(Print this page as a cover sheet for your printouts)

CSCE 121 HOMEWORK 3

Dr. Daugherity

Section: \_\_\_\_\_\_\_\_\_\_\_\_\_\_

Due: 11:59 P.M. Tuesday, September 26, 2017

"On my honor, as an Aggie, I have neither given nor received any unauthorized aid on any portion of the academic work included in this assignment."

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Typed or printed name of student Signature of student

NOTE: Please follow your lab instructor's directions for submitting your assignment through CSNET, which are posted at <https://piazza.com/class/j6nzibp5nh8pc?cid=43>. ONLY ASSIGNMENTS SUBMITTED TO CSNET WILL BE GRADED! Make a printout of each source file and staple it behind this cover sheet, unless your lab instructor directs otherwise. Sign it and give it to your TA in lab or put it in your TA's mailbox in the corner of the 3rd floor of HRBB, near room 312, by 5:00 P.M. Wednesday. IF YOU DO NOT TURN IN A SIGNED COVER SHEET YOUR WORK WILL NOT BE GRADED!

NOTE: Homework will be graded on build.tamu.edu using "g++ -std=c++17

-Wall -Wextra -pedantic -fsanitize=address,undefined" and should produce no

warnings or errors during compilation, linking, or execution. You are free to develop your programs on Visual C++ or any other platform, but it is your responsibility to make sure your programs also compile, link, and execute correctly on build.tamu.edu using the g++ command above.

NOTE: Each file submitted (h3p1, etc.--see below) must begin as follows:

//Your Name

//CSCE 121-xxx (fill in your section number)

//Due: September 26, 2017 (or whatever the due date is)

//h3p1\_012345678.cpp (or whatever this file name is, with your UIN)

NOTE: Every program from now on must include at least the exception-handling

framework in the last main() in section 5.6.3 of the Stroustrup book! If you add another catch clause, it must precede all more general clauses; for example, if you add **catch(Bad\_area)** it should come before the other two **catch** clauses.

The grade for this lab will be based on style (formatting, variable names, comments, etc.), syntax (no compilation or link errors), and correctness (passes all test cases). Follow the style guide at

<http://www.stroustrup.com/Programming/PPP-style.pdf>.

Your grade for this lab is:

Problem # 1 2 3 4

Style /2 /4 /4 /2

Syntax /3 /6 /6 /3

Correctness /5 /10 /10 /5

-------------------------------------------------------------------

Total /10 /20 /20 /10

Grand total \_\_\_\_\_/50

1. (10 points) One financial planner recommends dividing your income into the

following budget categories:

Give away 10% (e.g., to some cause you are committed to, or to help

others)

Save 10% (e.g., for unexpected expenses or planned large purchases)

Live on 80%

Write a program named h3p1 which repeatedly asks for an income amount in dollars and prints out the budget amounts in dollars and cents. Round off amounts to the nearest cent. A sample run should look like this:

Income in dollars? 1000

You should give away about $100.00, save about $100.00, and live on about $800.00.

Income in dollars? 2015.37

You should give away about $201.54, save about $201.54, and live on about $1612.30.

Income in dollars?^D

Hint: Look up **setprecision**, so your program doesn't print $201.537. Note: Don’t worry if the rounded-off amounts don’t add up to exactly the income amount; that’s why the output says “*about* $201.54,” etc.

2. (20 points) Write your own cube root function named **double my\_cbrt\_1(double n)** using the following Padé approximant:

≈

and then write a **main** which prints **n**, **cbrt(n)**, and **my\_cbrt\_1(n)** for **n** =

**π** times 10 to the **k**th power for **k** = -100, -10, -1, 0, 1, 10, and 100. Use this code:

**for(auto k : {-100, -10, -1, 0, 1, 10, 100}){**

**n = M\_PI \* pow(10.0, k);**

**//cout goes here**

**}**

Name your program h3p2. Note: The formula above is an approximation and so for some values of **n** it will not be very accurate!

Note: M\_PI is defined as **π** by including std\_lib\_facilities\_5.h.

3. (20 points) Modify problem 2 to also print the relative error as a per cent, by adding a column **relative\_error\_per\_cent = 100 \* ((my\_cbrt\_1(n) – cbrt(n)) / cbrt(n)**. Line up the columns using **setw()**, etc. Name your program h3p3.

OPTIONAL EXTRA CREDIT

4. (10 points) Modify problem 3 to be more accurate by defining a function

**double my\_cbrt\_2(double n)** using this pseudocode:

Set result to 1.

If n < 1.05 then divide result by 9/8 and multiply n by (9/8)3; repeat until

n ≥ 1.05.

If n > 1.55 then multiply result by 9/8 and divide n by (9/8)3; repeat until

n ≤ 1.55.

Return result \* my\_cbrt\_1(n) as the answer.

Now make further changes so that **my\_cbrt\_2(0.0)** returns 0 and **my\_cbrt\_2** is also correct for a negative argument. The main should print **n**, **cbrt(n)**, **my\_cbrt\_2(n)** and the relative error of **my\_cbrt\_2(n)** for the following values of **n**: -1e-100, -1e-10, -1, 0, 1, 1e10, and 1e100. It should be very accurate! Name your program h2p4.cpp.