CS471 Programming Language Structure I
Spring 2008

Homework 4

Goal

To understand the ideas behind higher-order functions, and use of functions as first-class values in Scheme and ML.

Problem Description

1. A finite sum of a series can be expressed as:

$$\sum_{n=a}^{b} f(n) = f(a) + \cdots + f(b)$$

where *f* is any function, and *a* and *b* are integers. Write a function in Scheme (call it sumSeries) that will return the value of the sum, given parameters *f*, *a* and *b*, and an increment value. Show how your function can calculate the sum of the first n integers, starting at 1, and also the sum of the squares of the first n even integers (e.g. 4 + 16 + 36, etc.)

Hints

- *a, b* and *f* will be parameters of the function, but you will also need a fourth parameter to hold the increment for *a*
- *a* will increase by the increment value for every recursive call until it is greater than *b*
- when the function reaches the base case of the recursion it will return 0

2. The definite integral of a function f between the limits a and b can be approximated by the

$$\int_a^b f = \left[f\left(a + \frac{dx}{2}\right) + f\left(a + dx + \frac{dx}{2}\right) + f\left(a + 2dx + \frac{dx}{2}\right) + \cdots \right] dx$$

formula:

Write a function in Scheme (call it integral) that will return the value of the integral, given parameters f, a, b. dx can be defined internally to the function. You must use your function sumSeries defined in part 1 to help you define this integral function. Show how the function works by having it evaluate:

$$\int_0^1 x^3 dx$$

and

$$\int_0^1 \left(2x^2 + 3x + 1\right) dx$$

Hints

- The function will have three parameters *a*, *b* and *f*; it will make an appropriate call to sumSeries
- The function needs to use real numbers – it is more accurate as *dx* tends towards 0, so make it very small – try 0.1, 0.01 etc. However, the smaller *dx* is, the slower it takes to converge to an answer.
- The value of the first integral is $1/4$, and of the second is $19/6$; your answers should be very close to these values
- If you want to use the let form for defining dx, use:

```
(let ((dx 0.01))
   …
 )
```

3. Write the functions in ML and try them out using the same set of calls.

Hints

- Each function definition starts with the keyword fun and ends with a semicolon
- Since ML is strongly typed, you will need two versions of sumSeries - one that handles integers, and one that handles reals – call the real version sumSeriesR. For the real version, instead of leaving the type inference system to find the type of the function, declare the type of each numeric parameter of sumSeries and integral, using the form (x:real)
- Since you are calculating a real result, the functions that you pass to integral (and thus to sumSeriesR) need also to be real functions. The function to cube a value will need a type declaration on the parameter; the other function to integrate over needs to use real constants, i.e. 2.0 instead of 2, 3.0 instead of 3 and 1.0 instead of 1
- If ML still complains that types don't match look carefully at the message and try to figure out what it is complaining about. Only when all the types match everywhere will ML be silent.
- If you want to use let to define a value for dx, use the form:

```
let val dx = 0.01 in
   …
end
```

  where the … is the expression to evaluate that contains dx in it. The keywords let, val, in and end are all necessary

Grading

This assignment is worth 50 points. Part 1 is worth 10 points, and part 2, 20 points, and part 3, 20 points.

Submission

This assignment must be submitted through the submission page. Submit two files, one for the Scheme functions and one for the ML functions.

Due Date

March 31st, before 5pm.

Notes on Running the Scheme and ML interpreters

The Scheme interpreter is available by typing:

%~rth/public/Scheme/scm/scm

You can put this as an alias in your .cshrc file:

alias scheme ~rth/public/Scheme/scm/scm

The complete definition of the standard version of the language is on the web site at:

http://www.cs.nmsu.edu/~rth/cs/cs471/r4rs.html

The original is at:

http://www.swiss.ai.mit.edu/~jaffer/r4rs_toc.html

A file of Scheme definitions may be loaded by typing (load "filename") at the prompt. Exit the interpreter by typing (quit) at the prompt.

The Standard ML interpreter may be executed, also on Solaris machines by first typing:

%source /local/config/cshrc.ml

and then

%sml

An introduction to ML is at:

http://www.cs.nmsu.edu/~rth/cs/cs471/sml.html

The original is at:

http://www.dcs.napier.ac.uk/course-notes/sml/manual.html

A file of ML definitions may be loaded by typing:

use "filename";

at the ML prompt. Exit the interpreter by typing control-D.