CMSI 585

PROGRAMMING LANGUAGES (GRADUATE LEVEL)

Fall 2006

Assignment 0912

Assignment 0912 follows up on the files you created in Assignment 0905 — *now* it's time to submit the code that you wrote, and we will do this via CVS.

Not for Submission

1. Read Chapter 2 in Scott.
2. Acquaint yourself with CVS. There are numerous resources on the Web for learning it; a starter link is available at the course Web site.

For Submission

Submit the 6 programs and the LaTeX source files that you wrote in Assignment 0905 to me via CVS. Since I will access your work electronically, you must follow the instructions below *to the letter, down to the capitalization*.

1. Your Keck lab account already comes with two subdirectories, *projects* and *homework*.
2. Caskey has prepared a script for setting up your CVS directories so that I can read/write them — ask him how to get to it, and run it.
3. On any computer that you will use for your school work, check out *projects*.
4. Under your local copy of *projects*, create a *cmsi585* subdirectory.
5. Place the LaTeX source files that you created in Assignment 0905 in *projects/cmsi585*.
6. Add then commit the files to CVS. When in doubt, invoke a *cvs update* to verify file statuses.
7. Follow a similar process with the *homework* directory: check it out, and create a *cmsi585* subdirectory within the checked-out *homework* directory. Then, create one more subdirectory under *cmsi585*, called *sixlanguages*.
8. Place the programs that you wrote in Assignment 0905 in *homework/cmsi585/sixlanguages*, each in its own subdirectory, named *c/*, *cpp/*, *java/*, *js/*, *ml/*, and *perl/*, respectively.
9. Add then commit your code to CVS.
10. Tag the files (both the LaTeX files in *projects/cmsi585* and the source code in *homework/cmsi585/sixlanguages*) in CVS as *hw-0912*.

Extra Credit

Install unit test frameworks for each of the six languages in the assignment, and add unit tests to your programs. This means that your algorithm is factored out cleanly, and the unit test framework runs "on top of" your algorithm to express and verify test fixtures.

You may adopt the same unit test frameworks or techniques that are used in the sample code, find something else (making sure that I can install and run it too), or roll your own. The unit test frameworks used by the sample code, except for ML, are listed on the course Web site. The ML sample code embeds its own unit test functions; however, the overall approach remains the same.

To get the extra credit, simply make sure that the programs in *homework/cmsi585/sixlanguages* that you commit and tag as *hw-0912* are written in a test-driven manner.

Note that this task is extra credit at this point primarily due to its timing. You *will* be required to do this eventually, as part of a future, non-extra-credit assignment. The extra credit is for those of you who are able to implement unit tests by the due date of *this* assignment.