

COSC 382 Organization of Programming Languages

Scheme Assignment 2, Fall 2009

credits: Neal Wagner

1. Write a recursive function `maxvec` that finds the maximum of a simple list of numbers.

[Hint: if the `cdr` is `nil`, just return the `car`, and otherwise return the maximum of the `car` and `(maxvec cdr)`.] For example, `(maxvec '(2 5 6 3))` returns 6.]

2. Try out the function `reverse` in Scheme:

```
(reverse '(a b c))
```

```
(reverse '((a b c)))
```

```
(reverse '(a (b c d) e)).
```

Define your own recursive function `myreverse` which will behave the same way as `reverse`, reversing the order of elements of a list at the top level only.

[Hint: Invoke `myreverse` recursively on the `cdr` and tack on the `car` at the end.]

3. Define a function `count-atoms` that will count the number of atoms in a list, at all levels. For example `(count-atoms '(a (b c) (d (e) f)))` should return 6.
4. Write a function `addvec` that takes a single list of numbers as input argument and returns the sum of the numbers. [Assume the input list has only one level. Return 0 for a `nil` list.]
5. Write a function `vecmul` that will take as inputs two simple lists of numbers. `vecmul` should multiply these lists coordinate-wise as one would multiply vectors. If one list is longer than the other, the result should be as if the shorter were padded with ones. [For example, `(vecmul '(2 3 4 5) '(1 4 5 2 14))` returns `(2*1 3*4 4*5 5*2 1*14)` or `(2 12 20 10 14)`.]
6. Use `addvec` and `vecmul` to define an inner product function `innprod`. [For example, `(innprod '(1 2 3) '(7 8 9))` returns `1*7 + 2*8 + 3*9` or 50.]
7. Define a function `tailtip` which returns the last element of a list (at the top level). Do not use the `reverse` already built in to Scheme.