Tipo Abstracto de Datos Lista

T.A.D. LISTA

ESTRUCTURAS DE DATOS
y ALGORITMOS
LCC TUPW

LISTAS

Las **listas** son estructuras de datos flexibles porque pueden crecer y contraerse, y los elementos accedidos, insertados y eliminados en cualquier posición de la lista.

T.A.D. LISTA – **Especificación (1)**

Lista: Secuencia de 0 o mas elementos de un tipo determinado, que puede crecer y contraerse sin restricción.

L = (a1, a2, ..., an) , n>=0

n : longitud de la lista

Si n=0, lista vacía : L=( )

Si n>0, entonces :

ai es el *i-esimo elemento*
a1 es el *primer elemento*
an es el *último elemento*
ai *precede a* ai+1, para 1<=i<n
ai *sucede a* ai-1, para 1<i<=n

Existe una relación de orden dada por la posición del elemento en la lista

T.A.D. LISTA – Orden

En matemáticas, un **orden total**, **orden lineal**, **orden simple**, o simplemente **orden** en un conjunto *X* es una relación binaria sobre *X* que es antisimétrica, transitiva, y total; esto es, si se denota una tal relación por ≤, lo siguiente vale para cualesquiera *a*, *b*, y *c* en *X*:

- Si $a \leq b$ y $b \leq a$, entonces $a = b$ (antisimetría).
- Si $a \leq b$ y $b \leq c$, entonces $a \leq c$ (transitividad).
- $a \leq b$ o $b \leq a$ (totalidad o *completitud : todo los pares de elementos son comparables bajo la relación*).

Ejemplos

- Las letras del alfabeto con el orden alfabético usual: "*A*" < "*B*" < "*C* " < "X"
- Los *naturales*, *enteros*, *racionales* y los *reales*, con el orden usual de las relaciones < o >, son conjuntos bien ordenados.

T.A.D. LISTA – **Especificación(2)**

L = (a1, a2, ...,, ai, ..., an)

1 2 i n

¿ p ?

Ingresar el elemento X, en la lista L, en la posición p

1<=p<=n+1

L = (a1, a2, ..., X, ai, ..., an)

1 2 i i+1 n+1

si p = i

L = (X, a1, a2, ..., ai, ..., an)

1 2 i+1 n+1

si p = 1

L = (a1, a2, ..., ai, ..., an, X)

1 2 i n+1

si p = n+1

T.A.D. LISTA – **Especificación(3)**

L = (a1, a2, ...,, ai, ..., an)

1 2 i n

¿ p ?

Eliminar de la lista L, el elemento que se encuentra en la posición p

1<=p<=n

L = (a1, a2, ..., ai+1, ..., an) X=ai si p = i

1 2 i n-1

L = (a2, ..., ai, ..., an) X=a1

1 i-1 n-1

si p = 1

L = (a1, a2, ..., an-1) X=an

1 2 n-1

si p = n

T.A.D. LISTA – **Especificación (4)**

Operaciones Abstractas

Sean **L**: Lista; **X**: elemento y **p,p1**: posiciones

NOMBRE	*ENCABEZADO*	*FUNCION*	*ENTRADA*	*SALIDA*
Insertar	Insertar(X,L,p)	Ingresa el elemento X, en la lista L, en la posición p	L , X y p	$L=(a_1,\ldots,a_{p-1},X,a_{p+1},\ldots,a_n)$ o $L=(a_1,\ldots,a_n,X)$ o $L=(X)$, si $1<=p<=n+1$; Error en caso contrario
Suprimir	Suprimir(L,p,X)	Elimina de la lista L, el elemento que se encuentra en la posición p	L y p	$L=(a_1,\ldots,a_{p-1},a_{p+1},\ldots)$ y $X=a_p$, si $1<=p<=n$; Error en caso contrario
Recuperar	Recuperar(L,p,X)	Recupera de la lista L, el elemento que se encuentra en la posición p	L y p	$X=a_p$, si $L=(a_1,\ldots,a_p,\ldots,a_n)$ y $1<=p<=n$; Error en caso contrario
Buscar	Buscar(X,L,p)	Localiza en la lista L, el elemento X	L y X	$p=i$, si $L=(a_1,\ldots,a_i=X,\ldots,a_n)$; Error en caso contrario

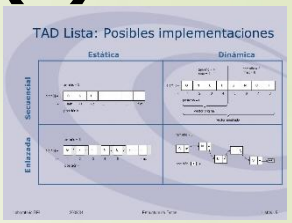
T.A.D. LISTA – **Especificación (5)**

Operaciones Abstractas

Sean **L**: Lista; **X**: elemento y **p**: posicion

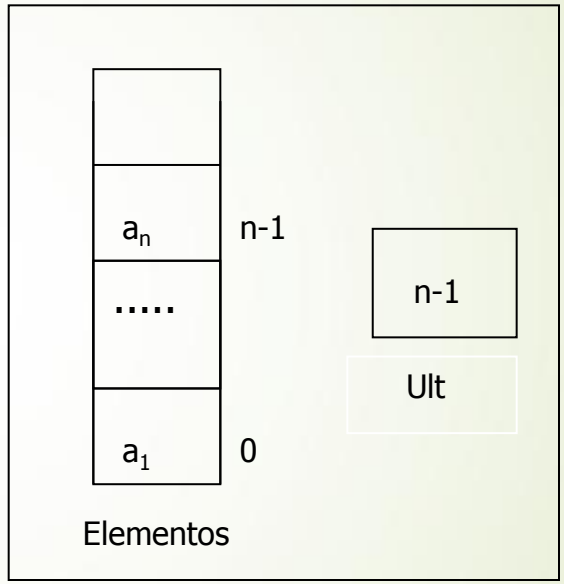
NOMBRE	*ENCABEZADO*	*FUNCION*	*ENTRADA*	*SALIDA*
Primer_elemento	Primer_elemento (L,X)	Reporta el primer elemento de la lista L	L	$X=a_{1,}$ si n>0 ; Error en caso contrario
Ultimo_elemento	Ultimo_elemento (L,X)	Reporta el último elemento de la lista L	L	$X=a_{n,}$ si n>0 ; Error en caso contrario
Siguiente	Siguiente(L,p,p_1)	Recupera de la lista L la posición (dirección) siguiente a p	L y p	p_1=direccion(p+1), si 1< =p<n; Error en caso contrario
Anterior	Anterior(L,p,p_1)	Recupera de la lista L la posición (dirección) anterior a p	L y p	p_1=dirección(p-1), si 1< p<=n; Error en caso contrario
Recorrer	Recorrer(L)	Procesa todos los elementos de la lista L	L	Está sujeta al proceso que se realice sobre los elementos de L
Crear	Crear(L)	Inicializa L	L	L=( )
Vacía	Vacía(L)	Evalúa si L tiene elementos	L	Verdadero si L No tiene elementos, Falso en caso contrario.

T.A.D. LISTA – **Representación(1)**



Representación secuencial

L = (a1, a2, ..., an)



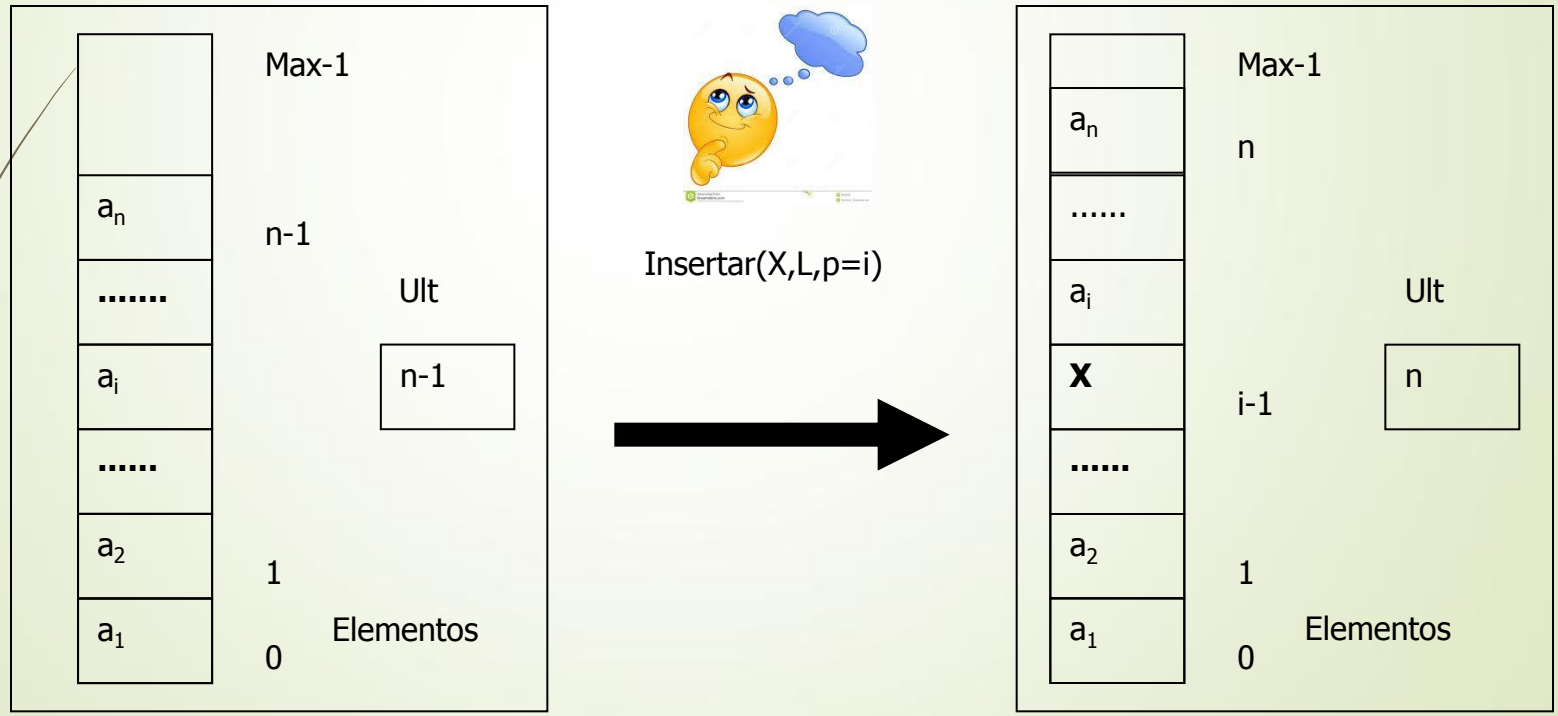
relación posición "lógica" ubicación "física"

operación insertar, debe prever el desplazamiento (shifteo) de los elementos almacenados, *overflow*.

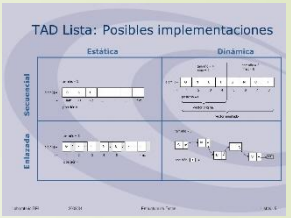
T.A.D. LISTA –
Construcción de operaciones abstractas(1)

Ver Video Insertar-representacion secuencial TAD Lista.mp4

REPRESENTACIÓN SECUENCIAL

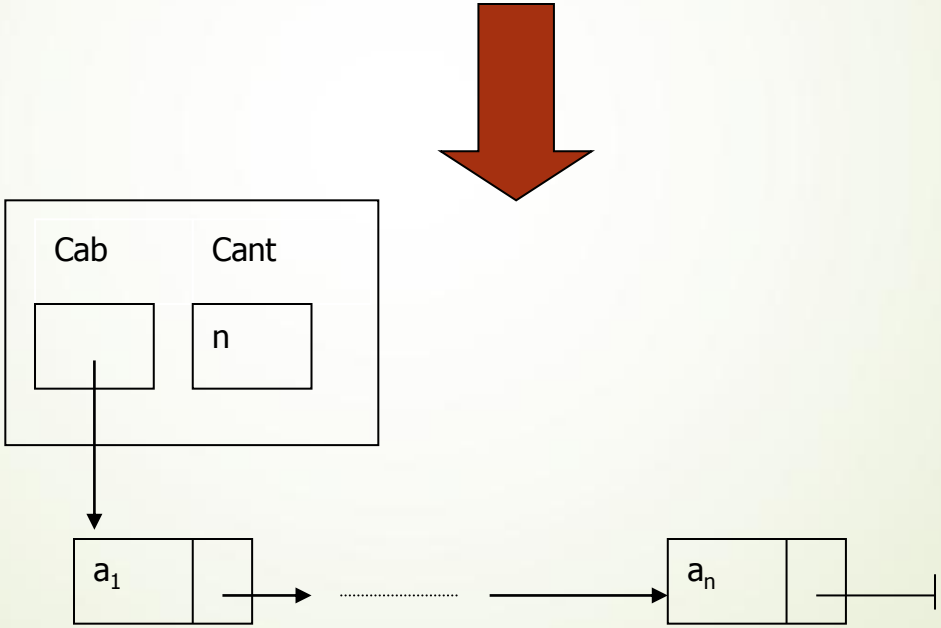


T.A.D. LISTA – Representación(2)



Representación encadenada:

L = (a1, a2, ..., an)

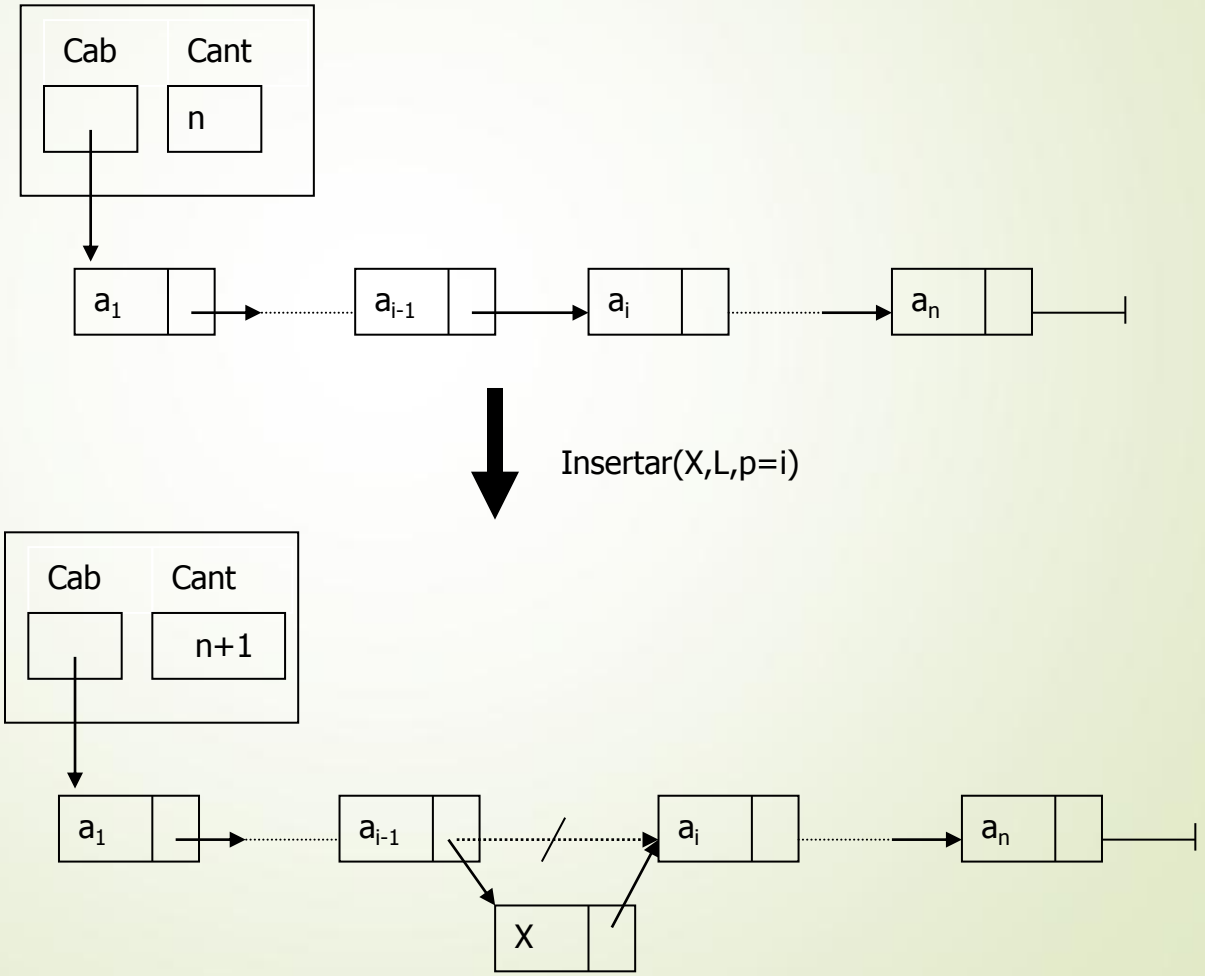


relación posición "lógica" no coincide con ubicación "física".

T.A.D. LISTA

Construcción de operaciones abstractas(2)

REPRESENTACIÓN ENCADENADA



T.A.D. LISTA

Complete la siguiente tabla con el tiempo de ejecución de cada operación abstracta – parámetro para evaluar la eficiencia-, en cada una de las representaciones trabajadas

Representaciones / Operaciones	SECUENCIAL	ENCADENADA
Insertar(X,L,p)		
Suprimir(X,L,p)		
Recuperar(L,p,X)		
Buscar(X,L,p)		
Primer_elemento (L,X)		
Ultimo_elemento (L,X)		
Siguiente(L,p,p_1)		
Anterior(L,p,p_1)		
Recorrer(L)		

T.A.D. LISTA – Representación(3)

REPRESENTACIÓN

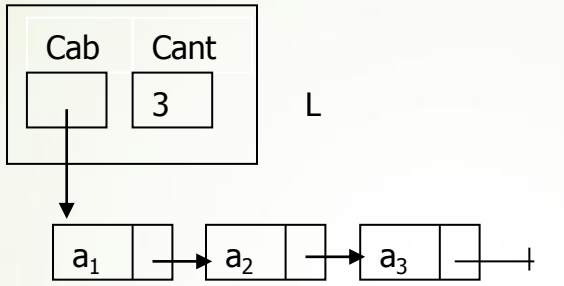
- *Representación Secuencial*
- *Representación Encadenada*
  - Variables dinámicas
  - ***Cursores***: *enteros que indican posiciones en un arreglo o en otros tipos de datos*

T.A.D. LISTA – Representación(4)

- **Cursores:** valores enteros que indican posiciones en un arreglo o en un archivo. Los cursores pueden ser usados, al igual que las variables dinámicas, para construir objetos de datos con representación vinculada.
- Para una lista con representación vinculada, cada celda es un registro con dos campos: **Elemento** y **Enlace** al siguiente elemento. Cuando trabajamos con **cursores**, **Enlace** es un valor entero.

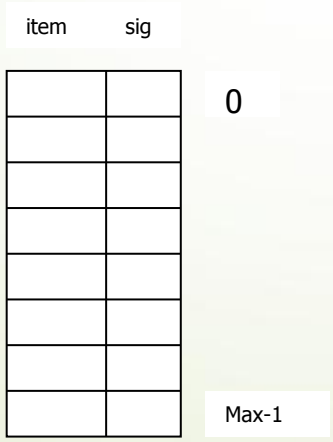
T.A.D. LISTA – **Representación(5)**

REPRESENTACIÓN ENCADENADA BASADA EN CURSOR



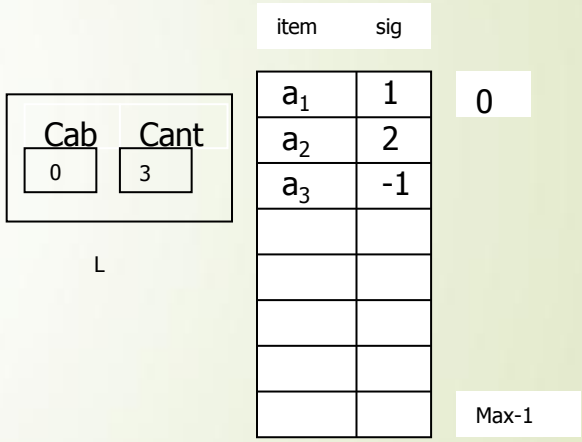
Lista L con representación enlazada

+



espacio

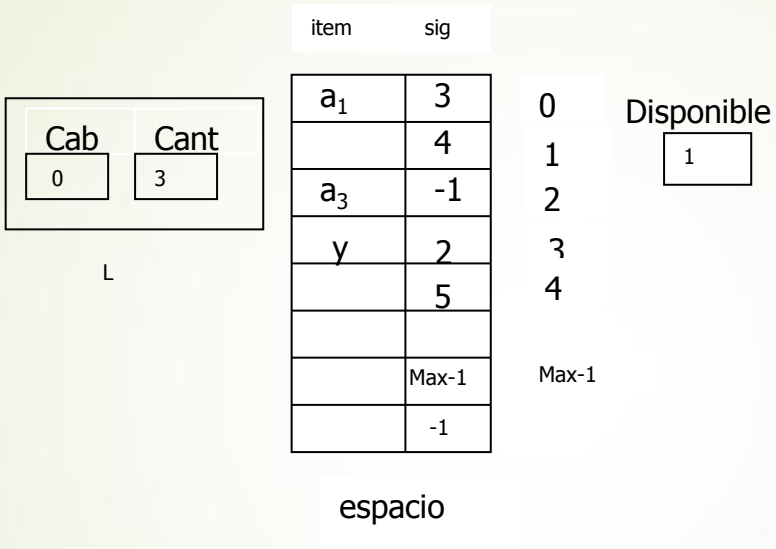
Espacio para cursores, definido sobre arreglo.



espacio

Lista L representada con cursores, sobre arreglo

T.A.D. LISTA – **Representación(6)**

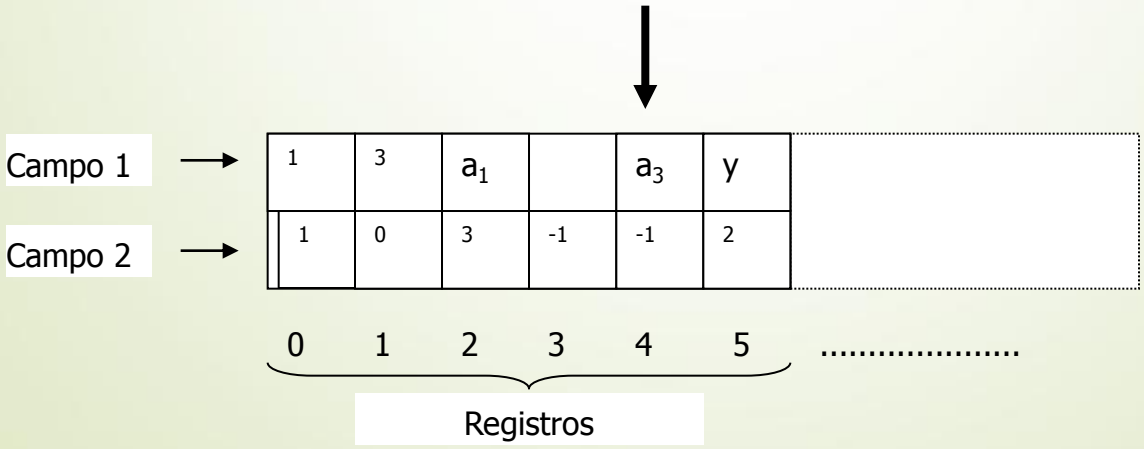


Registro 0: Cumple la función del campo ***Disponible***, contenido este en el *Campo 2*, pero en esta oportunidad se adiciona la cantidad de registros libres intercalados, contenida en *Campo 1*.

Registro 1: Este registro contiene ***cab*** en *Campo 2*, y ***cant*** en *Campo 1*, ambos correspondientes a la lista L.

Registros 2, 4 y 5: representan a las tres celdas de la lista **L=(a1, y, a3)**. En ellas *Campo 1,* es el recíproco de ***item***, mientras que *Campo* 2 lo es de ***sig***.

Registro 3: Este es el único registro que pertenece, en esta instancia, a la pila de espacios libres.



Archivo F conteniendo la lista L y la pila de espacios libres.

T.A.D. LISTA ORDENADA POR CONTENIDO

Lista ordenada por contenido: Lista en la que debe conservarse un orden lineal entre los valores de alguno de los atributos que conforman sus elementos.

Que modificación requiere la **Especificación** de Lista?

Insertar(X,L)

Entrada : L=(a1,….,an) n>=0 y X

Función : Insertar el elemento X manteniendo el orden lineal entre los valores de los elementos de L, esto es:

ai<=ai+1, para 1<=i<n.

Salida: L=(a1,..,ai,X,ai+1,....,an) si ai<=X<ai+1, 1<=i<n

L=(X,a1,....,an) si X<a1

L=(a1,....,an,X) si an<X

¿ Suprimir ?

T.A.D. LISTA – Aplicación

Diseñe el algoritmo que, **apoyado en el TAD Lista**, elimine elementos con valores repetidos de una lista de números naturales.

Entrada		Salida
L = (10, 5, 7, 5, 2, 10)	→	L = (10, 5, 7, 2)

Nota: No usar estructuras de datos adicionales, solo la lista de entrada