Find the length of a list

```
(define (length List)
  (if (null? List) 0
    (+ 1 (length (cdr List)))))
```

- (length ‘(2 4 6 1)) : 4

Find the sum of elements in a list

```
(define (sum List)
  (if (null? List) 0
      (+ (car List) (sum (cdr List)))))
```

- (sum '(2 4 6 1)) : 13

Find the maximum value in a list

```
(define (maximum List)
  (cond ((null? List) (display 'error))
        ((null? (cdr List)) (car List))
        (else (max (car List)
          (maximum (cdr List))))))
(define (max x y)
  (if (> x y) x y))
```

- (maximum '(2 4 6 1)) : 6

Find the minimum value in a list

```
(define (minimum List)
  (cond ((null? List) (display ‘error))
        ((null? (cdr List)) (car List))
        (else (min (car List)
          (minimum (cdr List))))))
(define (min x y)
  (if (< x y) x y))
```

- (minimum ‘(2 4 6 1)) : 1

Append a list to another list

```
(define (append L R)
  (if (null? L) R
    (cons (car L) (append (cdr L) R))))
```

- (append '(1 2 3) '(4 5 6)) :
  (1 2 3 4 5 6)

Reverse a list

```
(define (reverse List)
  (if (null? List) List
    (append (reverse (cdr List))
      (cons (car List) '() ))))
```

- (reverse '(2 4 6 1)) : (1 6 4 2)