CS 111 - Homework 8

Deadline

11:59 pm on **Friday, November 1, 2024**

Purpose

To practice with some C++ basics, including following design recipe steps for designing and writing simple C++ functions, tested with the help of a simple `main` function within a testing C++ program.

How to submit

You complete **Problems 1, 2, 3, and 4** on the course Canvas site (short-answer questions on various C++-related topics), so that you can see if you are on the right track.

Then, you will submit your work for **Problems 5** onward, in your files **`111hw8.cpp`** and **`111hw8-out.txt`**, on the course Canvas site.

Submit your **`111hw8.cpp`** file-in-progress early and often!

- Each time you submit a version of your **`111hw8.cpp`**, IF that version currently compiles, also submit a copy of the example output from running that latest version in file **`111hw8-out.txt`**.
- Be careful that each submitted **`111hw8-out.txt`** was created by running the compiled version of the **`111hw8.cpp`** file submitted along with it.

Important notes - 5 points

- You should structure your C++ program for this homework like the programs from the Week 8 and Week 9 Lab Exercises, **`lab8.cpp`** and **`lab9.cpp`**, and like the posted program **`111lect09-2.cpp`** posted in the In-Class Examples for Week 9 Lecture 2.
- You are still expected to follow the Design Recipe for all **functions** that you design/define.
  - Remember that a C++ "graphic design recipe helper" has been posted on the course Canvas site and on the public course web site, "translating" the design recipe steps into C++ syntax.
  - Remember, you will receive **significant** credit for the signature, purpose, header, and tests/test expressions portions of your functions.
  - Typically you'll get at least half-credit for a correct signature, purpose, header, and tests/test expressions, even if your function body is not correct.
  - (and, you'll **lose at least half-credit** if you omit these or do them poorly, even if your function body is correct).
- That said -- remember that, in C++:
  - use **C++ type names** in the signatures of C++ functions
  - write tests as `bool` expressions, typically using == or < . For example,

    ```
    my_funct(3) == 27
    abs(my_dbl(4.7) - 100.43) < 0.001
    ```

    ...and note that these tests are **EXPRESSIONS,** not statements -- do **NOT** end them with a semicolon!
- Be especially careful to include at least two tests/test expressions for every function, including at least one specific test/test expression for each "kind"/category of data, and (when there *are* boundaries) for

boundaries between data. You can lose credit for not doing so.

And, remember that tests should be:

- written as `bool` expressions within a non-`main` function's opening comment, after its purpose statement, **AND**
- written within parentheses `( )` within a `cout` in the testing `main` function.

- **STYLE REMINDER:** Indent your `return` statement by at least 3 spaces inside your function body!

  And curly braces go on their own line, lined up as shown in the posted class examples!

- Remember: in the CS50 IDE at https://cs50.dev:
  - each time you want to **compile** your program-in-progress:

    In a CS50 terminal, open to the folder CONTAINING `111hw8.cpp`, (**"Open in Integrated Terminal"**), type:

    `g++ 111hw8.cpp -o 111hw8`
  - when it has successfully compiled and created an executable program result, each time you want to run your program-in-progress:

    In that same CS50 terminal, open to the folder CONTAINING `111hw8.cpp`, type:

    `./111hw8`
  - when you are satisfied with the program's output for one of your functions, you can create an example output from running your program at that point to submit to Canvas by running :

    `./111hw8 > 111hw8-out.txt`

- Please let me know if you have any questions or concerns about the above requirements.

Problem 1 - 7 points

Problem 1 is correctly answering the "HW 8 - Problem 1 - Short-answer questions giving the types for C++ compound expressions - short-answer" on the course Canvas site.

Problem 2 - 8 points

Problem 2 is correctly answering the "HW 8 - Problem 2 - Short-answer questions on C++ compound expressions, arguments, and named constants" on the course Canvas site.

Problem 3 - 10 points

Problem 3 is correctly answering the "HW 8 - Problem 3 - Short-answer questions related to C++ and the design recipe" on the course Canvas site.

Problem 4 - 7 points

Problem 4 is correctly answering the 'HW 8 - Problem 4 - Short-answer questions on "translating" algebraic and boolean expressions into C++' on the course Canvas site.

Homework Program Setup for Problems 5 onward

- **Copy** the contents of the file **`111template.cpp`**, posted on the course Canvas site and on the public course web site, into a file named **`111hw8.cpp`** within the CS50 IDE (at https://cs50.dev/).

- See the comment that has `by:` and `last modified:` ?
  - START that comment with: `CS 111 - HW 8`
  - Then put your name after `by:` , and today's date after `last modified:` .
  - For example:

```
/*---
    CS 111 - HW 8
    by: Your Name
    last modified: 2024-10-28
---*/
```

Problem 5 - 14 points

In the "first `main.cpp` template" you pasted into your **`111hw8.cpp`**, find the comment:

```
/*--- PUT YOUR SIGNATURES, PURPOSES, TESTS, and FUNCTION DEFINITIONS HERE ---*/
```

AFTER this comment -- but **BEFORE** the function header for the function named `main` -- type a blank link, and then type the comment:

```
/*===
    Problem 5
===*/
```

Recall the function from Homework 2, Problem 6 that averages three measurements. Use the design recipe, in your file `111hw8.cpp`, to design a C++ version of this function named `avg_trio`. (`avg_trio` expects three measurements and returns the average of those three measurements.)

- Remember to include a **signature**, **purpose**, **function header**, **tests**, and then completed **function body** for `avg_trio`.
- Be sure to include your tests BOTH in a comment after your purpose statement, AND in `main`, as we have done in class.
- IF you would like, you can also include one or more `cout` statements that include JUST an example call of your function **after** these tests, so that you see the value those call(s) return.

Problem 6 - 14 points

After your function for Problem 5, type a blank link, and then type the comment:

```
/*===
    Problem 6
===*/
```

Recall the function from Homework 2, Problem 7 that returns the area of a triangle. Use the design recipe, in your file `111hw8.cpp`, to design a C++ version of this function named `tri_area`. (`tri_area` expects the height and width of a triangle, and returns the area of that triangle.)

- Remember to include a **signature**, **purpose**, **function header**, **tests**, and then completed **function body** for `tri_area`.
- Be sure to include your tests BOTH in a comment after your purpose statement, AND in `main`, as we have done in class.
- IF you would like, you can also include one or more `cout` statements that include JUST an example call of your function **after** these tests, so that you see the value those call(s) return.

Problem 7 - 14 points

After your function for Problem 6, type a blank link, and then type the comment:

```
/*===
    Problem 7
===*/
```

Recall the function from Homework 2, Problem 8 that asks a given person how they are doing. Use the design recipe to design a C++ version of this function named `ask_how_doing`. (`ask_how_doing` expects a person's name, and returns `How are you doing, ...that person's name... ?`)

(As in Homework 2, **optionally**, you may **change** the exact wording of the question, as long as it **includes** the person's name and **ends** with **at least one** question mark.)

- Remember to include a **signature**, **purpose**, **function header**, **tests**, and then completed **function body** for `ask_how_doing`.
- Be sure to include your tests BOTH in a comment after your purpose statement, AND in `main`, as we have done in class.
- IF you would like, you can also include one or more `cout` statements that include JUST an example call of your function **after** these tests, so that you see the value those call(s) return.

Problem 8 - 21 points

After your function for Problem 7, type a blank link, and then type the comment:

```
/*===
    Problem 8
===*/
```

Recall the function from Homework 2, Problem 10 that calculates the total cost for a coffee order. Use the design recipe to design a C++ version of this function named `order_total`. (`order_total` expects number of pounds of coffee in an order, and returns the total price for that order, including shipping.)

- Recall that this function uses named constants!
  - That is: The Tasty-Waking coffee roasters sells whole-bean coffee at $8.99 per pound plus the cost of shipping. Each order ships for $0.99 per pound plus $3.99 fixed cost for overhead.
  - Write C++ **named constant declaration statements** creating named constants:

    `COFFEE_PRICE_PER_LB`,

    `SHIP_PRICE_PER_LB`, and

    `SHIP_COST_FIXED`.
  - PLACE these declaration statements right *before* the function header for `order_total` (*after* the comment with the function's signature, purpose, and test `bool` expressions).
- Remember to include a **signature**, **purpose**, **function header**, **tests**, and then completed **function body** for `order_total`.
  - FOR FULL CREDIT, make sure that you USE the named constants `COFFEE_PRICE_PER_LB`, `SHIP_PRICE_PER_LB`, and `SHIP_COST_FIXED` appropriately within `order_total`'s function body.
- Be sure to include your tests BOTH in a comment after your purpose statement, AND in `main`, as we have

done in class.

- IF you would like, you can also include one or more `cout` statements that include JUST an example call of your function **after** these tests, so that you see the value those call(s) return.