

## CS 135 Homework for 25 Oct 2009

Submit one document with all answers. Submit a second file that contains the Scheme definitions (and any tests you want to include).

1. Define a Scheme function `expo` so that `(expo n i)` is  $n^i$  for natural numbers  $n$  and  $i$ . Use the standard definition

$$n^0 = 1 \quad \text{and} \quad n^i = n * n^{i-1} \quad \text{for } i > 0$$

2. Using the standard definition, prove that

$$n^i * n^j = n^{i+j}$$

for all natural numbers  $n, i, j$ . Use induction.

3. As a consequence of what you proved, we know that  $n^{2*i} = n^i * n^i$ . In other words, if  $j$  is even then  $n^j = n^{j/2} * n^{j/2}$ . Use this observation to add an additional case in your definition of `expo`. Call the new Scheme function `expo-fast`.

Justify the name by giving some test data and/or an explanation why it's faster.

4. Define another version, `expo-trec`, that works by calling a tail-recursive helper function.
5. Prove that `(expo-trec n i) = (expo n i)` for any natural  $n$  and  $i$ , by induction on  $i$ .
6. Recall this function from lecture 12:

```
(define powsum
  ; Takes a list, lon, of natural numbers. Returns the sum of
  ; the powers 2^n where n ranges over elements of lon.
  (lambda (lon)
    (cond [(null? lon) 0]
          [else (+ (expt 2 (car lon)) (powsum (cdr lon)))])))
```

Write another version, `powsum-trec`, that satisfies the same specification but works by calling a tail recursive helper. (It can use the built-in function `expt` or one of your `expo` functions, whatever you like.)

7. Write some tests for `powsum-trec`, using `powsum` to check the results.