CMSI 585

PROGRAMMING LANGUAGES (GRADUATE LEVEL)

Fall 2006

Assignment 1024

This assignment represents a number of control flow programming experiments and exercises on selected languages, as well as a few conceptual questions.

Not for Submission

1. Read the rest of Chapter 6 in Scott.
2. You will need Perl's test modules (Test::Simple, Test::Class, Test::More, and Test::Exception). If your Perl installation doesn't already have them, you can use the *cpan* utility to get them for you. The *gcd* and *roman* programs are examples of how to use these modules; of course you can always ask me, as well.
3. You will also need Andrew Wall's CUnit for this assignment; refer to the course Web site for the link. As with Perl, the *gcd* and *roman* programs are concrete examples of how to use CUnit, and you can ask me for help as needed.

For Submission

Commit the programming tasks under the specified directories with the given tag, and submit your answers to the assigned textbook exercises and questions on hardcopy:

1. Write a module + unit test suite that explores expression evaluation order in Perl and in C. The unit test format allows you to unambiguously express the expected results of your code in your individual installation. Here are the details — of course, implement these in the manner appropriate for each language:
   a. Define a module with two functions, *f()* and *g()*. These functions can do anything you wish; the criterion is that their use will expose the language's evaluation order. Specifically, the functions should behave in such a way that, if they are used in an expression, the final result of the expression differs depending on whether *f()* or *g()* gets evaluated first.
   b. Write a unit test that uses expressions involving *f()* and *g()* to determine the evaluation order policy in the programming language. You will probably want to first calculate an expression in a manner that enforces a particular evaluation order (e.g., sequential cumulative calculations), then compare that result to the "all-in-one" version of the expression.

   Commit the Perl version of your program under */homework/cmsi585/evalorder/perl*, and commit the C version under */homework/cmsi585/evalorder/c*. Tag the files with *hw-1024*.
2. Scott Exercise 6.1.
3. Scott Exercise 6.6.
4. Scott Exercise 6.11.
5. Is the expression evaluation order experiment applicable to Java? Why or why not?

Extra Credit

Perform the same expression evaluation order experiment in both ML and JavaScript. If you do perform the extra credit work, commit the code under */homework/cmsi585/evalorder/ml* and */homework/cmsi585/evalorder/javascript*, respectively. Also tag the files with *hw-1024*.

Extra Extra Credit

Perform this expression evaluation order experiment in Smalltalk. Instead of submitting code, submit a hardcopy description of what you did and describe the behavior that resulted.