EJERCICIOS DE RECURSIVIDAD EN LISP

- Hallar la profundidad máxima de un árbol, usando recursiva.

```
(defun profundidad_maxima(expresion)
(cond ((null expresion)0)
((atom expresion)1)
(t(+ 1 (apply #'max(mapcar #'profundidad_maxima expresion))))
))
```

Ejemplo: (profundidad_maxima '(1 2 (3 4 (5 6)) (7 8))) => 4

- Hallar el combinatorio de (x, y) de forma recursiva.

```
(defun combinatorio(x y)
(if(OR (= x y) (= y 1)) (if(= x y)1 x)
(+ (combinatorio (- x 1) (- y 1)) (combinatorio (- x 1) y))
))
```

Ejemplo: (combinatorio 4 3) => 4

- Transformar un número decimal a un número binario.

```
(defun decimal-binario (n)
(if (= (/ n 2) 0) 'fin
(progn (setf b (mod n 2)) (format t "~a~%" b) ))
(if(= (mod n 2) 0) (decimal-binario(/ n 2))
(decimal-binario(/ (- n 1) 2)))
)
```

Ejemplo: (decimal-binario 5)=> 1 0 1 FIN

- Escribir una función que tome una lista y un número natural “n” y retorne la lista original sin los últimos “n” elementos.

  Método tamano:

  ```
  (defun tamano(lista)
  (if(endp lista)
  0
  (if(atom(car lista))
  (+ 1 (tamano(cdr lista)))
  (+ (tamano(car lista)) (tamano(cdr lista)))
  )))
  ```

  Método cortarlista:

  ```
  (defun cortarlista (L n)
  (cond    ((<= (tamano L) n) nil)
  (t (cons (car L) (cortarlista (cdr L) n)))
  ))
  ```

  Ejemplo: (cortarLista ‘(1 2 3 4 5 6) 2) => (1 2 3 4)

- Mostrar todos los múltiplos de 3 de una lista.

  ```
  (defun multiplos3 (L)
  (if(null L) 'fin
  (if(= (mod (car L) 3) 0) (progn (setq b (car L)) (format t "~a~%"b )(multiplos3 (cdr L)))
  (multiplos3 (cdr L))
  )))
  ```

  Ejemplo: (multiplos3 ‘(1 2 3 4 5 6)) => 3 6 FIN