

Estructuras de Datos

TDAs y Estructuras Lineales

Gonzalo Gabriel Méndez, Ph.D.



TDA?



atributos +
comportamiento

TDA

Comportamiento definidos por:
valores + operaciones

Se llama ***abstracto*** porque proporciona un punto de vista independiente de la implementación.

<https://www.geeksforgeeks.org/abstract-data-types>

TDAs y Estructuras Lineales

Elementos están colocados uno detrás de otro

Cada elemento de una lista se conoce con el nombre de **nodo**

De acuerdo a su comportamiento, los conjuntos lineales se clasifican en

- Listas
- Pilas
- Colas
- Conjuntos y mapas

TDA Lista

Definición

Colección de 0 o más elementos

Todos los elementos son de un mismo tipo

Si la lista no tiene elementos, se dice que esta vacía

Mucho más flexibles que los arreglos porque permiten trabajo “*dinámico*” con un grupo de elementos

Dinámico = la estructura puede crecer o achicarse como sea necesario

Lista: Comportamiento deseado

A/con una lista se puede:

- Crear y Eliminar
- Conocer si esta vacía
- Añadir elementos y removerlos
- Consultar el primer y al ultimo elemento
- Imprimir sus elementos en pantalla
- Buscar un elemento con cierta información en la lista

TDA Lista

Java - Interface List

- En java la interfaz List corresponde a la implementación del TDA List

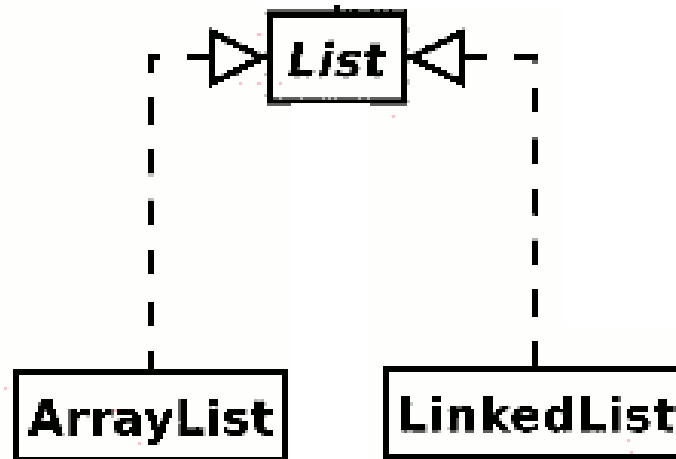
```
interface List {  
    boolean add(Object o)  
    void add(int index, Object element)  
    boolean addAll(Collection<? extends Object> c)  
    boolean addAll(int index, Collection<? extends  
Object> c)  
    void clear()  
    boolean contains(Object o)  
    boolean containsAll(Collection<?> c)  
    boolean equals(Object o)  
    Object get(int index)  
    ...  
}
```

TDA Lista Recordar

- TDA especifica el comportamiento pero no la implementación

TDA →

ESTRUCTURA DE
DATOS →

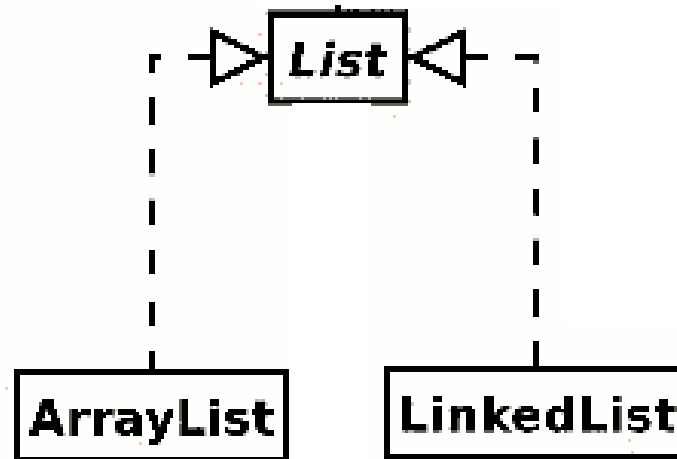


TDA Lista Recordar

- TDA especifica el comportamiento pero no la implementación

TDA →

ESTRUCTURA DE
DATOS →



```
List a = new  
ArrayList();  
List b = new  
LinkedList();
```

Tipos de listas

Conceptualmente, las **listas** se pueden clasificar en:

- Simplemente Enlazadas
- Doblemente Enlazadas
- Circulares
- Circular Doblemente Enlazada

Listas simples

- Conjunto de nodos uno detrás de otro
 - Siempre se puede conocer al nodo **inicial** y al **final**
- Cada nodo de la lista tiene
 - Un contenido
 - Información que almacena dentro y puede ser de cualquier tipo de dato
 - Un sucesor único
 - Excepto el último nodo de la lista que hace referencia a **null**

TDA Lista: Implementación

- Tenemos el concepto claro de lo que debe ser una lista
- ¿Cómo darle vida a una lista?
- Hay varias posibilidades de implementación
 - **Estática** usando arreglos de longitud “variable” (ArrayList en Java).
También llamada continua
 - **Dinámica** (LinkedList en Java)

Implementar un TDA es hacerlo **concreto**. Hay que preocuparse por el **cómo**. Esto deriva en una **Estructura de Datos**.

ArrayList

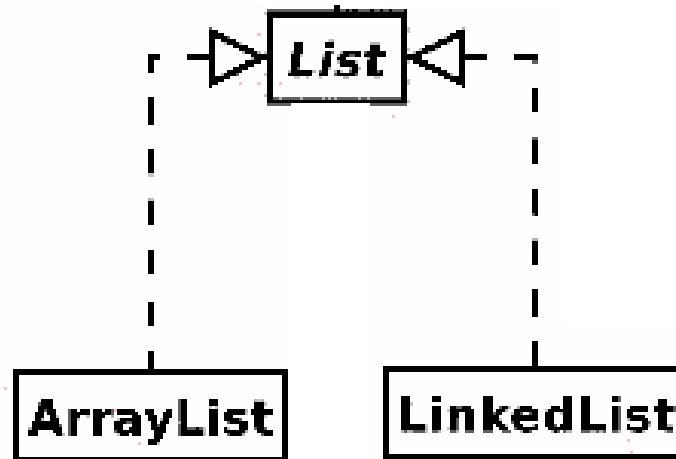
Implementación continua

Estructura de datos - **ArrayList**

- Utiliza un arreglo para implementar la funcionalidad de una lista

TDA →

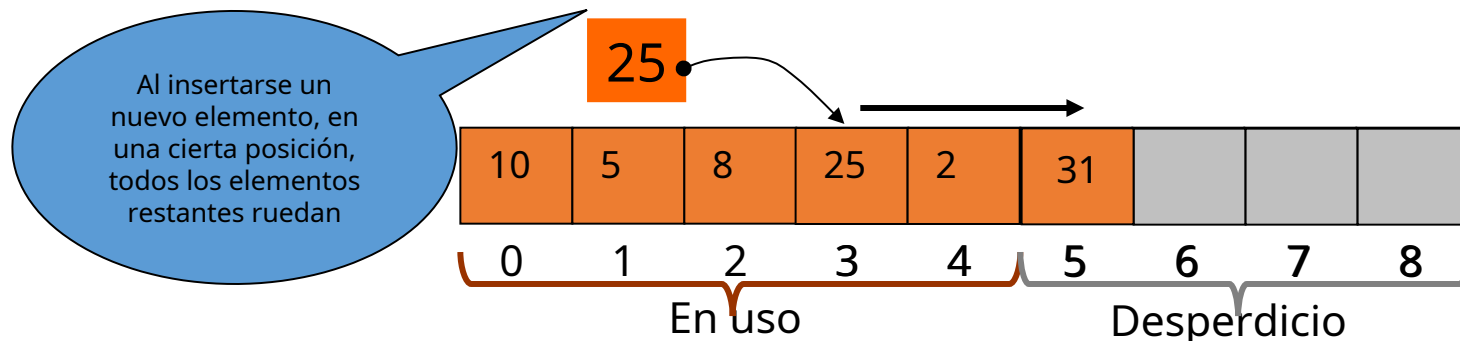
ESTRUCTURA DE
DATOS →



Implementación continua

Estructura de datos - ArrayList

- Utilizar un arreglo para implementar la interface List
 - Secuencia de elementos
 - Elementos continuos y adyacentes en memoria
 - Cada elemento tiene un índice
- Se puede acceder directamente a una ubicación en particular mediante el índice del arreglo
 - Tiempo de acceso constante
- Al insertar o remover un elemento,
 - Todos los elementos restantes avanzarán o retrocederán
- ¿Es ésta la implementación ideal para las listas simples?



Implementación continua

Estructura de datos -

ArrayList

- ¿Cuánto tiempo toma insertar un elemento al inicio de un ArrayList?
 - $O(1)$?
 - $O(n)$?

Implementación continua

Estructura de datos -

ArrayList

- El tamaño de un arreglo es fijo
- Para insertar un nuevo elemento
 - Crear un arreglo de tamaño igual al original más 1
 - Copiar el valor a ser insertado en la primera posición
 - Copiar los valores del arreglo original dentro de las posiciones restantes del nuevo arreglo

Implementación continua

Estructura de datos -

ArrayList

- El tiempo que se gasta copiando todos los valores depende linealmente del tamaño del arreglo original
 - $O(n)$
- El tiempo que toma acceder a un elemento (recuperar) en la implementación de ArrayList es $O(1)$
 - Los elementos están almacenado de forma continua en memoria
 - Pueden ser accedidas con un índice que accede directamente a cualquier elemento a un tiempo constante

Implementación continua

Estructura de datos -

ArrayList

- ArrayList implementa List
- Un TDA no especifica la eficiencia con la que se debe implementar
- La **Estructura de Dato** que implementa el TDA establece la eficiencia en la forma que lo hace
 - ArrayList no es eficiente para insertar nuevos elementos