CMPT 340

Assignment 1 Due: Thursday, January 31, 2017, 11:59pm

Simple Haskell Programming and Lambda Calculus

Total: 100 Points

[Note: For each of the problems 2 through 5 (a) you must explicitly include the function type signature.]

Problem 1. (5 + 5 + 5 + 5 Points) What are the types of the following functions?

a) swap (x, y) = (y, x)

b) pair x y = (x, y)

c) double x = x*2

d) twice f x = f (f (x))

Problem 2. (5 + 5 + 5 Points) Computing the exponent of a number (i.e., n^k) can take O(n) time if done by simply multiplying by n, k number of times. A logarithmic way for computing exponents is by using the idea of successive squaring. For instance, rather than computing n^8 as: n * n * n * n * n * n * n * n, we can compute it by repeatedly squaring, beginning with n, computing n^2, n^4 and finally n^8. In general, the algorithm would do the following:

n^k = (n^(k/2))^2 if k is even
n^k = n * n^(k-1) if k is odd

Implement functions in Haskell to do this in each of the following ways:

a) fastExp1: a conditional expression (i.e., using if-then-else)

b) fastExp2: guarded equations (i.e., using | to separate cases)

c) fastExp3: pattern matching

Problem 3. (5 + 10 Points) The Luhn algorithm is used to check bank card numbers for simple errors such as mistyping a digit, and proceeds as follows:

- Consider each digit as a separate number

- Moving right-to-left and beginning at the second last digit, double every other number

- Subtract 9 from each number that is now greater than 9

- Add all the resulting numbers together

- If the total is divisible by 10, the card number is valid

a) Define a function luhnDouble that doubles a digit and subtracts 9 if the result is greater than 9. For example:

> luhnDouble 3

6

> luhnDouble 6

3

b) Using luhnDouble, define a function luhn which decides if a four-digit bank card number is valid. For example:

> luhn 1 7 8 4

True

> luhn 4 7 8 3

False

Problem 4. (8 + 2 + 10 Points)

a) Implement a function averageThree to return the average of three integers.

Next, use averageThree to define another function howManyAboveAverage which returns how many of its three inputs are larger than their average value.

b) Use your averageThree function from above to write another function averageThreeInOne which receives the three input values as a triple rather than separately.

c) Implement a function orderTriple which accept a triple of integers as input, and evaluates to a triple with the three integers appearing in ascending order. Use functions written for (a) and/or (b) as helper functions scoped inside the main function to do this.

Problem 5. (10 + 10 Points)

a) Implement a function compose3, which takes three functions f(x), g(x) and h(x) (each of type Double -> Double) and evaluates to a function (also of type Double -> Double) to compute f(g(h(x))). Do NOT use the built-in composition function for this.

b) Show how the meaning of compose3 can be formalized in terms of a lambda expression. Use the backslash character (i.e., '\') to represent lambda.

Submission:

Create a directory with your nsid as its name. Inside this directory:

- For a problem not involving programming, include a file named like problem1.txt, under the top-level folder.

- For each problem with a programming component, create a separate sub-directory with names like problem2, problem3, etc. Under each of these folders, for each programming problem, include a separate file with your program, as well as a text file showing a transcript of your testing of the program. For non-programming components, include text files named after the problem part (such as problem5b.txt).

Once you have everything in your directory, create a zip file for the entire directory. If your nsid is <your_nsid> name the zip file <your_nsid>.zip. When opened, it should create a directory called <your_nsid>.

You may submit multiple times before the deadline, so you are advised not to wait till the last minute to submit your assignment to avoid system slowdown. You are encouraged to submit completed parts of the assignment early. Late submissions will not be accepted.