CMPT 898 **HOME EXERCISE 5**

LIFTING TO MONADS

We often want to apply binary operators (e.g., +, -, *, /, concat, etc.) to Option values.

1. What are some problems or inconveniences with defining these operators directly for the Option class (e.g., having a "plus" operator on Option that takes another option value, and returns an Option, and performs an addition operation. Similarly with "minus", "concat", etc.)?
2. Given some of the considerations above, what would be a disadvantage of separately defining e.g., "OptionalDouble" rather than using Option[Double], and defining such operators ("plus", "minus", etc.) on OptionalDouble, and similarly for Int, String, etc.?

In functional programming, we often avoid the need to define such methods directly by instead "lifting" functions that normally operate on values so that they operate instead on instances of the monad. For example, given Option[Double] values o1 and o2, rather than defining a "plus" operator explicitly in Option, we want to be able to say something like

```
map2ForOption(o1, o2, (x:Double,y:Double) => x+y)
```

or

```
map2ForOption(o1, o2, (x:Double,y:Double) => x*y)
```

and have an appropriate "Option[Double]" value returned. (As discussed in class, the resulting option will be "None" if either of o1 or o2 are None, and otherwise be Some(z) where z is the sum of the values associated with o1 and o2) More generally, we wish map2ForOption[T] to be of signature

```
map2ForOption[T,U](o1:Option[T], o2:Option[T], fn: (T,T) => U): Option[U]
```

3. Please define map2ForOption to "lift" binary function to operate on Option values. Hint: use flatMap and map, similar to how we used them in class when operating with Option values.

4. Now define map2ForTry to similarly "lift" binary function to operate on Try values. The signature of this will be
   ```
   map2ForTry[T,U](o1:Try[T], o2:Try[T], fn: (T,T) => U): Try[U]
   ```
   How does this differ when compared to the case with Option? What does this suggest about what we need to perform such a "lifting" with a map2 operator?

5. Please comment on the tradeoffs of “lifting” a function (using an operator0 such as map2ForOption when compared with hard-coding different binary operators as part of the Option type or its variants.

Due in class October 19, 2016