Tennessee TECH

CSC 1300 LAB 1

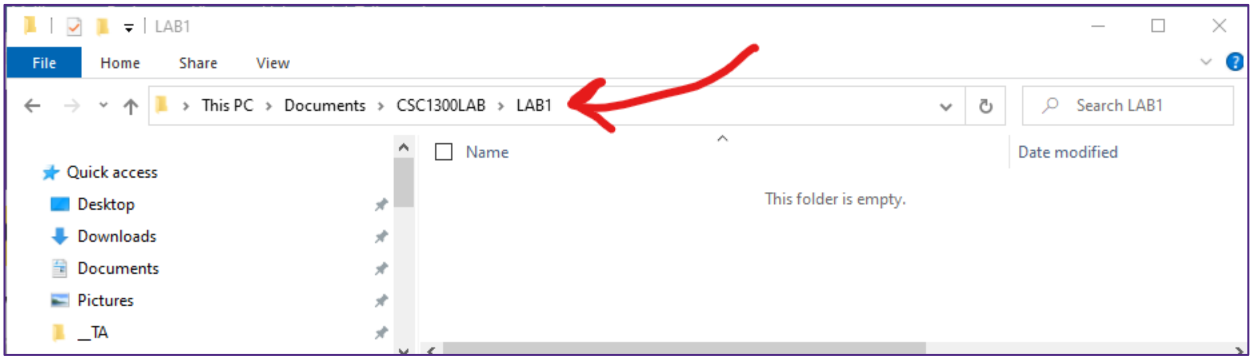
Fall 2023

STEP ONE: SET UP YOUR LAPTOP!

If you haven't already, make sure you have followed the directions in the "**1300_GETTING YOUR COMPUTER SET UP**" document. It may be in your lecture ilearn class, but also the document is in your lab 1 assignment, too. There is also a video available for Windows users: https://www.youtube.com/watch?v=hLo6vrsClT0&t=2s

STEP TWO: WRITE & RUN YOUR FIRST PROGRAM!

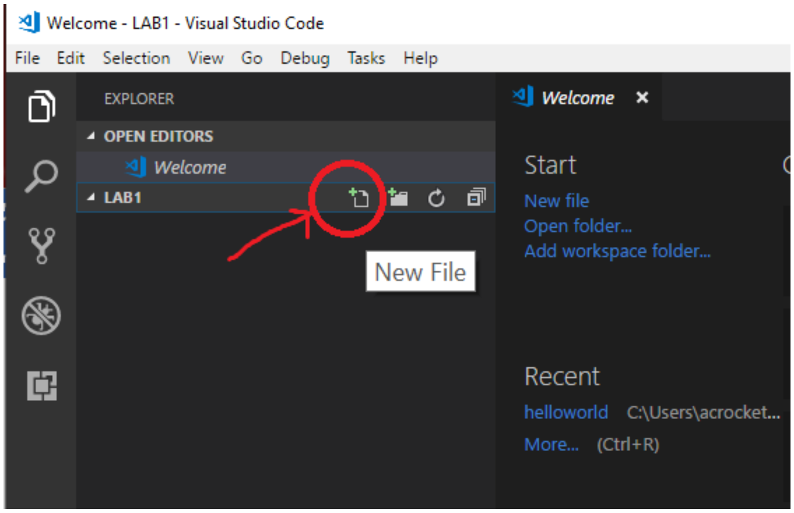
1. In your Documents folder on your computer, create a folder called **CSC1300LAB**. Then, inside the CSC1300LAB folder, create a **LAB1** folder.



2. Open **Visual Studio Code**.

3. Click on **File** and then select **Open Folder**. Select the **LAB1** folder that you just created.

4. Click on the New File icon to create a new file.



5. Type **lab1.cpp** to create a file with the name lab1.cpp.

6. Type the following lines of code exactly the way they are below in your lab1.cpp source file but **beside "Author", type your first and last name**.

```
/*
      Title:      lab1.cpp
      Author:
      Date:       January 2023
      Purpose:    Convert Celsius to Fahrenheit
*/

#include <iostream>
using namespace std;

int main()
{
      double fahrenheit, celsius;
      cout << "\n\nWhat is the temperature in Celsius?  ";
      cin >> celsius;
      fahrenheit = celsius * (9.0/5.0) + 32;
      cout << "\nRESULT:  " << celsius << " degrees Celsius is ";
      cout << fahrenheit << " degrees Fahrenheit.\n\n";
      return 0;
}
```

Sample Output

User input is highlighted in yellow.

```
What is the temperature in Celsius?  -4.8

RESULT:  -4.8 degrees Celsius is 23.36 degrees Fahrenheit.
```

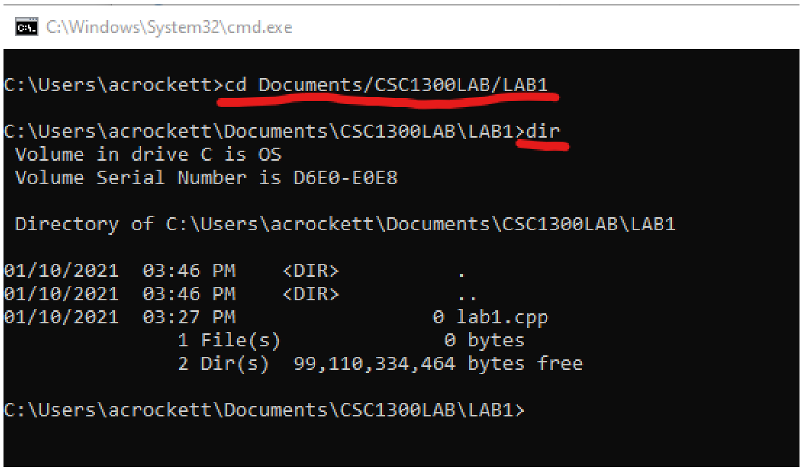
7. Now, open either the **Command Prompt** (on WINDOWS) or the **Terminal** (on MAC). Do not use an IDE for this assignment.

8. Using the appropriate DOS or LINUX commands, navigate to the location where your program is located on your computer in your Command Prompt/Terminal.
   a. You are likely already in your C: drive and so you now need to navigate to your Documents/CSC1300LAB/LAB1 folder. You can do this with the cd command, which stands for "change directory".
      i. Type **`cd Documents`**
      ii. Type **`cd CSC1300LAB`**
      iii. Type **`cd LAB1`**

Helpful Hints

You may also change directly into the LAB1 folder from the C: drive by typing **`cd Documents/CSC1300LAB/LAB1`**.
Also, if you ever need to back up a directory, instead of go forward, you can type **`cd ..`** and that will bring you back to the parent directory.

9. You should now be inside the LAB1 folder in the Command Prompt/Terminal and this should be the folder that contains your source file named **lab1.cpp**. To make sure, you can type:
WINDOWS- **`dir`**
MAC- **`ls`**
and this will list out the files and directories in your current folder. I can see by doing this that **lab1.cpp** is inside this directory.



```
C:\Users\acrockett>cd Documents/CSC1300LAB/LAB1

C:\Users\acrockett\Documents\CSC1300LAB\LAB1>dir
 Volume in drive C is OS
 Volume Serial Number is D6E0-E0E8

 Directory of C:\Users\acrockett\Documents\CSC1300LAB\LAB1

01/10/2021  03:46 PM    <DIR>          .
01/10/2021  03:46 PM    <DIR>          ..
01/10/2021  03:27 PM                 0 lab1.cpp
               1 File(s)              0 bytes
               2 Dir(s)  99,110,334,464 bytes free

C:\Users\acrockett\Documents\CSC1300LAB\LAB1>
```

10. Now you can compile. The compiler command to compile a single file into an executable (runnable) file is:
    `g++ sourcefilename.cpp –o executablefilename`

11. Where **`sourcefilename`** is the name of the file you want to compile and **`executablefilename`** can be anything you want it to be. I usually name my executable file the same name as the source file. You can also leave off the **`executablefilename`** and your executable file will be created as **`a.exe`** on windows or **`a.out`** on a Mac.
12. So, to compile this lab assignment, you type
    `g++ lab1.cpp –o lab1`

    If I typed **`dir`** now (or **`ls`**) then I will see both **`lab1.cpp`** and **`lab1.exe`** (Windows) or **`lab1.out`** (Mac).

13. Run your program.
    MAC command to run program- `./executablefilename`
    WINDOWS command to run program- `executablefilename`

    For this lab, you will replace **`executablefilename`** with **`lab1`**:
    MAC- `./lab1`
    WINDOWS- `lab1`

14. **Zip** your **LAB1** folder, which contains **`lab1.cpp`**. In ALL labs, you will submit your work in a zipped folder.
    a. **WINDOWS** – select the LAB1 folder you want to zip. Then, right-click on the folder, then point to **Send to** and select **Compressed (zipped) folder**. You can find the newly created .zip folder in the same directory.
    b. **MAC** – select the LAB1 folder you want to zip. Then, right-click on the folder and select **Compress Items**. You can find the newly created .zip folder in the same directory.

WHAT TO TURN IN

Upload your zipped file containing **lab1.cpp** to your LAB ilearn course under **Assignments**, and then **LAB 1**.

You do NOT have to turn in your executable (.exe) file.

LAB ASSIGNMENT RUBRIC

The following grading rubric will be used for **all** lab assignments. The maximum number of points you can earn on each assignment is 120 points.

Category	Excellent (20 points)	Good (15 points)	Fair (10 points)	Poor (5 points)
PROGRAM EXECUTION	Program executes correctly with no syntax or runtime errors.	Program compiles and executes but has a minor (easily-fixed) runtime error.	Program doesn't compile due to minor error or errors.	Program does not compile or execute (many errors).
CORRECT OUTPUT	Program displays correct output with no errors.	Output has minor errors.	Output has multiple errors.	Output is incorrect.
DESIGN OF OUTPUT	Program output is perfect with no misspellings and is visually easy-to-read and understand.	Program output is pretty good. There are one or two minor issues (Example: poor use of whitespace)	Program output fair. There are some issues.	Program output is very difficult to read and not user friendly at all.
FOLLOWED SPECIFICATIONS OF ASSIGNMENT	Completely followed all specifications in assignment and used all the correct programming constructs.	Followed specifications in assignment with a minor exception.	Mostly followed specifications. Had a few issues with not following the assignment.	Did not follow the specifications at all or there were multiple differences.
FOLLOWED GOOD PROGRAMMING PRACTICE	Programmer followed good programming practices.			Poor programming practices in code.
DOCUMENTATION & READABILITY OF CODE	Program is well commented and code is readable with appropriate indentions. Program source files contain comment block header at top with source code title, author, date, and purpose.	Program readability is good but there are a few issues with indentions or lacking some comments.	Program readability is fair. Maybe lacking comments or inconsistent indentions.	Horrible readability of code. Missing comments and inconsistent indentions.