NAME _____

EGR 2261 Lab 1

Entering and Running C++ Programs

OBJECTIVES

- Use Microsoft Visual Studio to enter some C++ programs.
- Compile and run these programs to verify that they work correctly.

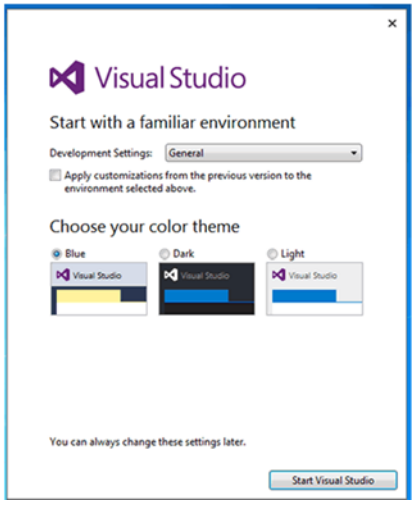
Part 1. Starting a New Project in Visual Studio

As you learned in class this week, for this course you will use Microsoft's Visual Studio to enter and run C++ programs. Visual Studio is a powerful and complicated program used by professional programmers. It contains many advanced features that we will not use in this course. The instructions below describe the steps you'll take to enter and run a simple program.

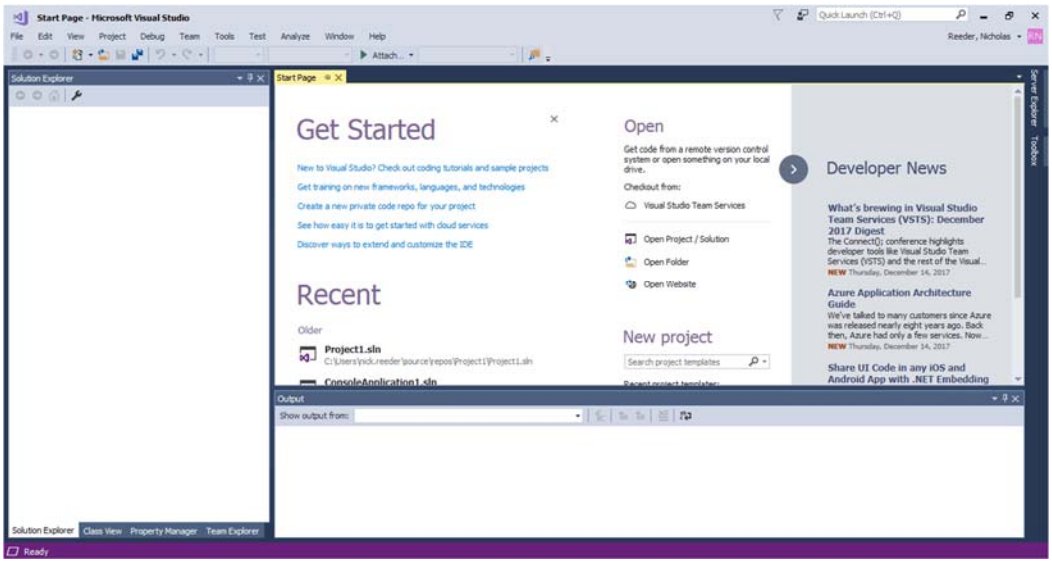
Every time you start working on a new program, you'll create a new **project** in Visual Studio. Each project consists of several nested folders (or directories) on your computer's drive. These folders will contain at least one file that you create yourself, along with many other files that Visual Studio creates for you automatically. At the end of each class you should copy your projects from the computer's hard drive to your flash drive for safekeeping.

Let's get started on this lab's first program, which is named **Lab01Hello**.

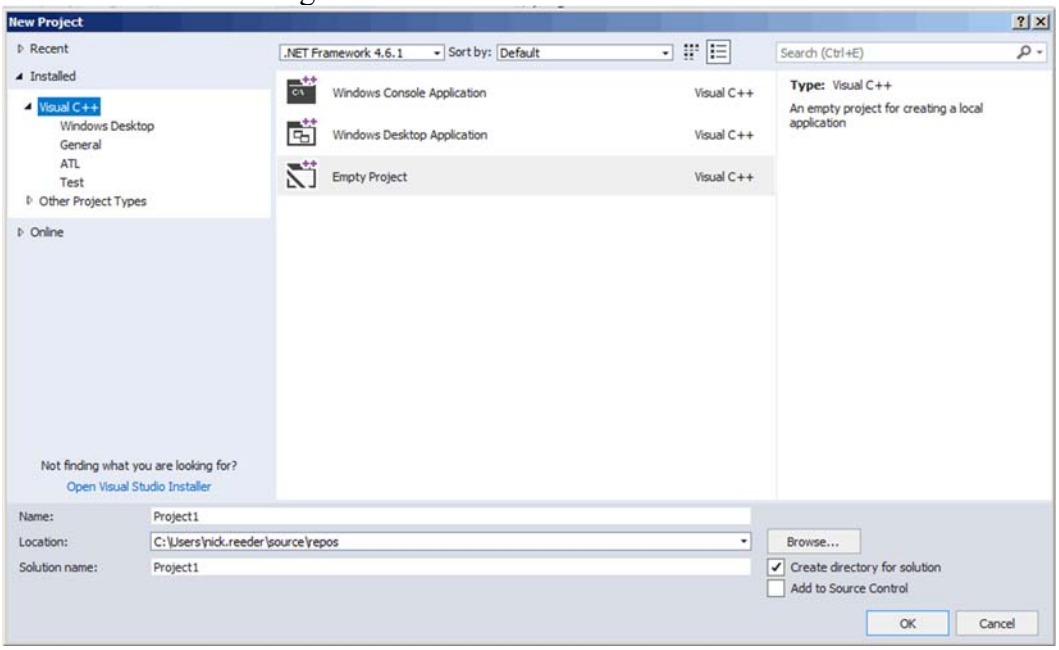
1. Start Visual Studio on your computer. If this is the first time you have used Visual Studio on this computer, you will see the following dialog box after a brief delay.



2. In the drop-down box named **Development Settings**, select **Visual C++**. Then press the **Start Visual Studio** button. After a delay, you should see a screen that looks like this:



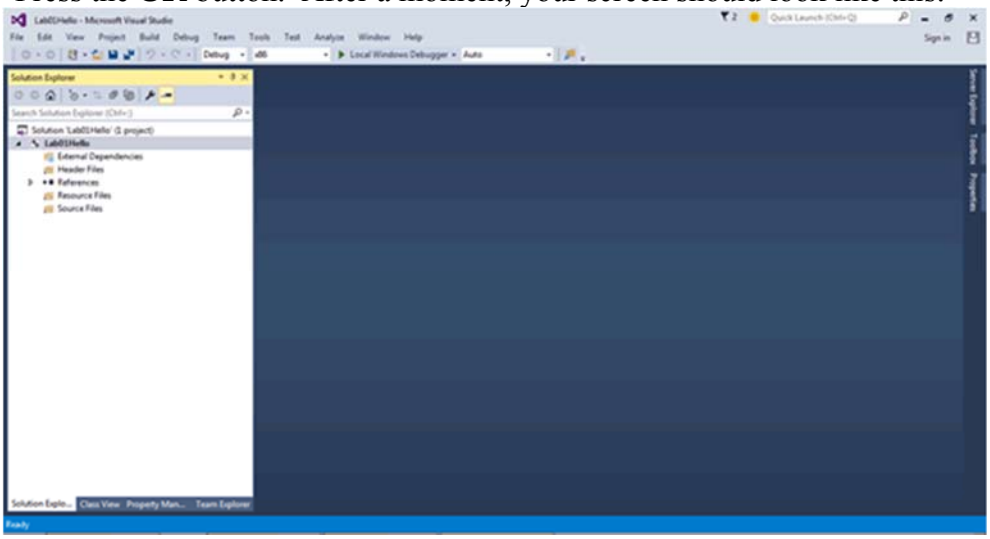
3. Notice that the "Solution Explorer" window in the left half of the screen is blank. This means that no project is currently opened. To start a new project, choose **File > New > Project…** on Visual Studio's menu bar. You'll see a dialog box that looks something like this:



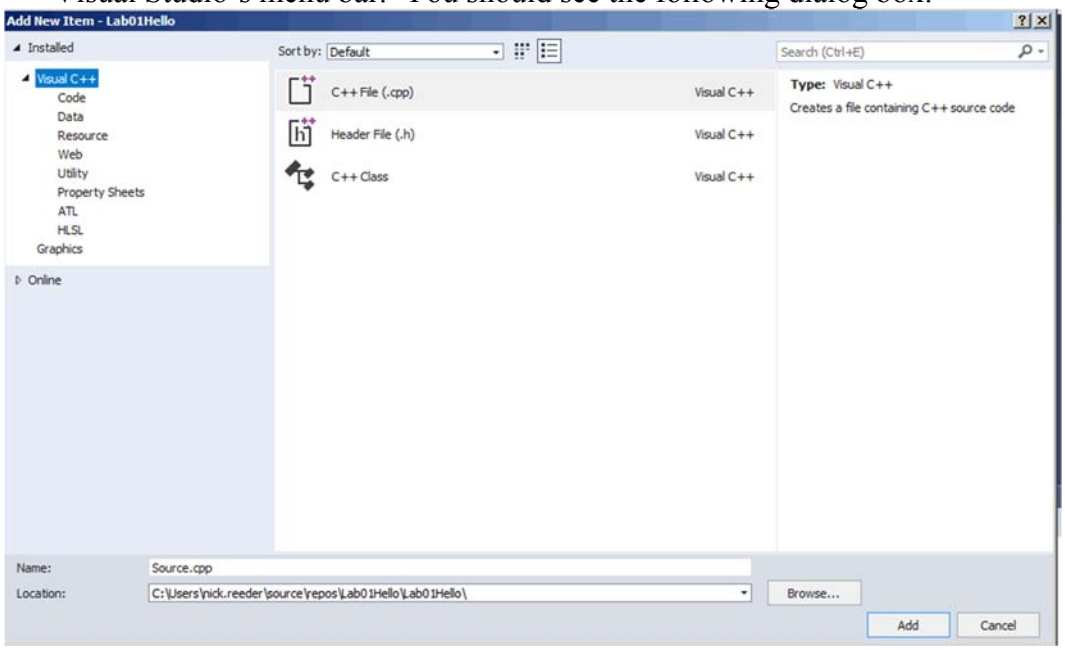
4. Make sure that **Visual C++** is selected (under **Installed**) in the dialog box's left-hand window. Next, make sure that **Empty Project** is highlighted in the dialog box's central window. Next, turn your attention to the three text fields near the bottom of the dialog box:
    - Notice that the **Location** is a folder named **repos** nested inside the **source** folder inside your **Users** folder on the computer's hard drive. This is what we want, so do not change the Location.

- Also, the fields labeled **Name** and **Solution Name** probably both say something like **Project1**. Change the **Name** to **Lab01Hello**. Notice that as you type the new **Name**, the **Solution Name** also automatically changes to **Lab01Hello**. This is what we want.

5. Press the **OK** button. After a moment, your screen should look like this:

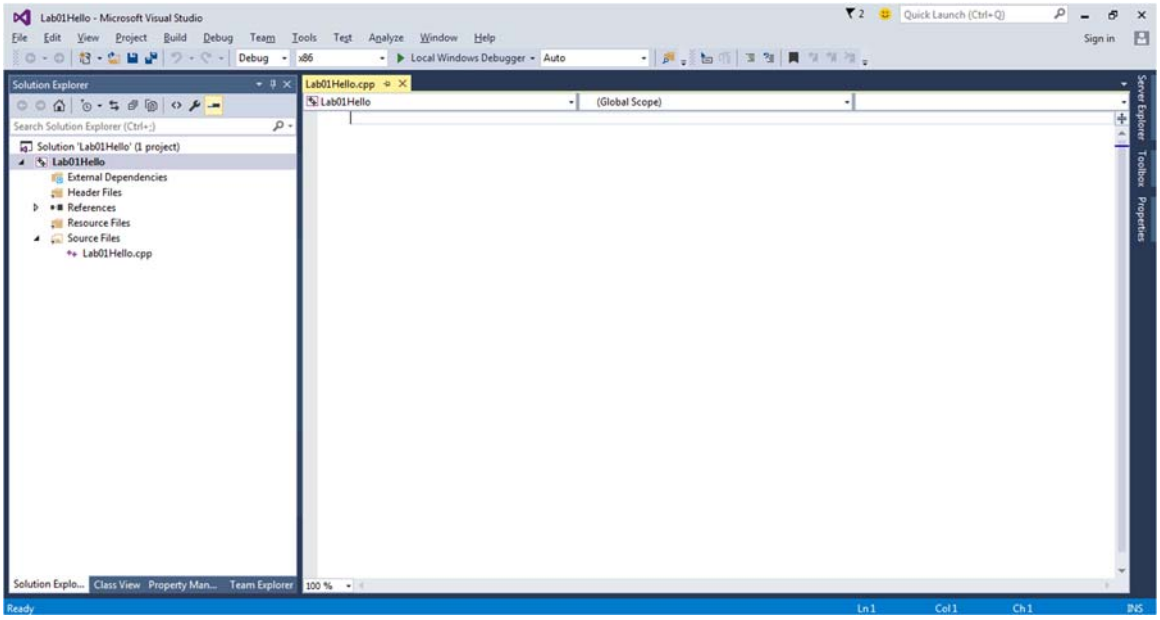


6. Notice that the left-hand **Solution Explorer** window, which was previously blank, now contains several items, some of which are labeled with your project's name (Lab01Hello). So you've created a new project, which already contains some files that Visual Studio created for you automatically. To create the file that will hold your code for this program, choose **Project > Add New Item…** on Visual Studio's menu bar. You should see the following dialog box:



7. Make sure that **C++ File (.cpp)** is highlighted in the dialog box's central window. Next, turn your attention to the two text fields near the bottom of the dialog box:
    - Notice that the **Location** is a folder named **Lab01Hello** nested deep inside your **Users** folder on the computer's hard drive. This is what we want, so do not change the Location.
    - Also, the **Name** is **Source.cpp**. This name would work just fine, but it will be more convenient for us to have this file name match the project's name, so change the **Name** to **Lab01Hello.cpp**.

8. Press the **Add** button. Your screen should look like this:

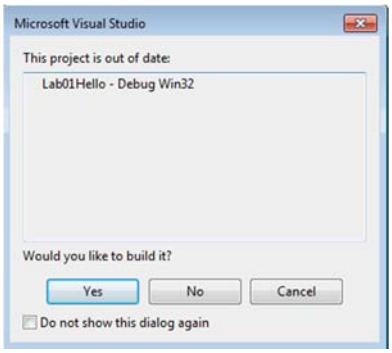


9. Notice two important changes.
    - First, your newly created file (Lab01Hello.cpp) now appears as the last item listed in the left-hand **Solution Explorer** window. The simple programs that you'll write initially will just have one source file, but more complicated programs will have many files listed here.
    - Second, most of the screen is now devoted to a blank window in which you will type your code. The cursor should already be blinking inside this window, ready for you to start typing.

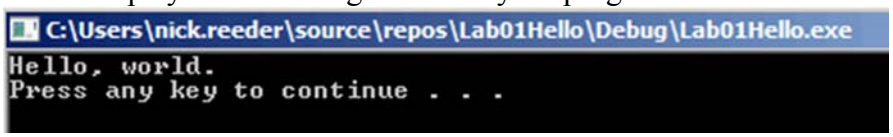
10. Type the following code (next page), **but type your own name as the Author, and use today's date**:

```
Lab01Hello.cpp
Lab01Hello                                  (Global Scope)
 1  //*****************************************************************
 2  //Name: Lab01Hello
 3  //Author: Nick Reeder
 4  //Date: 07/13/2015
 5  //This program prints the message "Hello, world" in a console window.
 6  //*****************************************************************
 7
 8  #include <iostream>
 9
10  using namespace std;
11
12  int main() {
13      cout << "Hello, world.\n";
14
15      system("pause");
16      return 0;
17  }
18
```

11. To compile and run your program, choose **Debug > Start Without Debugging** from Visual Studio's menu bar. You should see the following dialog box:



12. This is an annoying box that you probably don't want to see in the future, so I recommend checking the **Do not show this dialog again** checkbox. Then press the **Yes** button. If you typed the program's code correctly, a new window will open and display the following results of your program:



13. If your program worked, congratulations! Press any key to continue. If your program did not work, then examine the error messages in Visual Studio's **Output** window (at the bottom of the screen) to figure out what you did wrong.

14. When you are satisfied that your program works correctly, ask me to check it. I'll sign my initials below.

_____

15. Then close this project by choosing **File > Close Solution** from Visual Studio's menu bar.

Part 2. More Programs

Next you'll enter and run some longer programs. I don't expect you to understand exactly how these programs work, but you should be able to enter them, run them, and tell whether or not they're working correctly. All of the programs are taken from the textbook, and you will understand them better as we study later chapters in the book.

1. Following the same steps that you followed above (starting with Step 3), create a new project named **Lab01Rectangle**. After you've created the project, create a new source file named **Lab01Rectangle.cpp**. Then type the following code (but type your own name as the Author and use today's date). This code is from the textbook's Example 2−1.

```
//*************************************************************
// Name: Lab01Rectangle
// Author: Nick Reeder
// Date: 07/13/2015
// This program computes and outputs the perimeter and area of
// a rectangle based on hard-coded values of the length and width.
//*************************************************************

#include <iostream>

using namespace std;

int main()
{
    double length;
    double width;
    double area;
    double perimeter;

    cout << "Program to compute and output the perimeter and "
        << "area of a rectangle." << endl;

    length = 6.0;
    width = 4.0;
    perimeter = 2 * (length + width);
    area = length * width;

    cout << "Length = " << length << endl;
    cout << "Width =  " << width << endl;
    cout << "Perimeter = " << perimeter << endl;
    cout << "Area = " << area << endl;

    system("pause");
    return 0;
}
```

2. Run your program. When you are satisfied that it works correctly, ask me to check it.

3. Close this project by choosing **File > Close Solution** from Visual Studio's menu bar.

4. Do the same for the following program, using the name **Lab01DataTypes** for the project and the name **Lab01DataTypes.cpp** for the source file. This code is similar to the textbook's Example 2−13.

```
//*************************************************************
// Name: Lab01DataTypes
// Author: Nick Reeder
// Date: 07/13/2015
// This program illustrates the int, double, char, and
// string data types.
//*************************************************************

#include <iostream>
#include <string>

using namespace std;

int main()
{
    int num1, num2;
    double sale;
    char first;
    string str;

    num1 = 4;
    cout << "num1 = " << num1 << "\n";

    num2 = 4 * 5 - 11;
    cout << "num2 = " << num2 << "\n";

    sale = 0.02 * 1000;
    cout << "sale = " << sale << "\n";

    first = 'D';
    cout << "first = " << first << "\n";

    str = "It is a sunny day.";
    cout << "str = " << str << "\n";

    system("pause");
    return 0;
}
```

5. When your program works correctly, ask me to check it. Then close this project by choosing **File > Close Solution**.

6. Do the same for the following program, using the name **Lab01ConvertLength** for the project and the name **Lab01ConvertLength.cpp** for the source file. This code is from the textbook's Programming Example on page 96.

```
//*********************************************************************
// Name: Lab01ConvertLength
// Author: Nick Reeder
// Date: 02/25/2016
// Given a length in feet and inches, this program
// converts the length to inches, and also to centimeters.
//*********************************************************************

#include <iostream>

using namespace std;

//named constants
const double CENTIMETERS_PER_INCH = 2.54;
const int INCHES_PER_FOOT = 12;

int main()
{
    //declare variables
    int feet, inches;
    int totalInches;
    double centimeters;

    cout << "Enter two integers, one for feet and "
        << "one for inches: ";
    cin >> feet >> inches;
    cout << endl;

    cout << "The numbers you entered are " << feet
        << " for feet and " << inches
        << " for inches." << endl;

    totalInches = INCHES_PER_FOOT * feet + inches;

    cout << "The total number of inches = "
        << totalInches << endl;

    centimeters = CENTIMETERS_PER_INCH * totalInches;

    cout << "The number of centimeters = "
        << centimeters << endl;

    system("pause");
    return 0;
}
```

7. When you run this program, it will prompt you to enter two integers. Type two integers—let's say 15 and 7—separated by a space, and then press the keyboard's **Enter** key. When the program works correctly, ask me to check it.

8. After closing the previous project, do the same for the following program, using the names **Lab01MakingChange** and **Lab01MakingChange.cpp**. This code is from the textbook's Programming Example on page 99.

```
//*************************************************************
// Name: Lab01MakingChange
// Author: Nick Reeder
// Date: 07/13/2015
// Given an amount of change expressed in cents, this program computes the
// number of half-dollars, quarters, dimes, nickels, and pennies to be returned.
//*************************************************************
#include <iostream>
using namespace std;

const int HALF_DOLLAR = 50, QUARTER = 25;
const int DIME = 10, NICKEL = 5;

int main() {
    int change;

    cout << "Enter change in cents: ";
    cin >> change;
    cout << endl;

    cout << "The change you entered is " << change << endl;

    cout << "The number of half-dollars to be returned "
        << "is " << change / HALF_DOLLAR << endl;
    change = change % HALF_DOLLAR;
    cout << "The number of quarters to be returned is "
        << change / QUARTER << endl;
    change = change % QUARTER;
    cout << "The number of dimes to be returned is "
        << change / DIME << endl;
    change = change % DIME;
    cout << "The number of nickels to be returned is "
        << change / NICKEL << endl;
    change = change % NICKEL;
    cout << "The number of pennies to be returned is "
        << change << endl;

    system("pause");
    return 0;
}
```

9. When you run this program, it will prompt you to enter an amount of change in cents. Type an integer—let's say 583—and then press the keyboard's **Enter** key. When the program works correctly, ask me to check it.

10. After closing the previous project, do the same for the following program, using the names **Lab01ClassifyingNumbers** and **Lab01ClassifyingNumbers.cpp**. This code is from the textbook's Programming Example on page 303.

```
//*************************************************************
// Name: Lab01ClassifyingNumbers
// Author: Nick Reeder
// Date: 07/13/2015
// Given 20 integers, this program displays the number of even
// integers, the number of odd integers, and the number of zeros.
//*************************************************************
#include <iostream>
#include <iomanip>
using namespace std;
const int N = 20;
int main()
{
    int counter, number;
    int zeros = 0, odds = 0, evens = 0;
    cout << "Please enter " << N << " integers, "
        << "positive, negative, or zeros."
        << endl;
    cout << "The numbers you entered are:" << endl;
    for (counter = 1; counter <= N; counter++)
    {
        cin >> number;
        cout << number << " ";
        switch (number % 2)
        {
        case 0:
            evens++;
            if (number == 0)
                zeros++;
            break;
        case 1:
        case -1:
            odds++;
        }
    }
    cout << endl;
    cout << "There are " << evens << " evens, "
        << "which includes " << zeros << " zeros."
        << endl;
    cout << "The number of odd numbers is: " << odds
        << endl;
    system("pause");
    return 0;
}
```

11. When you run this program, it will prompt you to enter 20 integers. After you type 20 integers and press **Enter**, the program will tell you how many even integers you entered, how many odd integers, and how many zeros. When the program works correctly, ask me to check it.

12. Close this project by choosing **File > Close Solution**. Also close Visual Studio by choosing **File > Exit**.

Part 3. Uploading Your Source Files

If you followed my directions above, your projects are all saved in your Users folder on the computer's hard disk. To finish this lab, follow these steps to copy the projects to your flash drive for safekeeping and to upload your source-code files to the Lab 1 dropbox on the course website.

1. In Windows, navigate to the folder named **Users** folder. Within your **Users** folder, open the **source** folder. Within that folder, open the **repos** folder. Within that folder, you should see a folder for each of the programs you wrote in this lab. There should be six such folders, named **Lab01Hello**, **Lab01Rectangle**, and so on.

2. Copy these six folders onto your flash drive. I'll let you decide how to organize your flash drive, but you'll probably want to save these into a folder named something like **EGR2261** on your flash drive.

3. Still in Windows, open one of the folders—say, **Lab01Hello**—that you just copied to your flash drive. You should see several files and folders.
   - One of these files should be named **Lab01Hello.sln.** *This is the file that you would open if you wanted to re-open the entire Lab01Hello project.*
   - Also, one of the folders should be named **Lab01Hello**. (This may be confusing, because this is a folder named **Lab01Hello** inside another folder with exactly the same name.)

4. Open the **Lab01Hello** folder. (So you're now inside **Users > YOURNAME > source > repos > Lab01Hello > Lab01Hello**.) You should see several more files and folders. One of these files should be named **Lab01Hello.cpp**. *This is the file that contains the source code that you typed in by hand, and it's the one that I want you to upload to the course website.*

5. Now that you know where to find your projects and source-code files, upload all six source-code files (named **Lab01Hello.cpp**, **Lab01Rectangle.cpp**, **Lab01DataTypes.cpp**, **Lab01ConvertLength.cpp**, **Lab01MakingChange.cpp**, and **Lab01ClassifyingNumbers.cpp**) to the Lab 1 dropbox on the course website.

******* This lab had 6 named programs for me to check. If you didn't finish all of these during class, finish them after class. Then upload your source-code files for all 6 programs (including the ones that I checked in class) to the course website by the due date. Also turn in your lab sheets at the beginning of class.********