CSC 301 Assignment 7(Ch 9)

Baheem Ferrell

For each exercise, submit your code as well as evidence that it works (screen capture).

Exercise 9.6

Write the function sqsum of type int list -> int that takes a list of integers and returns the sum of the squares of those integers. For example, if you evaluate sqsum [1,2,3,4] you should get 1^2 + 2^2 + 3^2 + 4^2 = 30.

Do not use explicit recursion but use one of the fold functions in order to get full credit. Do not define any additional functions (not even in a let-expression); sqsum should be the only named function.

```
1  fun sqsum x = foldr (op +) 0 (map (fn x => x * x) x);
```

```
- val sqsum = fn : int list -> int
```

Exercise 9.26

Define a curried function mymap of type (‘a -> ‘b) -> ‘a list -> ‘b list that works just like the builtin function map. You are not allowed to use map to implement this function! You are allowed to define additional functions in order to implement this function (similar to the mergesort shown a few chapters ago). Use a let-expression to keep any additional functions private to mymap.

```
1  fun Mymap f[] = []
2  | Mymap f (num::numList) = f num :: Mymap f numList
3
4
5  fun squarelist intValues =
6  Mymap (fn num => num * num) intValues;
7
8
9  squarelist [1, 2, 3, 4];
10
11
12 val a = [1, 2, 3, 4];
13 val mylist = Mymap (fn (b) => b * b * b) a;
14
15 val boolValues = [true, false, false, true]
16
17
18 val boolComplimentList = Mymap (fn b => not b) boolValues;
```

```
- val Mymap = fn : ('a -> 'b) -> 'a list -> 'b list
val squarelist = fn : int list -> int list
val it = [1,4,9,16] : int list
val a = [1,2,3,4] : int list
val mylist = [1,8,27,64] : int list
val boolValues = [true,false,false,true] : bool list
val boolComplimentList = [false,true,true,false] : bool list
-
```