CS 2322 Laboratory Problem Set 6

Extra Credit Problem Set

These problems are to be done on an individual basis following the Trinity University Academic Integrity Policy or Trinity University Honor Code.

Academic Integrity and Honor Code

All students are covered by a policy that prohibits dishonesty in academic work. The Academic Integrity Policy (AIP) covers all students who entered Trinity before the Fall of 2004. The Academic Honor Code covers all those who entered the Fall of 2004 or later. The Integrity Policy and the Code share many features: each asserts that the academic community is based on honesty and trust; each contains the same violations; each provides for a procedure to determine if a violation has occurred and what the punishment will be; each provides for an appeal process. The main difference is that the faculty implements the AIP while the Honor Code is implemented by the Academic Honor Council. Under the Academic Integrity Policy, the faculty member determines whether a violation has occurred as well as the punishment for the violation (if any) within certain guidelines. Under the Honor Code, a faculty member will (or a student may) report an alleged violation to the Academic Honor Council. It is the task of the Council to investigate, adjudicate, and assign a punishment within certain guidelines if a violation has been verified. Students who are under the Honor Code are required to pledge all written work that is submitted for a grade: On my honor, I have neither given nor received any unauthorized assistance on this work and heir signature. The pledge may be abbreviated pledged with a signature.

Laboratory problems should be submitted electronically (e-mail to cs2322@cs.trinity.edu) on or before the due date and should contain a problem write-up, source code to any programs and data sets used in solving the problem. The submitted files should be ASCII text files having Unix end-of-line characters (please convert all Windows and Mac text files to Unix format–I have found that Emacs seems to do a reasonable job of such conversions). If several files need to be submitted, put them in a directory having name *your-last-name-problem-set-number* and create a tar archive of this file system and attach it to your e-mail problem submission.

Adding Records to a Database

In Laboratory Problem 4 we defined a simple flat-file database. Use this definition in this laboratory problem. The Class Files directory (`/users/jhowland/cs2322j`) contains a file `db.ijs` which you may use as a sample database for testing purposes. Create a monad, `add_record` which will add a record to a database.

You may want to maintain the database as a file rather than as a J object. To that end, there exist functions `read` and `write` which can be used to read and write text files. `read` is a monad whose argument is the name of the file to be read. For example:

```
   read '/etc/hosts'
127.0.0.1               localhost localhost.localdomain
131.194.131.13          Ariel.CS.Trinity.Edu Ariel
131.194.131.168         Sol.CS.Trinity.Edu Sol
```

on my workstation, *Ariel.CS.Trinity.Edu*, and:

```
   'this is some text to be written to a file called foo' write 'foo'
   system'cat foo'
this is some text to be written to a file called foo
```

Recall the existance of verbs:

```
   read_list
to_internal@read
   write_list
4 : '(to_char x.) write y.'
   to_internal
3!:2
   to_char
3!:1
```

which must be used when writing files which are not character lists.

Problem Set 6 Solution [HTML] [PS] [PDF]