Homework 6 — ML core language— due Wednesday 10 April

Total number of points available on this homework is 380. Full credit is equivalent to 100 points.

Reading assignment

Read Chapter 4 of *ML for the Working Programmer*.

6.1 Lists (30pts)

Do exercise 3.34 from *ML for the Working Programmer*.

6.2 Equality (30pts)

Do exercise 3.32 from *ML for the Working Programmer*.

6.3 Map (10pts)

If *xs* is a list, the evaluation of *map f* (*map g xs*) requires two list traversals. Simplify the expression so that only one traversal is needed.

6.4 Simple expression evaluator (20pts)

We can use the following data type declaration to introduce a language of simple arithmetic expressions:

```
datatype expr = Num of int
              | Add of expr * expr
              | Mul of expr * expr
```

Write a function `eval`, with type `expr -> int`, which returns the arithmetic value of an expression.

6.5 Simple parser (50pts)

Write a function `parse`, with type `string -> expr`, where `expr` is as in the preceding exercise. The function should return an expression corresponding to the text of the string, in accordance with the standard precedence of arithmetic. The input string may consist of digits, + and * signs, parentheses, and white space. For instance, `parse "5 + 3 * 4"` should evaluate to `Add (Num 5, Mul (Num 3, Num 4))`. In case the input string is not well-formed (for example, the string `"5 + 3 ) * 4"` is not), function `parse` should raise an exception `Failedbecause` (see p. 135 of *ML for the Working Programmer*).

6.6 Trees (30pts)

Do exercise 4.19 from *ML for the Working Programmer*.

6.7 Trees (30pts)

Do exercise 4.20 from *ML for the Working Programmer*.

6.8 Using lists for arithmetic (180pts)

This exercise is a continuation of exercises 5.9–5.14. Devise a suitable ML representation for floating-point numbers: a list may still contain the digits of the number (the mantissa), but an exponent and a sign bit will indicate that the number represented is really sign $\times$ mantissa $\times$ radix^{exponent}.

1. (10pts) Write a function for addition of numerals in any radix.
2. (30pts) Write a function for multiplication of numerals in any radix.
3. (50pts) Write a function for division of numerals in any radix.
4. (60pts) Write a function to compute the square root of a numeral in any radix.
5. (30pts) Write a function to convert numerals from any radix to any radix.

The functions may take an additional argument that gives the desired precision (number of digits in the mantissa) of the result.

How to turn in

Turn in your code by running

~dmykola/handin your-file

on a regular UNM CS machine.

You should use whatever filename is appropriate in place of your-file. You can put multiple files on the command line, or even directories. Directories will have their entire contents handed in, so please be sure to clean out any cruft.

Remember to submit extensive tests of your programs!

Homework must be accompanied by the following statement: *"I pledge my honor that in the preparation of this assignment I have complied with the University of New Mexico Board of Regents' Policy Manual, including Section 4.8, Academic Dishonesty."* The manual is available at `http://www.unm.edu/~brpm/index.html`.