Exercise I: A Brief Introduction to Visual C++ and Creating a New Project

Visual C++ is an integrated development environment (IDE). You are able to edit, compile, run and debug a program within the same development application. There are a variety of IDEs out there for the various platforms (PC, Mac, Unix variants) and languages (C, C++, Java, Pascal, etc.) This workbook is intended to introduce you to Microsoft Visual C++ 6.0 on the PC platform. It assumes that you are familiar with the C++ language itself.

Some Conventions and Tips

Capital letters in **bold face** will be used when referring to menu names. Menu selections will be written as they appear on the menu, also in bold letters. For example, you might be told to go to the **FILE** menu and select the **Print…** option in order to print the current document.

Mouse clicking directions will be presented in *italicized* **bold face** such as ***double click*** on **OK**.

New terminology specific to Visual C++ will also appear in **bold face** the first time it is used.

You will probably find it useful if you read over an exercise to get a general feeling for what will be done before actually doing the exercise.

First, launch Visual C++ on your computer. You should see a screen similar to the following one:

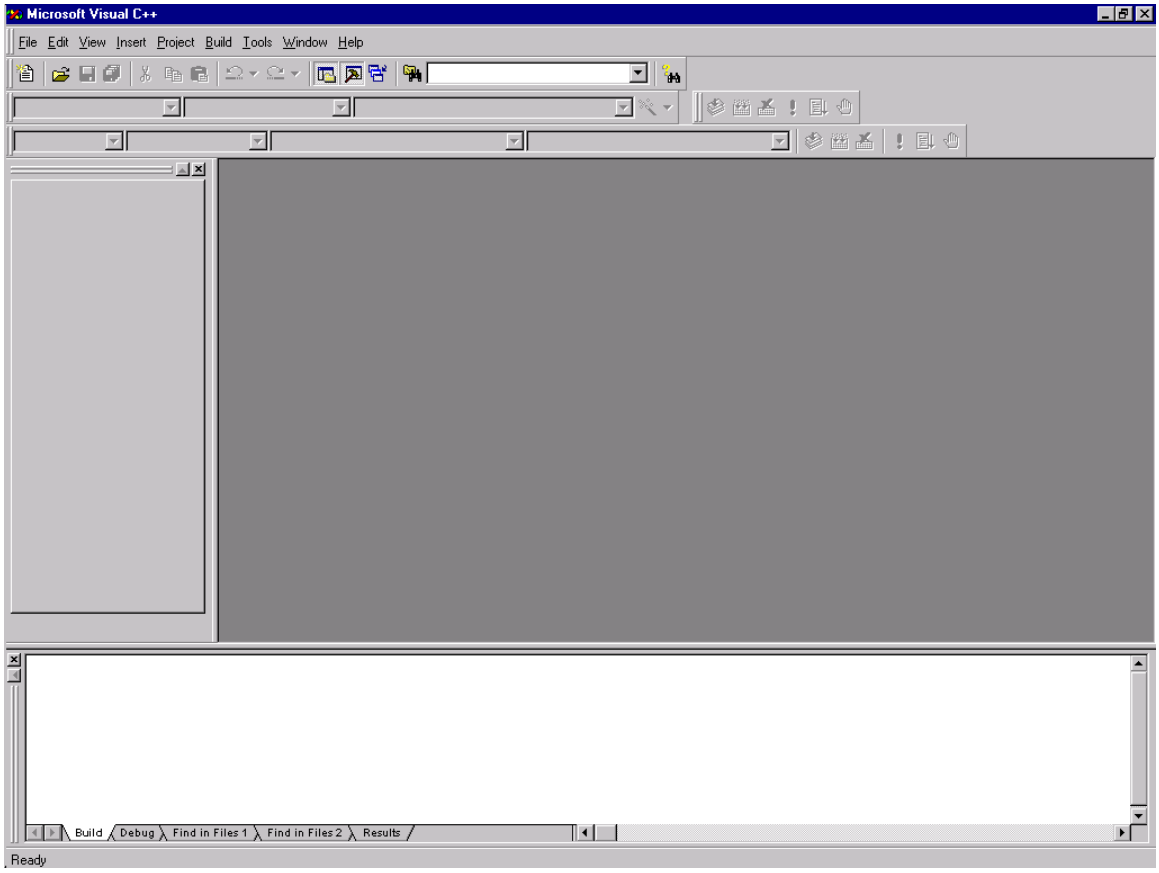

Figure I.1

In order to create a new project, you can go to the **FILE** menu and select **New…** from the menu. The dialog box that appears has several tabs. You want to choose the **Projects** tab.

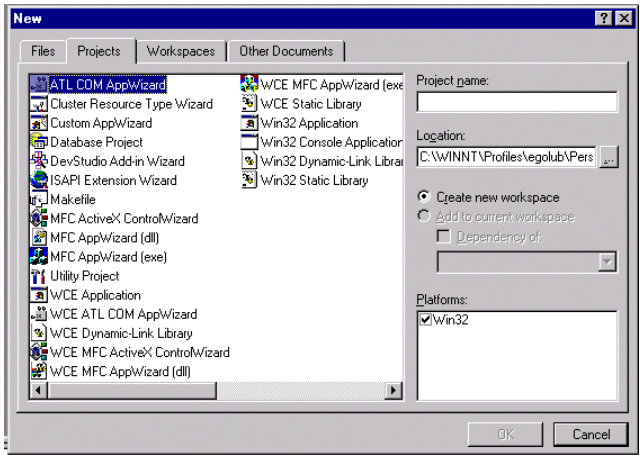

Figure I.2

At this point, there are many different types of projects that you can create. This workbook will only look at text based console applications. To create a new project of this type, ***single click*** on **Win32 Console Application**.

Next, you need to specify where this project will be created. You do this by choosing an existing directory in the **Location:** text entry box. The directory you specify will be the parent directory for a new directory that is created by Visual C++ to hold your project. You can use the **…** button next to the text entry box to bring up a directory browser with which you can choose an existing directory. You might find it helpful to create a new directory on your computer in advance that will hold all of your Visual C++ projects. (WAM users please see Appendix A for tips on using your WAM directory space when in the Visual C++ environment on campus.)

Specify the name of this project to be **exercise1** in the **Project name:** text entry box. Note that the path for the location now has the project name appended to it. At this point, you can ***single click*** on the **OK** button.

At this point, a new dialog box will appear with a list of choices with radio buttons next to them. You need to select what type of application Visual C++ should prepare for you.

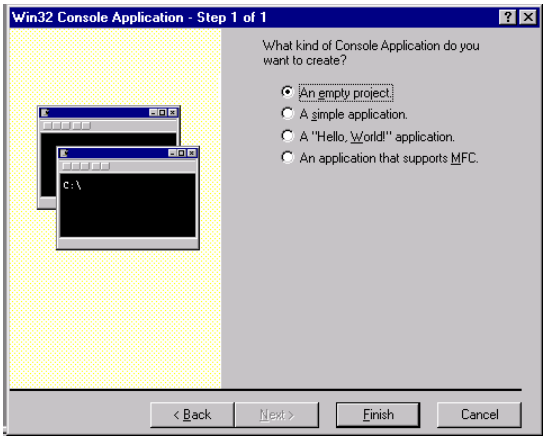

Figure I.3

Depending upon which type of project you are creating, different options will appear. For this workbook we are creating Win32 Console Applications and will mostly select **An empty project**. Header and source files will need to be created as you go. However, for this first exercise, we will select **A "Hello, World!" application** just to get started experimenting with the environment. ***Single click*** on the radio button next to **A "Hello, World!" application** and then ***single click*** on the **Finish** button.

After clicking on the Finish button, one last dialog box will come up informing you of what files (if any) were created for you. It may also have a brief description of what the files are designed to do (as it does in this case).

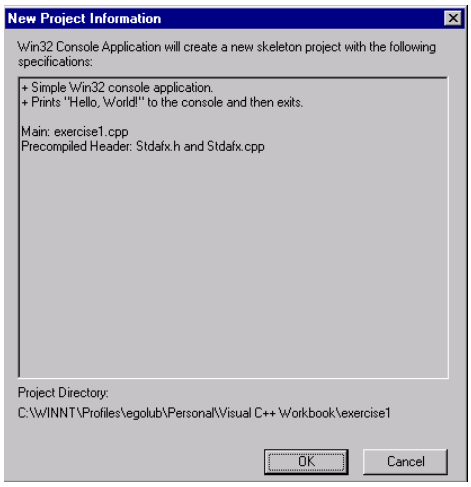

Figure I.4

If you now ***single click*** on the OK button, you will be brought into the development environment with this project loaded. Your screen should look similar to the following:

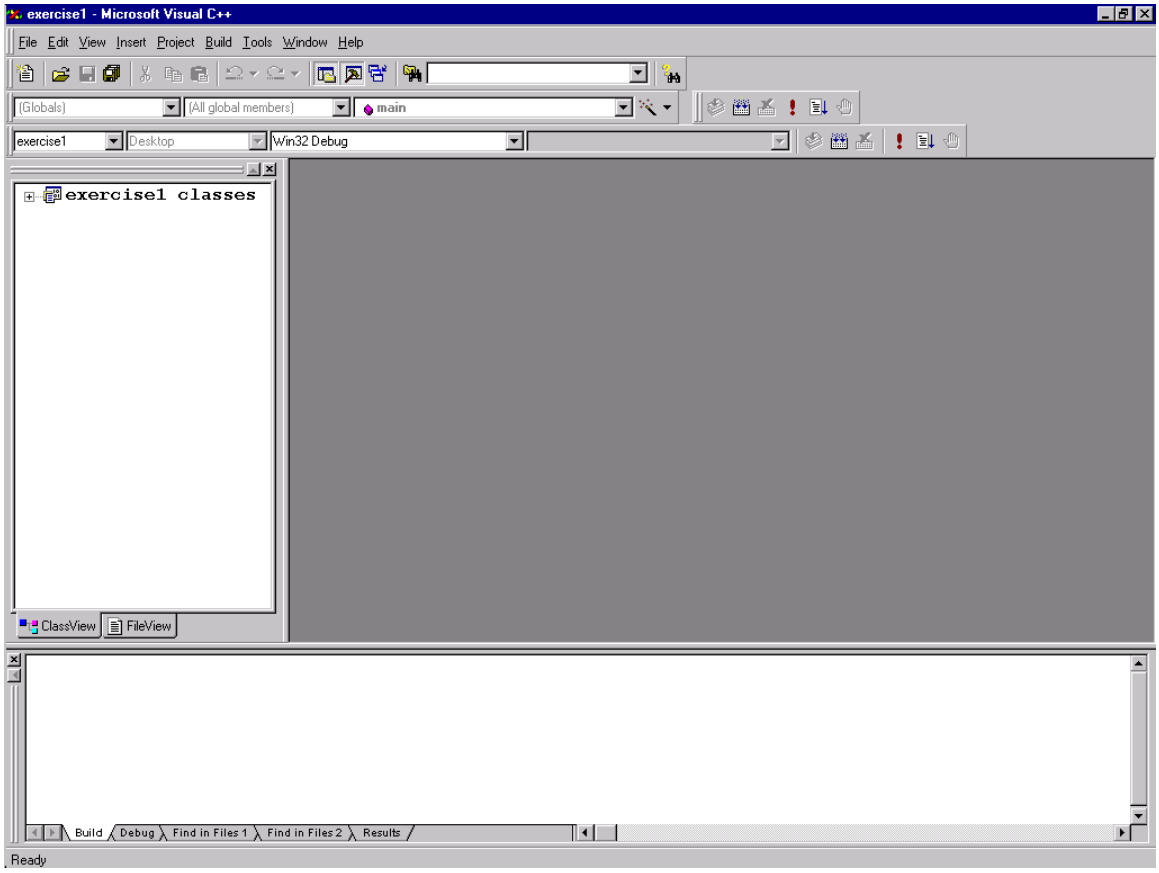

Figure I.5

You may or may not have the panel running across the width of the bottom of your screen. If you do not, it will appear the first time you compile a program. The two other panels shown in Figure I-5 are the workspace (left side of screen) and editor space (right side of screen).

Within the workspace, there are two different tabs available in a Win32 Console Application; **ClassView** and **FileView**. Under ClassView, you are able to view the various classes within the current project and well as details such as data members, methods and relationships to other classes within the project. Under FileView, you are able to view the list of files that are included within the current project.

First, single click on the ClassView tab. The ClassView tree begins with the name of the project followed by the word classes, so in this case, it will read **exercise1 classes**. To expand part of the tree, click on the + sign. In the following figure, I have fully expanded the tree.

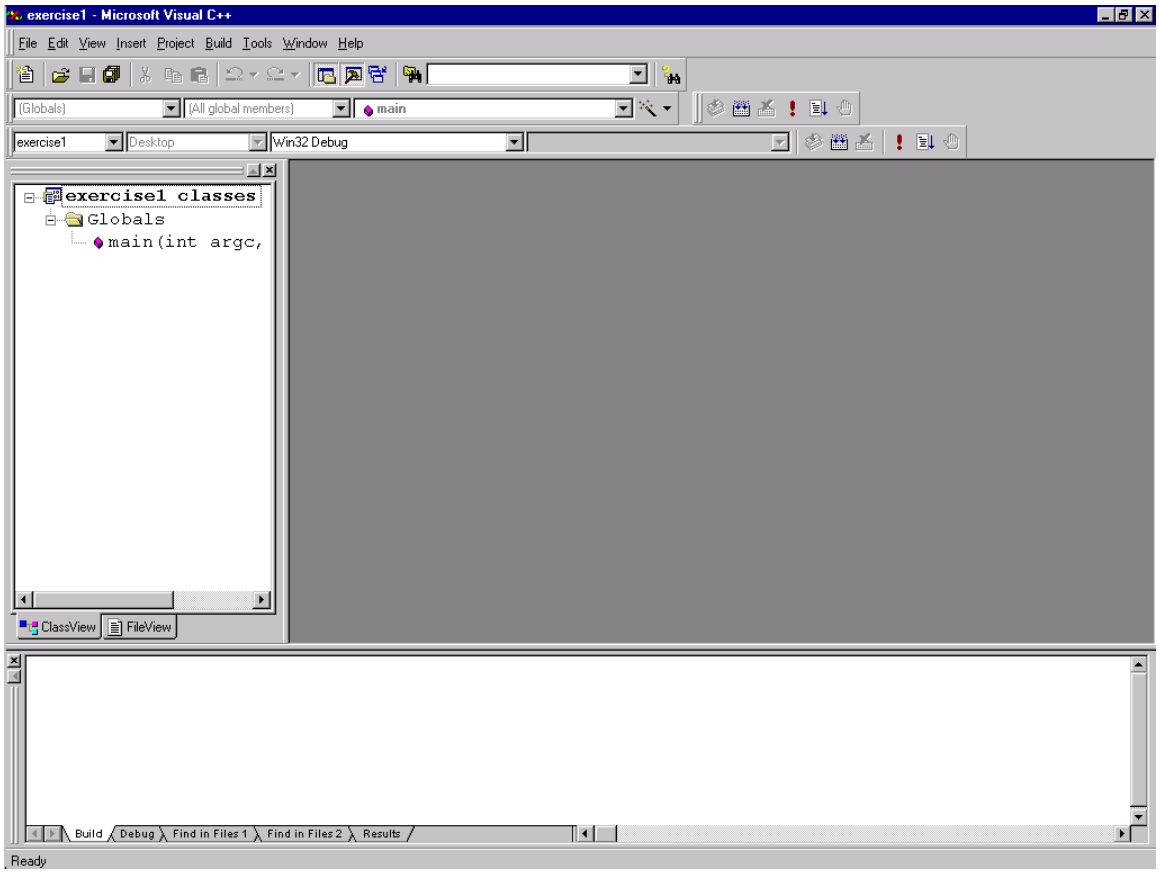

Figure I.6

This project does not have any classes, only a **main()** function. Since the main function is not a member of any class, it is grouped under the category of **Globals** in the ClassView. In Exercise III, we will add several classes to a project and see how they appear in the ClassView.

Next, click on the tab for FileView. You will again be shown the components of the project as a tree, but this time the tree is of the files rather than the classes. In the following figure I have fully expanded this tree as well.

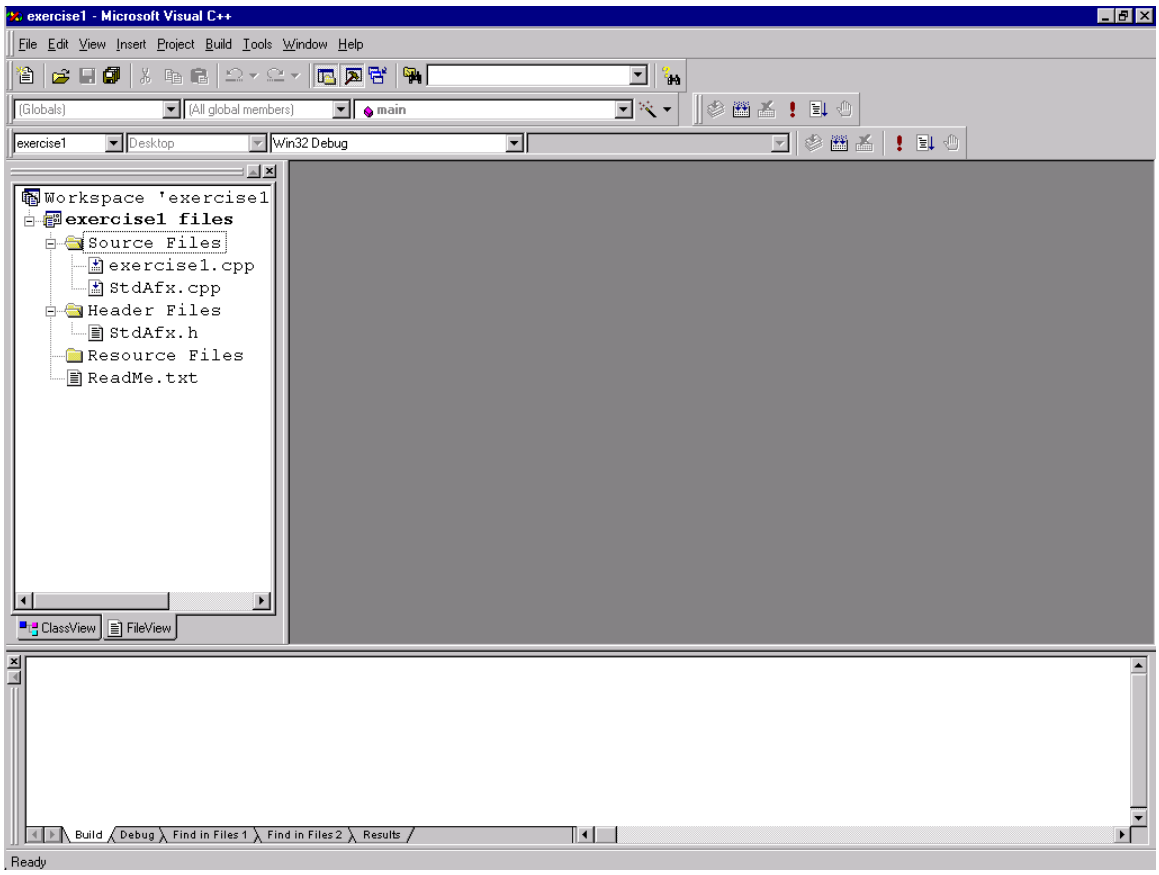

Figure I.7

The files within your project are grouped into categories such as **Source Files** and **Header Files**. This is organized by file suffix. For example, all files ending in **.cpp** will appear under Source Files and all files ending in **.h** will appear under Header Files. It is important to note that **.cc** is not a recognized suffix for source files in Visual C++.

In this example program created by Visual C++, there are three C++ files: **exercise1.cpp**, **StdAfx.cpp** and **StdAfx.h** and added them to this project. Let's discuss the **StdAfx** files briefly. Visual C++ creates the **StdAfx** files as a common area to place preprocessor directives that you intend to include in most or all source files in your project. With some project types, it will place various standard directives in there on your behalf. Since this is a basic console application, there is not much included in these files.

The relevant file is example1.cpp which contains the **main()** function, so let's view that file. One way to view a file is to ***double click*** on its name in the FileView panel. After ***double clicking*** on example1.cpp, your screen should appear similar to Figure I.8.

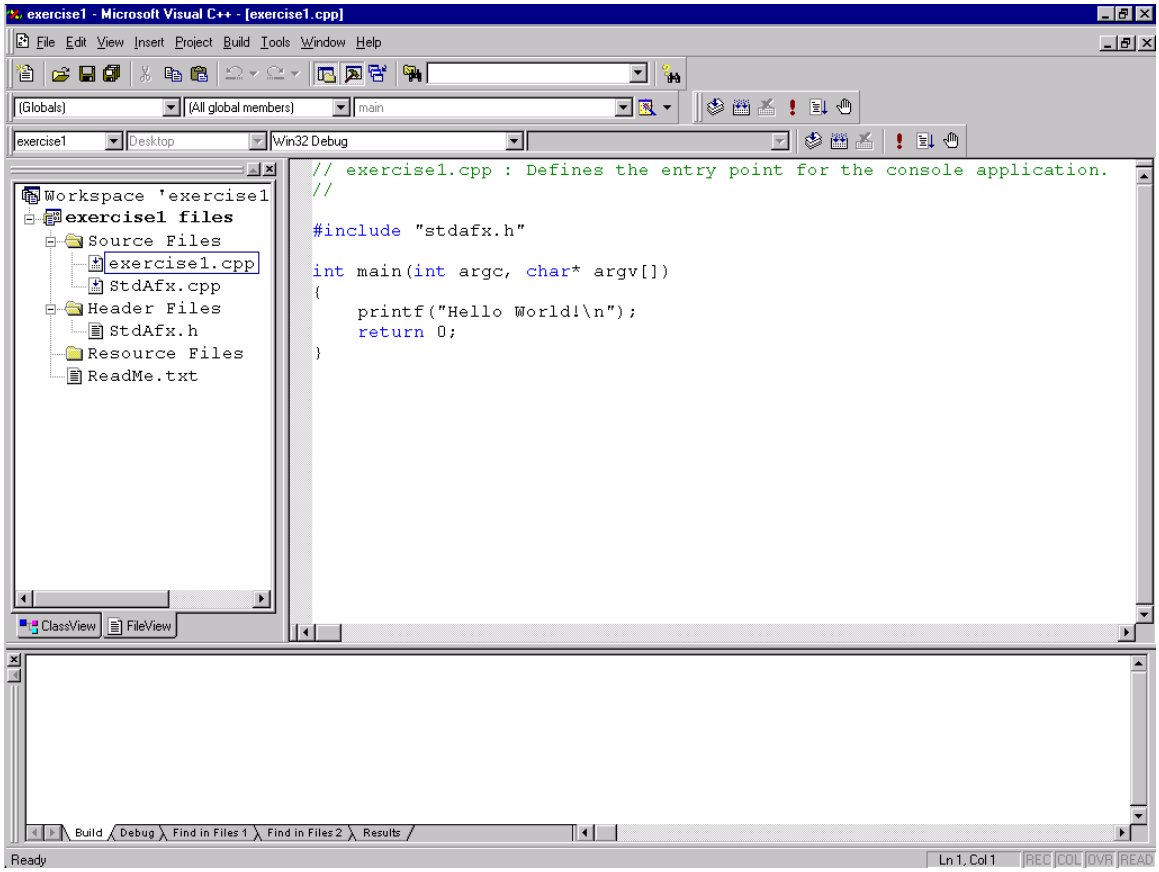

Figure I.8

This shows the simple C program which Visual C++ created for you, which prints the string **Hello World!** to the screen.

The next step is to compile the program. There are a variety of ways in which this can be accomplished. One thing to note is that Visual C++ creates its own internal makefile based on the files that have been specified as part of the project. In this case, those files are **example1.cpp**, **StdAfx.cpp** and **StdAfx.h**. If you instruct Visual C++ to build the project (we will look at the ways in which we can accomplish this shortly) it will save and compile any sources files that have not been compiled since the last time they or the headers files they depend upon were modified. It will then link them together to create an executable. The name of the executable will be the name of the project followed by a **.exe** suffix.

You can instruct Visual C++ to compile the project in any one of the following ways:

- Go to the **BUILD** menu and select **Build projectname.exe**
- Press the **F7** key
- Single click on the build button (shown in Figure I.9)

Figure I.9

Additionally, if you view a C++ source file, you can instruct Visual C++ to compile a single source file in any one of the following ways:

- Go to the **BUILD** menu and select **Compile currentfile.cpp**
- While holding down the **Control** key, press the **F7** key
- Single click on the compile button (shown in Figure I.10)

Figure I.10

However, even if you compile the individual source files, you will still need to use one of the build techniques to create an executable to run.

Build exercise1.exe now using any of the above mentioned techniques.

Notice that the panel going across the bottom of your Visual C++ window describes the steps that it is going through. If any problems occur during the compilation process, they will be listed in this panel. We will soon introduce an error to the provided code in order to experience this.

Now that you have an executable file created, you can instruct Visual C++ to execute (run) this program in any of the following ways:

- Go to the **BUILD** menu and select **Execute currentfile.exe**
- While holding down the **Control** key, press the **F5** key
- Single click on the execute button (shown in Figure I.11)

Figure I.11

Execute the program now using any of the above mentioned techniques.

A new window will have been opened, and your program executed within that window. It should have appeared similar to the following:

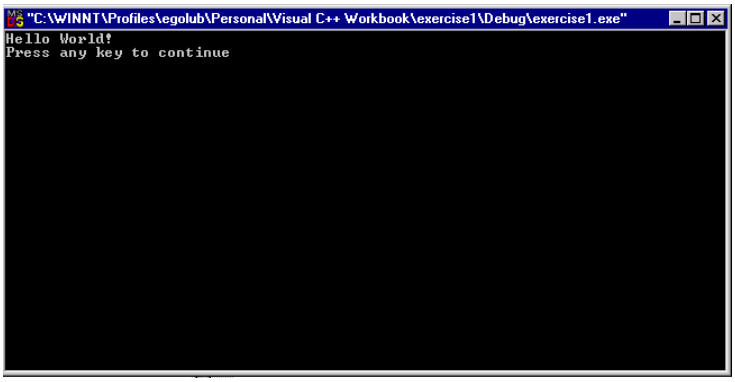

Figure I.12

Notice in Figure I.12 that after the program had completed its execution, the message **Press any key to continue** was displayed. This is there so that the window can stay open even after your program executes. This can be useful if you want to see the output of your program. If you press a key (the space bar is a nice choice) this window will then close.

Congratulations! You have now compiled and executed your first exercise.

To leave the Visual C++ environment, go to the **FILE** menu and select **Exit**.