



Análisis de Algoritmos y Estructura de Datos

Semana 6

Ejercicios: listas enlazadas simples y especiales

Prof. Violeta Chang C

Semestre 2 – 2023



TDA lista enlazada

- **Contenidos:**

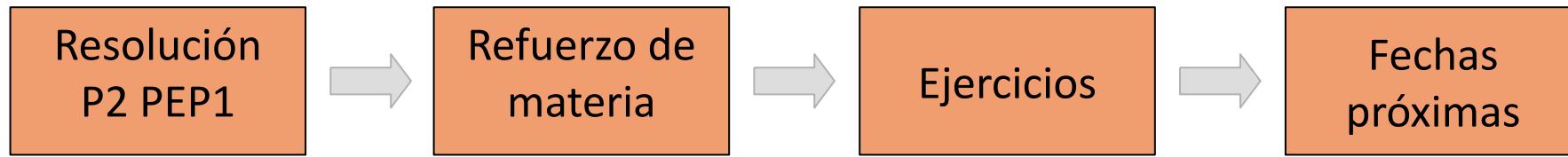
- Operaciones con listas enlazadas simples y especiales

- **Objetivos:**

- Comprender funcionamiento de operaciones con listas enlazadas simples y especiales
 - Determinar complejidad de operaciones con LE simples y especiales y aplicarlas para resolver problemas específicos



TDA lista enlazada





Resolución P2 de PEP1



Resolución PEP1

Pregunta 2 (24 puntos): Se tiene un arreglo ordenado de n enteros positivos y negativos y se necesita determinar si existe algún valor x tal que tanto x como -x están en el arreglo. Escribir un algoritmo que resuelva el problema, sin ocupar ni búsqueda lineal ni búsqueda binaria, ni arreglos auxiliares. Calcular y justificar la complejidad del algoritmo propuesto

I. El algoritmo propuesto respeta la restricción de NO uso de búsqueda lineal ni binaria	10	Respeto restricción de NO uso de búsqueda lineal ni binaria y apunta a resolver el problema	10
		Usa búsqueda lineal o binaria y/o no apunta a resolver el problema	0
II. El algoritmo resuelve correctamente el problema planteado.	8	El algoritmo resuelve correctamente el problema planteado	8
		El algoritmo tiene errores leves, o bien resuelve al menos el 75% del problema sin errores	6
		El algoritmo tiene errores graves, o bien resuelve al menos el 50% del problema sin errores	4
		No resuelve el problema	0
III. El algoritmo está escrito en seudocódigo ordenado y consistente, en el formato establecido, identificando entradas y salidas.	3	Seudocódigo ordenado y consistente, en el formato establecido, identificando entradas y salidas	3
		Seudocódigo ordenado y consistente, en el formato establecido, pero con algunos errores	2
		Seudocódigo no ordenado ni consistente, o sin el formato establecido, o no cumple el ítem II	0
IV. Calcula correctamente O() y justifica con argumentos correctos	3	Calcula correctamente O() de algoritmo propuesto y justifica con argumentos correctos	3
		Calcula correctamente O() de algoritmo propuesto pero no justifica con argumentos correctos, o argumentos son incorrectos	2



Revisión de materia



Especificación de TDA lista enlazada

- **Estructura de datos:**

- Una lista enlazada (LE) es una colección lineal de largo indeterminado con componentes homogéneos.
- Homogéneo: Todos los componentes son del mismo tipo
- Lineal: Componentes están ordenados en una línea por eso se conocen como listas enlazadas lineales

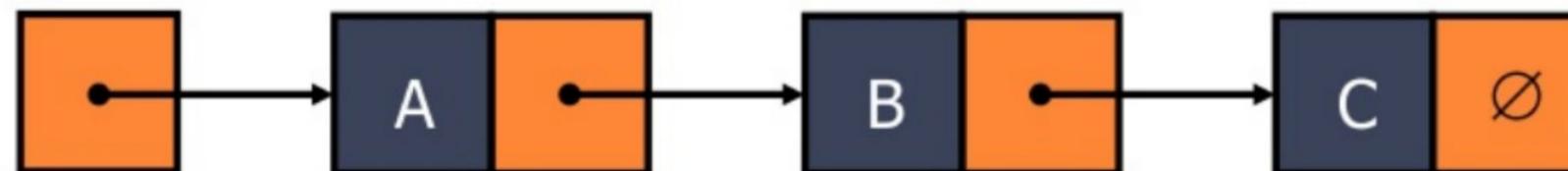
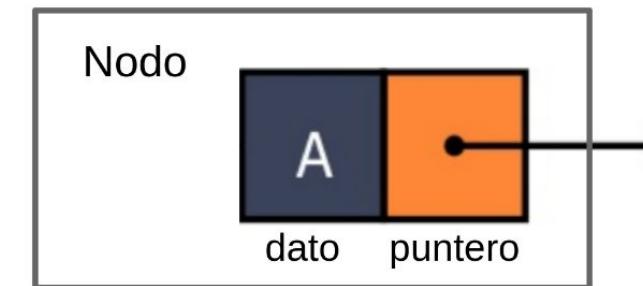




Especificación de TDA lista enlazada

- **Estructura de datos:**

- Una lista enlazada (LE) es una secuencia de nodos conectados
- A una lista con 0 nodos se le conoce como **lista vacía**
- Cada nodo contiene:
 - Una parte de datos (cualquier tipo)
 - Un puntero al siguiente nodo de la lista
- **Cabeza**: puntero al primer nodo
- El último nodo apunta a **nulo**





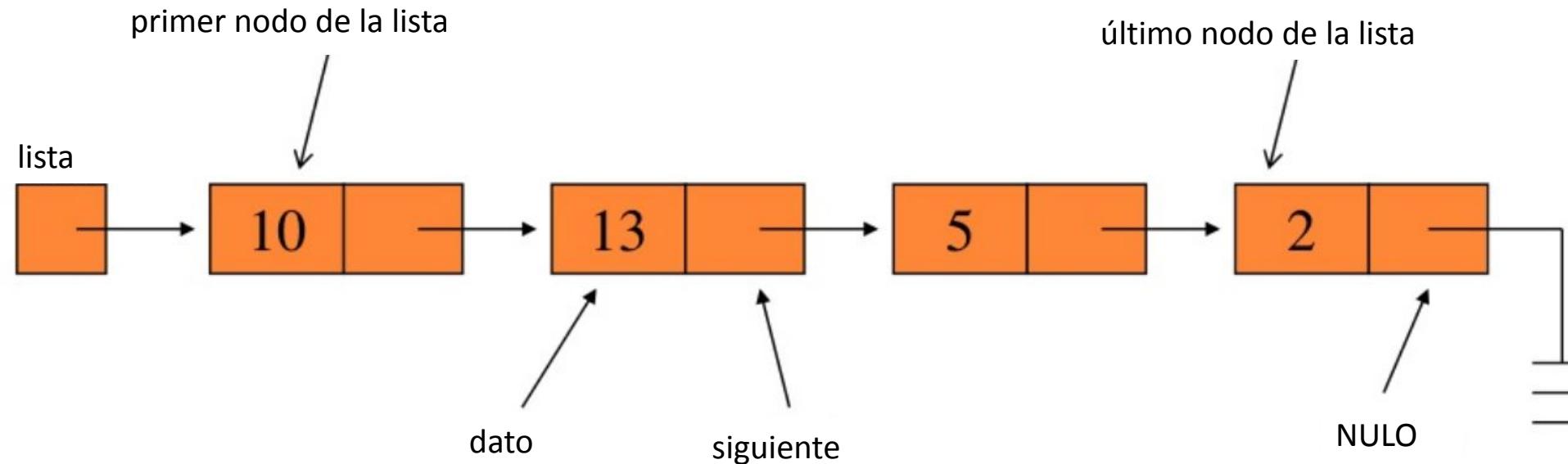
Especificación de TDA lista enlazada

- **Operaciones:**

- **esListaVacía (L)** : determina si lista L está vacía o no
- **insertarNodo (L, dato)** : inserta un nodo con dato en lista L
- **eliminarNodo (L, dato)** : elimina nodo con dato de lista L
- **buscarDato (L, dato)** : busca dato en lista L
- **recorrerLista (L)** : muestra contenido de cada nodo de lista L



Especificación de TDA lista enlazada



Lista enlazada de enteros



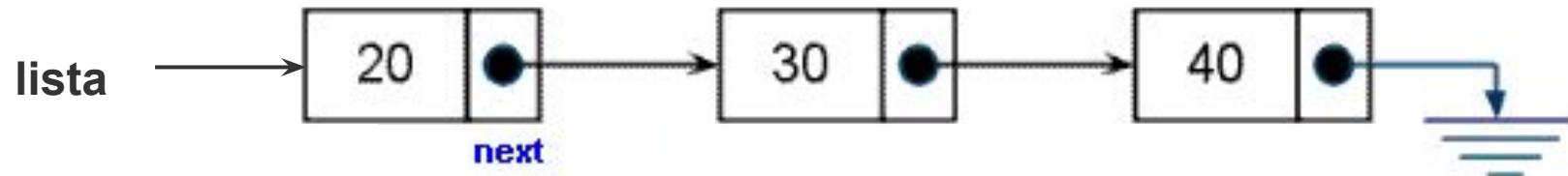
Operaciones con listas enlazadas simples

- **recorrerLista(lista)** → $O(n)$
- **insertarNodoInicio(lista,valor)** → $O(1)$
- **insertarNodoFinal(lista,valor)** → $O(n)$
- **insertarNodoDespues(lista,valor,datoAnterior)** → $O(n)$
- **eliminarNodoInicio(lista)** → $O(1)$
- **eliminarNodoFinal(lista)** → $O(n)$
- **eliminarNodoDatos(lista,dato)** → $O(n)$

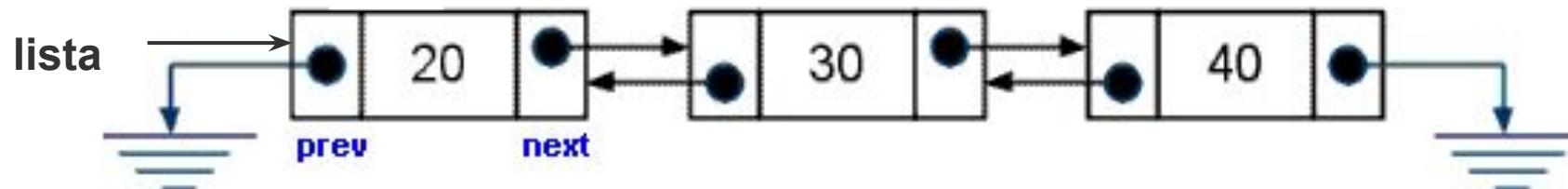


Tipos de listas enlazadas

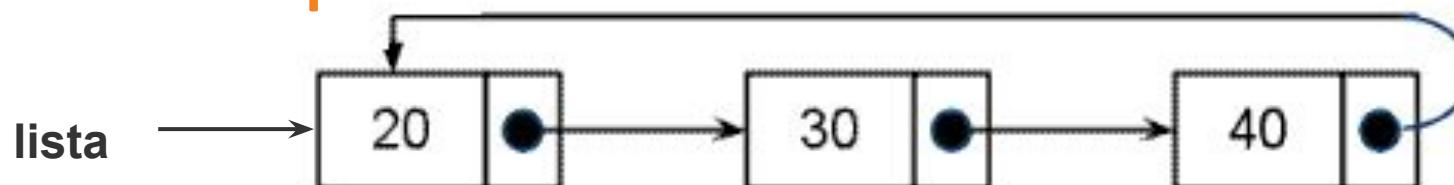
- **Lista enlazada simple**



- **Lista doblemente enlazada**



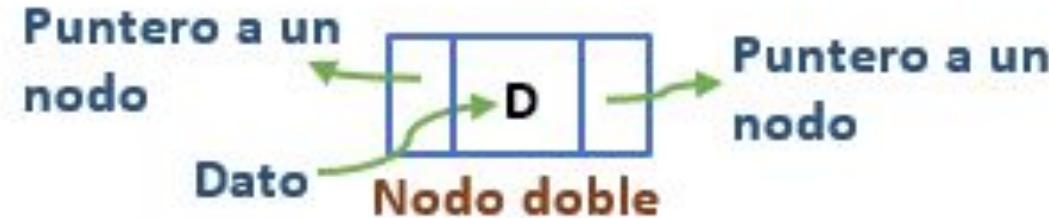
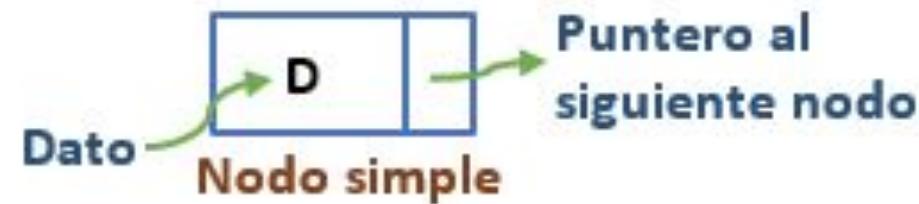
- **Lista enlazada simple circular**





Listas doblemente enlazadas

- Estructura de nodo doble:





Ejercicios



Actividad en conjunto - 10+5 minutos

- Escribir un algoritmo para que, dadas dos listas enlazadas simples, devuelva **verdadero** si ambas listas son iguales, y **falso** en caso contrario. Cuál es el orden de complejidad del algoritmo propuesto?



Falso

Después de comparar lista1 y lista2

Version 1

Comparar listas ($L \in L_1, L \in L_2$): booleano

$n_1 \leftarrow \text{obtener largo } L \in L_1$

$n_2 \leftarrow \text{obtener largo } L \in L_2$

Si: $n_1 < > n_2$ entonces

DESVOLVER FALSE

$\text{aux1} \leftarrow L_1$

$\text{aux2} \leftarrow L_2$

Mientras $\text{aux1} \neq \text{NULL}$ hacer

Si: $\text{aux1} \rightarrow \text{dato} \neq \text{aux2} \rightarrow \text{dato}$ entonces

DESVOLVER FALSE

$\text{aux1} \leftarrow \text{aux1} \rightarrow \text{puntero}$

$\text{aux2} \leftarrow \text{aux2} \rightarrow \text{puntero}$

DESVOLVER VERDADERO

}

$O(1)$

}

$O(n)$

}

Version 2

Comparar ($L \in L_1, L \in L_2$): booleano

$n_1 \leftarrow \text{obtener largo } L \in L_1$

$n_2 \leftarrow \text{obtener largo } L \in L_2$

Si: $n_1 < > n_2$ entonces

DESVOLVER FALSE

$\text{aux1} \leftarrow L_1$

$\text{aux2} \leftarrow L_2$

Mientras $\text{aux1} \rightarrow \text{dato} = \text{aux2 dato}$ hacer

Si: $\text{aux1} \rightarrow \text{puntero} = \text{NULL}$ entonces

DESVOLVER VERDADERO ... llegaras al último nodo

$\text{aux1} \leftarrow \text{aux1} \rightarrow \text{puntero}$

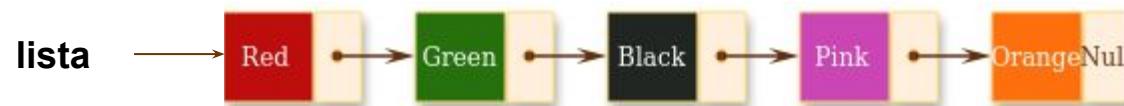
$\text{aux2} \leftarrow \text{aux2} \rightarrow \text{puntero}$

DESVOLVER FALSE



Actividad en parejas - 10+5 minutos

- Escribir un algoritmo para eliminar todos los elementos de una lista enlazada simple. Después de **eliminar** cada elemento, lo que queda debe seguir siendo una lista enlazada. Cuál es el orden de complejidad del algoritmo propuesto?



Lista original

lista →

Lista final

Ej. MinorLista (LE lista)

aux <- lista

$O(1)$

Mientras aux <> NULO hacer

nodo <- aux

$\{ O(m)$

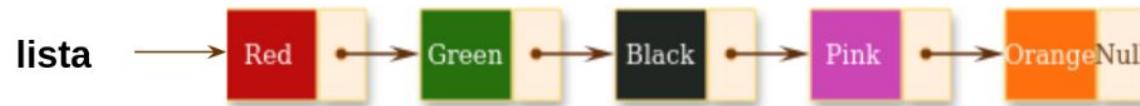
aux <- aux -> puntero

liberar (nodo)

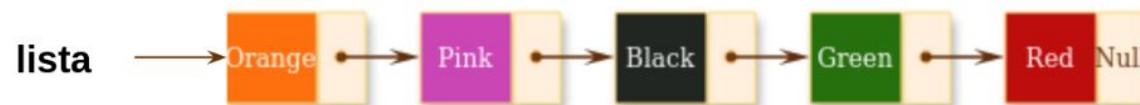


Actividad en parejas - 10+5 minutos

- Escribir un algoritmo para que, dada una lista enlazada simple, devuelva la misma lista pero con los nodos en **orden inverso** al original. Cuál es el orden de complejidad del algoritmo propuesto?



Orden original



Orden inverso

Invertir LE (LE lista): LE

aux \leftarrow lista
inversa \leftarrow crear lista vacia() } O(n)
mientras aux \neq NULO hacer
 insertar inicio (inversa, aux \rightarrow dato) } O(n)
 aux \leftarrow aux \rightarrow puntero
DE VOLVER inversa } O(1)



Actividad en parejas - 10+5 minutos

- Escribir un algoritmo para que dadas dos listas doblemente enlazadas, devuelva una lista nueva formada por los elementos de la primera lista, seguidos por los elementos de la segunda lista. ¿Cuál es el orden de complejidad del algoritmo propuesto?



Listas originales



Después de unir lista1 y lista2

Enlazar Listas Dblemente Enlazadas (LDE L1, LDE L2): LDE

aux1 ← L1

aux2 ← L2

Lsalida ← Crear Lista Doblemente Enlazada Vacío()

Mientras aux1 <> NULO entonces

insertarFinalNodoLDE(Lsalida, aux1 → dato)

Asume que esta función agrega nodo de LDE


aux1 ← aux1 → puntero

Mientras aux2 <> NULO entonces

insertarFinalNodoLDE(Lsalida, aux2 → dato)

aux2 ← aux2 → puntero

Otra idea que tenía era ir creando un nodo con todos los datos (valor, anterior, siguiente) y ir agregando su dirección de memoria a la nueva lista creada.



Actividad de cierre



- Ir a [menti.com](https://www.menti.com) e ingresar código 8352 4450



Próximas fechas...

- Resumen de la semana:
 - TDA lista enlazada
 - TDA lista enlazada circular
 - TDA lista doblemente enlazada

U2 - S6

~~cátedra~~ – ~~refuerzo1~~ – ~~refuerzo2~~ – ~~lab1~~ – ~~lab2~~

- Subsiguiente semana:
 - TDA pila
 - TDA cola

Octubre 2023						
Domingo	Lunes	Martes	Miércoles	Jueves	Viernes	Sábado
1	2	3	4	5	6	7
8	9	10	11	12	13	14
15	16	17	18	19	20	21
22	23	24	25	26	27	28
29						

Noviembre 2023						
Receso						
Domingo	Lunes	Martes	Miércoles	Jueves	Viernes	Sábado
5	6	7	8	9	10	11
12	13	14	15	16	17	18
19	20	21	22	23	24	25
26	27	28	29	30		