(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. **(nth n (reverse L)) = (nth (- (+ (length L) 1) n) L)**
   1. This means that L’n = L1 + (length L) – n
   2. This was proved in Assignment 4, which is reproduced below.

L is a list of length n > 0 with L = (L1, L2, …, Ln). We seek to prove that calling (reverse L) reverses the elements of the list: (reverse L) = L’ = (Ln, Ln-1, …, L2, L1) and that the ith element of (reverse L) is the (n + 1 – i)th element of L (L’i = Ln+1-i).

The Racket code for reverse is:

(define (reverse L)

(if (null? L)

null

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

)

)

Base case for list of length 1:

L = (L1)

L’ = (reverse L)

(rest L) = null

= (append (reverse null) (cons L1 null)) = (append null (L1)) = (L1)

L’ = (L1)

L’i=1 = L1+1-1

Inductive hypothesis for list of length k:

L = (L1, L2, …, Lk)

Let us assume that L’ = (reverse L ) = (Lk, Lk-1, …, L2, L1) and that L’i = Lk+1-i

Inductive proof for list of length k + 1

We must prove that for a list L = (L1, L2, …, Lk, Lk+1), L’ = (reverse L). L’ = (Lk+1, Lk, Lk-1, …, L2, L1) and that L’i = Lk+1+1-i

(reverse L) =

(rest L) = (L2, L3, …, Lk, Lk+1)

= (append (reverse (L2, L3, …, Lk, Lk+1)) (cons L1 null))

= (append (reverse (L2, L3, …, Lk, Lk+1)) (L1))

(L2, L3, …, Lk, Lk+1) is a list of size k. From the inductive hypothesis, we can assume that (reverse (L2, L3, …, Lk, Lk+1)) successfully reverses the list.

(reverse (L2, L3, …, Lk, Lk+1)) = (Lk+1, Lk, …, L3, L2)

= (append (Lk+1, Lk, …, L3, L2) (L1)) = (Lk+1, Lk, …, L3, L2, L­1)

L’ = (Lk+1, Lk, …, L3, L2, L­1)

To show that L’i = Lk+1+1-i

L’i=k+1 = the last element of L’ = L1

L’i, i < k + 1 = the ith element of (Lk+1, Lk, …, L3, L2) = the ith element of (reverse (L2, L3, …, Lk, Lk+1))

From the inductive hypothesis, the ith element of (reverse (L2, L3, …, Lk, Lk+1)) is the (k- i+1)th element of (L2, L3, …, Lk, Lk+1). The (k-i+1)th element is Lk+2-i because the list (L2, L3, …, Lk, Lk+1) starts at L2 rather than L1 as in our inductive hypothesis. Therefore, L’i = Lk+2-i.

The inductive proof is complete.

The proof is complete by induction.