**1.**

(defun suma (a)

(cond ((endp a) 0)

((numberp (car a))(+ (Car a)(suma (Cdr a))))

(T (suma (cdr a)))))

**2.**

(defun sumaAtom (a)

(cond ((endp a) 0)

((and (numberp (car a))(oddp (car a)))(+ 1(sumaAtom (Cdr a))))

(T (sumaAtom (cdr a)))))

**3.**

**(defun ingreso ()**

(let (lista)

(pprint "Ingrese lista heterogenea")

(setq lista (read))

(/ (**suma** lista)(**contador** lista))

))

**(defun suma (a)**

(cond

((endp a)0)

((numberp (Car a))(+ (car a)(suma (cdr a))))

(T (suma (cdr a)))))

**(defun contador (b)**

(cond

((endp b) 0)

((numberp (Car b))(+ 1 (contador(cdr b))))

(T (contador(cdr b)))))

**4.**

**(defun ejer4 (lista)**

(cond

((endp lista) NIL)

(T (cons (listp (car lista))(ejer4 (Cdr lista))))))

**5.**

**(defun ejer5 (lista)**

(cond

((endp lista) 'T)

((numberp (car lista))(ejer5 (cdr lista)))))

**6.**

(**defun aprobado (notas)**

(cond

((endp notas) 'T)

((and(numberp (car notas))(>= (car notas)6))

(aprobado (cdr notas)))

))

**7. LISTAS Y LONGITUD**

(defun ejer7 (a)

(cond

((endp a) nil)

((listp (car a)) (cons (list (car a) (length (car a))) (ejer7 (cdr a))))

(t (ejer7 (cdr a)))))

**8.ORDENADO**

**(defun ordenado (lista)**

**(cond**

**((or (endp lista) (endp (cdr lista))) 'T) ;** *Caso base: cuando la lista está vacía o tiene un solo elemento, está ordenada*

**((<= (car lista) (cadr lista)) (ordenado (cdr lista))) ;** *Llamada recursiva si los elementos están ordenados correctamente*

**(t NIL)))** *; Devuelve NIL si los elementos no están ordenado*

**9. DIFERENCIA DE LISTAS**

(defun ejer9 (a b)

(cond

((or(Endp a) (endp b)) '())

(t (cons (- (Car a)(car b)) (ejer9 (cdr a)(cdr b))))))

**10. ENTEROS (VER FOTO)**

**(defun noenteros (L)**

(cond

((endp L) nil)

((and (numberp (car L)) (not (integerp (Car L)))) (cons (Car L) (noenteros (cdr L))))

(T (noenteros (Cdr L)))))

**(defun enteros (L)**

(cond

((endp L) nil)

(( and (numberp (car L)) (integerp (car L))) (cons (car L) (enteros (cdr L))))

(T (enteros (cdr L)))))

**(defun ingreso ()**

(let (L1)

(pprint "Ingrese lista")

(setq L1 (read))

(if (consp L1) (armo L1) (pprint "MAL INGRESO"))))

**(defun armo (L)**

(list (enteros L) (noenteros L)))

**11. COPIE LA PIBITO**

(defun funcion (lista)

(cond

((Endp lista) nil)

((member (car lista)(cdr lista))(funcion (cdr lista)))

(T (cons (car lista) (funcion (cdr lista))))))

**12. INCOMPLETO**

**A.**

**(defun ingreso ()**

(let (lista)

(pprint "Ingrese lista")

(setq lista (read))

(if (consp lista)

(cuandoHabloMas lista)

(pprint "VALOR INGRESADO INVALIDO"))))

**(defun cuandoHabloMas (lista)**

(cond

((> (suma (car lista)) (suma (cadr lista))) (pprint "Se hablo mas en el horario NORMAL"))

((< (suma (car lista)) (suma (cadr lista))) (pprint "Se hablo mas en el horario REDUCIDO"))

(T (pprint "IGUALES"))))

**(defun suma (L1)**

(cond

((endp L1) '0)

((+ (car L1) (suma (cdr L1))))))

12 B.

**(defun maximo (lista)**

(cond

((endp lista) nil)

((null (cdr lista)) (car lista))

(T (max (car lista) (max (maximo (cdr lista)))))))

**(defun ingreso ()**

(let (lista)

(pprint "Ingrese lista")

(setq lista (read))

(if (consp lista)

(LLamadaMasLarga lista)

(pprint "VALOR INGRESADO INVALIDO"))))

**(defun LLamadaMasLarga (lista)**

(cond

((> (maximo (car lista)) (maximo (cadr lista)))

(pprint "LA LLAMADA MAS LARGA ES DEL HORARIO NORMAL"))

((< (maximo (car lista)) (maximo (cadr lista)))

(pprint "LA LLAMADA MAS LARGA ES DEL HORARIO REDUCIDO"))

((= (maximo (car lista)) (maximo (cadr lista)))

(pprint "IGUALES"))))

12.C

**(defun LLamadaMasCorta (lista)**

(cond

((< (minimo (car lista)) (minimo (cadr lista)))

(pprint "LA LLAMADA MAS CORTA ES DEL HORARIO NORMAL"))

((> (minimo (car lista)) (minimo (cadr lista)))

(pprint "LA LLAMADA MAS CORTA ES DEL HORARIO REDUCIDO"))

((= (minimo (car lista)) (minimo (cadr lista)))

(pprint "IGUALES"))))

**(defun minimo (lista)**

(cond

((endp lista) nil)

((null (cdr lista)) (car lista))

(T (min (car lista) (min (minimo (cdr lista)))))))

**(defun ingreso ()**

(let (lista)

(pprint "Ingrese lista")

(setq lista (read))

(if (consp lista)

(LLamadaMasCorta lista)

(pprint "VALOR INGRESADO INVALIDO"))))