CS 150 Lab 3
Arithmetic and the Debugger

The main objective of today's lab is to use some basic mathematics to solve a few real world problems. In doing so, you are to begin getting accustomed to using the Visual Studio debugger.

- Be sure your output looks exactly like the specified output.
- Be sure to submit the completed project to **CS150-02 Lab** when you are done.
- Be sure to use the program skeleton and add comments to your code!
- Show the instructor or TA your solution to each problem before submitting.
- You do not need to submit the challenge question!

Lab 3.0

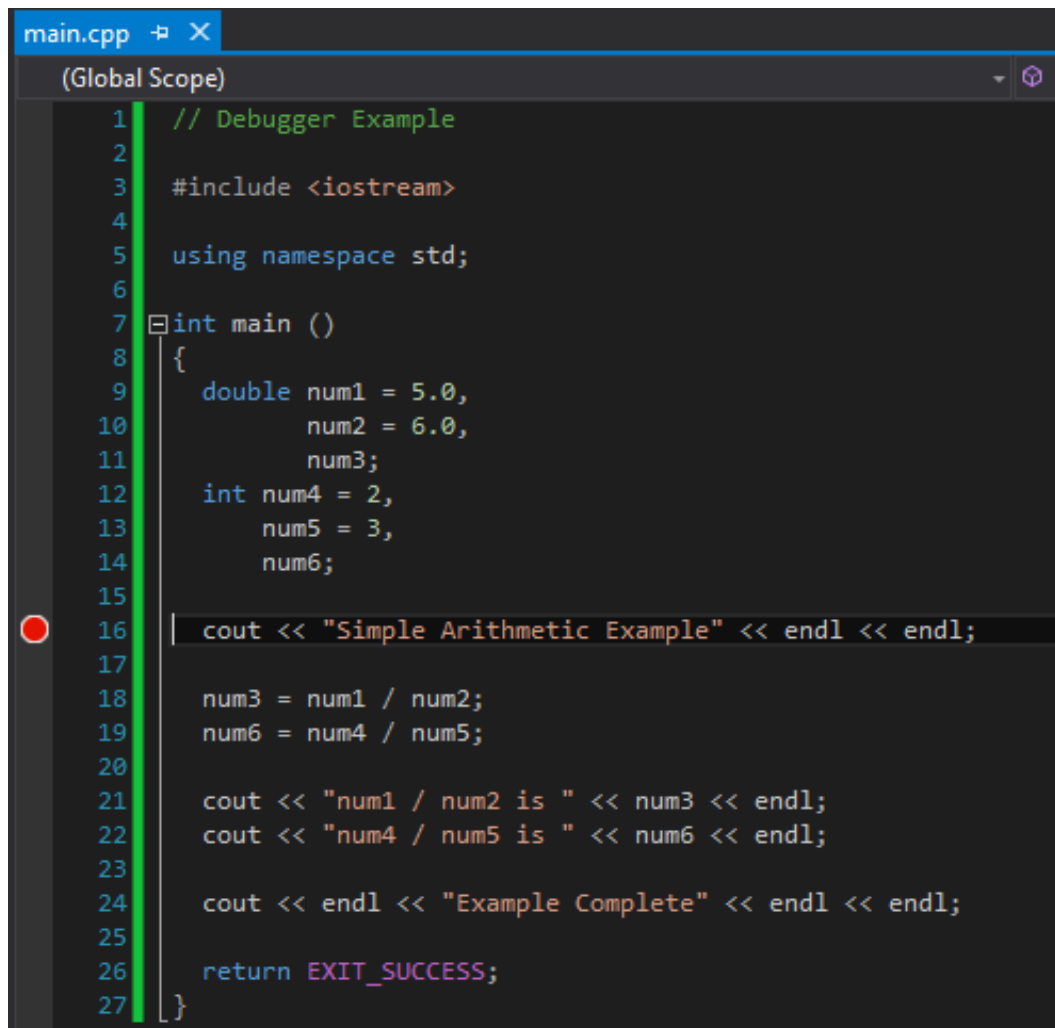We are going to begin using the Visual Studio 2013 debugger to aid with debugging programs.

1. Go to **CS150Public** folder on Grace and drag a copy of the solution **03DebuggerExample** to your desktop.

2. Open up main.cpp and look at the code. Make sure that you understand it.

3. Build your solution and then execute the program using the command "Start Without Debugging." You will see the following.

4. In looking at your output you wonder why num4 / num5 is 0 since both numbers are nonzero. What is going on? Let's debug!

5. Close the output window and go back to the code.

6. Set a breakpoint by clicking in the dark gray column to the left of the executable statement on line 16. This will place a large red circle in that spot. Your screen should look like the one below.
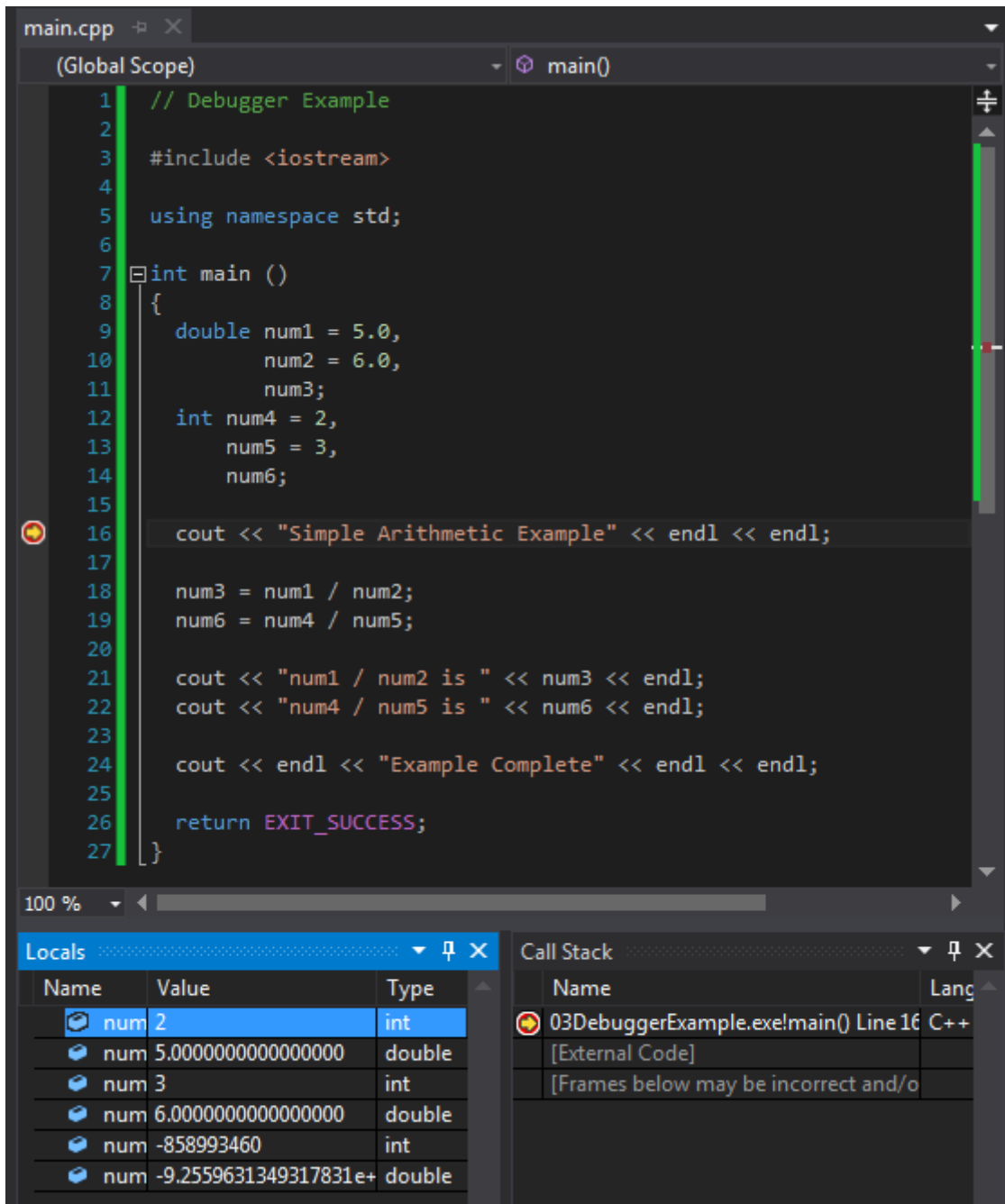
```cpp
// Debugger Example

#include <iostream>

using namespace std;

int main ()
{
   double num1 = 5.0,
          num2 = 6.0,
          num3;
   int num4 = 2,
       num5 = 3,
       num6;

   cout << "Simple Arithmetic Example" << endl << endl;

   num3 = num1 / num2;
   num6 = num4 / num5;

   cout << "num1 / num2 is " << num3 << endl;
   cout << "num4 / num5 is " << num6 << endl;

   cout << endl << "Example Complete" << endl << endl;

   return EXIT_SUCCESS;
}
```

7. Debugging starts running the program, but will stop at your breakpoint. This allows you to step through the program one statement at a time, and see the contents of the variables.

Let's start the debugger and step through our program one executable statement at a time. Start the debugger by going to Debug → Start Debugging. The debugger will stop at the first executable statement that has a breakpoint. The yellow arrow indicates the spot that the program will execute next. Your screen should look something like the following. At the bottom of the screen, make sure the Locals tab is selected and not the Autos tab.
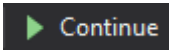
```cpp
main.cpp  ⊟ ✕
(Global Scope)                                    ⊕ main()
 1    // Debugger Example
 2
 3    #include <iostream>
 4
 5    using namespace std;
 6
 7  ⊟int main ()
 8    {
 9       double num1 = 5.0,
10              num2 = 6.0,
11              num3;
12       int num4 = 2,
13           num5 = 3,
14           num6;
15
16       cout << "Simple Arithmetic Example" << endl << endl;
17
18       num3 = num1 / num2;
19       num6 = num4 / num5;
20
21       cout << "num1 / num2 is " << num3 << endl;
22       cout << "num4 / num5 is " << num6 << endl;
23
24       cout << endl << "Example Complete" << endl << endl;
25
26       return EXIT_SUCCESS;
27    }
```

100 %

Locals				Call Stack	
Name	Value	Type		Name	Lang
num 2		int		⊕ 03DebuggerExample.exe!main() Line 16	C++
num 5.0000000000000000		double		[External Code]	
num 3		int		[Frames below may be incorrect and/o	
num 6.0000000000000000		double			
num -858993460		int			
num -9.2559631349317831e+		double			

8. Study the screen and answer the following questions:

   1. What is stored in the variable **num1**: _____

   2. What is stored in the variable **num4**: _____

   3. Why are they different? _____

   4. What is stored in the variable **num6**: _____

   5. Why does **num6** contain that value? _____

   6. Did the console (output) window open? If so, where is it? What does it contain?

   _____

   _____

9. While the debugger is running, look for the following toolbar at the top of the screen. Don't press any of the buttons yet.

   The buttons that most concern

   Continue: will continue running the program until the next breakpoint or the end of the program

   Stop Debugging: will stop running the program

   Step Over: will step over the program, one executable statement at a time

10. What happens when you click on **Step Over** once?

   _____

   _____

11. What happens when you click on **Step Over** once again?

   _____

   _____

12. Hover the mouse over **num4** on line 19. What happens? What is stored in **num4**?

_____

_____

13. Hover the mouse over **num5** on line 19. What happens? What is stored in **num5**?

_____

_____

14. What do you expect to be stored in **num6** after executing line 19?

_____

15. Click Step Over. What is stored in **num6** now?

_____

16. Modify the program so that it displays 0.666667 for **num4 / num5** instead of 0

17. ▶ STOP – Show the instructor or TA

Lab 3.1: Fahrenheit to Celsius

Open up your PUNetIDLabs solution that contains the lab projects from the last two weeks.

Add a project called **03_1_TemperatureConversion** to your **PUNetIDLabs** solution. This project is to convert a Fahrenheit temperature to both Celsius and Kelvin. The conversion formulas are:

$$C = \frac{5}{9}(F - 32)$$

$$K = \frac{5}{9}(F - 32) + 273$$

Sample Input/Output where the Fahrenheit temperature is the only input shown in bold below.

```
********************
Temperature Converter
********************

Enter a non-negative Fahrenheit temperature: 212

212 Fahrenheit is 100 Celsius

212 Fahrenheit is 373 Kelvin
```

1) Briefly describe the data that your program will need to use.

_____

_____

2) List the variable declarations necessary to store the data listed in 1.

_____

_____

3) Open up Visual Studio and write a complete program for the above problem.

4) Build, run and test your program.

5) Once it's working, show the instructor/TA that you can step through your solution with the debugger.

► **STOP – Show the instructor or TA**

Lab 3.2: Time Converter

Add a project called **03_2_TimeConverter** to your **PUNetIDLabs** solution. This project is to convert a number of seconds into hours, minutes, and seconds in a format like HH:MM:SS.

Sample Input / Output where the total number of seconds is the only input shown in bold below.

```
**************
Time Converter
**************

Enter seconds since day began: 100

The time is 0:1:40
```

1) Briefly describe the data that your program will need to use.

_____

_____

_____

2) List the variable declarations necessary to store the data listed in 1.

_____

_____

_____

3) Open Visual Studio and write a complete program for the above problem.

4) Build, run and test your program.

► **STOP – Show the instructor or TA**

Optional Challenge:

Always print your time as HH:MM:SS so the above time would show 00:01:40

Once both projects are completed, place your solution **PUNetIDLabs** into the **CS150-02 Drop** folder on Grace. Your solution is to have all six (2 from lab 1, 2 from lab 2, 2 from lab 3) projects completely working and correct.