CSE 341 – Haskell Mini-Exercises

1. Consider the following `append` function:

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

   Which of the following would be legal type declarations?

   - `[a] -> [a] -> [a]`
   - `[a] -> [b] -> [c]`
   - `[Integer] -> [Integer] -> [Integer]`
   - `Num a => [a] -> [a] -> [a]`

2. Write a Haskell function `rev` that reverses a list. What is its type?

3. Suppose that the following Haskell code has been loaded.

```
my_const k x = k

reverse_args f x y = f y x

my_map f [] = []
my_map f (x:xs) = f x : my_map f xs

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

double x = 2*x

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

mystery = 2 : my_map double mystery
```

What is the result of evaluting the following Haskell expressions? Remember that `:type x` asks Haskell to print the type of `x`, so in that case give just the type. Otherwise just give the value. If there is a compile-time type error, or a non-terminating computation, say so. If the result is an infinite data structure, give some initial parts of it. If Haskell would give an error of some sort rather than producing output, say so.

(a) `:type my_const`

(b) `:type my_map (my_const "ho")`

(c) `my_const "ho" (1/0)`

(d) `mystery`

(e) `:type my_map2`

(f) `:type reverse_args`

(g) `:type member`

(h) `:type reverse_args member`