COMP161 - Lab 4 & Homework 3

Spring 2014

Lab 4 and Homework 3 give you practice with writing and testing basic procedures.

Lab 4

For lab you'll complete the development of a single C++ procedure, and then complete the design phase for several more[1]. The problems you'll be solving come from the first edition of *How to Design Programs*.

[1] which will be completed as homework

Exercise 3.3.1. The United States uses the English system of (length) measurements. The rest of the world uses the metric system. So, people who travel abroad and companies that trade with foreign partners often need to convert English measurements to metric ones and vice versa.

Here is a table that shows the six major units of length measurements of the English system:

English		metric
1 inch	=	2.54 cm
1 foot	=	12 in.
1 yard	=	3 ft.
1 rod	=	5(1/2) yd.
1 furlong	=	40 rd.
1 mile	=	8 fl.

Develop the functions inches→cm, feet→inches, yards→feet, rods→yards, furlongs→rods, and miles→furlongs. Then develop the functions feet→cm, yards→cm, rods→inches, and miles→feet. Hint: Reuse functions as much as possible. Use variable definitions to specify constants.

First, let's work the whole design, implement, and test process for a single procedure.

1. (Design) Start the library header and declare and document an inches→cm procedure.
2. (Design) Start the library implementation file and stub the inches→cm procedure.
3. (Design) Compile the library implementation file to an object and fix any syntax errors and warnings[2].

[2] don't forget to use -Wall

4. (Design) Start the library tests file and write tests for the inches→cm procedure.
5. (Design) Compile the test file to an object and fix any syntax errors and warnings.
6. (Design) Link the tests with the library and run the tests[3].
7. (Implement) Now go back and complete the procedure definition.
8. (Test) When you think the procedure is done, recompile the library object, relink with the tests, and run the test. Debug your implementation[4] as needed.

[3] which should all fail

[4] or tests

Now for the remaining procedures, complete the design process only. Do not implement the procedure unless the procedure is fully documented and all tests are written. Write a new test case for every procedure; do not test multiple procedures within a single test case. If I see more than one complete procedure in your lab submission, then you'll lose points.[5]. I want documentation, stubs, and tests. That's it. Submit your source documents[6] as assignment *lab4* with *handin*. Do not submit objects, executables, temporary files created by emacs, or any other non-source file.

[5] I won't always be so strict, but I need to see you work the process at least once

[6] cpp and h files

Homework 4

Due by class time Wednesday 2/12. Submit as hwk4.

Complete the implementation and testing of all the remaining unit conversion procedures and then design and develop a procedure for the following problem[7]

[7] again, courtesy of *HtDP1e*

> **Exercise 4.4.3.** Some credit card companies pay back a small portion of the charges a customer makes over a year. One company returns
>
> .25% for the first $500 of charges,
>
> .50% for the next $1000 (that is, the portion between $500 and $1500),
>
> .75% for the next $1000 (that is, the portion between $1500 and $2500),
>
> and 1.0% for everything above $2500.
>
> Thus, a customer who charges $400 a year receives $1.00, which is 0.25 * 1/100 * 400, and one who charges $1,400 a year receives $5.75, which is 1.25 = 0.25 * 1/100 * 500 for the first $500 and 0.50 * 1/100 * 900 = 4.50 for the next $900.
>
> Determine by hand the pay-backs for a customer who charged $2000 and one who charged $2600.

Define the function pay-back, which consumes a charge amount and computes the corresponding pay-back amount.