INSTITUTO POLITÉCNICO NACIONAL
UNIDAD PROFESIONAL INTERDISCIPLINARIA DE INGENIERÍA CAMPUS ZACATECAS

PROGRAMACION ORIENTADA A OBJETOS

INVESTIGACION “DIFERENCIAS ENTRE ARRYLIST Y LINKEDLIST”

Nombre de lo(a) alumno(a):

Sofía Lizeth Jiménez Serna

Nombre del maestro:

Roberto Oswaldo Cruz Leija

Fecha de entrega:

24/10/ 19

ArrayList

La clase ArrayList en Java, es una clase que permite almacenar datos en memoria de forma similar a los Arrays, con la ventaja de que el numero de elementos que almacena, lo hace de forma dinámica, es decir, que no es necesario declarar su tamaño como pasa con los Arrays. ArrayList nos permiten añadir, eliminar y modificar elementos (que pueden ser objetos o elementos atómicos) de forma trasparente para el programador

Los principales métodos para trabajar con los ArrayList son los siguientes:

```
// Declaración de un ArrayList de "String". Puede ser de cualquier otro Elemento u Objeto (float, Boolean, Object, ...)
ArrayList<String> nombreArrayList = new ArrayList<String>();
// Añade el elemento al ArrayList
nombreArrayList.add("Elemento");
// Añade el elemento al ArrayList en la posición 'n'
nombreArrayList.add(n, "Elemento 2");
// Devuelve el numero de elementos del ArrayList
nombreArrayList.size();
// Devuelve el elemento que esta en la posición '2' del ArrayList
nombreArrayList.get(2);
// Comprueba se existe del elemento ('Elemento') que se le pasa como parametro
nombreArrayList.contains("Elemento");
// Devuelve la posición de la primera ocurrencia ('Elemento') en el ArrayList
nombreArrayList.indexOf("Elemento");
// Devuelve la posición de la última ocurrencia ('Elemento') en el ArrayList
nombreArrayList.lastIndexOf("Elemento");
// Borra el elemento de la posición '5' del ArrayList
nombreArrayList.remove(5);
// Borra la primera ocurrencia del 'Elemento' que se le pasa como parametro.
nombreArrayList.remove("Elemento");
//Borra todos los elementos de ArrayList
nombreArrayList.clear();
// Devuelve True si el ArrayList esta vacio. Sino Devuelve False
nombreArrayList.isEmpty();
// Copiar un ArrayList
ArrayList arrayListCopia = (ArrayList) nombreArrayList.clone();
// Pasa el ArrayList a un Array
Object[] array = nombreArrayList.toArray();
```

VENTAJAS Y DEVENTAJAS

Ventajas	Desventajas
Añadir elementos	Costos adicionales al añadir o remover elementos
Acceso a elementos	La cantidad de memoria considera la capacidad definida para el ArrayList, aunque no contenga elementos

COMPLEJIDAD

Operación	Complejidad Promedio
`get`	**O(1)**
`add(E element)`	**O(1)**
`add(int index, E element)`	O(n/2)
`remove(int index)`	O(n/2)
`Iterator.remove()`	O(n/2)
`remove(int index)`	O(n/2)

METODOS

Una manera de colacar elementos en un objeto ArrayList es utilizando el metodo de biblioteca add. Para averiguar que tan largo es un objeto ArrayList podemos usar el metodo de biblioteca size .

LINKEDLIST

se basa en la implementación de listas doblemente enlazadas. Esto quiere decir que la estructura es un poco más compleja que la implementación con ArrayList. Si tenemos una lista y lo que nos importa no es buscar la información lo más rápido posible, sino que la inserción o eliminación se hagan lo más rápidamente posible, LinkedList resulta una implementación muy eficiente y aquí radica uno de los motivos por los que es interesante y por los que esta clase se usa en la programación en Java. La clase Linkedlist no permite posicionarse de manera absoluta (acceder directamente a un elemento de la lista) y por tanto no es conveniente para búsquedas pero en cambio sí permite una rápida inserción al inicio/final de la lista

VENTAJAS Y DEVENTAJAS

Ventajas	Desventajas
Añadir y remover elementos con un iterador	Uso de memoria adicional por las referencias a los elementos anterior y siguiente
Añadir y remover elementos al final de la lista	El acceso a los elementos depende del tamaño de la lista

COMPLEJIDAD

Operación	Complejidad Promedio
`get`	O(n/4)
`add(E element)`	**O(1)**
`add(int index, E element)`	O(n/4)
`remove(int index)`	O(n/4)
`Iterator.remove()`	**O(1)**
`ListIterator.add(E element)`	**O(1)**

METODOS

Método: addLast

DEVOLUCIONES:

Si la lista en la que está agregando cambia retorna TRUE, de lo contrario retorna FALSE

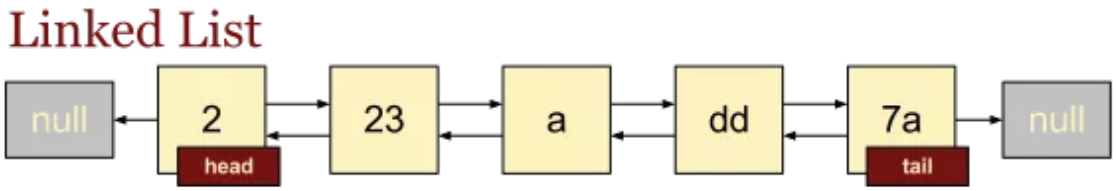
DIFERENCIAS ENTRE ARRAYLIST Y LINKEDLIST

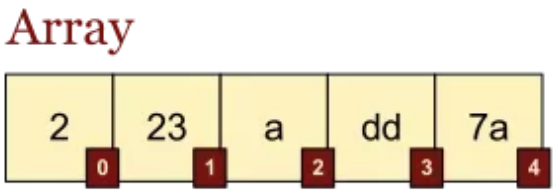
ArrayList:

- usa internamente un arreglo dinámico para almacenar los elementos.
- proporciona una manipulación lenta
- es la mejor opción para almacenar y acceder a datos o elementos consecutivos.

LinkedList

- proporciona una manipulación más rápida porque utiliza una lista doblemente enlazada.
- se puede utilizar como lista y cola porque implementa interfaz de List, Deque y Quede.
- es mejor para manipulación de elementos, es decir, para insertar y eliminar elementos.





Es posible que en un programa tengamos que usar una lista pero no sepamos si es más conveniente usar ArrayList ó LinkedList. En este caso podemos hacer lo siguiente:

Si a priori pensamos que es mejor utilizar una implementación con ArrayList porque pensamos que las búsquedas van a ser la mayoría de las operaciones, entonces pondremos algo así: List listaObjetos = new ArrayList();

Por el contrario si pensamos a priori que la mayoría de las operaciones sobre esta lista de objetos van a ser inserciones o eliminaciones sobre listas grandes escribiremos: List listaObjetos = new LinkedList();

Aquí queda reflejada la utilidad que tiene el uso de interfaces, porque usando los mismos métodos, podemos tener varias implementaciones que nos permitirán un mejor rendimiento dependiendo del uso que demos a nuestra aplicación.

En caso de no tener claro qué operación va a ser más frecuente en general usaremos ArrayList.

BIBLIOGRAFIA

http://www.enrique7mc.com/2016/07/diferencia-entre-arraylist-y-linkedlist/

https://emanuelpeg.blogspot.com/2018/01/diferencia-entre-arraylist-y-linkedlist.html

https://jarroba.com/arraylist-en-java-ejemplos/

https://www.aprenderaprogramar.com/index.php?option=com_content&view=article&id=605:interface-list-clase-linkedlist-api-java-ejercicio-diferencias-entre-arraylist-y-linkedlist-codigo-cu00921c&catid=58&Itemid=180

https://metodos-y-formas.webnode.mx/arraylist/