# EJERCICIOS

## Programación funcional en lenguaje Scheme

### RECURSIVIDAD

1. Escribir un procedimiento en Scheme que sirva para convertir un número decimal a su equivalente en binario. El procedimiento debera llamarse *dec-bin*, y tendrá como argumento al número a convertir. El procedimiento retornará la equivalencia en binario del número.
Ejemplos de las llamadas al procedimiento:

] (dec-bin 5)
*101*
] (dec-bin 11)
*1011*
] (dec-bin 256)
*100000000*

(HINT: El valor binario puede ser visto como una secuencia de 1's y 0's. Buscar desplegar unos y ceros, y NO el valor en binario.)

2. Escribir un procedimiento en Scheme que sirva para convertir un número decimal a su equivalente en en cualquier otra base numérica (entre 2 y 10 inclusive). El procedimiento debera llamarse *dec-base*, y tendrá como argumentos al número a convertir y la base. El procedimiento retornará el número representado en la base numérica dada.
Ejemplos de las llamadas al procedimiento:

] (dec-base 5 2)
*101* --> Convirtió el 5 decimal a base binaria.
] (dec-base 22 7)
*31* --> Convirtió el 22 decimal a base 7.
] (dec-base 22 8)
*26* --> Convirtió el 22 decimal a base octal.
] (dec-base 267 10)
*267* --> Convirtió el 267 decimal a la misma base decimal.

3. Hace tiempo, en algún lugar del mundo, el famoso matemático "Nerdo Troublemaker" creó lo que se conoce ahora como la SECUENCIA LOCA.
Esta secuencia consiste en una serie de números que se forman de la siguiente manera:
- Los primeros 3 números de la serie son: 1, 1, 1. (A, B, C)
- Los siguientes números de la serie se calculan en base a la siguiente fórmula: **C + 2B + 3A** , donde A, B, C son los 3 números anteriores al número que se quiere calcular.

Los primeros pocos números de la serie son: 1, 1, 1, 6, 11, 26, 66 .....
En base a la información anterior, realiza la implementación de un procedimiento en Scheme que sea útil para obtener el n-ésimo elemento de la SECUENCIA LOCA.

4. El algoritmo Euclidiano para obtener el Máximo Común Divisor (MCD) de dos números se muestra a continuación:

Sean **a** y **b** dos números enteros positivos y mayores a cero a los cuales se les desea obtener el Máximo Común Divisor (MCD). Para obtener el MCD de **a** y **b** realizar lo siguiente:

1.- Dividir **a** entre **b** para obtener un cociente **q** y un residuo **r**, de tal manera que: **a**=**bq**+**r**.

2.- Si **r** es diferente de cero, remplazar **a** por **b** y **b** por **r,** y volver a repetir los pasos 1 y 2.
Si **r** es igual a cero, el MCD de **a** y **b** es **b** .

Escribe la implementación de un procedimiento en Scheme que sirva para obtener el máximo común divisor de 2 números enteros positivos y mayores a cero.

5. La Función de ACKERMAN está definida para m>=0 y n>=0 de la siguiente forma:

Ack(0,n) = n+1
Ack(m,0) = Ack(m-1,1)
Ack(m,n) = Ack(m-1, Ack(m, n-1))

Escribe un procedimiento en Scheme que sirva para evaluar la función de Ackerman.

6. Escribe la defición de un procedimiento en Scheme que sirva para encontrar el n-ésimo dígito, de derecha a izquierda de cualquier número entero. Escribe un procedimiento equivalente pero que encuentre el n-ésimo dígito de izquierda a derecha.

7. Una aproximación del valor de $e^x$ puede ser calculada evaluando el valor de la siguiente serie infinita:

$$e^x = 1 + x + x^2/2! + x^3/3! + x^4/4! + .....$$

donde el factorial de cualquier número es: n! = n * (n-1) * (n-2) * .... * 1.

Escribe un procedimiento en Scheme que sirva para calcular el exponencial de un número dado por el usuario, aplicando la evaluación de la serie. La serie deberá ser evaluada hasta que el valor del último término sumado en la serie sea menor a 0.000001.

8. Escribe la definición de un procedimiento en Scheme, que sirva para sumar los dígitos que componen un número entero. Por ejemplo, para el número 12567, la suma de sus dígitos es 1+2+5+6+7 = 21.
Utiliza este procedimiento para comprobar que la suma de los dígitos de cualquier número que sea múltiplo de 9, es un múltiplo de 9, y por lo tanto, la suma de los dígitos a cualquier nivel, finalmente deber de ser 9.
Ejemplos: 1111104 es múltiplo de 9, pues 1+1+1+1+1+0+4 = 9
9999936 es múltiplo de 9, pues 9+9+9+9+9+3+6 = 54, y 5+4 = 9

9. Un método para calcular la depreciación de un artículo conforme pasan los años, es el método de la SUMA DE AÑOS DIGITOS. El método consiste en obtener la sumatoria desde 1 hasta la cantidad de años a depreciar y usar este valor ∑, en la siguiente fórmula:

cantidad a depreciar = valor real del artículo * ( cantidad de años que faltan por depreciar / ∑ )

Así por ejemplo, si se deseara depreciar en 5 años un artículo cuyo valor real es de $15,000 , se obtendrían las siguientes cantidades:

Cantidad a depreciar en el año 1 = 1500*(5/15) = 5000
Cantidad a depreciar en el año 2 = 1500*(4/15) = 4000
Cantidad a depreciar en el año 3 = 1500*(3/15) = 3000
Cantidad a depreciar en el año 4 = 1500*(2/15) = 2000
Cantidad a depreciar en el año 5 = 1500*(1/15) = 1000

Observar que ∑ = 15 pues la cantidad de años a depreciar es 5, y 15 = 1+2+3+4+5

Escribe la implementación de un procedimiento en Scheme, que dado el valor de un artículo y la cantidad de años a depreciar, muestre en pantalla las cantidades que en cada año se deberá depreciar el artículo.

10. Analizar cada una de las soluciones de los problemas anteriores, e indicar si utilizan recursividad terminal. En caso de que no sea así, convertir las soluciones de tal forma que utilicen recursividad terminal.

## MANIPULACIÓN GENERAL DE LISTAS

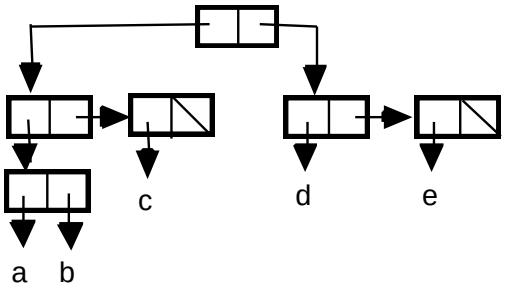
11. Indicar que responde el interpretador de Scheme a las siguientes llamadas:
NOTA: Suponer que previamente se hicieron las siguientes definiciones:

*(define lista1 (cons (cons 1 (cons 2 nil)) (cons 3 nil)) )*
*(define lista2 (list (list 4 5) 6 (list 7)) )*
*(define lista3 '((a) b c ((d)) e) )*

a) lista1
b) (append lista2 lista3)
c) (car (list 1 2 3))
d) (car '(list 1 2 3))
e) (car lista3)
f) (cdr lista2)
g) (list (car (+ 2 3)))
h) (list (car '(+ 2 3)))
i) (list '(car (+ 2 3)))
j) (list '(car '(+ 2 3)))
k) (car (cdr lista1))
l) (cdr (cdr (cdr lista3)))
m) (car (cdr (cdr '(cons '(a b c) '(d e)))))
n) (list 'a (append '(c d) ' (e f)))
o) (cons (list 'a 'b) (append '(c) '(d e)))
p) (cdadddr lista3)
q) (list (cddr lista2) (caar lista2))
r) (append (list (cons (* 2 3) ()) (- 8 1)) (cdr lista1))

12. A continuación se muestran listas en Scheme en diferentes formas de representación: visual, física (en forma gráfica), y con constructores. Para cada inciso encuentra las 2 representaciones faltantes equivalentes.

a) (cons (cons ‘a ‘b) ())
b) (a (b (c)))
c) ((a . 1) (b . 2))
d)




e) (list ‘(a b c) ‘d ‘(e))
f) ((a . b) c (d e))
g) (((a . b) . c) d e)

13. Para cada lista del problema anterior, indicar la forma en que se accesaría el elemento “b”.

## PROCEDIMIENTOS QUE RECIBEN LISTAS COMO ENTRADA Y GENERAN UN VALOR ATÓMICO COMO RESULTADO

14. Escribe la implementación en Scheme de un procedimiento llamado SUMA-DE-CUADRADOS, que sirva para calcular la sumatoria de los cuadrados de una lista numérica. Considerar que la lista de números es plana, es decir, no contine a sublistas como elementos.
Ejemplo de la llamada al procedimiento: (suma-de-cuadrados ‘(1 2 3 4)) -> 30

15. Escribir la definición de un procedimiento llamado CUAL-ES-MAYOR?, que reciba dos listas como entrada y que genere como salida un '1' si la primera lista tiene una longitud mayor o igual a la segunda, o bien, un '2' en otro caso.
Ejemplo de la llamada al procedimiento: (cual-es-mayor? '(a b c) '(1 2 3 4)) -> 2
(cual-es-mayor? '(a b c) '(1 2 3)) -> 1

16. Escribe la implementación en Scheme de un procedimiento llamado CUANTOS, que sirva para calcular cúantas veces aparece un átomo en una lista. Considerar que la lista en donde se buscará el átomo es plana, es decir, no contine a sublistas como elementos.
Ejemplo de la llamada al procedimiento: (cuantos 'a '(a b c b a a a b a a e)) -> 6

17. Escribe la implementación en Scheme de un procedimiento llamado ACCESA-N, que sirva para encontrar el N-ésimo elemento de una lista. Considerar que el primer elemento de la lista es el elemento #1 y así sucesivamente. En caso de querer accesar un elemento que no existe en la lista, el procedimiento regresará ().
Ejemplo de la llamada al procedimiento: (accesa-n 1 ‘(x y z (a b c) 7 8)) -> x
(accesa-n 4 ‘(x y z (a b c) 7 8)) -> (a b c)
(accesa-n 8 ‘(x y z (a b c) 7 8)) -> ()

18. Escribe la implementación en Scheme de un procedimiento llamado INDICE, que sirva para encontrar la posición de la primera ocurrencia de un elemento en una lista. Considerar que el primer elemento de la lista es el elemento #1 y así sucesivamente. En caso de que el elemento no exista en la lista, el procedimiento regresará (). Considerar que la lista es plana, es decir, no contine a sublistas como elementos.
Ejemplo de la llamada al procedimiento: (indice ‘y ‘(x y z a b c)) -> 2
(indice ‘w ‘(x y z a b c)) -> ()

19. Escribe la implementación en Scheme de un procedimiento llamado MIEMBRO?, que sirva para verificar si un átomo existe en una lista. El procedimiento funcionará como predicado, es decir, regresará un valor booleano como resultado. Considerar además, que la lista en donde se buscará el átomo es plana, es decir, no contine a sublistas como elementos.
Ejemplo de la llamada al procedimiento: (miembro? 'a '(a b c b a a a b a a e)) -> #t
(miembro? 'x '(a b c b a a a b a a e)) -> ()

20. Un conjunto es una estructura de datos cuya característica es que guarda a un grupo de datos, sin un orden especial; en este grupo de datos, NO deben existir repetidos, pues lo único que interesa es la pertenencia o no de los elementos en el conjunto. Escribe la implementación en Scheme de un procedimiento llamado ES-CONJUNTO? que sirva para verificar si los elementos de una lista conforman un conjunto. El procedimiento funcionará como predicado, es decir, regresará un valor booleano como resultado. Considerar además, que la lista en donde se buscará el átomo es plana, es decir, no contine a sublistas como elementos, y que se puede utilizar el procedimiento MIEMBRO? implementado anteriormente.
Ejemplo de la llamada al procedimiento: (es-conjunto? '(a b c b a a a b a a e)) -> ()
(es-conjunto? '(a b c x e)) -> #t

21. Escribe la implementación en Scheme de un procedimiento llamado PRODUCTO-PUNTO, que sirva para obtener el producto punto de 2 vectores que se encuentran simulados en 2 listas recibidas como entrada.
Ejemplo de la llamada al procedimiento: (producto-punto ‘(1 2 3) ‘(5 6 7)) -> 38

## PROCEDIMIENTOS QUE GENERAN UNA LISTA COMO RESULTADO

22. Escribir la definición de un procedimiento llamado CONVIERTE-DIGITOS, que reciba como entrada una lista de dígitos (valores entre 0 y 9) y que genere como salida la lista pero con los nombres de los dígitos (cero, uno, dos, tres, etc.).
Ejemplo de la llamada: (convierte-digitos '(9 4 6 1)) -> (nueve cuatro seis uno)

23. Escribir la definición de un procedimiento llamado ATOMOS-AL-FRENTE, que reciba una lista como entrada y que genere como salida la misma lista pero con los elementos acomodados de tal forma que primero aparezcan los elementos de la lista que son átomos.
Ejemplo de la llamada: (atomos-al-frente '((a b) (c) d (e) f)) -> (d f (a b) (c) (e))

24. Escribir la definición de un procedimiento llamado MEZCLA-LISTAS, que reciba dos listas de igual longitud como entrada y que genere como salida una lista que sea la unión de las 2 listas, de tal forma que el primer elemento sea el primero de la primer lista, el segundo el primero de la segunda lista, el tercero el segundo de la primera lista, el cuarto el segundo de la segunda lista , y así sucesivamente.
Ejemplo de la llamada: (mezcla-listas '(a b (c)) ´(1 2 3)-> (a 1 b 2 (c) 3)

25. Escribir la definición de un procedimiento llamado ELIMINA-IMPARES, que reciba una lista como entrada y que genere como salida la misma lista sin los elementos que se encuentran en las posiciones impares.
Ejemplo de la llamada: (elimina-impares '((a b) (c) d (e) f g)) -> ((c) (e) g)

26. Escribir la definición de un procedimiento llamado MAS-LARGA, que reciba dos listas como entrada y que genere como salida la lista de mayor longitud. En caso de que sean de igual longitud, regresar la primera lista.
Ejemplo de la llamada: (mas-larga '(a b c) '(1 2 3 4)) -> (1 2 3 4)
(mas-larga '(a b c) '(1 2 3)) -> (a b c)

27. Escribir la definición de un procedimiento llamado MISMA-POSICION, que reciba dos listas de igual longitud como entrada y que genere como salida una lista con aquellos elementos de las listas que aparezcan en las misma posición en ambas listas.
Ejemplo de la llamada: (misma-posicion '((a) b c) '(1 b 2)) -> (b)
(misma-posicion '(a b (c 2) d e) '(a 5 (c 2) e d)) -> (a (c 2))

## PROCEDIMIENTOS QUE MANEJAN LISTAS IMBRICADAS (RECURSIVIDAD PROFUNDA)

28. Escribe la implementación en Scheme de un procedimiento llamado CUENTA-ATOMOS, que sirva para calcular cúantos átomos contiene una lista. Considerar que la lista puede contener sublistas, y por lo tanto, se contarán los átomos sin importar en que nivel de anidamiento de listas se encuentran.
Ejemplo de la llamada al procedimiento: (cuenta-atomos '((a b (c) b) a ((a)((a b a) a)) e)) -> 11

29. Escribe la implementación en Scheme de un procedimiento llamado CUANTOS, que sirva para calcular cúantas veces aparece un átomo en una lista. Considerar que la lista en donde se buscará el átomo puede contener sublistas en donde también deberá buscarse el átomo.
Ejemplo de la llamada al procedimiento: (cuantos 'a '((a b (c) b) a ((a)((a b a) a)) e)) -> 6
Realiza la implementación en 2 versiones: una que haga uso de la recursividad terminal y otra no.

30. Escribe la implementación en Scheme de un procedimiento llamado PROFUNDIDAD, que sirva para encontrar la profundidad que tiene una lista imbricada. La profundidad es el número mayor de anidamientos de sublistas que posee una lista imbricada. Considerar que la profundidad de una lista plana es de 1(uno).
Ejemplos de la llamada al procedimiento: (profundidad ‘(a b c)) -> 1
(profundidad ‘((((a b c))))) -> 4
(profundidad ‘(((c))) d ((e))(a (b (c (d (e)))))) -> 6

31. Realiza la implementación de un procedimiento que sirva para obtener la cantidad de paréntesis (abiertos y cerrados) que posee una lista en su representación visual. (NOTA: la representación visual de una lista, es el formato en que se observa la lista en el interprete).
Ejemplos de llamadas al procedimiento: (parentesis '(1 2 3 4)) -> 2
(parentesis '(1 (3 (7 (8 1)) 2) 4 5)) -> 8
(parentesis '()) -> 2
(parentesis '(((a) (b) c) e ((((w))))))-> 16

32. Escribe la implementación en Scheme de un procedimiento llamado CONVIERTE-PLANA, que sirva para convertir una lista imbricada en una lista plana, es decir, que la lista resultante contenga a todos los átomos al mismo nivel de la lista principal.
Ejemplos de la llamada al procedimiento: (convierte-plana ‘(a b c)) -> (a b c)
(convierte-plana ‘((((a b c))))) -> (a b c)
(convierte-plana ‘(((c))) d ((e))(a (b (c (d (e)))))))
-> ( c d e a b c d e )

33. Escribe la implementación en Scheme de un procedimiento llamado INVIERTE, que sirva para invertir los átomos de una lista imbricada respetando los niveles de anidamiento de las sublistas.
Ejemplos de la llamada al procedimiento: (invierte '(a b c)) -> (c b a)
(invierte '((((a b c))))) -> ((((c b a))))
(invierte '(((c))) d ((e))(a (b (c (d (e)))))))
-> ((((((e) d) c) b) a) ((e)) d (((c))))

34. Escribir la definición de un procedimiento llamado DUPLICAR, que reciba una lista como entrada y que genere como salida la misma lista pero con todos los elementos de ella duplicados, sin importar en que posición de la lista aparezcan.
Ejemplo de la llamada: (duplicar '(((a) b) c)) -> (((a a) b b) c c)
(duplicar '((()) ((a) b) (()) c) -> ((()) ((a a) b b) (()) c c)

## ESTRUCTURAS DE DATOS EN LENGUAJE SCHEME

35. Cierto programa en Scheme ha sido realizado para administrar las calificaciones de los alumnos de los diferentes grupos en una escuela. El programa basa su operación en un lista de datos para cada grupo; esta lista guarda los datos de los alumnos del grupo correspondiente y tiene el siguiente formato:

**(**(matricula$_1$ (nombre$_1$) (calificaciones de los 3 parciales del alumno$_1$))
(matricula$_2$ (nombre$_2$) (calificaciones de los 3 parciales del alumno$_2$))
....
(matricula$_n$ (nombre$_n$) (calificaciones de los 3 parciales del alumno$_n$))**)**

a) Escribe un procedimiento en Scheme que utilice RECURSIVIDAD TERMINAL y que sirva para determinar cuántos alumnos tiene un grupo.

b) Escribe un procedimiento en Scheme que dada una lista con los datos de UN alumno (sublista de la lista de un grupo), genere como resultado el promedio de las calificaciones de los 3 parciales.

c) Escribe un procedimiento en Scheme que dada una lista de un grupo, genere como resultado una lista que contenga los promedios de las 3 calificaciones parciales de cada uno de los alumnos del grupo. Utiliza el procedimiento que se implementó en el inciso anterior.

d) Escribe un procedimiento que dada una lista de un grupo, la matrícula de un alumno y la calificación del examen final, genere como resultado su calificación final. La calificación final se conforma con el 60% del promedio de las calificaciones parciales y el 40% de la calificación del examen final. Utiliza el procedimiento implementado en el inciso b.

36. Una matriz de números puede ser representada en Scheme, utilizando una lista de sublistas, en donde cada sublista contiene los elementos de un renglón de la matriz. Asi por ejemplo,

la matriz: $\begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{pmatrix}$ sería representada en Scheme con la siguiente lista: ((1 2 3) (4 5 6) (7 8 9)).

Bajo esta representación, escribe la implementación de los siguientes procedimientos en Scheme:

a) Procedimiento que sirva para obtener la suma de 2 matrices.

b) Procedimiento que sirva para encontar la sumatoria de todos los elementos de una matriz.

c) Procedimiento que sirva para encontrar la sumatoria de los elementos de un renglón indicado como parámetro.

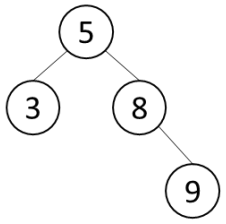
d) Procedimiento que sirva para encontrar la sumatoria de los elementos de una columna indicada como parámetro.

e) Procedimiento que sirva para encontrar la sumatoria de los elementos de la diagonal principal de una matriz cuadrada.

f) Procedimiento que sirva para verificar si una matriz es cuadrada.

g) Procedimiento que sirva para verificar si una matriz es simétrica.

37. Un ARBOL BINARIO puede ser representado en Scheme utilizando una lista de sublistas, en donde para cada nodo del árbol se tiene una lista con 3 elementos: el primero representando el campo de información del nodo, el segundo siendo la lista que representa al subarbol izquierdo, y el tercero siendo una lista que representa al subárbol derecho. Así por ejemplo, el árbol :




será representado por la lista:

(5 (3 () ()) (8 () (9 () ())))

Bajo esta representación, escribe la implementación de los siguientes procedimientos en Scheme:

a) Procedimiento que sirva para recorrer en PREORDEN a un árbol binario.

b) Procedimiento que sirva para encontrar el elemento mayor de un árbol binario de búsqueda.

c) Procedimiento que sirva para insertar un nuevo elemento en un árbol binario de búsqueda.

d) Procedimiento que sirva para encontrar la altura de un árbol binario.

e) Procedimiento que sirva para encontrar la cantidad de nodos hoja de un árbol binario.

## PROCEDIMIENTOS COMO ELEMENTOS DE PRIMER ORDEN EN LENGUAJE SCHEME

38. En geometría, un polígono es una figura plana de 3 o más lados, por lo tanto, un triángulo, un cuadrado o un hexágono son ejemplos típicos de polígonos.
En Scheme, una opción para representar a un polígono, consiste en una lista de pares en la que cada par representa a la coordenada (x, y) en la que se encuentra un vértice del polígono. El orden en que se encuentran los pares en lista es tal, que al unir los vértices (de dos en dos) con una línea, se forman los lados del polígono. Así por ejemplo, la definición de un cuadrado, cuyo centro está en el origen de un plano cartesiano, sería: (define cuadrado '((2 . 2) (2 . -2)(-2 . -2)(-2 . 2)))
(Nota: Observar que los pares son listas impropias, y que este cuadrado tiene una longitud de 4 en sus lados)

a) Escribir en Scheme, un procedimiento llamado LADOS que sirva para obtener las parejas de vértices que conforman los lados de un polígono. El procedimiento recibirá la lista de los vértices de un polígono y la transformará para generar una lista cuyos elementos son sublistas que contienen una pareja de vértices. Así por ejemplo, al llamar a este procedimiento con el cuadrado definido anteriormente, se obtendría la lista: (((2 . 2)(2 . -2)) ((2 . -2)(-2 . -2)) ((-2 . -2)(-2 . 2)) ((-2 . 2)(2 . 2))) . Observar que la lista se forma haciendo la pareja del primer vértice con el segundo, el segundo con el tercero, y así sucesivamente hasta formar la pareja del último vértice con el primero.

b) Escribir en Scheme, un procedimiento llamado DISTANCIA que sirva para obtener la distancia entre 2 vértices. El procedimiento recibirá como entrada una lista que contiene 2 pares (listas impropias) con la información de los vértices de un lado del polígono.
Recordar que la distancia entre 2 puntos es: $\sqrt{(x_2-x_1)^2 + (y_2-y_1)^2}$

c) Utilizando los procedimientos obtenidos en los incisos anteriores, y haciendo uso de los primitivos: MAP y APPLY, escribir la implementación en Scheme de un procedimiento llamado PERIMETRO, que sirva para obtener el perímetro de un polígono. Recordar que el perímetro de un polígono es la suma de las longitudes de todos sus lados.

39. Sean f1 y f2 dos funciones. Se dice que g es una función compuesta si g(x) = f2(f1(x)), es decir, el resultado de g(x) es el resultado de aplicar f2 sobre el resultado de aplicar f1 en x. Escribir en Scheme, la definición de un procedimiento llamado FUNCION-COMPUESTA, que sirva para generar como resultado un procedimiento que corresponda a la función compuesta que se forma a aplicar 2 funciones (procedimientos) que se reciban como entrada.
Ejemplificar el uso de este procedimiento con las funciones f1(x) = x/2 y f2(x) = $x^2$, generando la función g(x) = f1(f2(x)) = f1($x^2$) = $x^2$/2.

40. Escribir la definición de un procedimiento llamado FILTRO que reciba como entrada a un predicado y a una lista y que genere como salida una lista con aquellos elementos de la lista original que cumplen con el predicado.
Ejemplo de la llamada: (filtro negative? '(1 2 3 4)) -> ()
(filtro negative? '(5 -9 100 -2) -> (-9 -2)

41. Escribir la definición de un procedimiento llamado PREDICADOS-EXITOSOS, que reciba como entrada una lista de predicados y un objeto y que genere como salida una lista con aquellos predicados que se cumplen para el objeto dado. Ejemplo de la llamada: (predicados-exitosos (list integer? negative? even?) 100) -> (#<Procedure INTEGER> #<Procedure EVEN> )

42. Escribir la definición de un procedimiento llamado MAPEA-PARES, que reciba como entrada un procedimiento y una lista y que genere como salida una lista con los resultados de aplicar el procedimiento por parejas de elementos en la lista.
Ejemplo de la llamada: (mapea-pares + '(1 2 3 4)) -> (3 7)

43. Escribir la definición de un procedimiento llamado TRANSPUESTA, que reciba como entrada una matriz en el formato de lista de sublistas y que genere como salida la matriz transpuesta.

44. Escribir la definición de un procedimiento llamado MULT-MAT, que reciba como entrada dos matrices en el formato de lista de sublistas y que genere como salida la multiplicación de las 2 matrices.