

CSE 332 Studio Session on C++ Development Environments

These studio exercises are intended to (re-)acquaint you with Visual C++ and with some basic programming concepts and techniques that are likely already familiar from your previous programming experience, and to make sure that the programming tools and environment we'll use this semester are working correctly for you.

In this studio (as in the others this semester) you will work in self-selected groups of 2 or 3 people, and will report the results of your work on the studio exercises on the course message board. Students who are more familiar with the material are encouraged to help those who are less familiar with it, and asking questions of your classmates, professor and teaching assistants during the studio sessions is highly encouraged as well.

Please record your answers you work through the following exercises. After you have finished please post your answers to the required exercises, and to any of the enrichment exercises you completed, to the course message board, as a reply to my posting titled "Development Environment Studio". The enrichment exercises are optional but are a good way to dig into the material a little deeper, especially if you breeze through the required ones.

PART I: REQUIRED EXERCISES

1. Form a team of 2 or 3 people of your choice and write down the names of the team members as the answer to this first exercise.
2. Log into one of the studio lab's Windows machines, and open up Visual Studio 2008 from the Start→Programming→Microsoft Visual Studio 2008 menu (note that Visual Studio 2005 is also installed on the machines: please make sure you open 2008 not 2005). When the program opens, choose Microsoft Visual C++ as your development environment. Click on the Help→About Microsoft Visual Studio menu item at the top of the main window, and write down the serial number for Microsoft Visual C++ 2008 from the Installed products list of the dialog window that appears, as the answer to this exercise. Close the dialog window.
3. Click on the File→New→Project menu item at the top of the main window, and from the list under Visual C++ that appears on the left of the New Project window, select Win32. Select the Win32 Console Application template that should appear to the right of that list. Enter a name for the project (for example, **DevEnvStudio** or something similar that identifies which studio this is for). As a workaround for an issue in Visual Studio 2008 that's currently being looked at by the SEAS IT staff, type **h:** into the location field and then click the **Browse** button which will then give you a network drive directory in which Visual Studio 2008 can create the directory for your new project. Click the **OK** button, and if the Win32 Application Wizard dialog appears, just click on the **Finish** button. After a while, Visual Studio will bring up the newly created project, and a list of the files in it will appear in the Solution Explorer window. Write down the name you chose for the project and the names of all of the files that appear on the left as the answer to this exercise.

4. Click on the Build→Build Solution menu item at the top of the main window, and watch the output messages that are generated by Visual C++ when it builds the project. Write down the names of the source files that were compiled when you built your project, as the answer to this exercise.

5. On your computer's main menu, click on the Start→Run menu item, make sure the entry to be opened says **cmd**, and click **OK**. In the terminal window that appears, use the **cd** command to change to the directory on the **H:** drive where your new project was created, and then use **cd** again to change into the **Debug** directory under it. Type **dir** and hit enter to list the contents of that directory. You should see a file with a **.exe** extension, which is the executable file the compiler created (on Linux, executable files usually have no extension at all). Run the built program by typing in the name of that file and hitting enter. Write down what output (if any) is produced by running the program, as the answer to this exercise.

6. Switch back to the Visual C++ development environment and add the following line just above the line that says **return 0;** in your project's main source file (which should be named something similar to **DevEnvStudio.cpp**):

```
cout << "hello, world!" << endl;
```

Build your project again. Write down the names of the unrecognized symbols (as given by the error messages that should appear), as the answer to this exercise.

7. Below the line that says **#include "stdafx.h"** (a required inclusion for Windows, which must be the first non-comment line of the program) add the necessary pre-compiler directives to the top of your project's main source file (e.g., **DevEnvStudio.cpp**) to allow the symbols from exercise 6 above to be recognized when you try to build the program (**hint:** see the C++ program structure slides from today's lecture). After successfully building the program, go back to the terminal window and run the updated executable program. Write down what output is produced, as the answer to this exercise.

PART II: ENRICHMENT EXERCISES (optional, feel free to skip some and do ones that interest you)

8. Add a directive to include C++ style strings as a recognized type in your program (**hint:** see the C++ string Class slide in the set of slides that is provided in the syllabus for this studio) and replace the **cout << "hello, world!" << endl;** line with a declaration of a C++ style string variable (that is initially empty). Use the assignment operator **=**, the concatenation (addition) operator **+**, and/or the "concatenate (add) and assign" operator **+=** to add the names of your team members to that string variable. Following that code, add a line to your program that prints out the contents of that variable as program output. Build your program again, fixing any errors and/or warnings you encounter, until it builds successfully. Run the program, and write down the program's output as the answer to this exercise.

9. Declare an unsigned integer variable, and assign it the length of the string (once you've concatenated all your team members' names into it), which can be obtained by calling the string variable's **length()** method. Add a line to your program that prints out the value of this unsigned integer variable. Build your program, and make sure there are no warnings or errors (if there are, fix them) and then switch back to the terminal window and run the program. Write down the program's output as the answer to this exercise.