

Diferencias entre Listas

POO

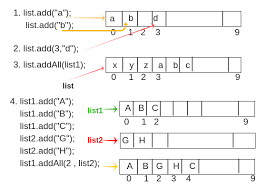


24 de octubre de 2019

IPN

## 

## List

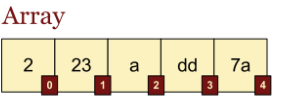


El uso de esta interfaz precisa el control sobre donde está cada elemento insertado. Las listas por lo regular permiten pares de elementos e1 y e2 tal que e1 = e2, y por lo regular permiten múltiples elementos nulos.

### Métodos

Add(E e), add(int index, E element), addAll(Collection<? Extends E> C)

## ArrayList

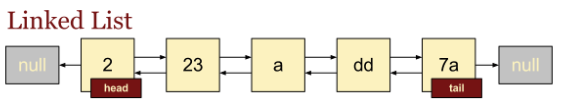


ArrayList ofrece **acceso en tiempo constante**, pero si quieres añadir o remover un elemento en cualquier posición que no sea la última es necesario mover elementos. Además, si el arreglo ya está lleno es necesario crear uno nuevo con mayor capacidad y copiar los elementos existentes. Con el ArrayList se pueden añadir elementos de manera sencilla y se pueden acceder a elementos de igual manera pero tiene costos adicionales al añadir o remover elementos y la cantidad de memoria considera la capacidad definida para el ArrayList (que por lo regular es 10).

### Métodos

get, add(e element), add(int index, e element), remove(int index), Iterator.remove(), remove(int index)

## LinkedList



LinkedList permite **eliminar e insertar elementos en tiempo constante** usando iteradores, pero el acceso es secuencial por lo que encontrar un elemento toma un tiempo proporcional al tamaño de la lista.  
Normalmente la complejidad de esa operación promedio sería O(n/2) sin embargo usar una lista doblemente ligada el recorrido puede ocurrir desde el principio o el final de la lista por lo tanto resulta en O(n/4). Añade y remueve elementos con un iterador y es al final de la lista. Se usa más memoria por las referencias a los elementos anterior y siguiente, el acceso a los elementos depende del tamaño de la lista.

### Métodos

get, add(e element), add(int index, e element), remove(int index), Iterator.remove(), ListIterator.add(e element)