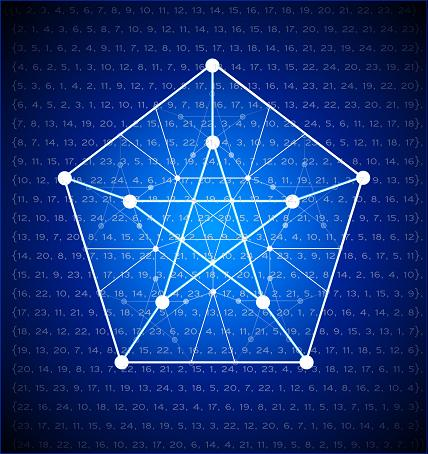
****

**Tecnológico de Monterrey**

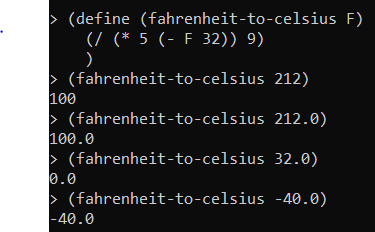
**TC2037: Implementación de métodos computacionales**

Actividad 2.1: Programación funcional – Parte 1

****

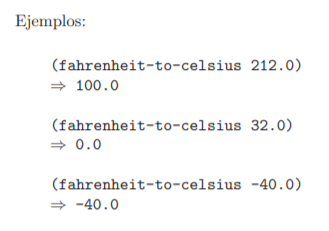
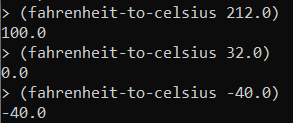
* Carlos Estrada Ceballos A01638214
* Abigail Velasco García A01638095
* Natalia Velasco García A01638047
* **1**

La función fahrenheit-to-celsius toma como entrada una temperatura f en grados Fahrenheit y la convierte a su equivalente en grados Celsius usando la siguiente fórmula:

(define (fahrenheit-to-celsius F) 

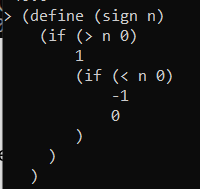
(/ (\* 5 (- F 32)) 9)

)

* **2**

La función sign recibe como entrada un valor entero n. Devuelve -1 si n es negativo, 1 si n es positivo mayor que cero, o 0 si n es cero.

(define (sign n) 

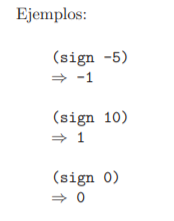
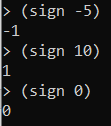
(if (> n 0)

1

(if (< n 0)

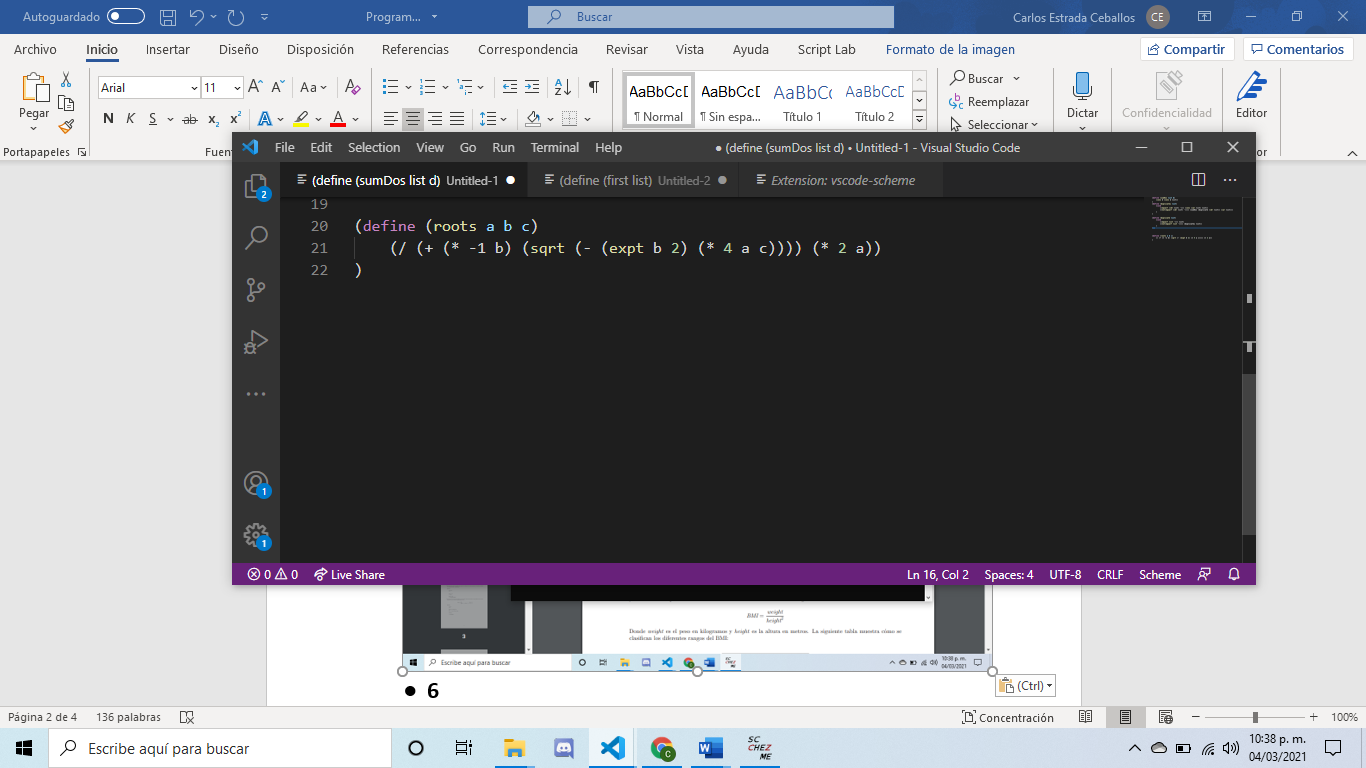
-1

0)))

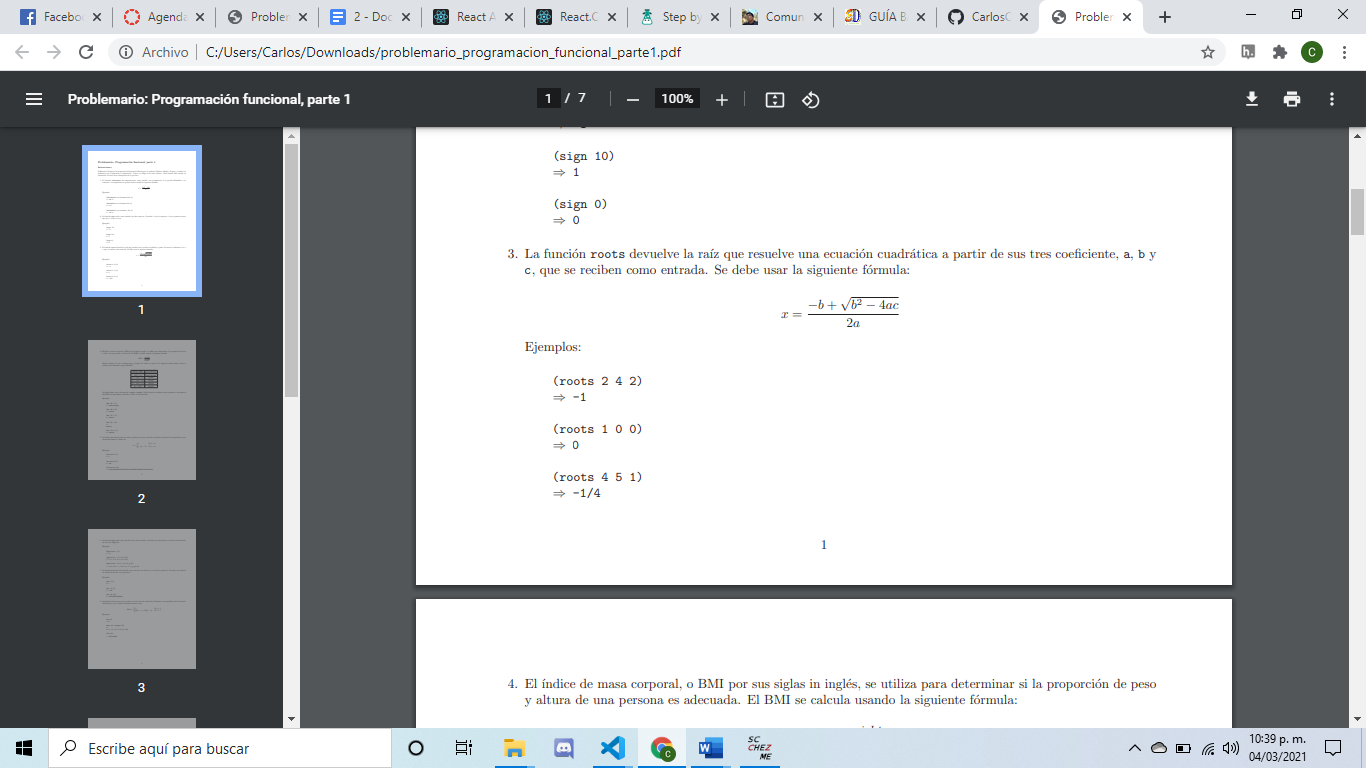
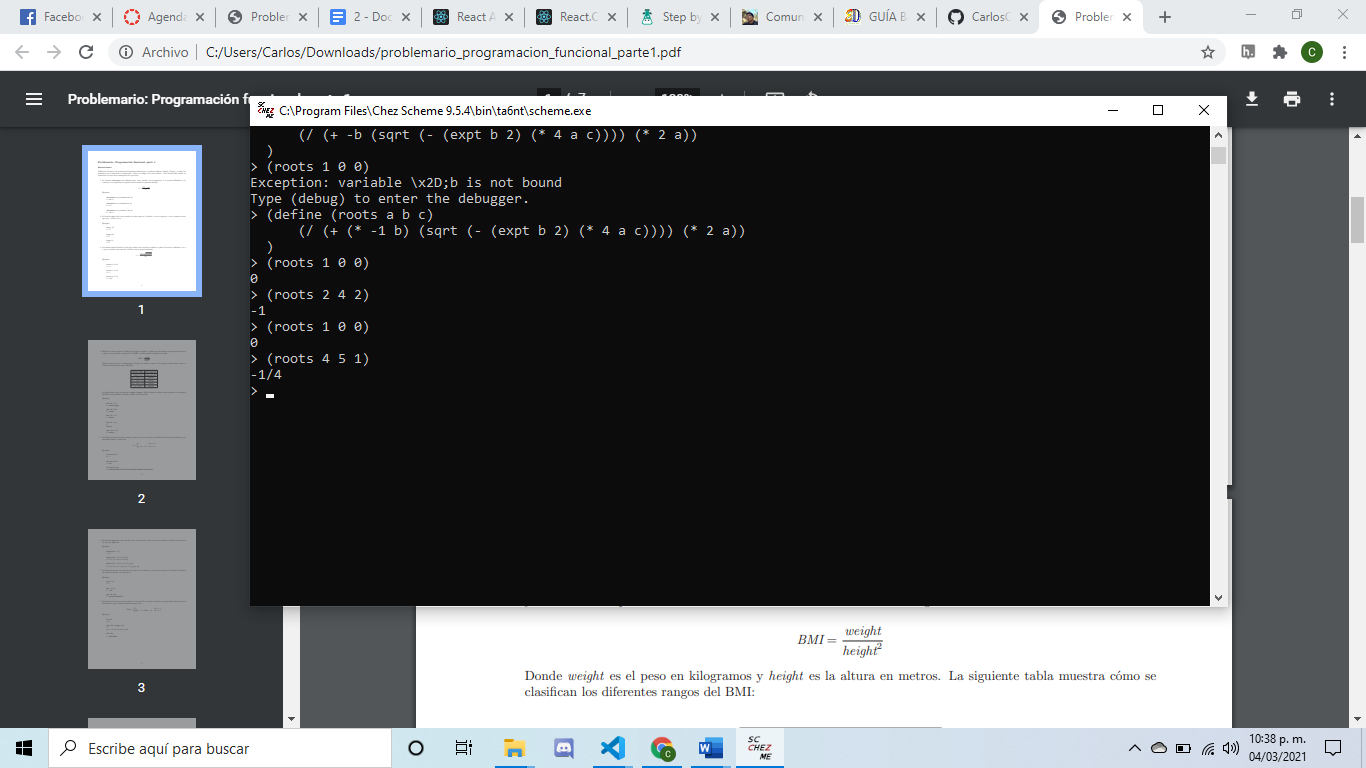
* **3**

La función roots devuelve la raíz que resuelve una ecuación cuadrática a partir de sus tres coeficientes, a, b y c, que se reciben como entrada. Se debe usar la siguiente formula:



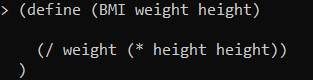
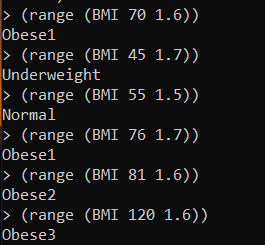
(define (roots a b c)

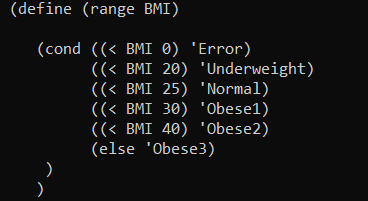
(/ (+ (\* -1 b) (sqrt (- (expt b 2) (\* 4 a c)))) (\* 2 a))

)

* **4**

La función bmi recibe dos entradas: weight y height. Debe devolver un símbolo que represente la descripción del BMI correspondiente calculado a partir de sus entradas.

****



> (define (BMI weight height)

(/ weight (\* height height))

)

> (define (range BMI)

(cond ((< BMI 0) 'Error)

((< BMI 20) 'Underweight)

((< BMI 25) 'Normal)

((< BMI 30) 'Obese1)

((< BMI 40) 'Obese2)

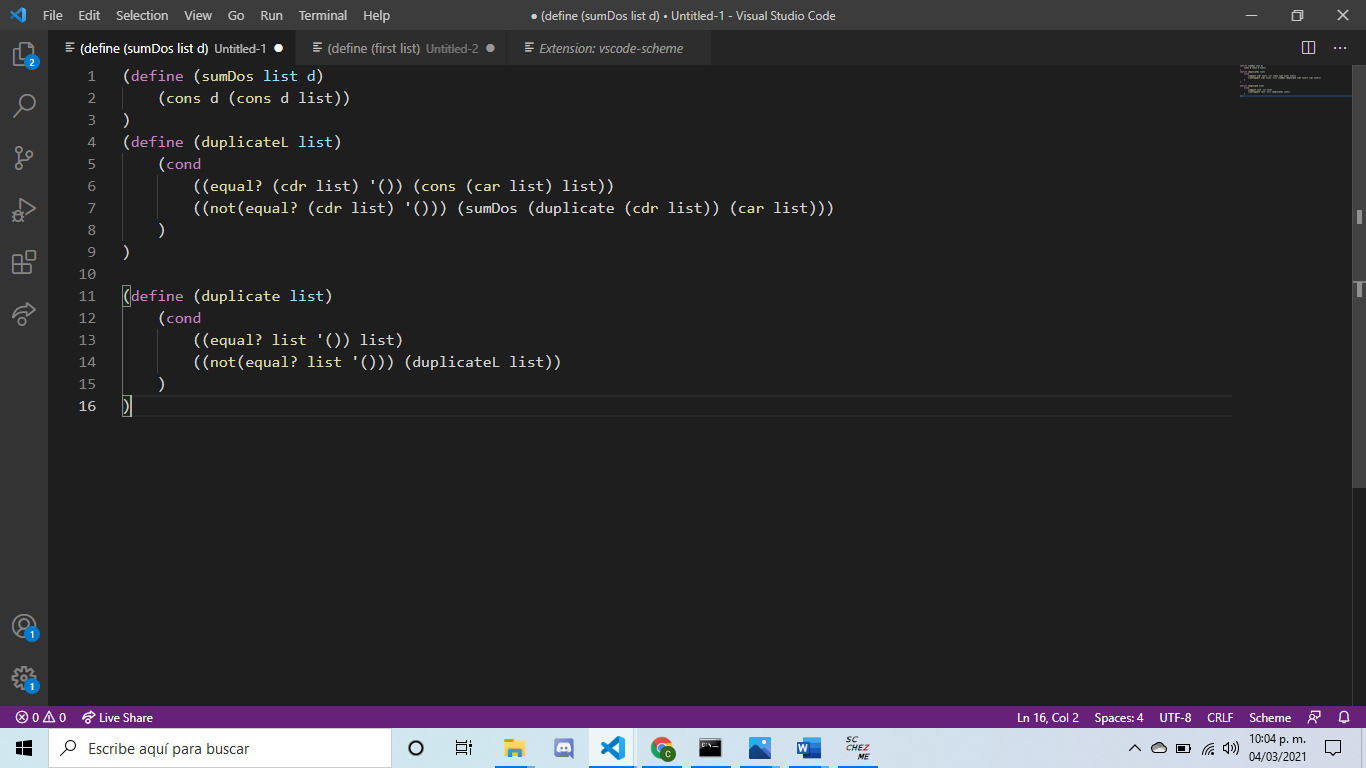
(else 'Obese3)

)

)

* **6**

La función ‘duplicate’ toma una lista ‘lst’ como entrada y devuelve una nueva lista en donde cada elemento de ‘lst’ está duplicado.



(define (sumDos list d)

(cons d (cons d list))

)

(define (duplicateL list)

(cond

((equal? (cdr list) '()) (cons (car list) list))

((not(equal? (cdr list) '())) (sumDos (duplicate (cdr list)) (car list)))

)

)

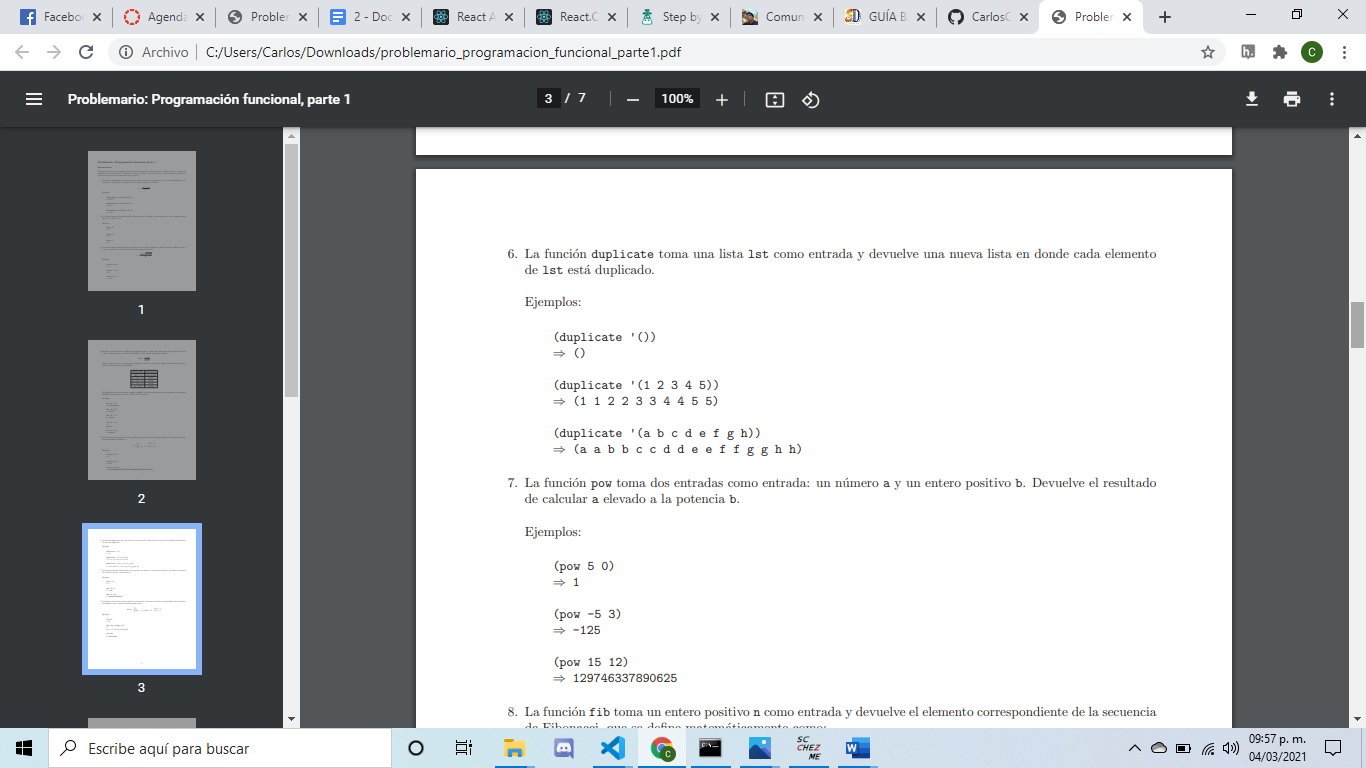
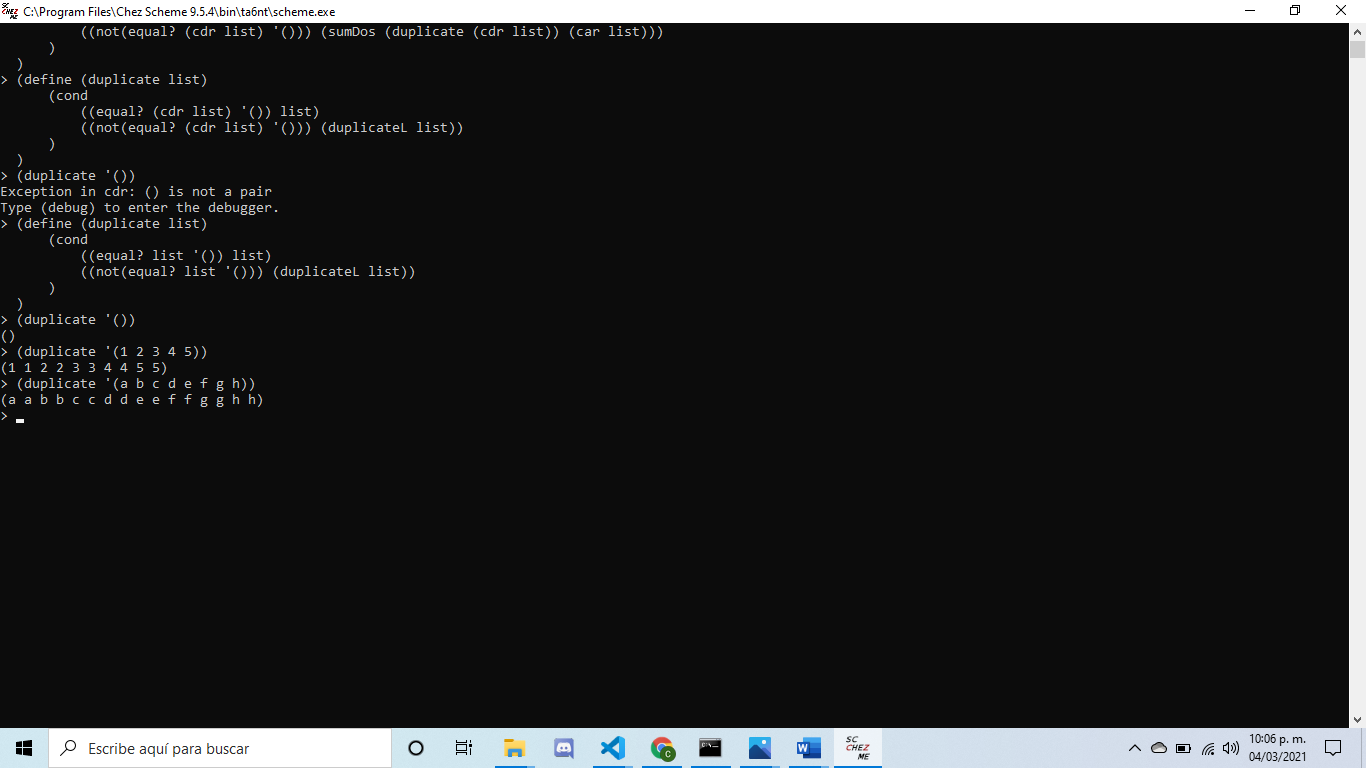
(define (duplicate list)

(cond

((equal? list '()) list)

((not(equal? list '())) (duplicateL list))

)



* **10**

La función positives toma una lista de números lst como entrada y devuelve una nueva lista que solo contiene los números positivos de lst.

> (define (positives2 list list2)

(if (null? list)

list2

(if (> (car list) 0)

(positives2 (cdr list) (append list2 (cons (car list) '())))

(positives2 (cdr list) list2)

)

)

)

> (positives '(12 -4 3 -1 -10 -13 6 -5))

(12 3 6)

> (positives '())

()

> (positives '(-4 -1 -10 -13 -5))

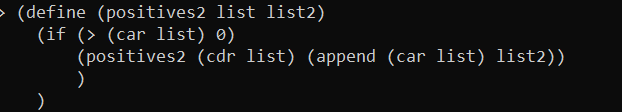
()

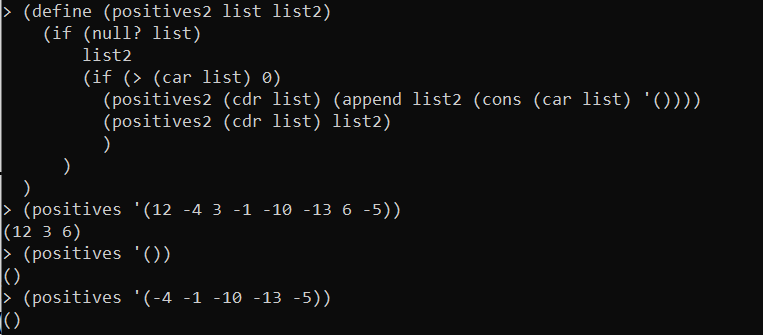
>

(define (positives list)

(positives2 list '())

)

****

****