

# Estructuras Lineales

## Listas

Verónica E. Arriola-Rios

Facultad de Ciencias, UNAM

7 de julio de 2021



# Listas

- 1 Listas
- 2 Implementaciones
- 3 Listas circulares
- 4 Listas múltiples
- 5 Bibliografía

# Temas

## 1 Listas

- Definición recursiva
- Definición secuencial

# Definición de lista

## Recursividad estructural

$$\text{Lista} = \begin{cases} \text{Lista vacía} \\ \text{Dato seguido de otra lista} \end{cases}$$

# Lista recursiva en Java

## Código 1: Lista recursiva

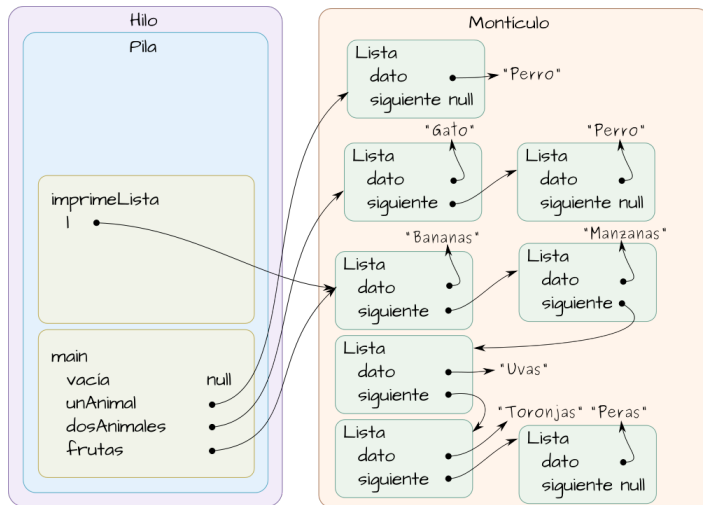
```
1 public class Lista {
2     private Object dato;
3     private Lista siguiente;
4
5     /** Construye una lista con un dato, seguida de otra lista. */
6     public Lista(Object dato, Lista siguiente) {
7         this.dato = dato;
8         this.siguiente = siguiente;
9     }
10
11     public Object getData() {
12         return dato;
13     }
14
15     /** Versión funcional de un método que trabaja con la lista. */
16     public static void imprimeLista(Lista l) {
17         if(l == null) return;           // Caso base, escrito explícitamente.
18         else {
19             System.out.println(l.dato);
20             imprimeLista(l.siguiente);
21         }
22     }
23 }
```

# Construcción manual de listas

- $\emptyset$ .
- "Perro"  $\rightarrow \emptyset$ .
- "Gato"  $\rightarrow$  "Perro"  $\rightarrow \emptyset$ .
- "Bananas"  $\rightarrow$  "Manzanas"  $\rightarrow$  "Toronjas"  $\rightarrow$  "Uvas"  $\rightarrow$  "Peras"  $\rightarrow \emptyset$ .

## Código 2: Uso de listas

```
1 public class UsoLista {
2     public static void main(String[] args) {
3         Lista vacía = null;
4         Lista unAnimal = new Lista("Perro", null);
5         Lista dosAnimales = new Lista("Gato", new Lista("Perro", null));
6         Lista frutas = new Lista("Bananas",
7                                 new Lista("Manzanas",
8                                             new Lista("Toronjas",
9                                                         new Lista("Uvas", new Lista("Peras", null)))));
10        imprimeLista(frutas);
11    }
12 }
```



# Construcción manual de listas

- $\emptyset$ .
- "Perro"  $\rightarrow \emptyset$ .
- "Gato"  $\rightarrow$  "Perro"  $\rightarrow \emptyset$ .
- "Conejo"  $\rightarrow$  "Gato"  $\rightarrow$  "Perro"  $\rightarrow \emptyset$ .

## Código 3: Uso de listas

```
1 public class UsoLista {
2     public static void main(String[] args) {
3         Lista vacía = null;
4         Lista unAnimal = new Lista("Perro", null);
5         Lista dosAnimales = new Lista("Gato", new Lista("Perro", null));
6         Lista animales = new Lista("Conejo", dosAnimales);
7         imprimeLista(animales);
8     }
9 }
```



# Temas

## 1 Listas

- Definición recursiva
- Definición secuencial

# Definición de lista 2

- Una *lista* es una secuencia de cero a más elementos **de un tipo determinado** (que por lo general se denominará tipo-elemento). Se representa como una sucesión de elementos separados por comas:

$$a_0, a_1, \dots, a_{n-1} \quad (1)$$

donde  $n \geq 0$  y cada  $a_i$  es del tipo **tipo-elemento**.

- Al número  $n$  de elementos se le llama *longitud* de la lista.
- $a_0$  es el *primer elemento* y  $a_{n-1}$  es el *último elemento*.
- Si  $n = 0$ , se tiene una **lista vacía**, es decir, que no tiene elementos. Aho, Hopcroft y Ullman 1983

# Lista

Una propiedad importante de una lista es que sus elementos pueden estar ordenados en forma lineal de acuerdo con sus posiciones en la misma.

- Se dice que  $\alpha_i$  *precede* a  $\alpha_{i+1}$  para  $i = 0, 1, \dots, n-2$  y  $\alpha_i$  *sucedee* a  $\alpha_{i-1}$  para  $i = 1, 2, \dots, n-1$ .
- Se dice que el elemento  $\alpha_i$  está en la posición  $i$ .

# Operaciones

- instanciar:  $\emptyset \rightarrow \text{ListaVacía}$
- insertar:  $\text{Lista, Posición, Elemento} \rightarrow \text{Lista}$
- recuperar:  $\text{Lista, Posición} \rightarrow \text{Elemento}$
- borrar:  $\text{Lista, Elemento} \rightarrow \text{Lista}$
- siguiente:  $\text{Lista, Posición} \rightarrow \text{Elemento}$
- anterior:  $\text{Lista, Posición} \rightarrow \text{Elemento}$
- destruir:  $\text{Lista} \rightarrow \emptyset$
- imprimir:  $\text{Lista} \rightarrow \text{Cadena}$

# Implementaciones

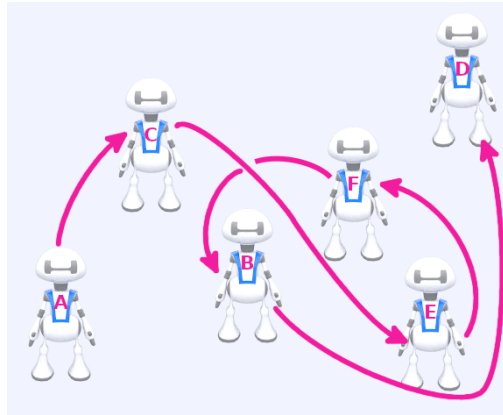
- 1 Listas
- 2 Implementaciones**
- 3 Listas circulares
- 4 Listas múltiples
- 5 Bibliografía

# Temas

## 2 Implementaciones

- Lista ligada
- Lista doblemente ligada
- Lista en un arreglo

# Lista ligada



**Figura:** Cada robot sabe quién va detrás de él. La lista define la secuencia de robots: ACEFBD.

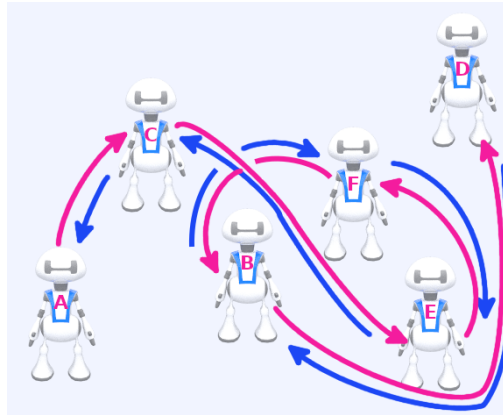
# Temas

## 2 Implementaciones

- Lista ligada
- Lista doblemente ligada
- Lista en un arreglo



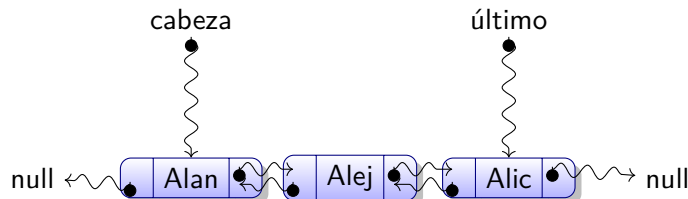
# Lista doblemente ligada



**Figura:** Cada robot sabe quién va detrás de él. La lista define la secuencia de robots: ACEFBD.

# Nodo doble

- Cada elemento contiene referencias al elemento anterior y al elemento posterior.



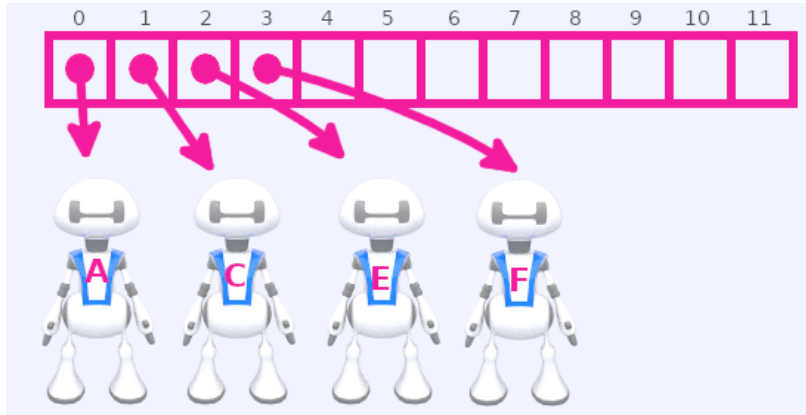
NodoDoble<E>
-dato: E -siguiente: NodoDoble<E> -anterior: NodoDoble<E>
+NodoDoble(dato:E) +NodoDoble(dato:E,siguiente:NodoDoble<E>) +NodoDoble(dato:E,anterior:NodoDoble<E>, siguiente:NodoDoble<E>) +getDato(): E +setDato(dato:E): void +getSiguiente(): NodoDoble<E> +setSiguiente(sig:NodoDoble<E>): void +getAnterior(): NodoDoble<E> +setAnterior(ant:NodoDoble<E>): void

# Temas

## 2 Implementaciones

- Lista ligada
- Lista doblemente ligada
- Lista en un arreglo

# Lista en un arreglo



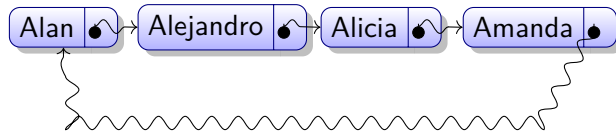
**Figura:** Los elementos de la lista se guardan en las posiciones correspondientes de un arreglo.

# Listas circulares

- 1 Listas
- 2 Implementaciones
- 3 Listas circulares**
- 4 Listas múltiples
- 5 Bibliografía

# Listas circulares

- El elemento final hace referencias al elemento inicial, por lo que para todo elemento existe un elemento siguiente y un elemento anterior.



# Listas múltiples

- 1 Listas
- 2 Implementaciones
- 3 Listas circulares
- 4 Listas múltiples**
- 5 Bibliografía

# Listas múltiples

- Representan relaciones muchos a muchos Vargas Villazón, Lozano Moreno y Levine Gutiérrez 1998.
- Se representan con referencias.
- Casi siempre es más fácil usar tablas, pero llegan a aparecer.



# Listas múltiples. Ejemplo.

Estudiantes\Cursos	CS101	CS202	CS203
Alan			✓
Alejandro	✓	✓	
Alicia			✓
Amanda	✓		
Andrés		✓	✓
Angélica	✓		✓

Registro de  
Estudiantes

Alan

Alejandro

Alicia

Amanda

Andrés

Angélica

Registro de  
InscripciónRegistro de  
Cursos

CS101



CS202

CS303

# Bibliografía

- 1 Listas
- 2 Implementaciones
- 3 Listas circulares
- 4 Listas múltiples
- 5 Bibliografía**

# Bibliografía I

-  Aho, Alfred V., John E. Hopcroft y Jeffrey D. Ullman (1983). *Data Structures and Algorithms*. Addison-Wesley.
-  Vargas Villazón, América, Jorge Lozano Moreno y Guillermo Levine Gutiérrez, eds. (1998). *Estructuras de datos y Algoritmos*. 438 pp.

# Licencia

Creative Commons  
Atribución-No Comercial-Compartir Igual

