Diego Velázquez
Sebastián Rojas
Pablo Blanco

Tarea 2 - Actividad 01

```
; Ejercicio 01 - Sebas
; El codigo de la funcion de conversión funciona directamente implementando
la fórmula de conversión donde la entrada de la función es el valor en
fahrenheit a convertir y la salida es la temperatura en celsius.
; Se resuelven las operaciones directamente
(define (fahrenheit-to-celsius f)
   (/ ( * (- f 32) 5) 9)
)
```

```
; Ejercicio 02 - Sebas
; El codigo de la funcion para definir el signo del numero funciona mediante
conditional donde se definen los distintos casos que podrian suceder.
(define (sign n)
  (cond
    ((= n 0) 0)
    ((< n 0) -1)
    ((> n 0) 1)
  )
)
```

```
; Ejercicio 03 - Diego
; El codigo de la funcion de roots realiza la solucion de una ecuacion
cuadratica por medio de los tres coeficientes 'a b c' ademas se uso sqrt para
la raiz cuadrada y expt para elevar al cuadrado
(define (roots a b c)
  (/ (+ (- 0 b) (sqrt (- (expt b 2) (* 4 (* a c))))) (* 2 a))
)
```

```
; Ejercicio 04 - Diego
; El codigo de la funcion ibm esta dividido en dos funciones una que calcula
el ibm y dentro del otro hace la comparacion mediante los diferentes casos
que existen
(define (calculator w h)
  (/ w (expt h 2))
)

(define (bmi weight height)
(cond
    ((< (calculator weight height) 20) 'Underweight )
    ((< (calculator weight height) 25) 'Normal )
    ((< (calculator weight height) 30) 'obese1 )
    ((< (calculator weight height) 40) 'obese2 )
    ((> (calculator weight height) 39) 'obese3 )
  )
)
```

```
; Ejercicio 06 - Pablo
; Primero se revisa que la lista que se entrega no venga vacía. Si viene vacía se regresa vacía, sino se pasa a otra función que lo que hace es llamarse recursivamente hasta que se dupliquen todos
;   los valores dentro de la lista (como si fuera un while, o un forEach). Ya que se duplicaron cada uno de los valores entonces se pasa a una función que voltea toda la lista porque habían acabado volteados.
;   Ya que se voltean entonces se regresa la lista con los duplicados.
(define (reverse l)
    (if (null? l)
        '()
        (append (reverse (cdr l)) (list (car l)))
    )
)

(define (d new old)
    (if (null? (cdr old))
        (reverse (append (cons (car old) old) new))
        (d (cons (car old) (cons (car old) new)) (cdr old)))
)

(define (duplicate l)
    (if (null? l)
        '()
        (d '() l))
)
```

```
; Ejercicio 10 - Pablo
; Primero se revisa que la lista que se entrega no venga vacía. Si viene vacía se regresa vacía, sino se pasa a otra función que lo que hace es llamarse recursivamente hasta que cada uno de los números positivos
;   de esa lista pasa a otra lista, como si fuera un forEach. No se hace la revisión de si el dato es un número ya que como no tenía casos de prueba así dentro del archivo de tarea entonces se supuso que no
;   se requerían. Ya que pasaron todos los positivos a la otra lista se usa la función de reverse para volear el orden de la última lista, porque como se habían estado insertando en la cabeza de la otra lista habían
;   acabado voleados. Después se regresa la lista final.
(define (reverse l)
    (if (null? l)
        '()
        (append (reverse (cdr l)) (list (car l)))
    )
)

(define (p new old)
    (if (null? old)
        (reverse new)
        (if (> (car old) 0)
            (p (cons (car old) new) (cdr old))
            (p new (cdr old))))
)

(define (positives l)
    (if (null? l)
        '()
        (p '() l))
)
```