CSCI 1113: Introduction to C/C++ Programming for Scientists and Engineers Homework 1 Fall 2015

Due Date: Friday, September 25th before 3:35pm (before class begins).

Purpose: The purpose of Homework 1 is to practice your C++ program writing skills by solving problems requiring C++ arithmetic operators, simple input and output, differing variable types, and selection (i.e. `if`) statements. Computational problem-solving is a bonus.

Instructions: This is an individual homework assignment There are two problems worth 20 points each. Solve each problem by yourself and submit each solution as a separate C++ source code file via Moodle.

A few more important details

1. Unlike the computer lab exercises, this is not a collaborative assignment. You must design, implement, and test the solution to each problem on your own without the assistance of anyone other than the course instructor or TAs. In addition, you may not include solutions or portions of solutions obtained from any source other than those provided in class: examples from the textbook, lectures, or code you and your partner write to solve lab problems. Otherwise obtaining or providing solutions to any homework problems for this class is considered academic misconduct. See the "collaboration rules" file on the Moodle page for more details, and ask the instructor if you have questions.
2. Because all homework assignments are submitted and tested electronically, the following are important:
   - You follow any naming conventions mentioned in the homework instructions.
   - You submit the correct file(s) through Moodle by the due deadline.
   - You follow the example input and output formats given in each problem description.
   - Regardless of how or where you develop your solutions, your programs compile and execute on cselabs computers running the Linux operating system.
3. Here are some hints on this assignment:
   - Start early. One common problem in this class is students waiting until soon before the assignment is due, and then running out of time. By the time this homework is posted you will have seen enough C++ in lecture and the textbook reading to begin the problems here.

- For each problem, make sure you understand it and design a step-by-step solution before starting to write C++ code.
- Ask questions during office hours as needed.
- If you have lab early in the week, remember what you did in lab: it will usually help in doing the homework. If you have lab later in the week, start working on the homework before lab. That way you can identify parts of the lab that will be similar to what is needed in the homework. (Notes: (i) there are advantages and disadvantages to having labs early or later in the week. So students having lab early don't have an inherent advantage over students having labs later, and vice versa. (ii) Labs are for doing lab problems, not homework problems. If you need help with homework problems please stop by during any of the TAs' office hours. (iii) Homework problems will often be based on the material in previous weeks' labs rather than the current week's lab. So start working on homework well in advance of the due date, not the day before the homework is due.)

4. Please make your source code easy to read. In particular, start each file with comments giving the program name, your name, and the date you completed the program. And use good coding style (as discussed in lecture and the textbook) to ensure that the graders can understand your code.

5. The problem descriptions will usually show at least one test case and the resulting correct output. However, you should test your program on other test cases (that you make up) as well. Making up good test cases is a valuable programming skill, and is part of ensuring your code solution is correct.

6. Have fun! Solving computational problems and writing code can be difficult and, at times, frustrating. But it can also be fun and satisfying.

Problem A: Partial Pyramid Base (20 points)

Often in science, engineering, and related fields we need to work with areas or volumes. A mechanical engineer might want to know the surface area of an engine part; a hydrologist might want to know the volume of water in a pond; an aerospace engineer might want to know the surface area of an aircraft part.

Suppose you have a mold for a truncated square pyramid. It has an top side length of `b`, an base side length of `a`, and height `h`. What is the volume of the mold? What is the surface area of the mold?

Write a C++ program that asks the user to input the upper and base side lengths and the height (in meters). The calculate and output, using the values, equations, and format below, the following values:

- Volume of the truncated square pyramidal mold: $\frac{1}{3}(a^2 + ab + b^2)h$
- Surface area: of the mold: $a^2 + b^2 + 2(a + b)\sqrt{(\frac{a-b}{2})^2 + h^2}$
- Cost of concrete to fill mold (per m^3): \$120
- Cost of labor to build (per m^3): \$85
- Total cost of job

Here is an example. In your input and output, follow this input and output format exactly (i.e. same input and output messages, **whitespace**, two digits to the right of the decimal place, etc.). Remember that the number formatting commands are given in the (9th edition of the) textbook on page 56.

```
Input the top side length (in m):  .5
Input the base side length (in m):  1
Input the height (in m):  1

Volume of the mold (in cubic meters):  0.58
Surface area of the mold (in square meters):  4.34
Cost of concrete for mold:  $70.00
Cost of labor:  $49.58
Total cost:  $119.58
```

Test your program using not only the example data above, but other cases as well. And revise your program until you are sure it is correct.

When you are done, name the source code file <username>_1A.cpp. Here you replace <username> with your U of M email address; for example, if your email address is smithx1234@umn.edu, your file should be named smithx1234_1A.cpp. Then submit your program using the HW 1 Problem A submission link in Moodle.

It is important that you follow the file naming conventions very carefully. For example, note your username should be all in lowercase, you should not include "@umn.edu", the file name should contain an underscore (not a dash), the 'A' in the "1A" part is upper case, the extension is .cpp, etc. Following rigorous naming conventions is something computer programmers often must do in "real life" programming, and so submitting your program with the correct name is part of doing this assignment correctly.

Problem B: Imperial vs. Metric (20 points)

As discussed, the choice of Imperial vs. Metric measurement units is an important one. While the U.S. tends to use Imperial units (much to my chagrin), the rest of the world (mostly) uses metric. What's more, there are two typical measurements in Imperial volume: cubic feet and cubic yards.

Let's revisit Problem A and extend its capability. We will add the ability for the user to select which measurement type to use, and perform calculations based on this choice.

Modify your C++ program from Problem A by adding the functionality that the program first asks the user if they wish to input Imperial ('i') or Metric ('m') unit values. If the user select the Imperial measurement type, the program should ask if the user would like to use cubic feet or cubic cards. Then your program should proceed as before, but using the correct formula for the user's measurement type selection.

The measurement type does not effect the formula's for calculating the volume or surface area, but does effect the price's of concrete. The price of concrete is \$90 / yd^3, and the price of labor is \$71 / yd^3. If the user selects to input Imperial measurements (regardless of whether they choose to in put in yards or feet), the output should be in cubic **yards**.

Your program's input and output should follow the format shown in the examples below, including the exact input and output prompts, use of whitespace, comma separated output, and float output format with two places to the right of the decimal point.

Example 1:

```
Are your measurements in Imperial or Metric (i/m)?  m
Input the top side length (in m):  .5
Input the base side length (in m):  1
Input the height (in m):  1

Volume of the mold (in cubic meters):  0.58
Surface area of the mold (in square meters):  4.34
Cost of concrete for mold:  $70.00
Cost of labor:  $49.58
Total cost:  $119.58
```

Example 2:

```
Are your measurements in Imperial or Metric (i/m)?  i
Are your measurements in yards or feet (y/f)?  y
Input the top side length (in yd):  .55
Input the base side length (in yd):  1.09
Input the height (in yd):  1.09

Volume of the mold (in cubic yards):  0.76
Surface area of the mold (in square yards):  5.17
Cost of concrete for mold:  $68.35
Cost of labor:  $53.92
Total cost:  $122.26
```

Example 3:

```
Are your measurements in Imperial or Metric (i/m)?  i
Are your measurements in yards or feet (y/f)?  f
Input the top side length (in ft):  1
Input the base side length (in ft):  3
Input the height (in ft):  2.5

Volume of the mold (in cubic yards):  0.40
Surface area of the mold (in square yards):  3.50
Cost of concrete for mold:  $36.11
Cost of labor:  $28.49
Total cost:  $64.60
```

Example 4:

```
Are your measurements in Imperial or Metric (i/m)?  j
Invalid option.
```

Test your program using not only the example data above, but other cases as well. And revise your program until you are sure it is correct. When you are done, name the source code file <username>_1B.cpp. Here you replace <username> with your U of M email address; for example, if your email address is smithx1234@umn.edu your file should be named smithx1234_1B.cpp. Remember to follow this naming convention diligently. Then submit your file using the Homework 1 Problem B link on Moodle.