**Structures and Interpretation of Computer Program**

**Exercise Chapter 2.1 Name:** Wan Huzaifah bin Wan Azhar

**Exercise 2.1.1 Arithmetic Operation for Rational Number**

1. Define better make-rat

(define (abs x)

(if (< x 0)

(\* -1 x)

x))

(define (make-rat n d)

(if (< (/ (\* n 1.0) d) 0)

(cons (\* -1 (abs n)) (abs d))

(cons n d)))

(define (numer x) (car x))

(define (denom x) (cdr x))

(define (print-rat x)

(newline)

(display (numer x))

(display "/")

(display (denom x)))

(define (add-rat x y)

(make-rat (+ (\* (numer x) (denom y))

(\* (numer y) (denom x)))

(\* (denom x) (denom y))))

(define t (make-rat 1 -3))

(print-rat t)

1. Represent line segment in a plane

(define (make-point x y)

(cons x y))

(define (x-point point)

(car point))

(define (y-point point)

(cdr point))

(define (make-segment x1 y1 x2 y2)

(cons (make-point x1 y1) (make-point x2 y2)))

(define (start-segment segment)

(car segment))

(define (end-segment segment)

(cdr segment))

(define (midpoint-segment segment)

(make-point

(/ (+ (x-point (end-segment segment)) (x-point (start-segment segment))) 2)

(/ (+ (y-point (end-segment segment)) (y-point (start-segment segment))) 2))

)

(define (print-point p)

(newline)

(display "(")

(display (x-point p))

(display ",")

(display (y-point p))

(display ")"))

(define line (make-segment 1 2 3 4))

(print-point (midpoint-segment line))

1. Make two different rectangle representation and make it work with perimeter and area
2. Rectangle using segment

(define (make-point x y)

(cons x y))

(define (x-point point)

(car point))

(define (y-point point)

(cdr point))

(define (make-segment x1 y1 x2 y2)

(cons (make-point x1 y1) (make-point x2 y2)))

(define (start-segment segment)

(car segment))

(define (end-segment segment)

(cdr segment))

(define (midpoint-segment segment)

(make-point

(/ (+ (x-point (end-segment segment)) (x-point (start-segment segment))) 2)

(/ (+ (y-point (end-segment segment)) (y-point (start-segment segment))) 2))

)

(define (print-point p)

(newline)

(display "(")

(display (x-point p))

(display ",")

(display (y-point p))

(display ")"))

(define (make-rect width height)

(cons (make-segment 0 0 width 0)

(make-segment 0 0 0 height)))

(define (rect-width rect)

(x-point (end-segment (car rect))))

(define (rect-height rect)

(y-point (end-segment (cdr rect))))

(define (perimeter rect)

(+ (\* (rect-width rect) 2)

(\* (rect-height rect) 2))

)

(define (area rect)

(\* (rect-width rect) (rect-height rect)))

(display (area (make-rect 3 4)))  
(display (perimeter (make-rect 3 4)))

1. Rectangle using point

(define (make-point x y)

(cons x y))

(define (x-point point)

(car point))

(define (y-point point)

(cdr point))

(define (make-segment x1 y1 x2 y2)

(cons (make-point x1 y1) (make-point x2 y2)))

(define (start-segment segment)

(car segment))

(define (end-segment segment)

(cdr segment))

(define (midpoint-segment segment)

(make-point

(/ (+ (x-point (end-segment segment)) (x-point (start-segment segment))) 2)

(/ (+ (y-point (end-segment segment)) (y-point (start-segment segment))) 2))

)

(define (print-point p)

(newline)

(display "(")

(display (x-point p))

(display ",")

(display (y-point p))

(display ")"))

(define (make-rect width height)

(cons (make-point width 0)

(make-point 0 height)))

(define (rect-width rect)

(x-point (car rect)))

(define (rect-height rect)

(y-point (cdr rect)))

(define (perimeter rect)

(+ (\* (rect-width rect) 2)

(\* (rect-height rect) 2))

)

(define (area rect)

(\* (rect-width rect) (rect-height rect)))

(display (perimeter (make-rect 3 4)))

(display (area (make-rect 3 4)))