CSE 341 — Haskell Mini-Exercises # 2

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

1. Suppose that we have the following definition of the `member` function in Haskell:

```haskell
member x [] = False
member x (y:ys) | x==y = True
| otherwise = member x ys
```

Circle each type declaration that is a correct type for `member`. (Not necessarily the most general type, just a correct one.)

- `member :: Integer -> Integer -> Bool`
- `member :: (Ord a) => a -> [a] -> Bool`
- `member :: (Integer -> Integer) -> [Integer -> Integer] -> Bool`
- `member :: (Eq a) => a -> [a] -> Bool`
- `member :: a -> [a] -> Bool`
- `member :: (Eq a) => [a] -> [[a]] -> Bool`
- `member :: Bool -> [Bool] -> Bool`

Which of the above types, if any, is the most general type for `member`?

2. What are the first ten elements in the following list?

```haskell
mystery = 1 : map (*2) mystery
```

3. Define a variable `ints` whose value is the infinite list of all integers. It should be ordered in such a way that you can find any given integer after searching a finite number of elements in `ints`. In other words, this isn’t going to work:

```haskell
ints = [1 ..] ++ [-1, -2 ..]
```

4. Write a Haskell action `capitalize` that reads in a line of text and prints it out in all capitals. (Hint: use the function `Data.Char.toUpper`.)

```haskell
capitalize = do
  putStrLn "calculating the square root of x"
  putStrLn (show (sqrt x))
```

5. Convert the following functions into equivalent ones that don’t use `do`:

```haskell
printsqrt2 = do
  putStrLn "the square root of 2 is "
  putStrLn (show (sqrt 2))
```

```haskell
calcsqrt = do
  x <- readLn
  putStrLn "calculating the square root of x"
  putStrLn (show (sqrt x))
```