CSE 341 — Miranda Discussion Questions

These are questions for discussion in class. (You don't need to hand in anything.) The solutions are on the class web page.

1. Write a Miranda function to find the cube of a number. What is the type of this function?

2. Write a Miranda function to find the sum of three numbers. What is the type of this function?

3. Write a Miranda function to find the sum of a list of numbers. What is the type of this function?

4. Write a Miranda function to find the maximum of two numbers. What is the type of this function?

5. Write a Miranda function to find the value of the quadratic expression $ax^2 + bx + c$ for parameters a, b, c, and x. What is the type of this function?

6. Write a Miranda function to find the two roots of the quadratic equation $ax^2 + bx + c = 0$ for parameters a, b, and c. What is the type of this function?

7. Write a Miranda function to reverse a list. What is the type of this function?

8. Write a function `my_map2` that is analogous to `map` but works for functions of two arguments rather than one. What is its type? For example,

   ```
   map2 (+) [1,2,3] [10,11,12]
   ```

   should evaluate to [11,13,15]

9. Tacky true/false questions!

   (a) In Miranda, programs would give the same answers if we replaced lazy evaluation with call-by-name.
   (b) In Miranda, programs would give the same answers if we replaced lazy evaluation with call-by-value.

10. Write a Miranda function to return the infinite list of amounts of money you have every year, assuming you start with $1 and get paid 5% interest, compounded yearly. (Ignore inflation, deflation, taxes, the possibility of collapsing currencies, and other such details.)

11. Suppose that the following Miranda script has been filed in.

```
plus x y = x+y

append [] ys = ys
append (x:xs) ys = x : append xs ys

my_map2 f [] [] = []
my_map2 f (x:xs) (y:ys) = f x y : my_map2 f xs ys
```

What is the result of evaluating the following Miranda expressions? If there is a compile-time type error, or a run-time error, or a non-terminating computation, say so. If the result is infinite, give the first several values. If the expression is followed by `::`, then give the type, instead of the value.

(a) `plus ::`

(b) `plus 5 ::`

(c) `append ::`

(d) `append [] ::`

(e) `append [3,4] ::`

(f) `append [] [3,4] ::`

(g) `my_map2 plus ::`

(h) `my_map2 append ::`

(i) `my_map2 plus [1..] [1..]`