

# Tema 2.1 TAD lineales

Estructura de Datos y Algoritmos (EDA)

# Contenidos

---

- ▶ **2.1. ¿Qué es un TAD Lineal?**
- ▶ 2.2. TAD Pila
- ▶ 2.3. TAD Cola
- ▶ 2.4. TAD Lista
  - ▶ 2.4.1 Implementación con una Lista Simplemente Enlazada
  - ▶ 2.4.2 Implementación con una Lista Doblemente Enlazada

# Objetivos

---

- ▶ Al final de la clase, los estudiantes deben ser capaces de:
  - ▶ Definir un TAD lineal
  - ▶ Explicar las principales ventajas y desventajas de usar arrays para implementar un TAD lineal
  - ▶ Explicar cómo las estructuras de datos dinámicas son una alternativa a los arrays
  - ▶ Implementar una Cola usando una estructura dinámica
  - ▶ Explicar el concepto de nodo e implementarlo como una clase

# TAD Lineal

---

- ▶ Representa una secuencia de elementos de algún tipo
- ▶ Ejemplos:
  - ▶ Nombre se personas (strings): María, Pepe, Juan, ...
  - ▶ Números enteros: 5,6,1,-3,0,2,1,...
  - ▶ Objetos (instancias) de la clase Punto
  - ▶ Objetos (instancias) de la clase Empleado
- ▶ Todos los elementos deben pertenecer al mismo tipo de datos

# Arrays

- ▶ Ejemplo: un array (tamaño 6) of números enteros.
- ▶ Un entero toma 4 bytes en espacio de memoria.
- ▶ Si se conoce la dirección inicial de un array y el índice de un elemento, puede calcular fácilmente su dirección

<b>Dirección de memoria</b>	100	104	108	112	116	120
<b>Elementos</b>	<b>21</b>	<b>33</b>	<b>-5</b>	<b>8</b>	<b>12</b>	<b>2</b>
<b>Posición respecto al array</b>	0	1	2	3	4	5

# Arrays

- ▶ **Ventajas:** acceso directo a cada elemento del array a través de su índice.
- ▶ **Desventajas:**
  - ▶ Las operaciones de borrado e inserción pueden requerir que algunos elementos se muevan. Estas operaciones son **lentas**.

Insertar 8 en la posición 4



1	3	5	0	1	10
---	---	---	---	---	----

1	3	5	8	0	1
---	---	---	---	---	---

*Perdemos el último elemento: 10*

# Arrays

---

## ▶ Desventajas:

- ▶ Los arrays tienen tamaños fijos (no se pueden modificar en tiempo de ejecución)
- ▶ ¡Algunas veces el tamaño puede ser insuficiente!
- ▶ Un tamaño muy grande puede conllevar a un uso ineficiente de la memoria
- ▶ A veces no se puede saber el tamaño necesario para tu problema

# Arrays

---

- ▶ Ejemplo:
  - ▶ Un array para almacenar la temperatura media diaria del último año. Tu puedes usar un array de tamaño 365 (366 es un año bisiesto).



# Estructuras de datos dinámicas

---

- ▶ Los elementos no se almacenan en posiciones de memoria consecutivas, sino también con espacios entre ellos.
- ▶ Puede crecer o reducirse en tiempo de ejecución
- ▶ Uso eficiente de la memoria (solo pueden ocupar la memoria necesaria)
- ▶ Alternativa a los arrays para implementar TAD lineales. Vamos a verlo !!!

# Array vs Estructura dinámica

**Array**

memoria

Mary
John
Jim
Arthur
Martin

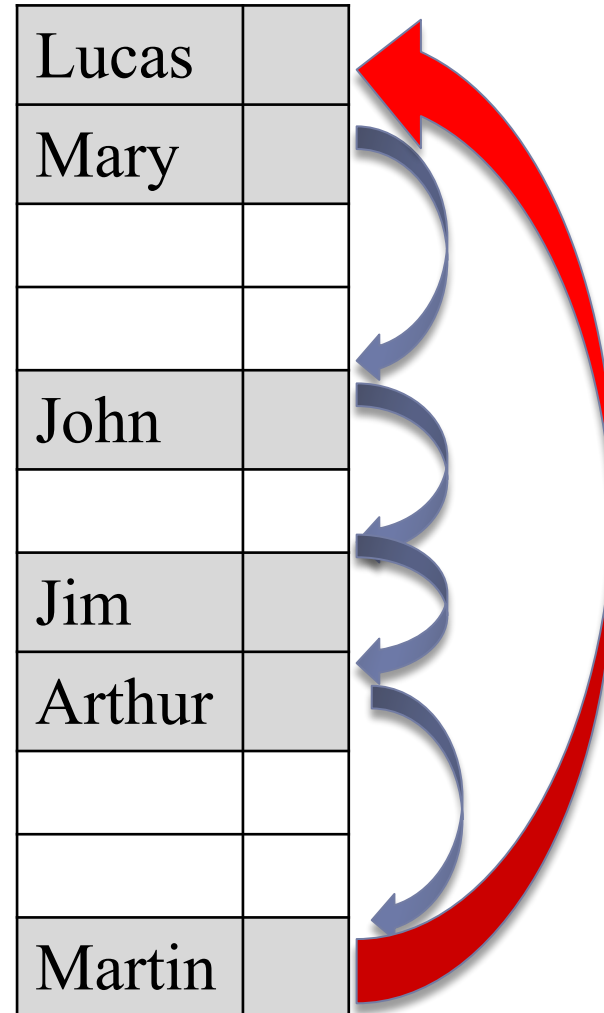
**Estructura dinámica**

Mary	
John	
Jim	
Arthur	
Martin	

# Implementación de un TAD lineal utilizando una estructura dinámica

## Estructura dinámica

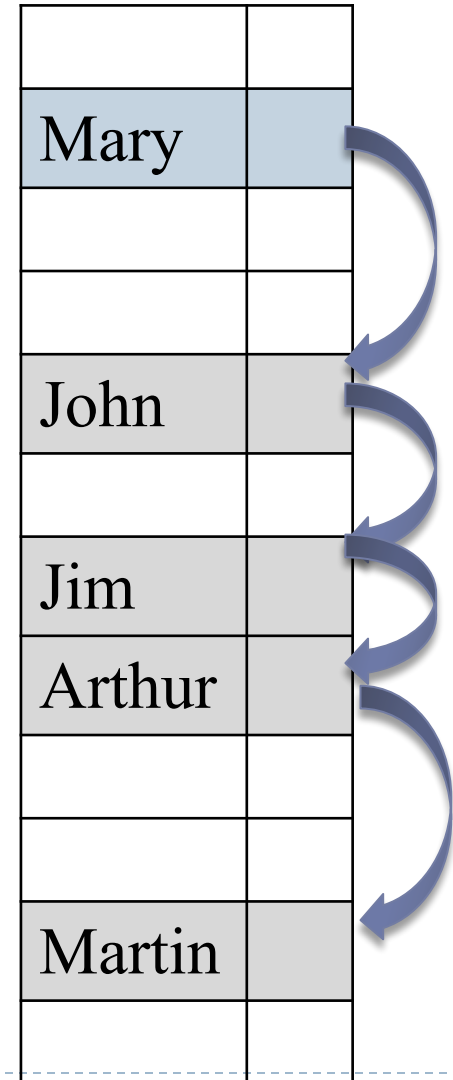
Los espacios en la memoria permiten que las órdenes físicas y lógicas puedan ser diferentes



# Implementación de un TAD lineal utilizando una estructura dinámica

## Estructura dinámica

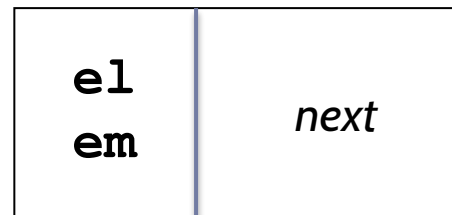
Cada ubicación no solo almacena (una referencia a) un objeto, sino también la referencia (dirección) a su sucesor en la Lista



# Implementación de un TAD lineal utilizando una estructura dinámica

- ▶ Necesitamos definir una clase, **Node**, para representar cada ubicación. La clase **Node** tiene dos atributos:
  - ▶ **elem**: es la referencia a un elemento almacenado en la Lista. Es decir, almacena la dirección de memoria donde se almacena el elemento. El tipo de datos de **elem** debe ser del mismo tipo que los elementos de la Lista
  - ▶ **next**: es la referencia al nodo que almacena el siguiente elemento en la Lista. Su tipo de datos debe ser **Node**

Node



# Class Snode (simplemente node)

---

```
public class SNode {  
  
    public String elem;  
    public SNode next;  
  
    public SNode(String e) {  
        elem = e;  
    }  
}
```

Permite almacenar un objeto de tipo String. Sin embargo, puede definir un node que almacene cualquier tipo de objeto.

**uc3m** | Universidad **Carlos III** de Madrid