1. **NUMEROS**

(Defun esnumero ()  
          (let (lista)  
             (pprint "Ingrese lista")  
            (setq lista (Read))  
            (mapcar 'numberp lista)))

**4.**

(defun esnumero ()

(pprint "Ingrese lista: ")

(setq lista (read))

(mapcar (lambda (x) (if (numberp x) "SI" "NO")) lista))

**5. POTENCIA CORREGIR**

(defun ejer5 ()

(let((numero) (lista))

(pprint "ingrese numero")

(Setq numero (read))

(pprint "Ingrese lista")

(setq lista (read))

(mapcar (lambda (x) (exp lista x) numero))))

**6.**

**(defun ingreso()**

(let (lista)

(pprint "Ingrese lista")

(setq lista (read))

(if(consp lista)

(mapcar(lambda (x) (cond

((> x 0) (list x '>0))

((< x 0) (list x '<0))

((= x 0) (list x '=0))

)) lista))))

**7.**

**((defun ejer7 (lista)**

(mapcar(lambda (x) (if(listp x)

(length x)0)) lista))

**8.REVISAR (foto)**

(defun ejer8 (lista1 lista2)

(if (consp lista1)

(mapcar (lambda (x) (if

(not (numberp x)) (list x (car(last lista2))))

) lista1)))

**9. SUMO AMBOS**

**(defun sumo-ambos ()**

(let ((lista1)(lista2))

(pprint "Ingrese lista 1")

(setq lista1 (Read))

(pprint "Ingrese lista 2")

(setq lista2 (read))

(mapcar '+ lista1 lista2)))

10.

(mapcar(lambda(x y) (if(and(integerp x)(integerp y))

(list x y 'T)

(list x y 'NIL)))

lista1 lista2)