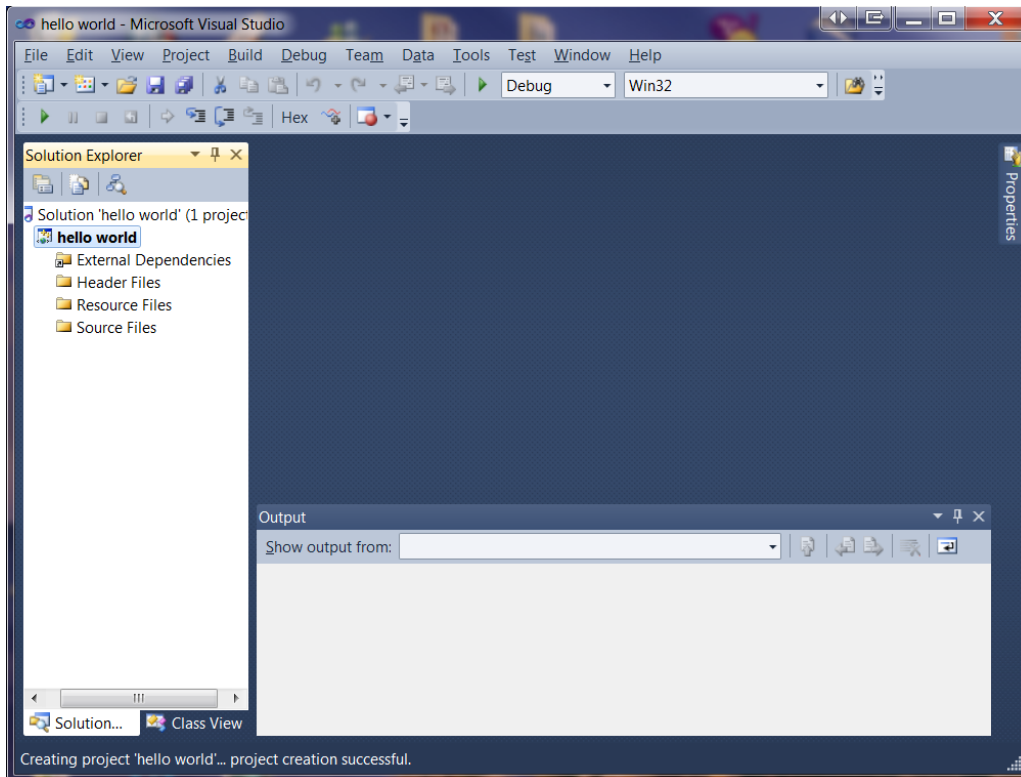


Click the **Next** button to display the following dialog box:

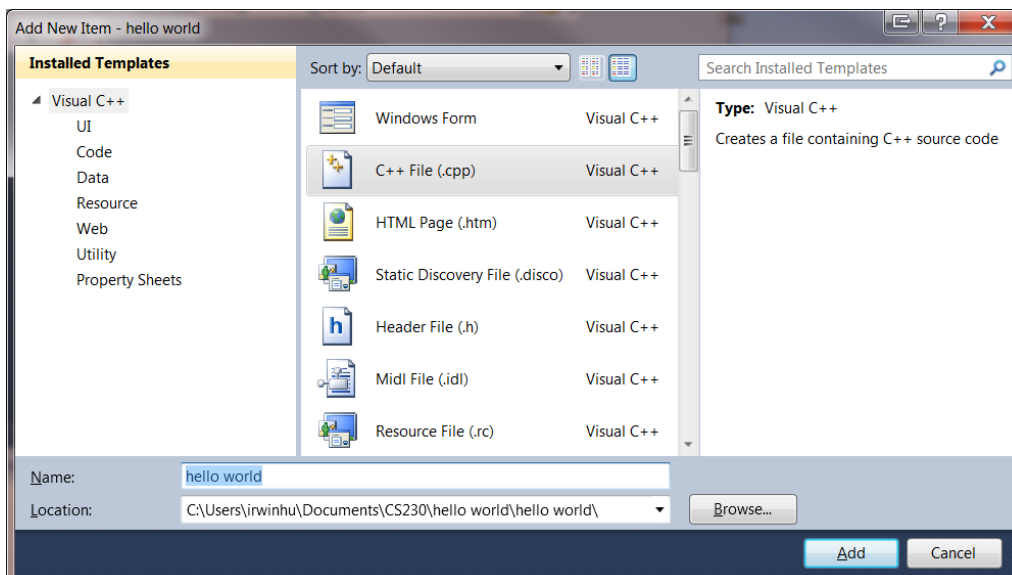


Check the **Empty project** box as shown above and click on the **Finish** button to proceed to the next step.

Step 3: Now the system displays the following window.

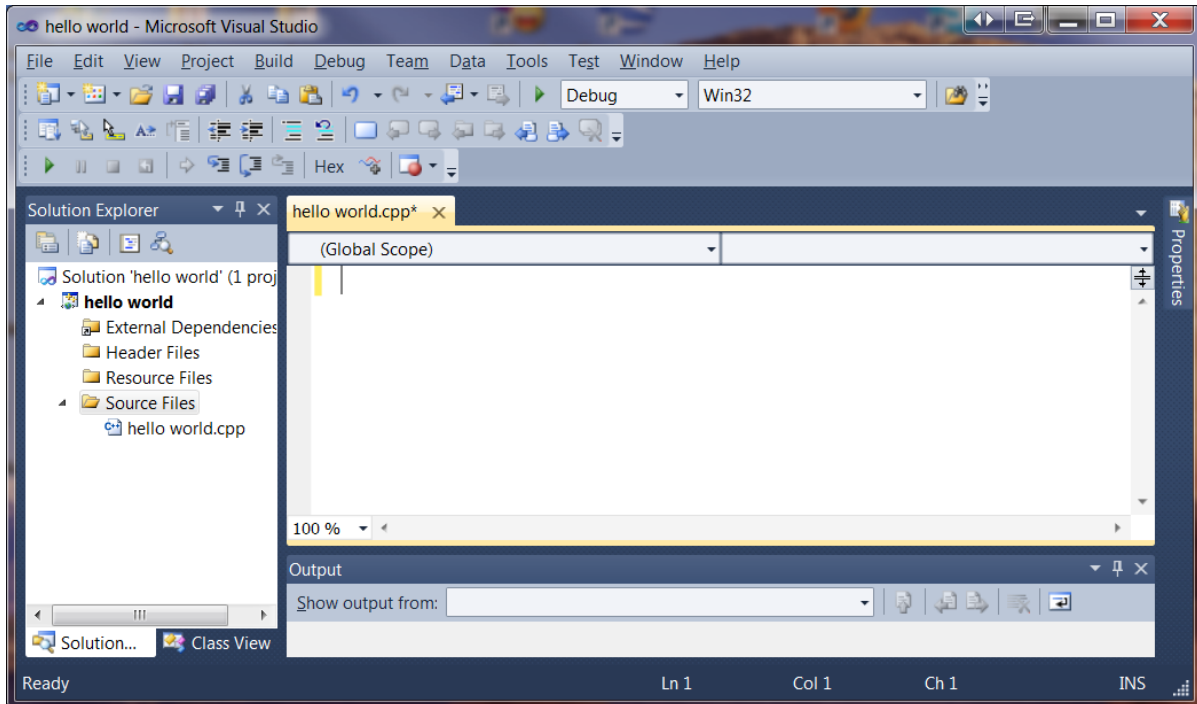


Right click on the **Source Files** folder in the **Solution Explorer** pane. In the popup menu, click **Add** then **New Item...** to display the following **Add New Item – hello world** dialog box:

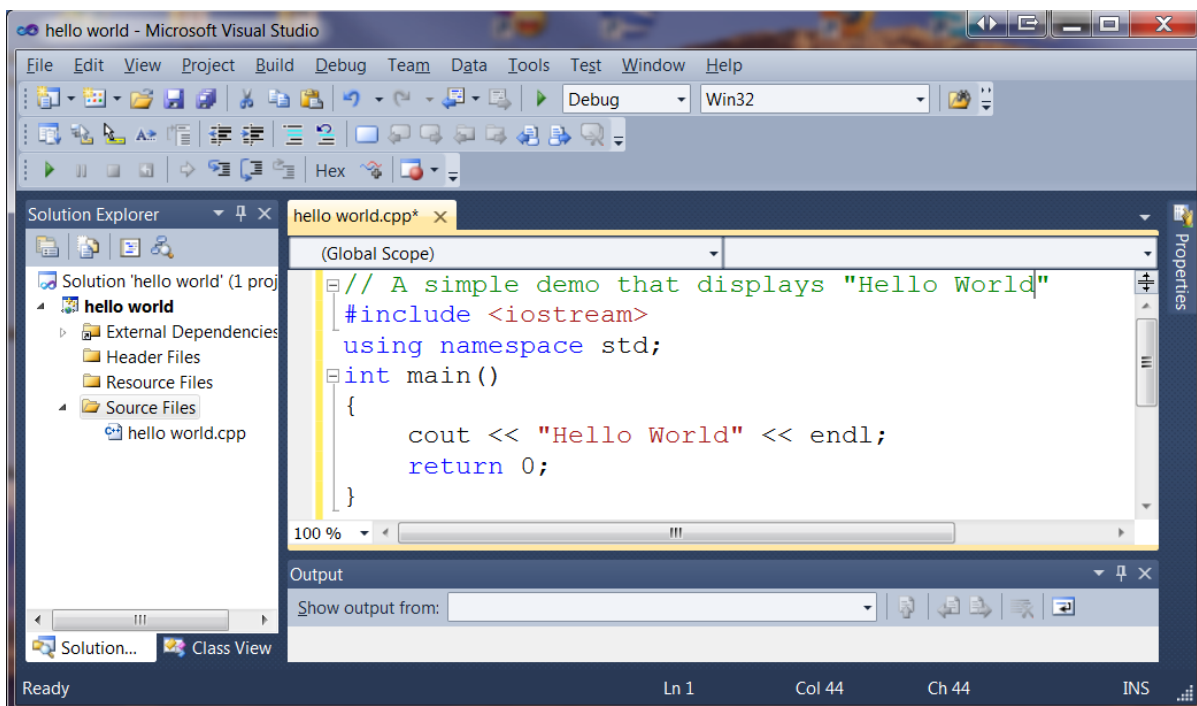


Select **C++ Files (.cpp)** by clicking on it in the middle pane and enter an arbitrary file name (e.g., *hello world*, the same name we used for the project is chosen here). Click **Add** to proceed to the next step.

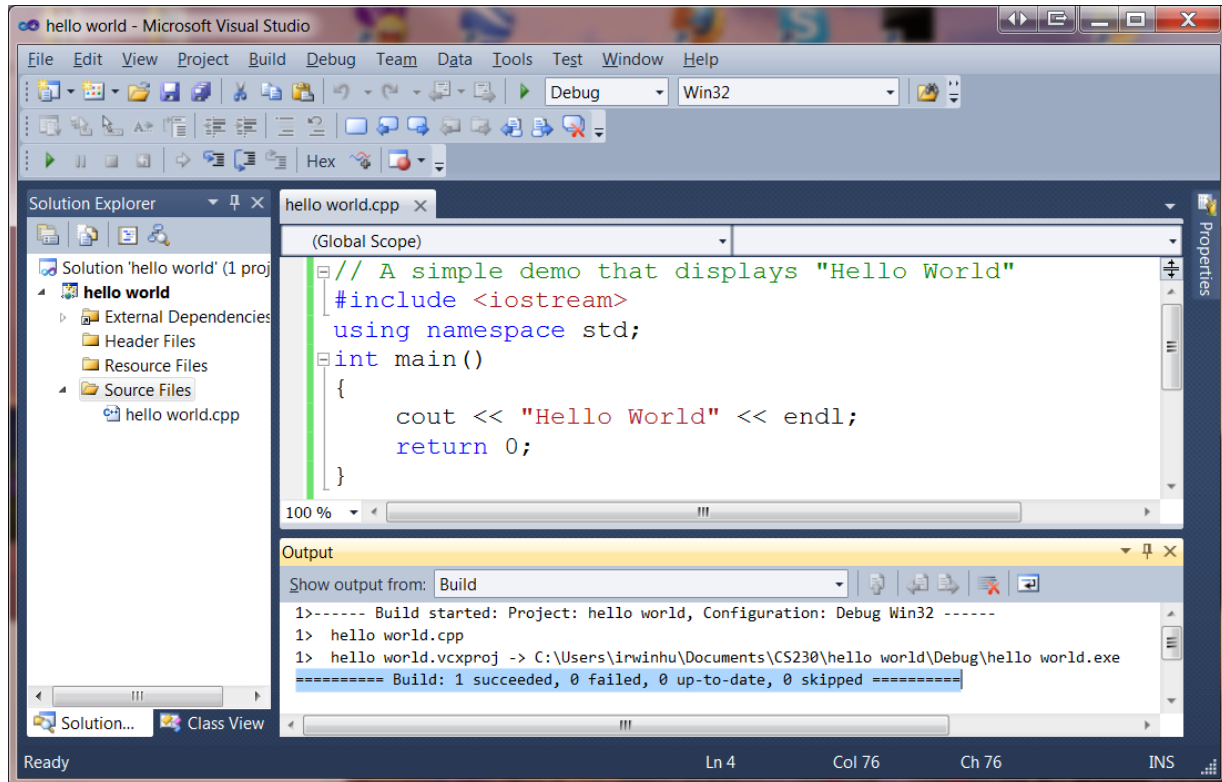
Step 4: The system displays the below window. Notice that 1) the **Source Folder** in **Solution Explorer** pane (make sure the **Solution Explorer** instead of **Class View** tab is selected) contains the *hello world.cpp* file that we just added and 2) the blank editing area/board is displayed with a *hello world.cpp* tab for you to enter your C++ source code.



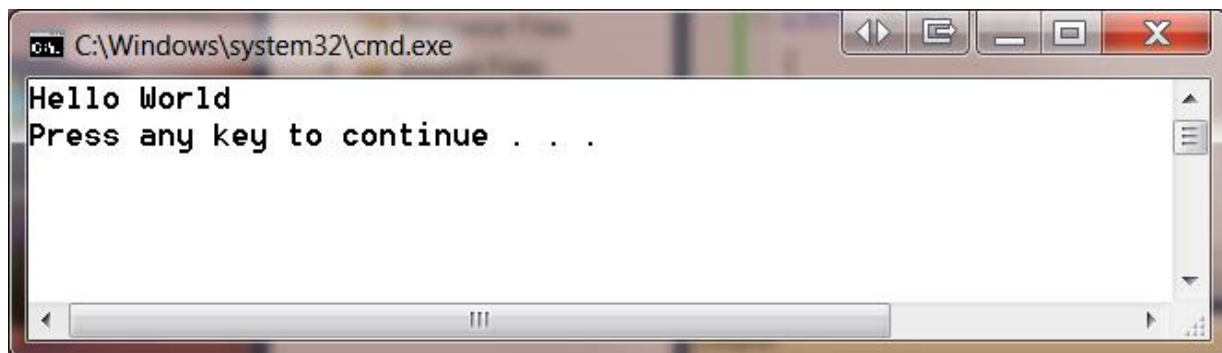
The same window after the source code is entered:



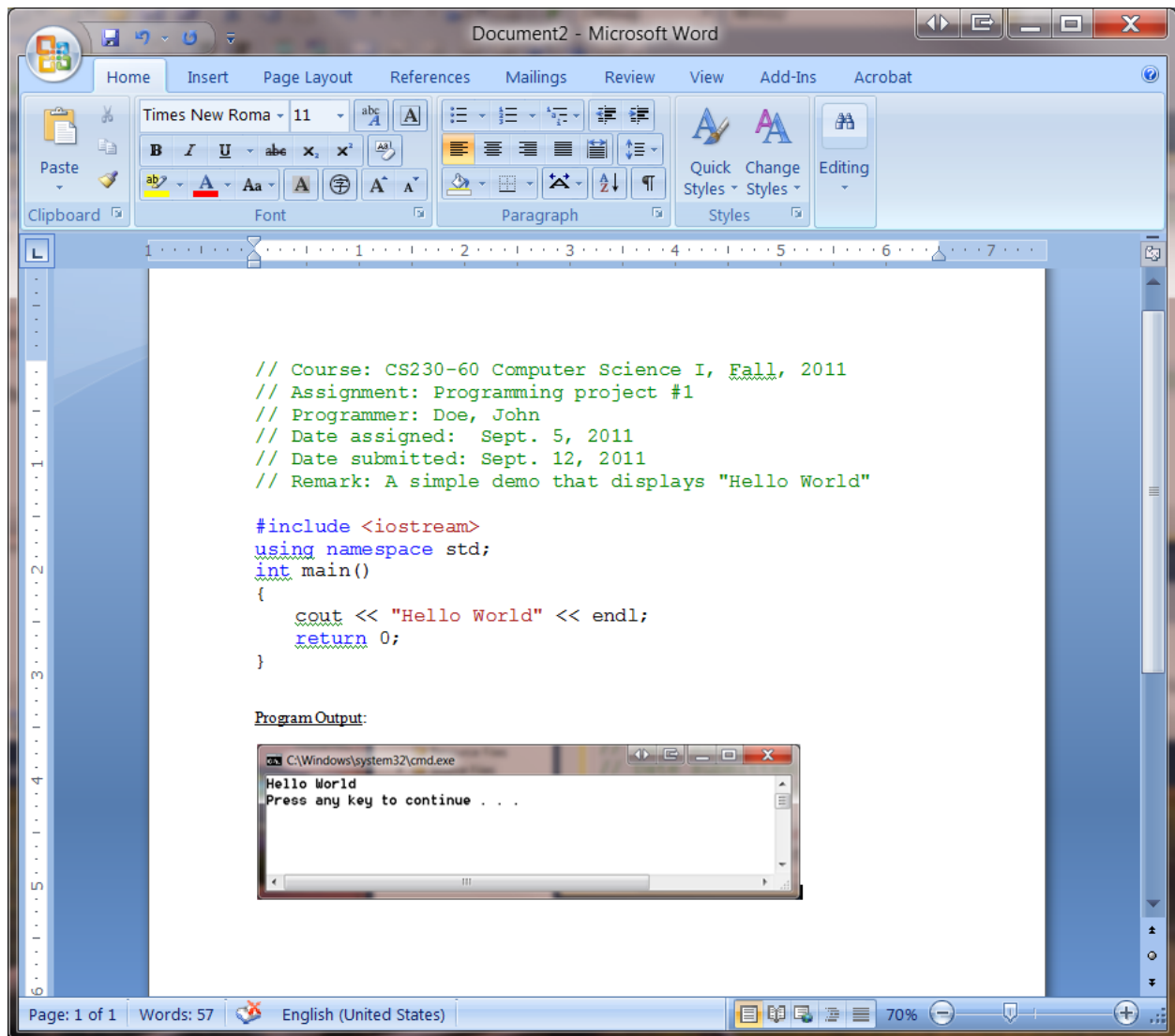
Step 5: To compile, link, load, and execute a program in a single step, click Debug in the menu bar and then click Start Without Debugging. If there is no error in your project, the below message ===== Build: 1 succeeded, 0 failed, 0 up-to-date, 0 skipped ===== is displayed in the Output window as shown below.



Also displayed in a separate window (C:\Windows\system32\cmd.exe) is the program output:



Step 6: You are normally required to turn in both the source code and the program output in printed form. One way to do it is to copy the source code from Visual C++ 2010 and paste it onto a Word document. Immediately after the source code, do a screen capture of the program output window (click on the window to activate it, hold down the **Alt** key and press the **Print Screen** key) and paste it on the Word document as shown below.

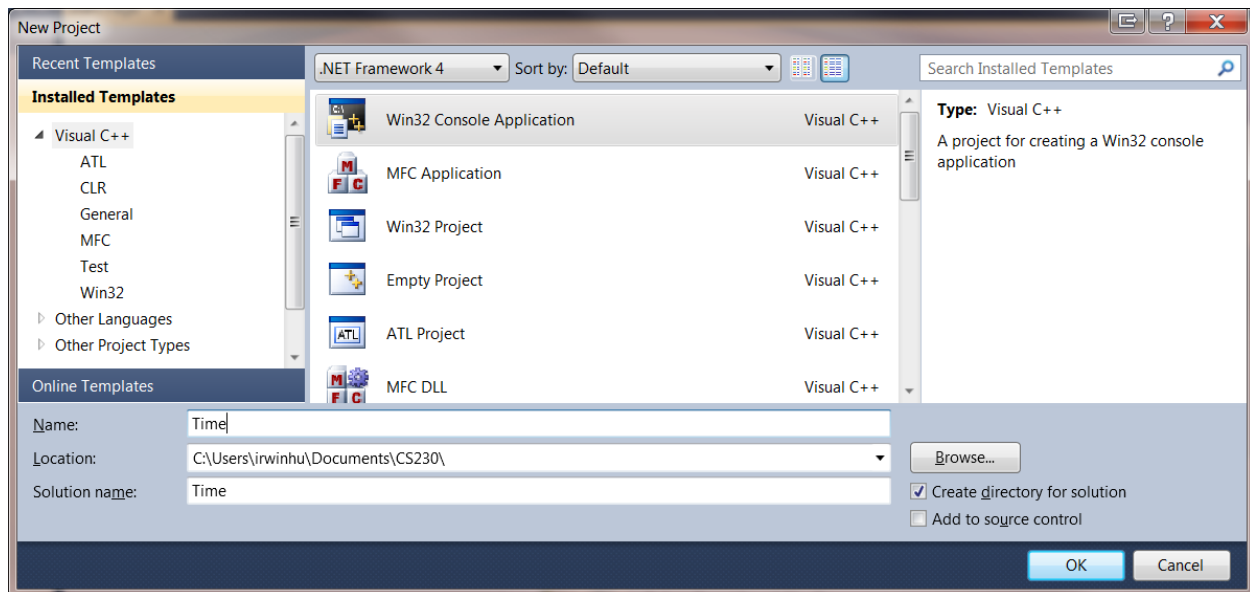


Note that comment statements have been added to the program code as part of the documentation requirement for a programming project. Check with your instructor regarding program documentation and submission rules.

A Multiple-File Project: The *Time* Program

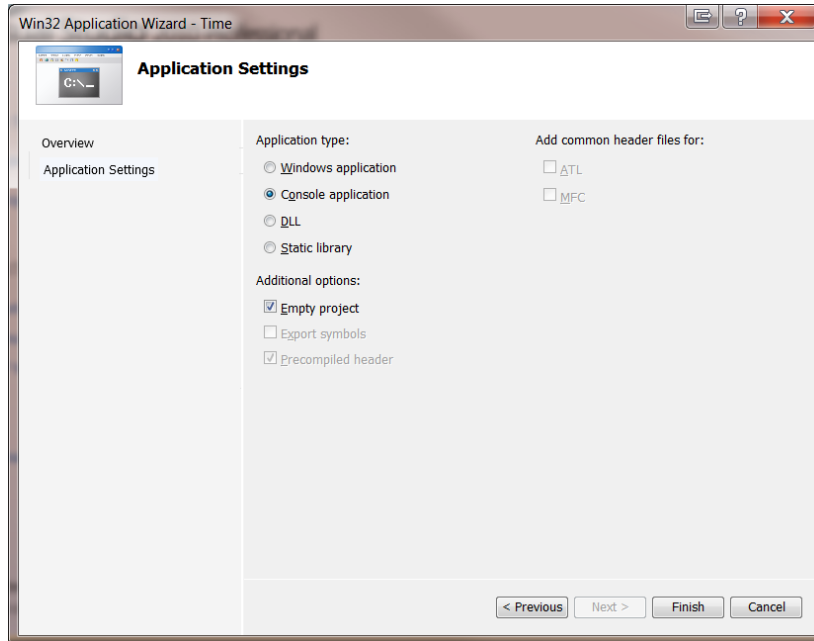
In this demonstration, the project or program that prints a given time of the day in both military as well as standard time formats. The project contains three files created by the programmer: two source files (*timeDemo.cpp* and *Time.cpp*) and a header file (*Time.h*). Again we'll build the project from scratch.

Step 1: Launch Visual C++ 2010, click **File** → **New** → **Project...** to display the **New Project** dialog box. In the dialog box, select **Visual C++** in **Installed Template** pane and **Win32 Console Applications** in the middle pane. Enter project name (e.g., *Time* in this demo) in the **Name** box and select the folder in which you'd like files for this project to be stored (e.g., *C:\Users\irwinhu\Documents\CS230* in this demo) by clicking the **Browse...** button as shown below:



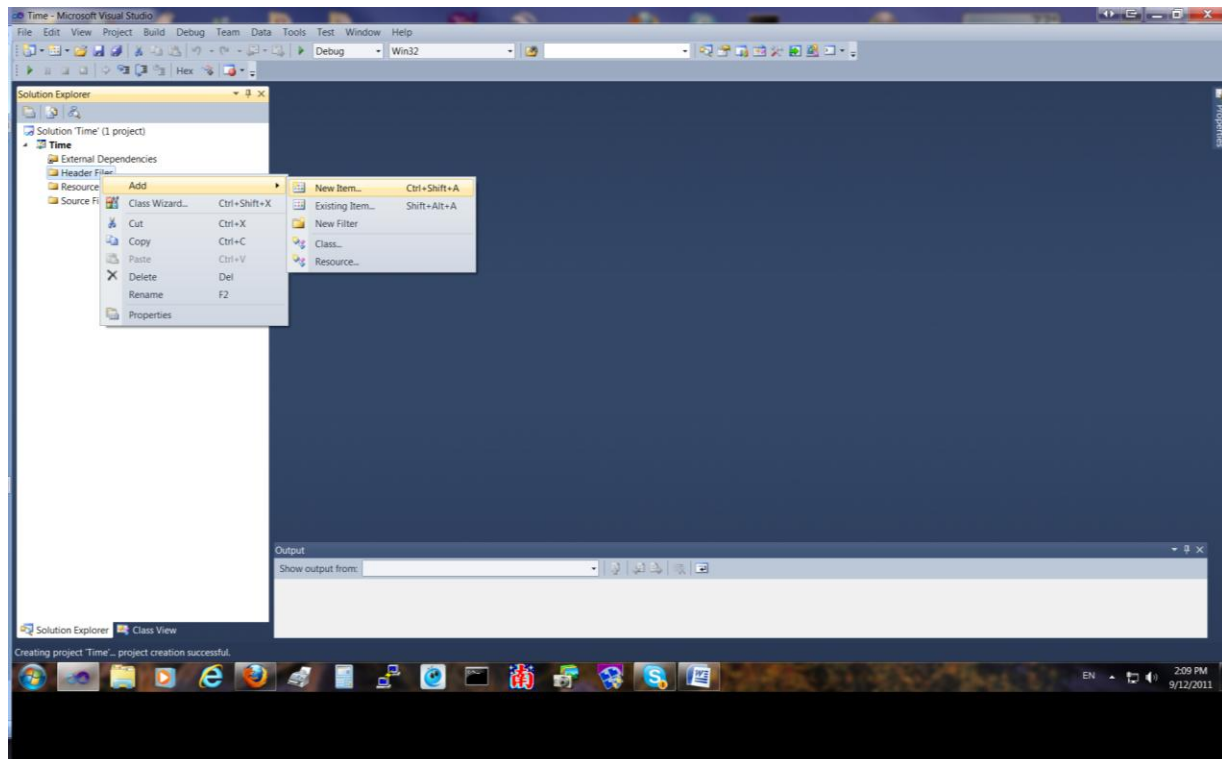
Click **OK** to display the **Win32 Application Wizard** – *Time* dialog box.

Step 2: In the dialog box, click **Next** (do not click **Finish**) and check the **Empty project** box as shown below:

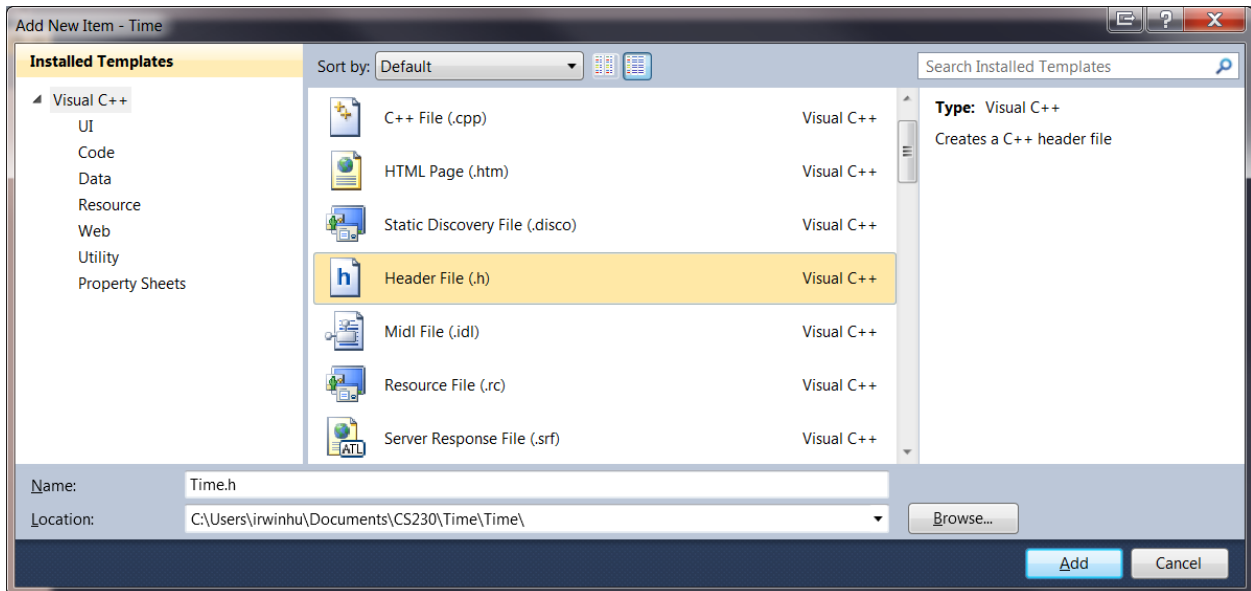


Click Finish to move on.

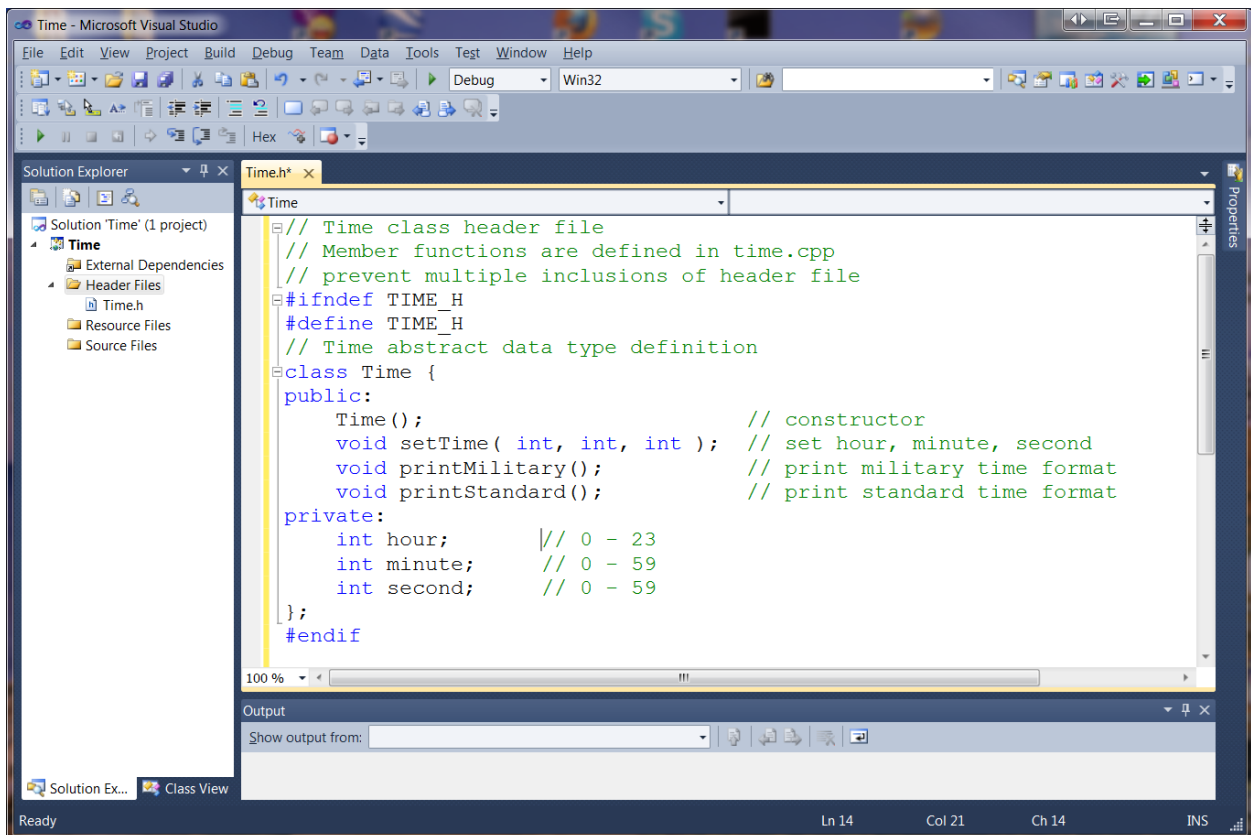
Step 3: Add header file to Header Files folder by right clicking the folder and then click **Add** → **New Item...**



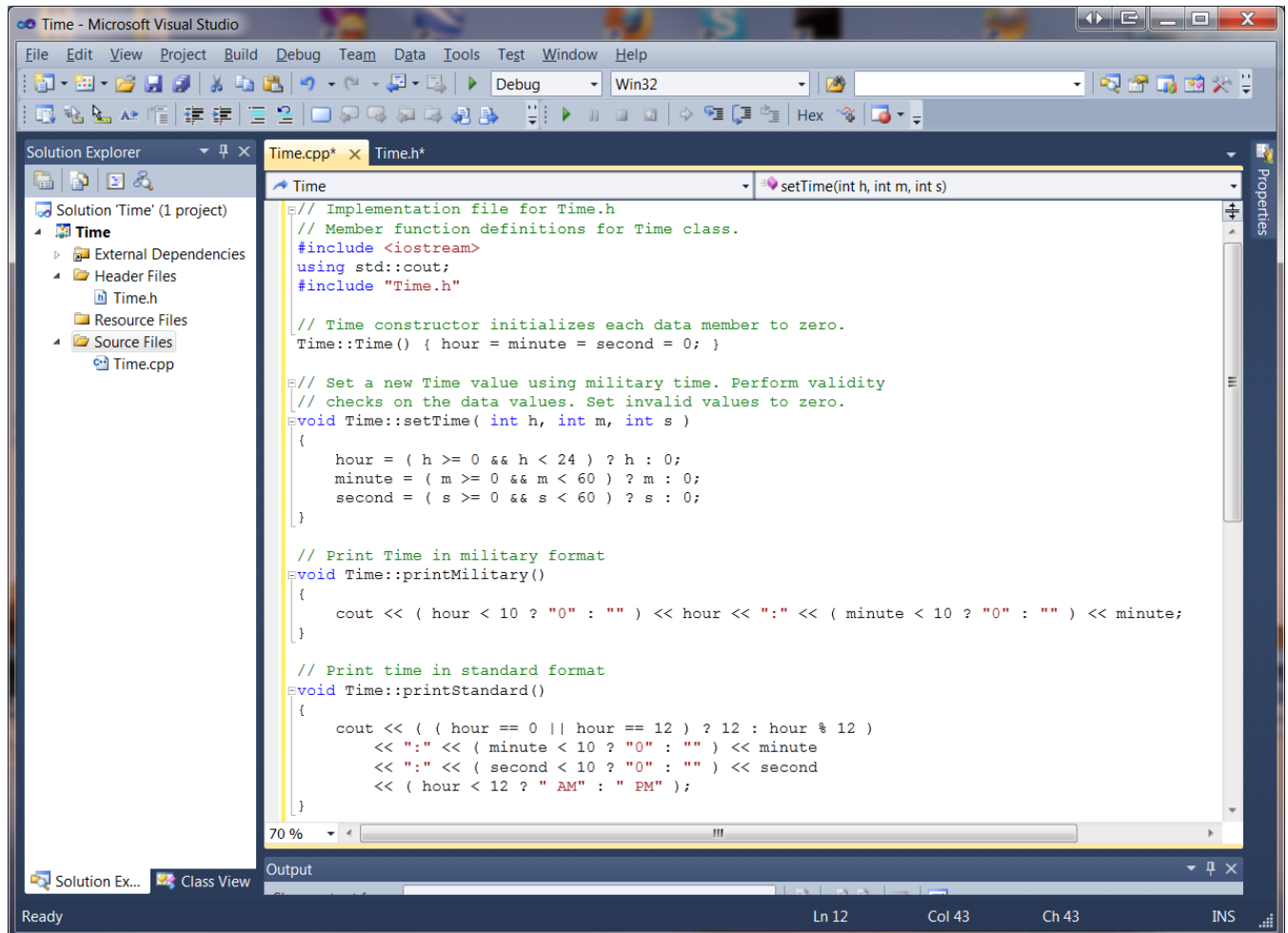
In the **Add New Item –Time** dialog box, select **Header File (.h)** in the middle panel and enter an arbitrary name (e.g., Time in this demo) for the header file:



Click the **Add** button, the empty *Time.h* file appears in the **Header Files** folder in the **Solution Explorer** window. Enter source code in the editing area as shown:



Step 4: Similar to step 3 above, add two source files (must be *.cpp* type; but the file names can be arbitrary, e.g., *Time.cpp* and *TimeDemo.cpp* in this demo), one at a time, to the **Source Files** folder by right clicking the **Source Files** folder. The below screen shows the source code for *Time.cpp* file has been entered:



```
Time - Microsoft Visual Studio
File Edit View Project Build Debug Team Data Tools Test Window Help
Debug Win32
Solution Explorer Time.cpp* Time.h*
Solution 'Time' (1 project)
  Time
    External Dependencies
    Header Files
      Time.h
    Resource Files
    Source Files
      Time.cpp
      Time.cpp*
      Time.h*
      Time
        setTime(int h, int m, int s)
// Implementation file for Time.h
// Member function definitions for Time class.
#include <iostream>
using std::cout;
#include "Time.h"

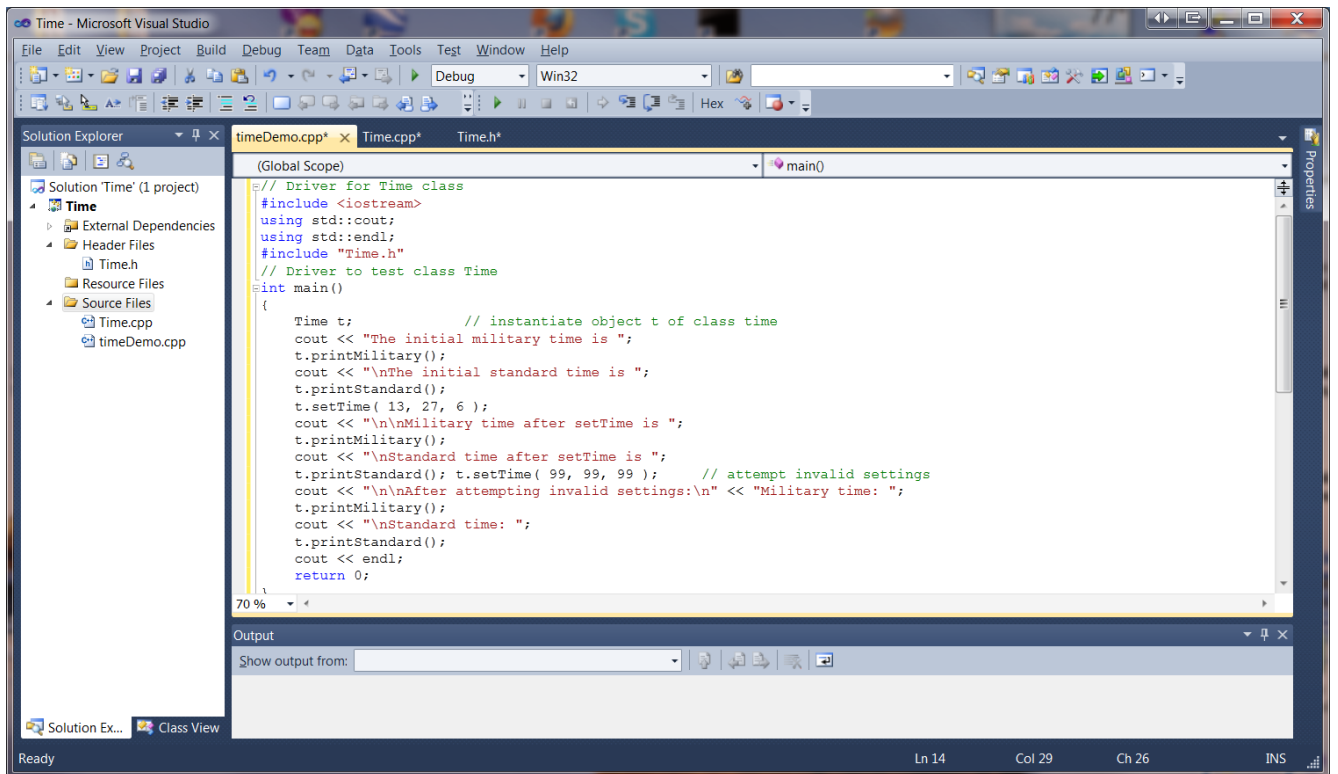
// Time constructor initializes each data member to zero.
Time::Time() { hour = minute = second = 0; }

// Set a new Time value using military time. Perform validity
// checks on the data values. Set invalid values to zero.
void Time::setTime( int h, int m, int s )
{
    hour = ( h >= 0 && h < 24 ) ? h : 0;
    minute = ( m >= 0 && m < 60 ) ? m : 0;
    second = ( s >= 0 && s < 60 ) ? s : 0;
}

// Print Time in military format
void Time::printMilitary()
{
    cout << ( hour < 10 ? "0" : "" ) << hour << ":" << ( minute < 10 ? "0" : "" ) << minute;
}

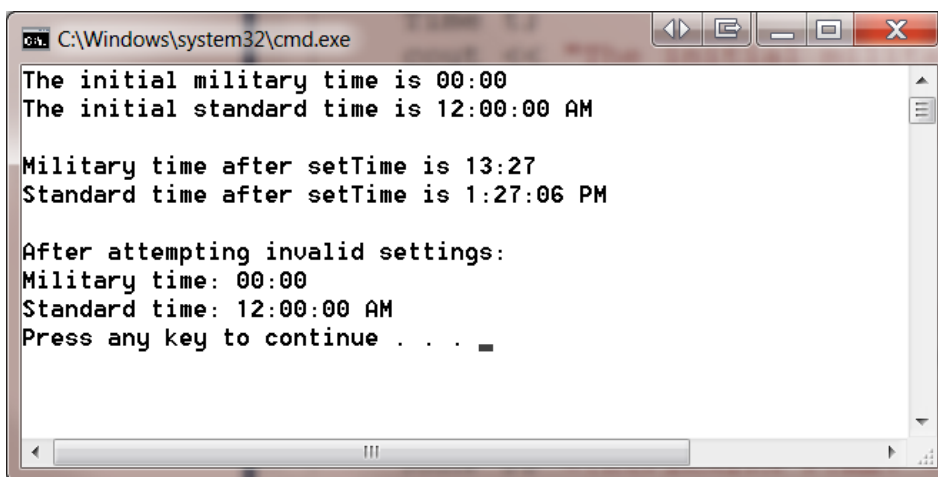
// Print time in standard format
void Time::printStandard()
{
    cout << ( ( hour == 0 || hour == 12 ) ? 12 : hour % 12 )
    << ":" << ( minute < 10 ? "0" : "" ) << minute
    << ":" << ( second < 10 ? "0" : "" ) << second
    << ( hour < 12 ? " AM" : " PM" );
}
70 %
Output
Ready Ln 12 Col 43 Ch 43 INS
```

The below screen shows that the source file timeDemo.cpp (the main driver function) has been added:



Notice that in the source code editing area, there are three named tabs, each is associated with a file previously entered. To view the source of any of these files, just click the tab with desired name on it.

Step 5: To compile, link, load, and execute or run the program, click **Debug** → **Start Without Debugging**. Since there is no error or bug in the code, the output window displays the message ===== Build: 1 succeeded, 0 failed, 0 up-to-date, 0 skipped =====. In a separate window, the program output is display below.



Appendix: The Source Code

Single-file Demo: The Hello World Program

File name: Hello World.cpp

```
#include <iostream>
Using namespace std;
int main()
{
    cout << "Hello World!" << endl; return 0;
}
```

Multiple-file Demo: the Time Program

File 1: Time.h

```
// Time class header file
// Member functions are defined in time.cpp
// prevent multiple inclusions of header file
#ifndef TIME1_H
#define TIME1_H
// Time abstract data type definition
class Time {
public: Time(); // constructor
void setTime( int, int, int ); // set hour, minute, second
void printMilitary(); // print military time format
void printStandard(); // print standard time format
private:
    int hour; // 0 - 23
    int minute; // 0 - 59
    int second; // 0 - 59
};
#endif
```

File 2: Time.cpp

```
// Implementation file for Time.h
// Member function definitions for Time class.
#include <iostream>
using std::cout;
#include "time.h"

// Time constructor initializes each data member to zero.
Time::Time() { hour = minute = second = 0; }

// Set a new Time value using military time. Perform validity
// checks on the data values. Set invalid values to zero.
```

```

void Time::setTime( int h, int m, int s )
{
    hour = ( h >= 0 && h < 24 ) ? h : 0;
    minute = ( m >= 0 && m < 60 ) ? m : 0;
    second = ( s >= 0 && s < 60 ) ? s : 0;
}
// Print Time in military format
void Time::printMilitary()
{
    cout << ( hour < 10 ? "0" : "" ) << hour << ":" << ( minute < 10 ? "0" :
"" ) << minute;
}

// Print time in standard format
void Time::printStandard()
{
    cout << ( ( hour == 0 || hour == 12 ) ? 12 : hour % 12 )
        << ":" << ( minute < 10 ? "0" : "" ) << minute
        << ":" << ( second < 10 ? "0" : "" ) << second
        << ( hour < 12 ? " AM" : " PM" );
}

```

File 3: timeDemo.cpp (main function - the driver)

```

// Driver for Time class
#include <iostream>
using std::cout;
using std::endl;
#include "time.h"
// Driver to test class Time
int main()
{
    Time t;
    // instantiate object t of class time
    cout << "The initial military time is ";
    t.printMilitary();
    cout << "\nThe initial standard time is ";
    t.printStandard();
    t.setTime( 13, 27, 6 );
    cout << "\n\nMilitary time after setTime is ";
    t.printMilitary();
    cout << "\nStandard time after setTime is ";
    t.printStandard(); t.setTime( 99, 99, 99 ); // attempt invalid settings
    cout << "\n\nAfter attempting invalid settings:\n" << "Military time: ";
    t.printMilitary();
    cout << "\nStandard time: ";
    t.printStandard();
    cout << endl;
    return 0;
}

```