```
(define (reverse l)
     (if (null? l)
          null
          (append (reverse (rest l)) (cons (first l) null))))
```

# 1. (list? L) → (list? (reverse L))

a. Assume (list? L) = #t
b. Base Case: L = '()
   i. (reverse '()) = (append (reverse (rest '())) (cons (first '()) null)) = (append (reverse '()) (cons '() null)) = (append '() '()) = '()
   ii. (list? '()) = #t
c. Inductive Hypothesis: Assume (list? B) = #t, (list? (reverse B)) = #t
d. Inductive Proof: Prove that (list? (reverse (cons a B))) = #t
   i. (reverse (cons a B)) = (append (reverse (rest (cons a B))) (cons (first (cons a B)) null)) = (append (reverse B) (cons a null) = (append (reverse B) '(a))
   ii. (list? (reverse B)) = #t by Inductive Hypothesis
   iii. (list? '(a)) = #t
   iv. (and (list? (reverse B)) (list? '(a))) → (list? (append (reverse B) '(a)) by Property 1 of append
   v. (list? (reverse (cons a B))) = #t
e. Proof complete by induction

# 2. (length (reverse x)) = (length x)

a. Base Case: x = '()
   i. (length '()) = 0
   ii. (reverse '()) = '()
   iii. (length (reverse '())) = (length '()) = 0
b. Inductive Hypothesis: assume (list? B) = #t, (length B) = (length (reverse B)) = n
c. Inductive Proof: Prove that (length (reverse (cons a B))) = (length (cons a B))
   i. (length (cons a B)) = (+ 1 (length (rest (cons a B)))) = (+ 1 (length B)) = (+ 1 n)
   ii. (length (append (reverse B) '(a))) = (+ (+ n (+ 1 0))) = (+ n 1) because:
      1. (reverse (cons a B)) = (append (reverse (rest (cons a B))) (cons (first (cons a B)) null)) = (append (reverse B) (cons a null) = (append (reverse B) '(a))
      2. (length (append (reverse B) '(a))) = (+ (length (reverse B)) (length '(a))) by Property 5 of append
      3. (+ (length (reverse B)) (length '(a))) = (+ n (length '(a))) by Inductive Hypothesis
      4. (length '(a)) = (+ 1 (length (rest '(a)))) = (+ 1 0) = 1
   iii. (length (reverse (cons a B))) = (length (cons a B)) = (+ n 1)

```
(define (append x y)
     (if (null? x)
     y
   (cons (first x) (append (rest x) y))))
```

# 3. (reverse (append x y)) = (append (reverse y) (reverse x))

a. Base Case: x = '()
   i. (reverse (append '() y)) = y
b. Inductive Hypothesis: (reverse (append B y)) = (append (reverse y) (reverse B))
c. Inductive Proof: (reverse (append (cons a B) y)) = (append (reverse y) (reverse (cons a B)))
   i. (append (cons a B) y) = (append (append '(a) B) y) = (append '(a) (append B y))
      1. (append '(a) B) = (cons a B)
   ii. (reverse (append '(a) (append B y))) = (append (reverse (append B y)) '(a)) because:
      1. (rest (append '(a) (append B y))) = (append B y)
      2. (first (append '(a) (append B y))) = '(a)
   iii. (append (reverse (append B y)) '(a)) = (append (append (reverse y) (reverse B)) '(a)) because:
      1. (reverse (append B y)) = (append (reverse y) (reverse B)) by Inductive Hypothesis
   iv. (append (append (reverse y) (reverse B)) '(a)) = (append (reverse y) (append (reverse B) '(a))) by Property 5 of append
   v. (append (reverse y) (append (reverse B) '(a))) = (append (reverse y) (reverse (cons a B))) because:
      1. (reverse (cons a B)) = (append (reverse (rest (cons a B))) (cons (first (cons a B)) null)) = (append (reverse B) (cons a null) = (append (reverse B) '(a))
   vi. (append (cons a B) y) = (append (reverse y) (reverse (cons a B)))

# 4. (reverse (reverse x)) = x

a. Base Case: x  = '()
   i. (reverse '()) = '()
   ii. (reverse (reverse '())) = '()
b. Inductive Hypothesis: (reverse (reverse B)) = B

c. Inductive Proof: Prove that (reverse (reverse (cons a B))) = (cons a B)

i. (reverse (reverse (cons a B))) = (reverse (append (reverse B) '(a))) because:

1. (reverse (cons a B)) = (append (reverse (rest (cons a B))) (cons (first (cons a B)) null)) = (append (reverse B) (cons a null) = (append (reverse B) '(a))

ii. (reverse (append (reverse B) '(a))) = (append (reverse '(a)) (reverse (reverse B))) by Property 3 of append

iii. (append (reverse '(a)) (reverse (reverse B))) = (append '(a) B) = (cons a B)

1. (reverse (reverse B)) = B by Inductive Hypothesis
2. (append '(a) B) = (cons a B)

# 5. (list? L) → ( and (length (reverse L)) == (length L)) ((nth n (reverse L)) == (nth (- (+ (length L) 1) n) L)))

a. (nth n (reverse L)) == (nth (- (+ (length L) 1) n) L)) means that $L'_n$ ( L' = (reverse L) ) = $L_{1 + (\text{length } L) - n}$

i. This was proved in Assignment 4