(define (reverse l)

(if (null? l)

null

(append (reverse (rest l)) (cons (first l) null))))

1. **(list? L) 🡪 (list? (reverse L))**
   1. Assume (list? L) = #t
   2. Base Case: L = ‘()
      1. (reverse ‘()) = (append (reverse (rest ‘())) (cons (first ‘()) null)) = (append (reverse ‘()) (cons ‘() null)) = (append ‘() ‘()) = ‘()
      2. (list? ‘()) = #t
   3. Inductive Hypothesis: Assume (list? B) = #t, (list? (reverse B)) = #t
   4. Inductive Proof: Prove that (list? (reverse (cons a B))) = #t
      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. (list? (reverse B)) = #t by Inductive Hypothesis
      3. (list? ‘(a)) = #t
      4. (and (list? (reverse B)) (list? ‘(a))) 🡪 (list? (append (reverse B) ‘(a)) by Property 1 of append
      5. (list? (reverse (cons a B))) = #t
   5. Proof complete by induction

1. **(length (reverse x)) = (length x)**
   1. Base Case: x = ‘()
      1. (length ‘()) = 0
      2. (reverse ‘()) = ‘()
      3. (length (reverse ‘())) = (length ‘()) = 0
   2. Inductive Hypothesis: assume (list? B) = #t, (length B) = (length (reverse B)) = n
   3. Inductive Proof: Prove that (length (reverse (cons a B))) = (length (cons a B))
      1. (length (cons a B)) = (+ 1 (length (rest (cons a B)))) = (+ 1 (length B)) = (+ 1 n)
      2. (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
      3. (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))))

1. **(reverse (append x y)) = (append (reverse y) (reverse x))**
   1. Base Case: x = ‘()
      1. (reverse (append ‘() y)) = y
   2. Inductive Hypothesis: (reverse (append B y)) = (append (reverse y) (reverse B))
   3. Inductive Proof: (reverse (append (cons a B) y)) = (append (reverse y) (reverse (cons a B)))
      1. (append (cons a B) y) = (append (append ‘(a) B) y) = (append ‘(a) (append B y))
         1. (append ‘(a) B) = (cons a B)
      2. (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)
      3. (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
      4. (append (append (reverse y) (reverse B)) ‘(a)) = (append (reverse y) (append (reverse B) ‘(a))) by Property 5 of append
      5. (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))
      6. (append (cons a B) y) = (append (reverse y) (reverse (cons a B)))
2. **(reverse (reverse x)) = x**
   1. Base Case: x = ‘()
      1. (reverse ‘()) = ‘()
      2. (reverse (reverse ‘())) = ‘()
   2. Inductive Hypothesis: (reverse (reverse B)) = B
   3. Inductive Proof: Prove that (reverse (reverse (cons a B))) = (cons a B)
      1. (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))
      2. (reverse (append (reverse B) ‘(a))) = (append (reverse ‘(a)) (reverse (reverse B))) by Property 3 of append
      3. (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)
3. **(list? L) 🡪 ( and (length (reverse L)) == (length L)) ((nth n (reverse L)) == (nth (- (+ (length L) 1) n) L)))**
   1. (nth n (reverse L)) == (nth (- (+ (length L) 1) n) L)) means that L’n ( L’ = (reverse L) ) = L1 + (length L) – n
      1. This was proved in Assignment 4