



# Análisis de Algoritmos y Estructura de Datos

Semana 6

Ejercicios 2: listas enlazadas simples y especiales

Prof. Violeta Chang C

Semestre 2 – 2023



# TDA lista enlazada

- **Contenidos:**

- Operaciones con listas enlazadas simples y especiales

- **Objetivos:**

- Aplicar operaciones de TDA lista enlazada, TDA lista enlazada circular y TDA lista doblemente enlazada para resolver problemas específicos



# TDA lista enlazada





# Ejercicios



## Actividad en parejas - 15 minutos

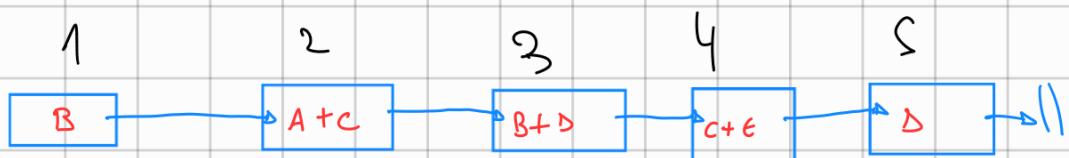
Escribir un algoritmo en seudocódigo que reciba una lista enlazada L1 de enteros desordenados y que devuelva una nueva lista enlazada L2, tal que cada elemento de L2 sea la suma de sus vecinos en L1. Los vecinos de un nodo son el nodo anterior y el nodo siguiente, siempre que existan(\*). Las listas L1 y L2 tienen la misma cantidad de elementos. El algoritmo propuesto debe ser de complejidad proporcional a la cantidad de elementos de L1. Explicar cómo el algoritmo propuesto cumple con la restricción de complejidad indicada.

(\*) si no existen ambos tomar solo el siguiente o solo el anterior

idea



Resultado



Suma Vecinos (LE lista): LE

$O(1)$  { aux <-- lista

$n \leftarrow$  obtenerCantidadNodos(lista)

YES <-- crearListaVacia()

PARA  $i < 1$  HASTA  $n$  PASO 1

aux <-- aux->puntero ... Veras de inmediato al nodo siguiente.

Si  $i = n$  ENTONCES

AGREGAR final (YES, 0)

Sino

AGREGAR final (YES, aux->dato)

aux.Res <-- YES

aux2 <-- lista

PARA  $i < 1$  HASTA  $n$  PASO 1

aux.Res <-- aux.Res -> puntero ... Veras al nodo siguiente resultado.

Si  $i = n$  Entonces

DEVOLVER YES ... Ya se complete la LE\_res.

Sino

aux.Res -> dato <- aux.Res -> dato + aux2 -> dato

aux2 <-- aux2 -> puntero.

⚠️ CREO QUE ESTÁ BIEN

El Vlad usó obtener nodo para hacer este algoritmo.



# Actividad en parejas - 15 minutos

Escribir un algoritmo en seudocódigo para eliminar el nodo central (ubicación del medio) de una lista simplemente enlazada circular de enteros. Calcular y justificar el orden de complejidad del algoritmo propuesto.

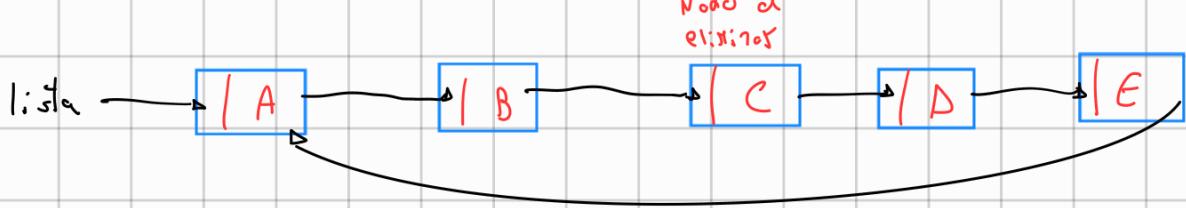
**Entrada:** lista simplemente enlazada circular L de valores enteros

**Salida:** la misma lista simplemente enlazada circular L, sin el nodo del medio

OBS: el elemento del medio en una lista L con un número n de nodos corresponde al nodo en la ubicación  $\lceil n/2 \rceil$  (entero superior)

↳ Ej.: 5 nodos  $\Rightarrow 5/2 = 2.5$ , luego 3  $\lceil$  superior

idea



algoritmo (LEC lista)

indicado <- lista

anterior <- NULO

n <- cantidadNodosCircular (lista)

posicion\_borrar <- ⌈n/2⌉ ... entero superior

PARA i <= 1 HASTA posicion\_borrar PASO 1

anterior <- aux

indicado <- indicado → puntero

anterior → puntero ← indicado → puntero

liberar (indicado)

anexo

CantidadNodosCircular (LEC lista): num

aux <- lista

count <- 1 ... partimos de 1 pq no cuenta el ultimo nodo

MIENTRAS aux → puntero <> lista HACER

count <- count + 1

aux <- aux → puntero

DEVOLVER count.

esta bien (?)



# Actividad en parejas - 15 minutos

Kidd Voodoo

- Después de obtener su grado de doctorado, Cristina se ha convertido en toda una celebridad en su universidad, y su perfil de una determinada red social está lleno de solicitudes de contacto. Con el deseo de quedar bien con todas y todos, Cristina ha aceptado todas las solicitudes. Sin embargo, con el correr de los días, entrar a su cuenta de la red social se ha transformado en una avalancha de noticias y posteos. Simplemente, no alcanza a mirar todo y muchos de sus nuevos contactos se resienten porque no les pone 'like' a sus posteos, mientras que a otros sí. Para evitarse problemas, Cristina ha decidido eliminar a algunos contactos de su red social. Como ella sabe la popularidad de cada uno de los contactos que tiene, ha decidido eliminar a quienes tengan popularidad menor a la media. **Escribir un algoritmo que ayude a Cristina a identificar exactamente cuáles contactos debe eliminar, según su requerimiento.** Solamente se puede utilizar el TDA lista enlazada. Se debe ingresar el número N de contactos que Cristina tiene actualmente, el número K de contactos que Cristina quiere eliminar como máximo y la popularidad de los actuales contactos de Cristina. Se deben mostrar los números que representan la popularidad de los contactos de Cristina después de filtrar según la restricción indicada. Además, se debe **calcular y justificar el orden de complejidad del algoritmo propuesto.**

N: idea wñ

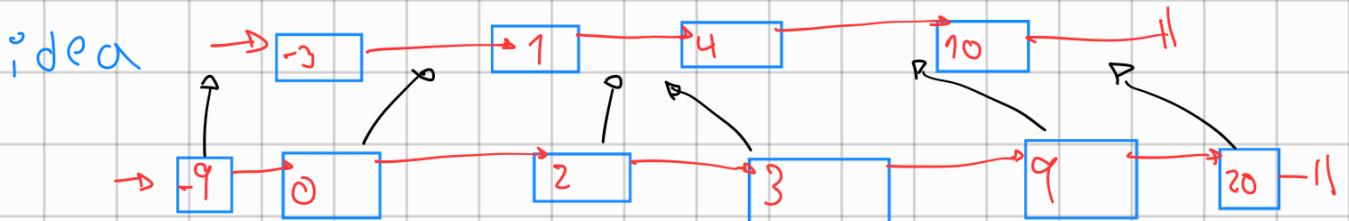


## Incentivo - 20+5 minutos

Escribir un algoritmo para que, dadas dos listas enlazadas ordenadas L1 y L2, devuelva una nueva lista L luego de mezclar las listas L1 y L2, manteniendo la condición de ordenamiento (es decir, L debe estar ordenada bajo el mismo criterio que L1 y L2). Además se debe calcular y justificar la complejidad del algoritmo propuesto.

▷ puede ser decreciente  
o creciente

IDEA: Creo una LF de las 2 listas, luego busco en todo la lista un nodo con el dato menor, lo elimino y agrego al final de una nueva lista (No sirve cuando es orden decreciente)



Voy agregando cada nodo en su posic correspondiente

algoritmo ( $LG L_1, LE L_2$ ):  $LE$

$aux \leftarrow L_1$

$salida \leftarrow \text{crearListaVacia}()$  ... haremos copia de  $L_1$ .

MIENTRAS  $aux \neq \text{NULO}$  HACER

$\text{insertarNodoFinal}(salida, aux \rightarrow \text{dato})$

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

$auxL_2 \leftarrow L_2$

MIENTRAS  $auxL_2 \neq \text{NULO}$  HACER

$auxS \leftarrow salida$

MIENTRAS  $auxS \neq \text{NULO}$  HACER

Si  $auxS \rightarrow \text{puntero} \neq \text{NULO}$  Entonces ↗

$\text{datoSiguiente} \leftarrow (auxS \rightarrow \text{puntero}) \rightarrow \text{dato}$  Entonces ↗

Si  $auxS = \text{salida}$  Y  $auxL_2 \rightarrow \text{dato} \leq \text{datoSiguiente}$

$\text{insertarNodoInicio}(salida, auxL_2 \rightarrow \text{dato})$

Entonces ↗

Si  $auxL_2 \rightarrow \text{dato} \geq \text{auxS} \rightarrow \text{dato}$  Y  $auxL_2 \rightarrow \text{dato} \leq \text{datoSiguiente}$

$\text{insertarNodoDespues}(salida, auxL_2 \rightarrow \text{dato}, auxS \rightarrow \text{dato})$

Entonces ↗

Si  $auxS \rightarrow \text{puntero} = \text{NULO}$  Y  $auxL_2 \rightarrow \text{dato} \geq \text{auxS} \rightarrow \text{dato}$

$\text{insertarNodoFinal}(salida, auxL_2 \rightarrow \text{dato})$

Creo que esto  
esta bien.

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

$auxL_2 \leftarrow auxL_2 \rightarrow \text{puntero}$



# Incentivo 3 - 15+5+5 minutos

## Puntaje

I.	El algoritmo propuesto apunta a resolver el problema planteado. (SI 1 punto / NO 0 punto)	
II.	El algoritmo resuelve correctamente el problema planteado. (SI 1 punto / NO 0 punto)	
III.	El algoritmo está escrito en pseudocódigo ordenado (SI 1 punto / NO 0 punto)	
IV.	El algoritmo está escrito en el formato establecido (SI 1 punto / NO 0 punto)	
V.	El algoritmo identifica entradas correctamente (SI 1 punto / NO 0 punto)	
VI.	El algoritmo identifica y declara salidas de manera correcta. (SI 1 punto / NO 0 punto)	
VII.	Calcula correctamente la complejidad (SI 1 punto / NO 0 punto)	
VIII.	Justifica la complejidad del algoritmo propuesto. (SI 1 punto / NO 0 punto)	
	PUNTOS	



# Actividad de cierre



- Ir a [menti.com](https://www.menti.com) e ingresar código 2612 1214



# 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		