```
Expression
prefix notation
(+ 137 349)
486

(- 1000 334)
666

(* 5 99)
495

(/ 10 5)
2

(+ 2.7 10)
12.7

(+ 21 35 12 7)
75

(* 25 4 12)
1200

(+ (* 3 5) (- 10 6))
19

(+ (* 3 (+ (* 2 4) (+ 3 5))) (+ (- 10 7) 6))

(+ (* 3
        (+ (* 2 4)
           (+ 3 5)))
   (+ (- 10 7)
        6))

define variable
(define size 2)
size
2

(* 5 size)

(define pi 3.14159)
(define radius 10)
(* pi (* radius radius))
314.159
(define circumference (* 2 pi radius))
circumference
62.8318
```

```
Evalutaing using tree representation
(* (+ 2 (* 4 6))
   (+ 3 5 7)

procedure definition
(define (square x) (* x x))
(define (<name> <formal parameters>)
 (<body>)

(square 21)
441
(square (+ 2 5))
49
(square (square 3))

(define (sum-of-squares x y)
  (+ (square x) (square y)))
(sum-of-squares 3 4)
25(define (abs x)
(cond ((> x 0) x)
((= x 0) 0)
((< x 0) (- x))))

(define (f a)
  (sum-of-squares (+ a 1) (* a 2)))
(f 5)
136

conditional expression
(define (abs x)
  (cond ((> x 0) x)
        ((= x 0) 0)
        ((< x 0) (- x))))

(define (abs x)
  (cond ((< x 0) (- x))
        (else x)))

predicate
(define (abs x)
  (if (< x 0)
      (- x)
      x))

(if <predicate> <consequence> <alternative>)
(and <e1> … <en>)
(or <e1> … <en>)
(not <e>)
```

```
(and (> x 5) (< x 10))
(define (>= x y) (or (> x y) (= x y)))
(define (>= x y) (not (< x y)))

recursive lisp
(define (factorial n)
  (if (= n 1)
      1
      (* n (factorial (– n 1)))))


(factorial 6)
(* 6 (factorial 5))
(* 6 (* 5 (factorial 4)))
(* 6 (* 5 (* 4 (factorial 3))))
(* 6 (* 5 (* 4 (* 3 (factorial 2)))))
(* 6 (* 5 (* 4 (* 3 (* 2 (factorial 1))))))
(* 6 (* 5 (* 4 (* 3 (* 2 1)))))
(* 6 (* 5 (* 4 (* 3 2))))
(* 6 (* 5 (* 4 6)))
(* 6 (* 5 24))
(* 6 120)
720

(define (factorial n)
  (fact-iter 1 1 n))
(define (fact-iter product counter max-count)
  (if (> counter max-count)
      product
      (fact-iter (* counter product)
                 (+ counter 1)
                 max-count)))


(factorial 6)
(fact-iter 1 1 6)
(fact-iter 1 2 6)
(fact-iter 2 3 6)
(fact-iter 6 4 6)
(fact-iter 24 5 6)
(fact-iter 120 6 6)
(fact-iter 720 7 6)
720
```