Ejercicios obligatorios:

1. Definir una variable que contenga una lista con los nombres de cuatro personas. Indica los comandos necesarios para modificar esa variable con el objeto de:

```
(setq lista '(Pepe jose juan luis))
```

a. Añadir dos nombres al principio.

```
(setq lista (append lista '(carlos teorodo)))
```

b. Añadir un nombre al final.

```
(setq lista (reverse (append '(david) (reverse lista))))
```

c. Sustituir el penúltimo nombre por el segundo.

```
(setq lista (cons (first lista) (cons (second (reverse lista)) (nthcdr 2 lista))))
```

d. Eliminar los dos últimos nombres.

```
(setq lista (reverse (nthcdr 2 (reverse lista))))
```

e. Que contenga sólo el primer y el tercer nombre.

```
(setq lista (cons (first lista) (third lista)))
```

2. Poner dos ejemplos para cada uno de los predicados que se indican a continuación, uno que devuelva verdadero y otro que devuelva falso. Si es posible, añade un tercer ejemplo que devuelva error:

a. <
   1. (< 1 2) T
   2. (< 2 1) NIL

b. <=
   1. (<= 1 2) T
   2. (<= 2 1) NIL

c. zerop
   1. (zerop 0) T
   2. (zerop 1) NIL

d. numberp
   1. (numberp 3) T
   2. (numberp (cons 4 2)) NIL

e. symbolp
   1. (symbolp ‘letra) T
   2. (symbolp 1) NIL

f. atom
    1. (atom 23) T
    2. (atom '(1 2)) NIL
g. constantp
    1. (constatnp 1) T
    2. (constantp 'letra) NIL
h. listp
    1. (listp '(1 2)) T
    2. (listp 1) NIL

3. Escribe las funciones: x/2, x/3, ... x/10 que calculen la mitad, un tercio, etc. de un número. Haz uso siempre que puedas de las funciones previamente desarrolladas.

```
(defun x/2 (x)
        (/ x 2)
)
(defun x/3 (x)
        (/ x 3)
)
(defun x/4 (x)
        (/ (/ x 2) 2)
)
(defun x/5 (x)
        (/ x 5)
)
(defun x/6 (x)
        (/ (/ x 2) 3)
)
(defun x/7 (x)
        (/ x 7)
)
(defun x/8 (x)
        (x/2 (x/4 x))
)
(defun x/9 (x)
        (x/3 (x/3 x))
)
(defun x/10 (x)
        (x/2 (x/5 x))
)
```

4. Considera la siguiente definición de función:

```
(defun quien-lo-sabe (lst1 lst2)
        (cond ((= (nth 2 lst1) (nth 3 (reverse lst2)))
                (- (length lst1) (* 2 (length lst2))))
                ((<= (length lst1) (length lst2)) (length lst2))
                (t (length lst1))))
```

a. Pon un ejemplo para que devuelva -5.

Para que la función quien-lo-sabe devuelva un 5 hay que pasarle una lista de tamaño 5 y otra menor o igual.

b. ¿Cómo harías que esta función falle? (devuelva un ERROR) Ten cuidado en explicar porqué ocurre.

   Para que falle, hay que pasarle una de las listas vacías, ya que intentará acceder a un campo inexistente.

Ejercicios Obtativos:

5. Escribe una función denominada CIRCULA-DIRx2 que pueda circular una lista en ambas direcciones 2 posiciones. Es decir, algo como:

> (circula-dir '(1 2 3 4 5) 'left) (4 5 1 2 3)
> (circula-dir '(1 2 3 4 5) 'right) (3 4 5 1 2)

La función debe incorporar tratamiento de errores.

```
(defun circula-dir (lista dir)
  (cond ((eq dir 'left)
           (setq liste  (append (last lista) (cddr (reverse lista))))
           (setq lista (append (list (second (reverse lista))) liste))
         )
         ((eq dir 'right)
           (setq liste (append (list (first lista)) (list (second lista))))
           (setq lista (append (cddr lista) liste))
         )
         (t
           (format t "La direcciÛn dada es errÛnea.~%")
         )
  )
)
```