CSE 341 Assignment 5

February 3, 2006

The programming portion (Part III) is due on Thursday, February 9th at 11pm. Use the online turnin form to turn in this part of the assignment. Written problems (Parts I and II) are due on Friday, February 10th, at the *beginning* of class. No late assignments will be accepted.

Most of the written questions can, of course, be solved by typing code into an ML interpreter and writing down the results. However, if you just do this, you won't learn what you need from this assignment and won't have these experiences to draw on later.

Part I: Records as ADTs

This portion of the assignment consists primarily of written problems. We have provided a file called *hw5_provided.sml* that contains the definition for an `Employee` datatype. This datatype makes use of a record to store useful information about an employee, including their pay rate, hours worked, and managerial status. It contains several functions that allow these pieces of information to be updated.

The definition of the datatype is followed by several lines of code that perform various operations using these sets. Between these lines of code are comments that indicate that you should fill out their correnponding sections on the worksheet by evaluating the statements provided on the worksheet. Fill out the worksheet with the results of evaluating those expressions (on paper, not in the ML interpreter).

Part II: Type Inferencing (and a bit of mystery solving)

More paper problems. Fill in the blanks in the worksheet. Code used in this part of the assignment is also provided in *hw5_provided.sml*

1. What are the types of the following functions? Briefly explain why and how you arrived at these answers. ("That's what the computer says" is not an adequate answer.)

```
fun mystery1 p u A B = (p (u A B))/(p B)
fun mystery2 (f:real→real) d i = f(d-i)
fun mystery3 (f:real→real, g) x = f(x)*g(x)
fun mystery4 f r j = def_integral (mystery3(f, (mystery2 r j))) ~1E2 1E2
```

   (the *def_integral* function is provided)

2. What are the mathematical functions (equations) that are computed by *mystery2* and *mystery3*? (Optional: write down the equation for *mystery4*, omitting the upper and lower bounds for the integral since they're just used for the numerical approximation.)

3. (Optional, but you can play with this even if you don't want to solve it). Take the *box* function provided and substitute it for f and r in *mystery*4. Evaluate the following expression using SML:

```
val g = mystery4 box box
```

Try testing it on different floating point values (ex: `g 0.0, g 0.001, g 1.0, g 2.1`). Can you tell (i) what the shape of the function looks like, or even (ii) what equation it computes? (If you can figure out the equation, then you probably already know the answer!)

You can also play with the additional functions provided, like *sawtooth*, *triangles*, and *sinc*. To import the *Grapher* package provided, type:

```
use "graph.sml"
```

To draw a function, type:

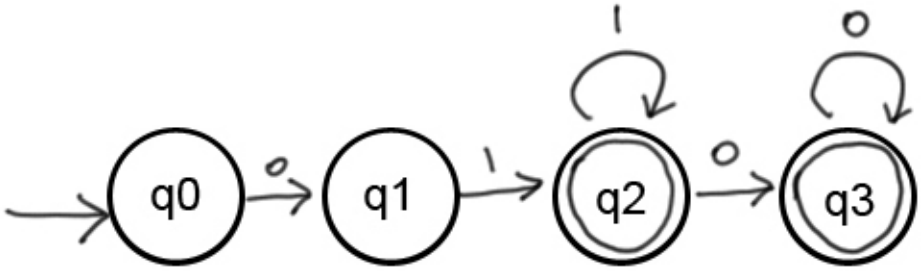
```
Grapher.graph(<function to draw>)
```

Try mixing different things with *mystery*4. For example, what would you predict the graph to look like if you graph:
`Grapher.graph(mystery4 triangles box)` ?
(*note*1: It may take some time for the graph to draw because of the numerical integration)

Part III: A little finite automata

And now, a little coding.

This section deals with regular expressions, languages, and finite automata. For this problem, implement a function *simpleFA* that simulates a finite automaton(FA) that recognizes the language described by the expression 01^+0^*. Some of the strings that belong to this language are: 01, 010, 011, 0111, 011110, ...
A finite automaton that recognizes this language is:



Your *simpleFA* function should take an input string that consists of a combination of 0s and 1s, and return *true* if the input belongs to the language 01^+0^* and $false$ if it does not. You should use mutually recursive functions to represent the states and transitions from one state to the other.

For more information on regular expressions, languages, and automata, look in any standard text like Sipser's *Introduction to the Theory of Computation* or look up these topics on Wikipedia.

Turnin Instructions

You should turn in two things for this assignment:

(1) The written worksheet. You can type your answers on the worksheet then print them out, or fill them in by hand on a printed copy.

(2) The *simpleFA* function in part III should be put in a sml file named *hw*5.*sml* and turned in electronically.

Type Summary

Here is the type summary for part III (the rest of the assignment asks you to compute the types of other functions, so including them here is not appropriate).

val simpleDA = fn : string → bool

Sample Log Execution

- simpleFA("0100")
val it = true : bool
- simpleFA("1000000")
val it = false : bool

Assessment

- There are two components to this assignment, a written one and a programming one. In order to get full credit, you must submit both parts. The written component of the assignment should be turned in in class on Friday, February 10th, while the programming component must be turned in via the online course web on Thursday, February 9th.
- Your solution should generate correct bindings and give correct results.
- You must use pattern matching. Do not use the functions `hd, tl, null, #1, #2`, etc. in your code.
- You are expected to use good style when coding. Name your functions as specified in the assignment, place helper functions inside let statements if you do not plan to use them in more than one function, name your variables well, and comment your code if you think that you've done something that isn't immediately obvious. There are other style guidelines that have been mentioned in class, and you will be expected to follow those as well.