CS1580 Lab 2: Errors Strike Back

1 Lab Objectives

This week we will:

1. See common errors when writing C++ programs and how to fix them.
2. Practice good coding standards, and follow the CS1570 Coding standard.
3. Try out some basic user input commands.
4. Submit your results using the cssubmit tool.

2 Lab Preparation

First, this start by making a folder for this lab. Only material for this week's lab should be placed in this folder. For example, if you made a CS1580 folder last week, you'd probably want to issue the following commands:

- cd SDRIVE
- cd cs1580
- mkdir lab2
- cd lab2

The above commands changed into your CS1580 folder, made a new subfolder for lab2 and entered that folder. Remember, you can always see the contents of a directory by issuing the ls command.

3 Correcting Common C++ Errors

1. On Blackboard, under Lab 2, you will find a BMICalcWithErrors.cpp.
2. Download BMICalcWithErrors.cpp from Blackboard and place it in your lab2 folder. (You may use "mv src dest" command)
3. Open BMICalcWithErrors.cpp and put your name in the header.

(Use "jpico BMICalcWithErrors.cpp" command)

4.Save and rename file to lab2.cpp

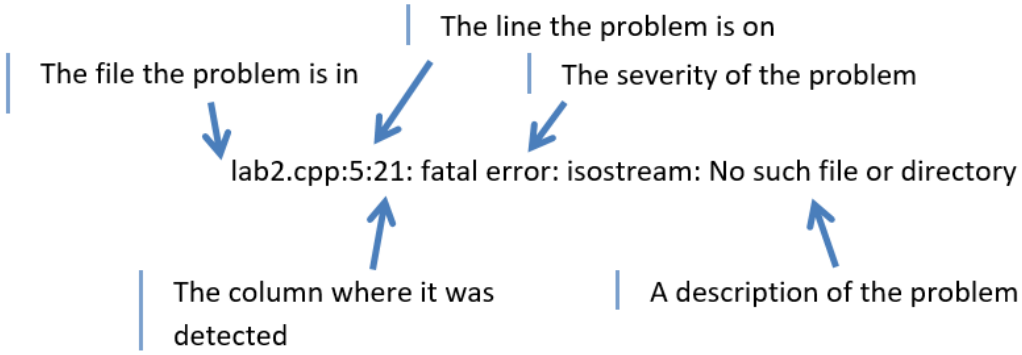
4. Correct the compiler errors in the program. (fg++ lab2.cpp –o lab2)
5. Run the program. Does it calculate the BMI correctly? (lab2)
6. Make sure the program meets the coding standard.
7. Read the grading section for instructions on how to submit the program.

This file is a simple program that calculates BMI based on a simple formula. It has several common mistakes. If you try to compile this program, you will see a variety of compilation errors.

3.1 Compiler Errors

Hint: Always start with the first compiler error. It helps to attempt to compile after each fix as well, as many errors can cascade, resulting in several error reports from one mistake. Here is a list of some errors and what they mean to help you fix the program:

A compiler error in g++ (Our C++ compiler) follows a certain structure:



- isostream: No such file or directory

  A #include statement is referencing a file or library that does not exist.

- expected ';' before ...

  A line is missing a semi-colon.

- No match for operator ...

  A *type* doesn't support an operation being performed on it. (See cin and cout below!)

- missing terminating " character

  A closing quote is missing.

3.2 cin and cout

cin and cout are special variables in C++ that take input from the user and display information to the user, respectively. They are *streams*. The << and >> show how data flows into or out from these streams.

- Data comes in from a cin and the >> characters should always be used with cin.
- Data goes out to the screen with a cout and the << characters should always be used with a cout.

3.3 Debugging

Once your program runs, try it out with some numbers you know how to multiply.

C++ uses symbols to represent math operations:

+	Add
-	Subtract
*	Multiply
/	Divide
%	Modulo

Do you see anywhere in the program that looks like we are doing a math operation? Is the right symbol being used?

3.4 Coding Standard

The coding standard is an 11-page document listed on Price's website. Eventually, you'll be expected to follow the entire coding standard. For now, however, make sure your lab meets the following requirements:

- The comment section at the top of your lab is filled out and all the appropriate sections are listed (see below).
- The contents of the main function are indented 2 spaces.
- Curly braces { and } go on their own lines and the pair of braces are aligned vertically.

3.4.1 Top Section

A correctly formatted top section should look similar to this:

```
/* Programmer: Your Name
   Instructor: Maria
   Section: G
   Date: Today's date
   Description: What the function of the program is.
*/
```

3.4.2 Indentation and Curly Braces

Anything between a pair of curly braces { and } should be indented 2 spaces. Each time you open a curly brace, indent over two more spaces. Do not use tabs—these print differently than they are viewed, and will result in line wraps!

Additionally, curly braces should always go on their own line.

```
int main()
{
  // I'm inside some curly braces, so I've indented two spaces!
```

```
  cout<<"I'm going to change the world!"<<endl;
  return 0;
}
```

3.5 Grading

To get a 100% on this lab your code should:

- Have your name entered in the header.
- Compile successfully.
- Calculate BMI correctly.
- Meet the portion of the coding standard listed in this lab

To submit this lab, you will use the cssubmit tool. To do this, in the folder with your lab2.cpp issue this command:

cssubmit 1580 g 2