"Racket Exercises"

Mobin Kheibary
Student No: 99442097

نام و نام خانوادگی: مبین خیبری
شماره دانشجویی: 99442097

تمرین اول:

#lang racket

(define a 5) a |-> 5

(define b (* a a)) b |-> 25

(define c (+ (* 2 a) (- b a))) c |-> 30

(define d (+ a b c)) d |-> 60

(define e (+ a))

↳ error -- +: (+) requires at least 1 argument, but only 0 given. The expression (+ a) tries to add the value of a to an empty list of arguments.

(define f (+)) f: error -- (+) requires at least 1 argument, but only 0 given. The expression (+) tries to add an empty list of arguments.

(define g (*)) g: error -- (*) requires at least 1 argument, but only 0 given. The expression (*) tries to multiply an empty list of arguments.

(define h b) h |-> 25

(define i (if (< a c) 7 (quotient 43 a))) i |-> 8

(define j (if (= a 0) 11 (remainder 43 a))) j |-> 43 (when a is not equal to 0, the remainder of 43 divided by a is 43, and thus j is bound to 43)

(define k (/ i j))

k |-> 0.186046511627906977 (when a is not equal to 0, i is 8 and j is 43, so k is bound to 8/43)

(define l (< a b)) l |-> #t (since 5 is less than 25)

(define m (= b c)) m |-> #f (since 25 is not equal to 30)

(define n (< b c)) n |-> #t (since 25 is less than 60)

(define o (+ a m)) o |-> 30 (since m is false, the if expression in the definition of o evaluates to (+ a 0) which is equal to a, and a is 5, so o is bound to 30)

(define p (< a m)) p |-> #f (since m is bound to 30, and 5 is not less than 30)

(define q (if a b c)) q |-> 25 (since a is 5, b is 25, and c is 30, the condition a evaluates to true, so q is bound to b, which is 25)

(define r (if l m n)) r |-> #f (since l is true, and m is false, the if expression evaluates to n, which is #t)

(define s (if (if (< i j) m l) s |-> 30 (since the condition (if (< i j) m l) evaluates to m which is false, the if expression evaluates to (if (< a b) (+ a b) (* a b)), which is (+ a b), which is 30)
(if (> a c) (+ a c) (* a c))
(if (< a b) (+ a b) (* a b))))

```
(define+ (= (if (< b c)
                (* (remainder (- (* a b) (* 2 a)) c) (- quotient d b))
                (* (- (+ d (* (* a b) (quotient d a))))))
            (- (* (/ (- c a) (- a 1)) (quotient b (if (> a c) i j)))))))
```

+: error -- (-) requires at least 1 argument, but only 0 given. The expression (- (* (/ (- c a) (- a 1)) (quotient b (if (> a c) i j)))) tries to subtract an empty list of arguments from the value of the condition in the outermost define expression.

```
(define u (if (> i 8) (< a 0) (< (< b c) a)))
```

u |-> #f (since i is 6 and 8 is 43, (> i 8) evaluates to #f, a is 5, b is 25, and c is 30, so the condition (< b c) evaluates to #t, and thus u is bound to #f)

```
(define v (if (<= i 8) (< a 0) (< a 0) (< (< b c) a)))
```

v |-> #t (since i is 6 and 8 is 43, (<= i 8) evaluates to #t, a is 5, b is 25, and c is 30, so the condition (< b c) evaluates to #t, and thus v is bound to #t)

envs, (+ (if (< a b) (* a (+ b c)) (+ b #t))
(if (if (> a c) (< c b) (- c 4)) (* 2 b) a))

--> [B-PLUS]

envs, (if (< a b) (* a (+ b c)) (+ b #t))
+ [B-PLUS]

envs, (if (if (> a c) (< c b) (- c 4)) (* 2 b) a)

--> [B-IF-TRUE]

envs, (* a (+ b c))

--> [B-STAR]

envs, a

--> [B-IF-FALSE]

envs, (+ b #t)

--> [B-ADD]

envs, b

--> [B-IF-TRUE]

envs, (* 2 b)

--> [B-STAR]

envs, 12

--> [B-IF-FALSE]

envs, a

--> [B-ADD]

envs, 15

تمرین دوم:

B_PLUS: evaluates the sum of two expressions

B-IF_TRUE: evaluates the consequent of an if expression when the predicate evaluates to true.

B-IF-FALSE: evaluates the alternate of an if expression when the predicate evaluates to False

B-STAR: evaluates the product of two expressions.

B-ADD: evaluates the sum of two value.

env5, (+ (if (< a b) (* a (+ b c)) (+ b #t))
(if (if (> a c) (< c b) (- c 4)) (* 2 b) a))
--> {B-PLUS}
env5, (if (< a b) (* a (+ b c)) (+ b #t))
+ (if (if (> a c) (< c b) (- c 4)) (* 2 b) a)
--> {B-IF-TRUE}
env5, (* a (+ b c))
+ (if (if (> a c) (< c b) (- c 4)) (* 2 b) a)
--> {B-STAR}
env5, (* 3 (+ b c))
+ (if (if (> a c) (< c b) (- c 4)) (* 2 b) a)
--> {B-ADD}
env5, (* 3 b)
+ (if (if (> a c) (< c b) (- c 4)) (* 2 b) a)
--> {B-IF-TRUE}
env5, (* 3 b)
+ (* 2 b)
+ a
--> {B-STAR}
env 5, 30
+ (* 2 b)
+ a
--> {B-ADD}
env6, 30
+ 12
+ a
--> {B-ADD}
env5, 42
+ a
--> {B-ADD}
env 5, 45

B-PLUS: evaluates the sum of two expressions.
B-IF-TRUE: evaluates the consequent of an if expression when the predicate evaluates to true.
B-IF-FALSE: evaluates the alternate of an if expression when the predicate evaluates to false.
B-STAR: evaluates the product of two expressions.
B-ADD: evaluates the sum of two values.