HW 4 **Prof. Caldwell**
Due: 9 September 2008 **COSC 3015**

1

In class we discussed the following code.

plus (x,y) = x + y
plusc x y = x + y
curry f x y = f(x,y)
uncurry f (x,y) = f x y
(f . g) x = f (g x)
id x = x

Recall the extensionality rule for proving functions $f, g : A \to B$ are equal.

$$f = g \stackrel{\text{def}}{=} \forall x : A.\ fx = gx$$

We gave a proof in class of the following identity.

Lemma 1.1. For every function f where $f :: A \to B$, the following equality holds

$$f.\ id = f$$

Proof: To show these functions are equal, by extensionality, we must show that

$$\forall x : A.(f\ .\ id)x = f\ x$$

Choose an arbitrary $x \in A$ and show $(f\ .\ id)x = f\ x$. Starting on the left side:

$$(f.\ id)\ x \stackrel{\langle\langle \text{def. of } (.)\rangle\rangle}{=} f(id\ x) \stackrel{\langle\langle \text{def. of } id\rangle\rangle}{=} f\ x$$

So the identity holds.
□
Following the proofs given in class, prove the following equalities using extensionality.

Problem 1.1.
i.) *curry plus = plusc*
ii.) *curry (uncurry plusc) = plusc*
iii.) *uncurry (curry plus) = plus*

Problem 1.2. Haskell has a built-in function called `flip` having the following type:

```
(a -> b -> c) -> b -> a -> c
```

Write a bit of Haskell code that has this type.