CS 457/557 Homework 1 – due 2pm, Tuesday, October 4, 2005

Hand in your solutions on paper *and* email them to `cs457acc@cs.pdx.edu`. All the programs should be placed in a single file `hw1.hs`, which should be sent as an *attachment* to your email message. It is *not* necessary to show evidence that you have loaded and tested your programs, but this is of course the only sensible way to make sure that you have found correct answers!

(A number of these exercises are taken from Simon Thompson, *Haskell: The Craft of Functional Programming*, Addison-Wesley, 1996.)

0. If you haven't already done so, Learn how to run the Hugs interpreter on the machine(s) of your choice.

1. (Thompson) Give a definition of the function

```
power :: Int -> Int -> Int
```

such that `(power` k n`)` is k^n. (What happens if n is negative?) Your function should run in at worst $O(n)$ time; for extra credit, design it to run in $O(\log n)$ time.

2. (Thompson) Give a function

```
duplicate :: String -> Int -> String
```

which takes a string s and an integer n. The result is n copies of s joined together. If $n \leq 0$, the result should be the empty string, `""`, and if $n = 1$, the result will be s. (Hint: A `String` is just a list of characters; see Hudak, p. 39)

3. (Thompson)

a. Define a function

```
memberNum :: [Int] -> Int -> Int
```

such that `memberNum` l s returns the number of times the item s appears in the list l.

b. Use `memberNum` to define a function

```
member :: [Int] -> Int -> Bool
```

such that `member` l s returns `True` iff s is in l.

c. Use `memberNum` to define a function

```
unique :: [Int] -> [Int]
```

that returns a list of the numbers that occur exactly once in the argument list. For instance,

```
unique [2,4,2,1,4] = [1]
```

4. Do Hudak Exercise 2.1 (p. 25). (Note: The necessary `data` declarations can be imported (or cut-and-pasted) from file `Shape.lhs`, available on the course web page.)

5. Do Hudak Exercise 2.5 (p. 33).