Eric Wan – ezw23@drexel.edu – Lab 9

Question 1:
(list? L) -> (list? (reverse L))

1. Prove k=0, 1 case is true
   L is a null list:
   (list? null)
   = #t
   (list? (reverse null))
   = (list? (null? null))
   = (list? null)
   = #t
   L is a single value list:
   (list? ‘(x))
   = #t
   (list? (reverse ‘(x))
   = (list? (null? ‘(x))
   = (list? (append (reverse (rest ‘(x))) (cons (first ‘(x)) null))))
   = (list? (append (reverse null) (cons ‘(x) null)))
   = (list? (append null ‘(x))
   = (list? ‘(x))
   = #t
2. Assume k=n case is true
   L is some arbitrary list:
   (list? L) is true
   (reverse L) will output L’, and (list? L’) is assumed true
3. Prove k=n+1 case is true
   L is assumed to be ‘(x L):
   (list? ‘(x L))
   = #t
   (list? (reverse ‘(x L)))
   = (list? (null? ‘(x L)))
   = (list? (append (reverse (rest ‘(x L))) (cons (first ‘(x L)) null)))))
   = (list? (append (reverse L) (cons ‘(x) null)))
   = (list? (append (reverse B) ‘(x)))
   (list? (reverse B)) = #t by Inductive hypothesis
   (list? ‘(x)) = #t by base case/def of list?
   By property 1 of append:
   (and (list? (reverse B)) (list? ‘(x))) = #t
   (list? (append (reverse B) ‘(x))) = #t

Question 2:
(length (reverse x)) = (length x)

1. Prove k=0, 1 case is true
   (length null) = 0

(reverse null) outputs null, and (length null) is still 0
(length ‘(x)) = 1
(reverse ‘(x)) outputs ‘(x), and (length ‘(x)) is still 1

2. Assume k=n case is true
   (list? L) = #t
   (length L) = n is true
3. Prove k=n+1 case is true
   Assuming new list is ‘(x L)
   (length ‘(x L)) = L + 1
   (length (reverse ‘(x L))
   (length (append (reverse L) ‘(x)))
   By property 5 of append
   (+ (length (reverse L) (length ‘(x)))
   By Inductive hypothesis
   (+ n (length ‘(x)))
   (+ n 1) = n+ 1

Question 3:
(reverse (append x y)) = (append (reverse y) (reverse x))

1. Prove k=0, 1 case is true
   Both cases are null:
   (reverse (append null null)) = (reverse null) = null
   (append (reverse null) (reverse null)) = (append null null)) = null
   x is null, y is ‘(y):
   (reverse (append null ‘(y))) = (reverse ‘(y)) = ‘(y)
   (append (reverse ‘(y)) (reverse null)) = (append ‘(y) null)) = ‘(y)
   x is ‘(x), y is null:
   (reverse (append ‘(x) null)) = (reverse ‘(x)) = ‘(x)
   (append (reverse null) (reverse ‘(x))) = (append null ‘(x))) = ‘(x)
   x is ‘(x), y is ‘(y):
   (reverse (append ‘(x) ‘(y))) = (reverse ‘(y x)) = ‘(x y)
   (append (reverse ‘(y)) (reverse ‘(x))) = (append ‘(y) ‘(x))) = ‘(x y)
2. Assume k=n case is true
   (reverse (append X Y)) = (reverse ‘(Y X)) = ‘(X’ Y’)
   (append (reverse Y) (reverse X)) = (append (Y’ X’)) = ‘(X’ Y’)
3. Prove k=n+1 case is true
   (reverse (append ‘(x X) ‘(y Y)) = (reverse ‘(y Y x X)) = ‘(X’ x Y’ y)
   (append (reverse ‘(y Y) (reverse ‘(x X)) = (append ‘(Y’ y) ‘(X’ x)) = ‘(X’ x Y’ y)

Question 4:
(reverse (reverse x)) = x

1. Prove k=0,1 case is true
   (reverse (reverse null)) = (reverse null) = null

(reverse (reverse x)) = (reverse x) = x

2. Assume k=n case is true
   (list? L) = #t
   (reverse (reverse L)) = (reverse L’) = L
3. Prove k=n+1 case is true
   (reverse (reverse ‘(x L)) = (reverse ‘(L’ x)) = ‘(x L)
   =(reverse (append (reverse L) ‘(x)))
   by property 3 of reverse
   = (append (reverse (reverse L) (reverse ‘(x)))
   = (append L (reverse ‘(x)))
   = (append L ‘(x))
   = ‘(x L)

Question 5:
Equation to prove:
(nth x L) = (nth (- (length L) (- x 1)) (reverse L))
Assuming (reverse L) works and outputs L’

Prove k=1 is true:
(nth x L)
= (nth 1 L)
= L(1)
= first element of L
(nth (- (length L) (- x 1)) (reverse L))
= (nth (- (length L) (- 1 1)) (reverse L))
= (nth (- (length L) 0) (reverse L))
= (nth (length L) (reverse L))
= (nth (length L) L’)
= L(1)
= first element of L

Assume k=n is true:
(nth x L)
= (nth n L)
= L(n)
= nth element of L
(nth (- (length L) (- x 1)) (reverse L))
= (nth (- (length L) (- n 1)) (reverse L))
= (nth (- (length L) (-n 1)) L’)
= L(n)
= nth element of Ln

Prove k=n+1 is true:
(nth x L)

= (nth (+ n 1) L)
= L(n+1)
= n+1 element of L
(nth (- (length L) (- (+ n 1) 1)) (reverse L))
= (nth (- (length L) n) (reverse L))
= (nth (- (length L) n) (reverse L))
= (nth (- (length L) n) L’)
= (nth (- (+ 1 n) n) L’)
= (nth 1 L’)
= L(n+1)
= n+1 element of L