CS 457/557 Homework 5 – due 2pm, Tuesday, November 8, 2005

Hand in all your solutions on paper *only* (there is no need to email your proofs).

1. Given the usual definitions

```
map f [] = []                          (map1)
map f (x:xs) = f x : map f xs          (map2)
(f . g) x = f (g x)                    (comp)
```

prove that for any finite list `xs` and appropriately typed functions `f` and `g`,

```
map (f.g) xs = (map f . map g) xs
```

2. Given the definitions

```
reverse [] = []                        (rev1)
reverse (x:xs) = reverse xs ++ [x]     (rev2)
[] ++ ys = ys                          (++1)
(x:xs) ++ ys = x:(xs ++ ys)            (++2)
```

prove that for any finite list `xs`,

```
reverse (reverse xs) = xs
```

Hint: You'll need to prove (by a separate induction) an auxiliary lemma relating `reverse` and `++`. You may take as given (without the need for further proof), the two properties of `(++)` listed at the top of Hudak Table 11.2 (call them `(++assoc)` and `(++nil)` respectively).