
Clase 10: Collections - Listas

— Programación & Laboratorio III —

Ejemplo

```
public static void main(String [] args) {
```

```
    Perro [] perros = new Perro[4];
```

```
    Perro p1 = new Perro("Boby");
```

```
    Perro p2 = new Perro("Ayudante de Santa");
```

```
    Perro p3 = new Perro("Tom");
```

```
    Perro p4 = new Perro("Oliver");
```

```
    perros[0] = p1;
```

```
    perros[1] = p2;
```

```
    perros[2] = p3;
```

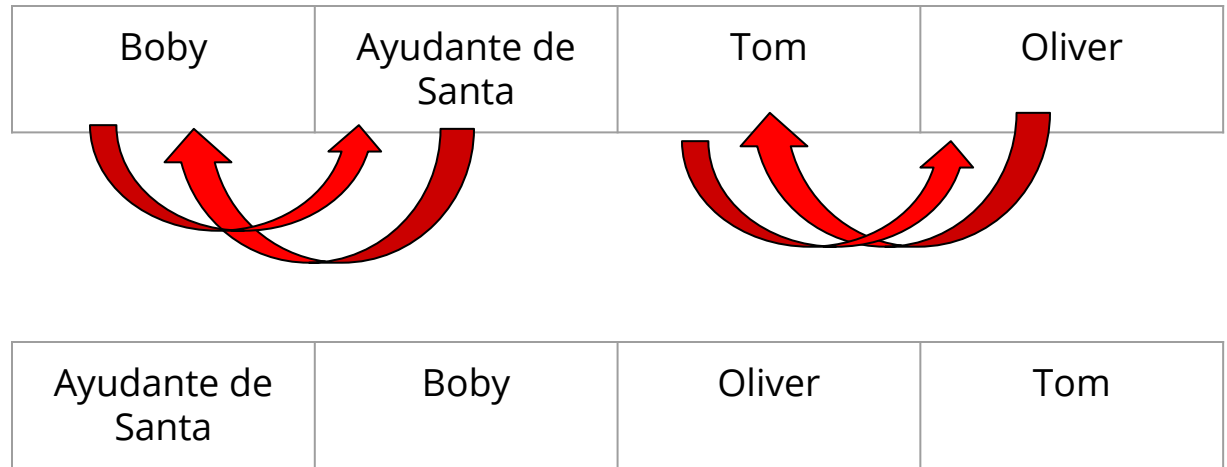
```
    perros[3] = p4;
```

```
}
```

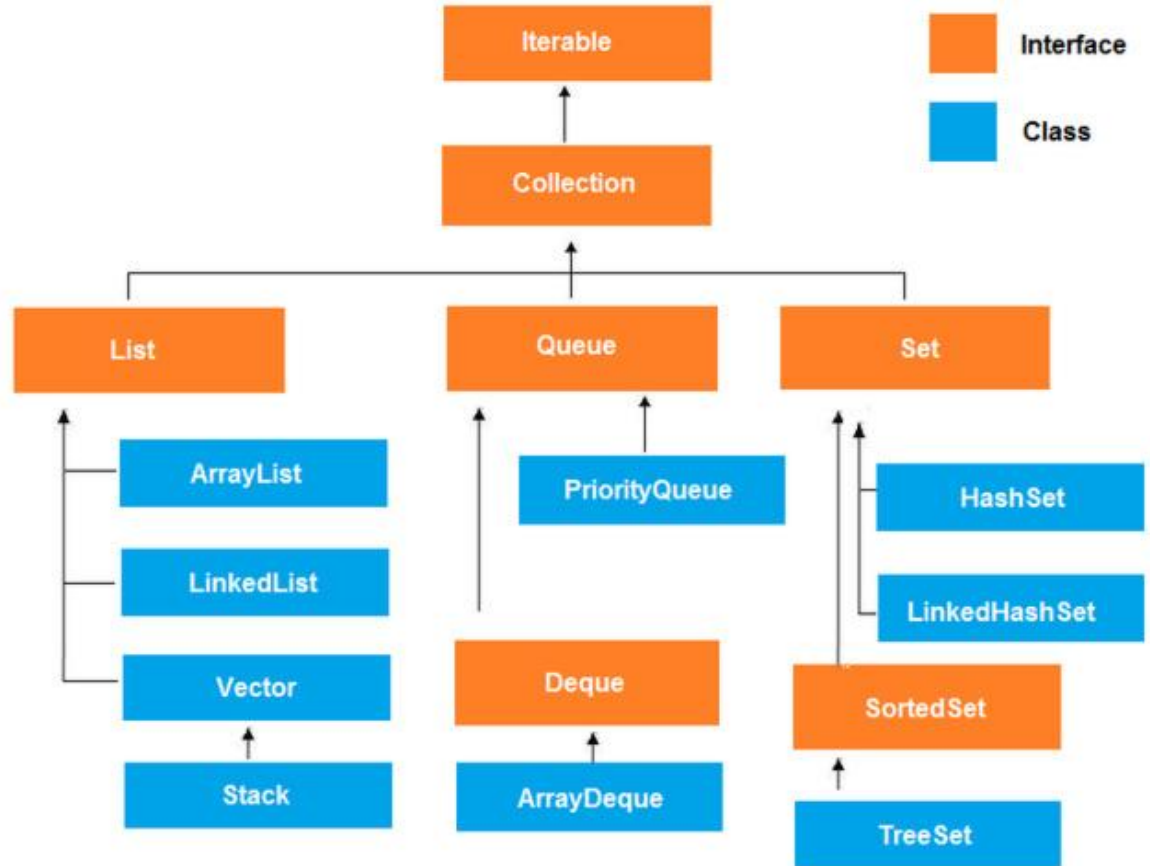
PREVIOUSLY ON...

Problema: Ordenar el arreglo?

- Posible solución:
 - Buscar algoritmos de ordenamiento que utilicen variables auxiliares e implementarlos.

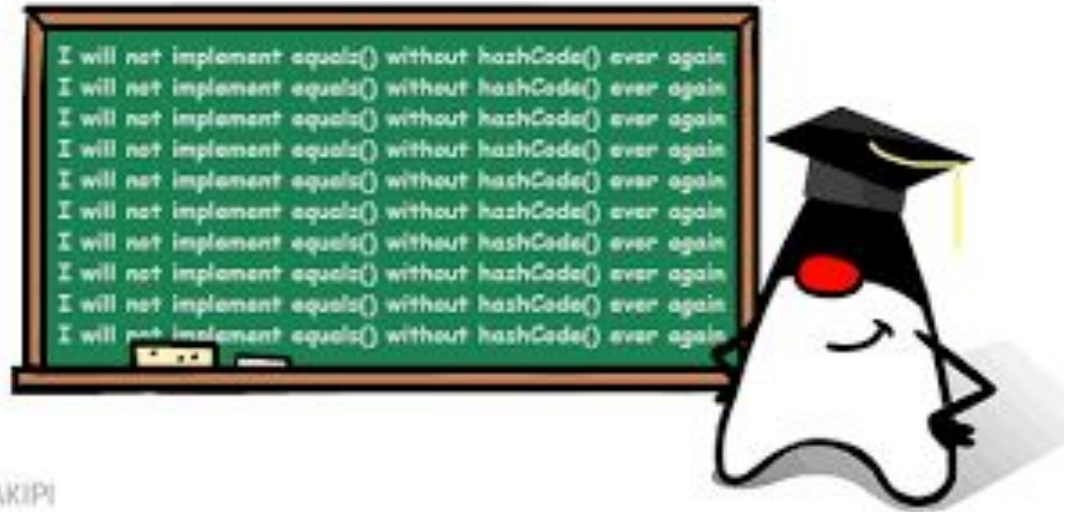


Solución: Collection API



Agenda

- **Collection API**
- Interfaz Collection
- List
- ArrayList
 - Insertar elemento
 - Lectura de un elemento
 - Eliminar elemento
- Método equals()
- Método hashCode()
- Método equals() - Ejemplo



TAKIPI

Collection API

- Una **Collection** es un objeto que representa un grupo de objetos.
- El framework Collection es una arquitectura unificada para representar y manipular colecciones, lo que permite manipularlas independientemente de los detalles de la implementación.



- Reduce los esfuerzos de programación al proveer **estructuras de datos y algoritmos** que no tenemos que escribir.
- Provee **implementaciones** de alta performance.
- Fomenta la reutilización del software al proporcionar una **interfaz** para colecciones y algoritmos para manipularlas.

Agenda

— ~~Collection API~~

- **Interfaz Collection**

- List

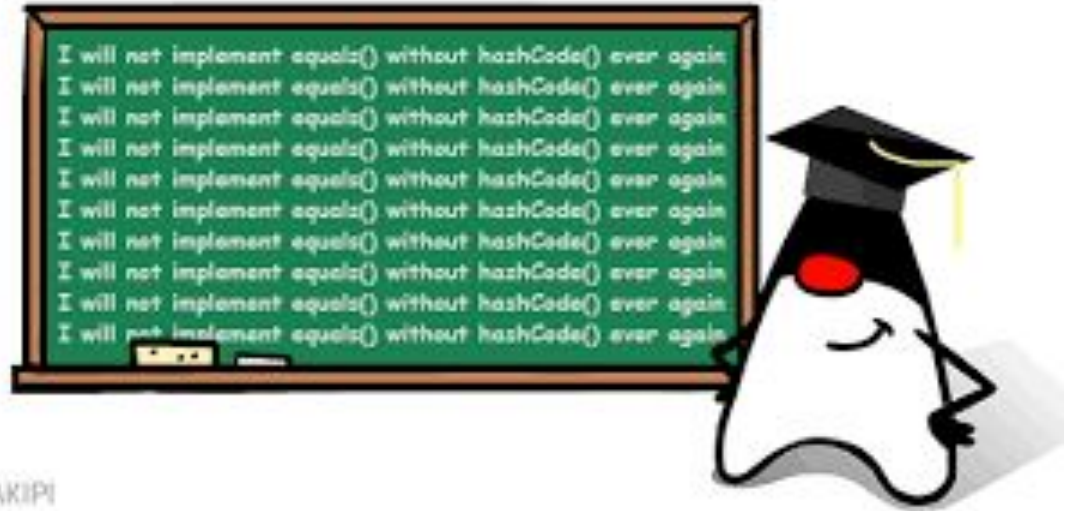
- ArrayList

- Insertar elemento
 - Lectura de un elemento
 - Eliminar elemento

- Método equals()

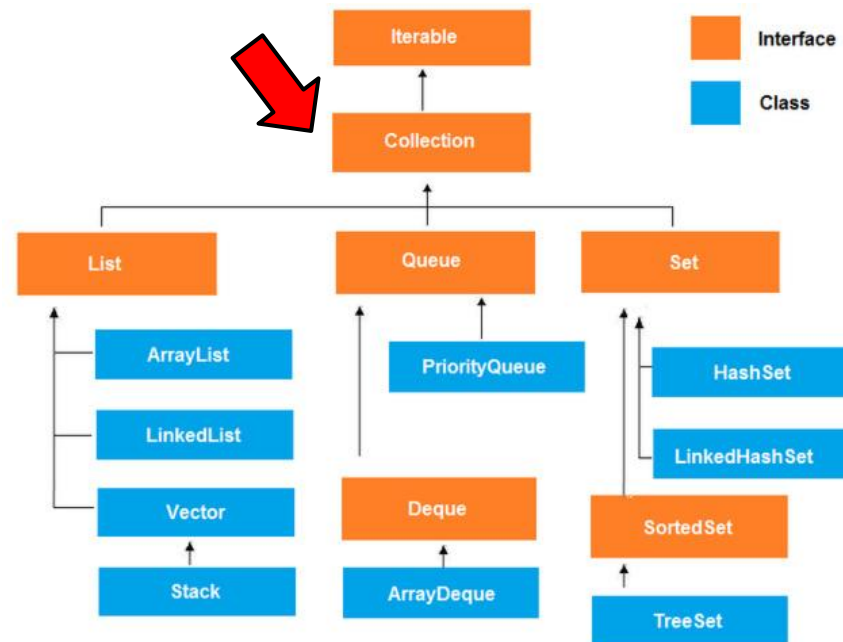
- Método hashCode()

- Método equals() - Ejemplo



Interfaz Collection

- Es la raíz de todas las interfaces y clases relacionadas con colecciones de elementos.
- Algunas colecciones permiten elementos **duplicados** mientras que otras no.
- Otras colecciones pueden tener los elementos **ordenados** mientras que en otras no existe orden alguno.



Interfaz Collection (2)

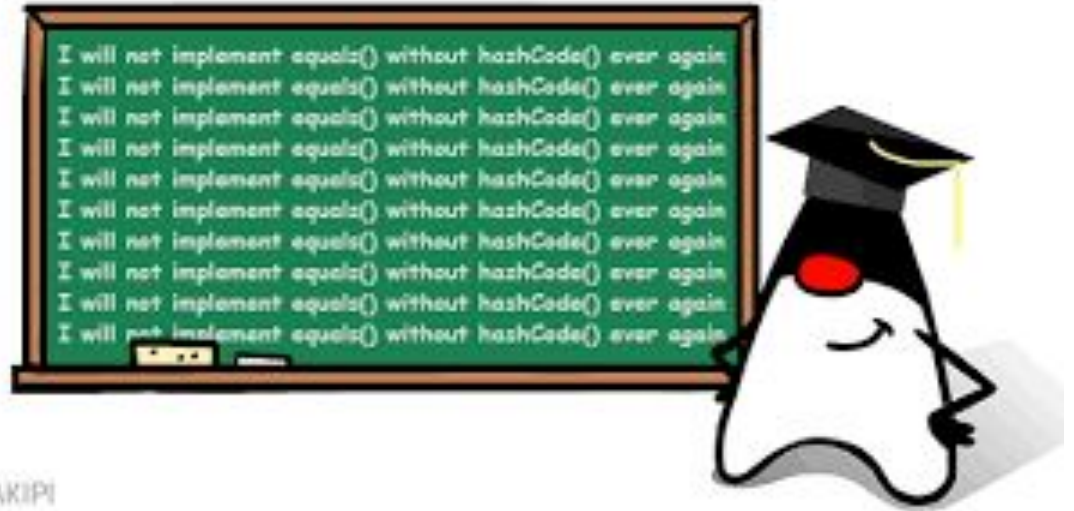
- **Por qué es una interfaz?**
- Es la manera más genérica para representar un grupo de elementos.
- Puede ser usada para pasar colecciones de elementos o manipularlas de la manera más general.
- Gracias a esta interfaz, podemos almacenar cualquier tipo de objeto y podemos usar una serie de métodos comunes, como pueden ser: añadir, eliminar, obtener el tamaño de la colección.
- Se trata de métodos definidos por la interfaz que obligatoriamente han de implementar las subinterfaces o clases que hereden de ella.

Interfaz Collection - Algunos métodos definidos

- `boolean add(E e)`
- `int size()`
- `boolean isEmpty()`
- `boolean remove(E e)`
- `void clear()`
- `Object[] toArray()`

Agenda

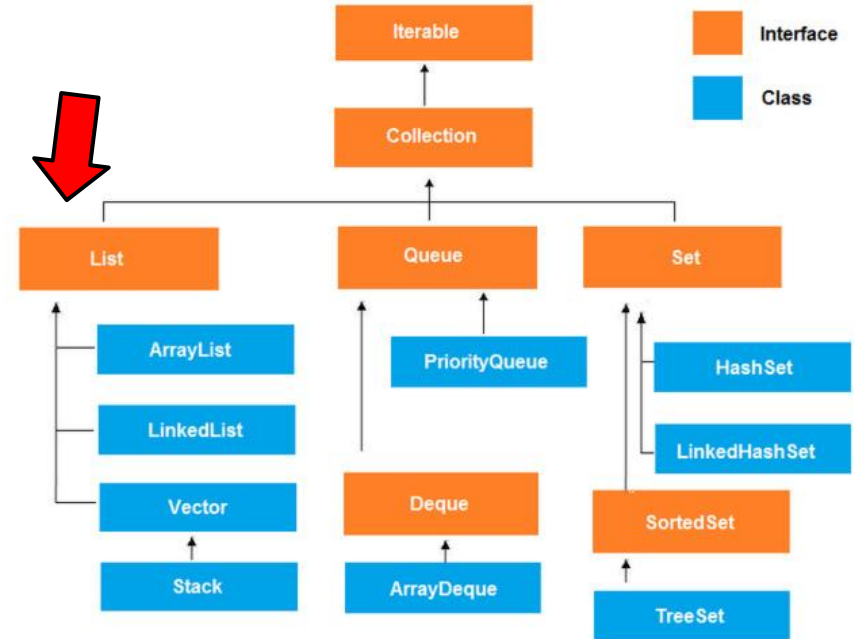
- ~~Collection API~~
- ~~Interfaz Collection~~
- **List**
- ArrayList
 - Insertar elemento
 - Lectura de un elemento
 - Eliminar elemento
- Método equals()
- Método hashCode()
- Método equals() - Ejemplo



TAKIPI

List

- Es la interfaz encargada de agrupar una colección de elementos en forma de lista, es decir uno detrás de otro.
- Acepta elementos **duplicados**.
- Al igual que en los arreglos, el primer elemento está en la posición 0.

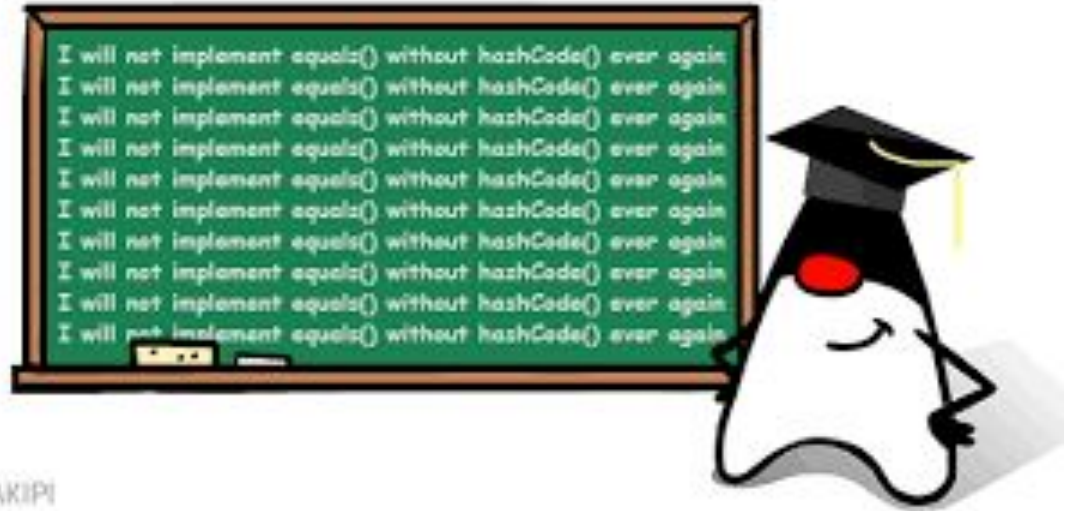


List - Algunos métodos definidos

- `E get(int index)`
- `E set(int index, E element)`
- `E add(int index, E element)`
- `E remove(int index)`
- `int indexOf(Object o)`
- `Object[] toArray()`

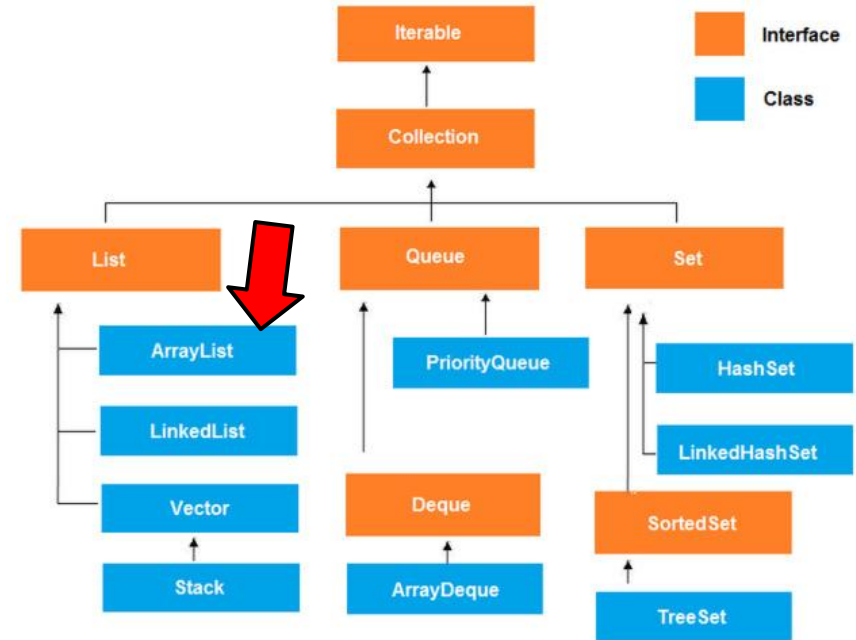
Agenda

- ~~Collection API~~
- ~~Interfaz Collection~~
- ~~List~~
- **ArrayList**
 - Insertar elemento
 - Lectura de un elemento
 - Eliminar elemento
- Método equals()
- Método hashCode()
- Método equals() - Ejemplo



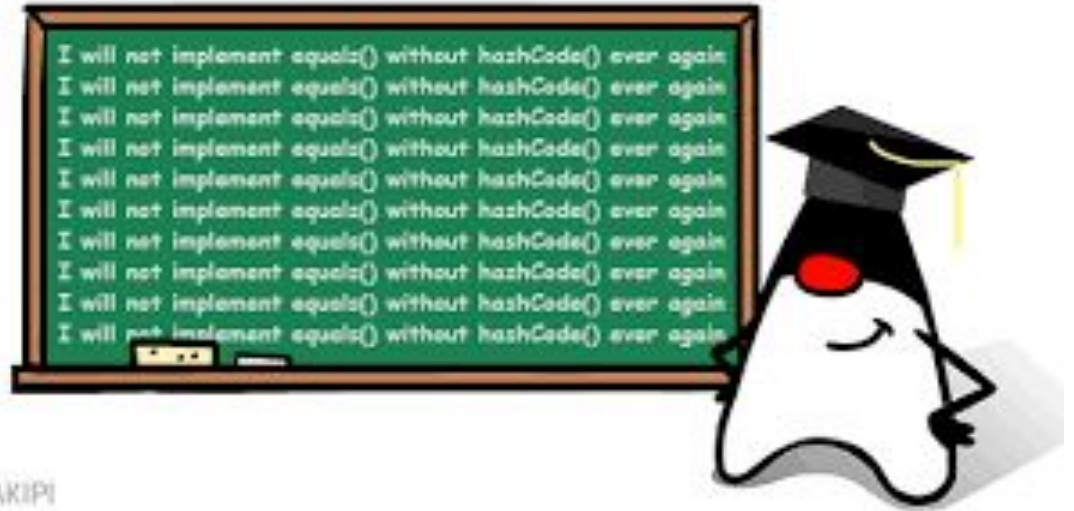
ArrayList

- Basa la implementación de la lista en un array de tamaño variable.
- Un beneficio de usar esta implementación es que las operaciones de acceso a elementos, capacidad y saber si es vacía se realizan de forma eficiente y rápida.
- Todo arraylist tiene una propiedad de capacidad, aunque cuando se añade un elemento esta capacidad puede incrementarse.



Agenda

- ~~Collection API~~
- ~~Interfaz Collection~~
- ~~List~~
- **ArrayList**
 - **Insertar elemento**
 - Lectura de un elemento
 - Eliminar elemento
- Método equals()
- Método hashCode()
- Método equals() - Ejemplo



ArrayList - Insertar elemento (2)

```
public static void main(String [] args) {
```

```
    List<Perro> perros = new ArrayList<>();
```

```
    Perro p1 = new Perro("Boby");
```

```
    Perro p2 = new Perro("Ayudante de Santa");
```

```
    ...
```

```
    Perro p11 = new Perro("Rob")
```

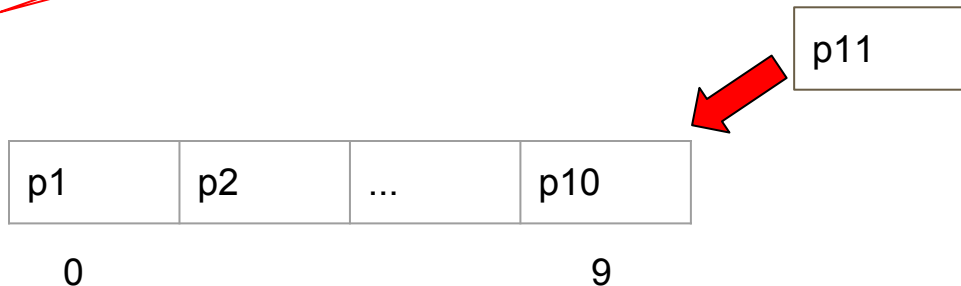
```
    perros.add(p1);
```

```
    ...
```

```
    perros.add(p11);
```

```
}
```

Se valida la capacidad del arreglo y como no es suficiente se crea un nuevo arreglo con el doble de la capacidad y se copia el anterior.



ArrayList - Inserir elemento (3)

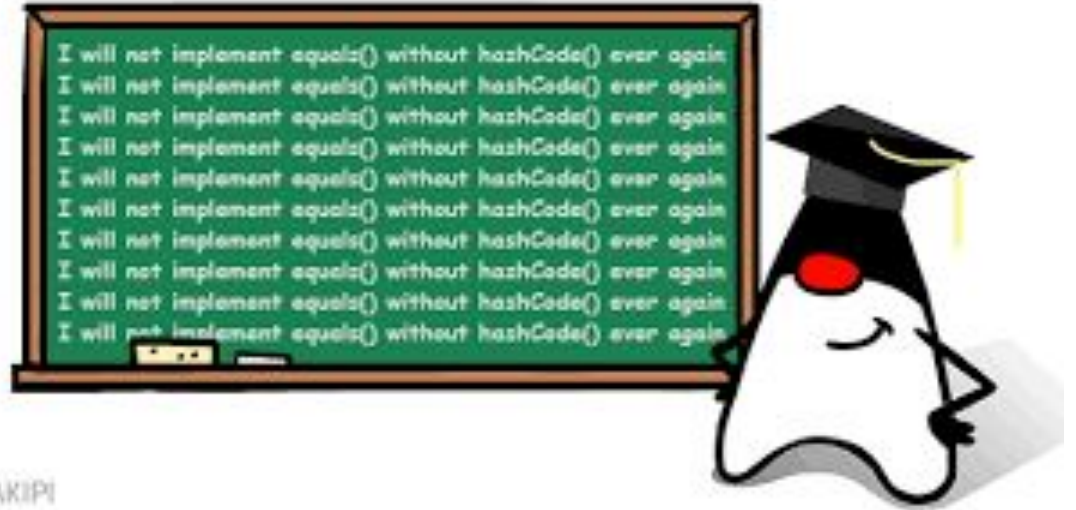
p1	p2	...	p10
----	----	-----	-----



p1	p2	p10	p11								
0									9	10								19

Agenda

- ~~Collection API~~
- ~~Interfaz Collection~~
- ~~List~~
- **ArrayList**
 - Insertar elemento
 - **Lectura de un elemento**
 - Eliminar elemento
- Método equals()
- Método hashCode()
- Método equals() - Ejemplo



ArrayList - Lectura de un elemento

- Como la estructura interna es un arreglo, se accede al elemento a través del índice correspondiente.

```
public static void main(String [] args) {  
  
    List<Perro> perros = new ArrayList<>();  
  
    Perro p1 = new Perro("Boby");  
    Perro p2 = new Perro("Ayudante de Santa");  
  
    perros.add(p1);  
  
    System.out.println(perros.get(0).getNombre());  
}
```

p1	p2
----	----	-----	-----

0

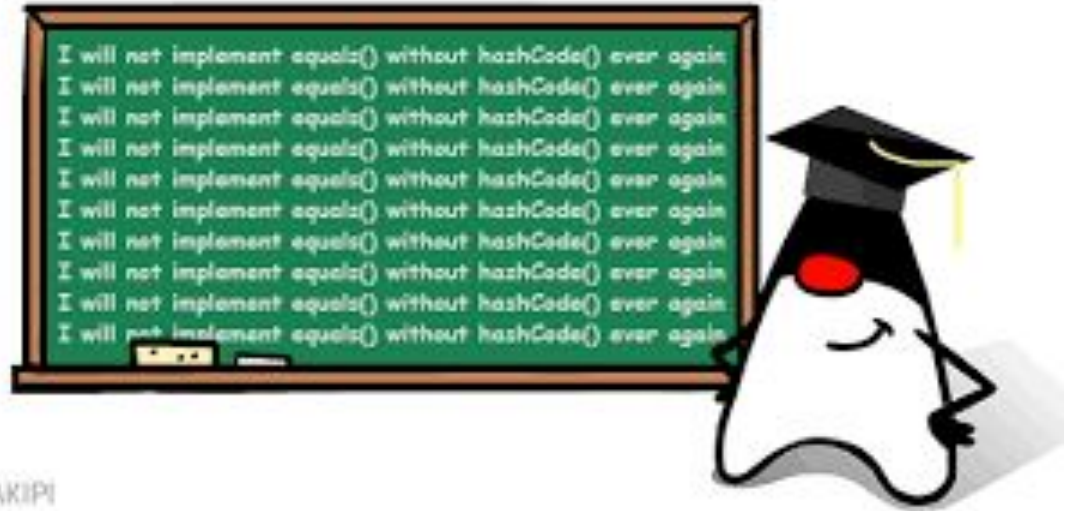
10



Es lo mismo que: p1.getNombre()

Agenda

- ~~Collection API~~
- ~~Interfaz Collection~~
- ~~List~~
- **ArrayList**
 - ~~Insertar elemento~~
 - ~~Lectura de un elemento~~
 - **Eliminar elemento**
- Método equals()
- Método hashCode()
- Método equals() - Ejemplo



ArrayList - Eliminar elemento

1) A través del índice:

```
public static void main(String [] args) {  
  
    List<Perro> perros = new ArrayList<>();  
  
    Perro p1 = new Perro("Boby");  
    Perro p2 = new Perro("Ayudante de Santa");  
  
    perros.remove(0);  
  
}
```

ArrayList - Eliminar elemento (2)



Se deben correr los elementos



ArrayList - Eliminar elemento (3)

2) A través de igualdad:

```
public static void main(String [] args) {  
  
    List<Perro> perros = new ArrayList<>();  
  
    Perro p1 = new Perro("Boby");  
    Perro p2 = new Perro("Ayudante de Santa");  
  
    perros.remove(p1);  
  
}
```

ArrayList - Eliminar elemento (4)



Se verifica que cada elemento sea igual al que se quiere eliminar.
`if (e.equals(p1))`



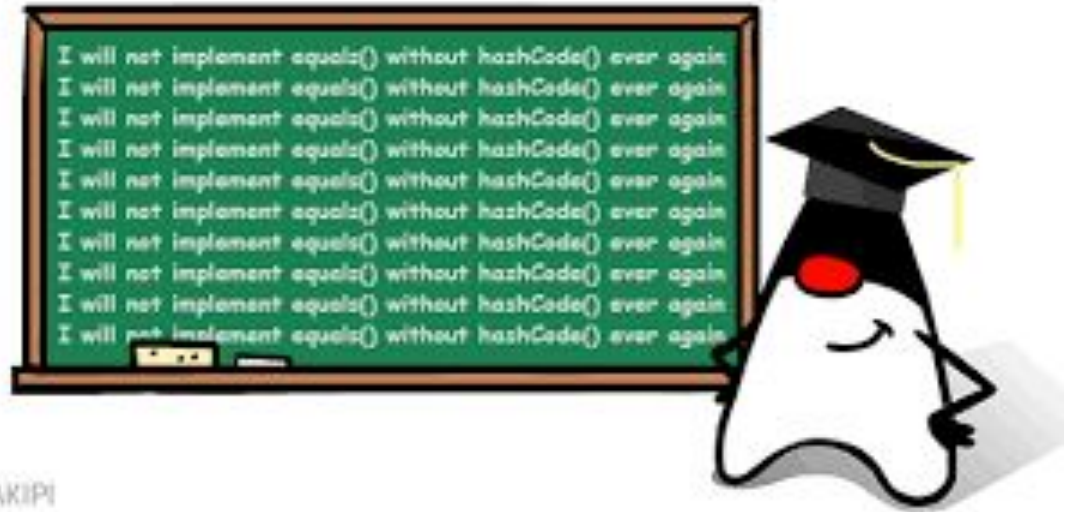
Se deben correr los elementos



IMPORTANTE!!!
Método equals()

Agenda

- ~~Collection API~~
- ~~Interfaz Collection~~
- ~~List~~
- ~~ArrayList~~
 - ~~Insertar elemento~~
 - ~~Lectura de un elemento~~
 - ~~Eliminar elemento~~
- **Método equals()**
- Método hashCode()
- Método equals() - Ejemplo



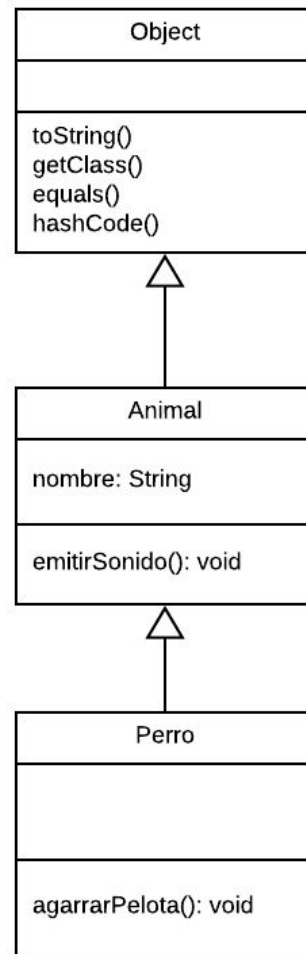
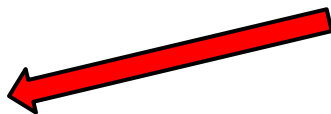
Método equals()

```
public boolean equals(Object obj) {  
    return (this == obj);  
}
```

Se compara la identidad

- Se debe redefinir el método equals() para comparar el contenido de los objetos que queremos evaluar.

- **Igualdad != Identidad**



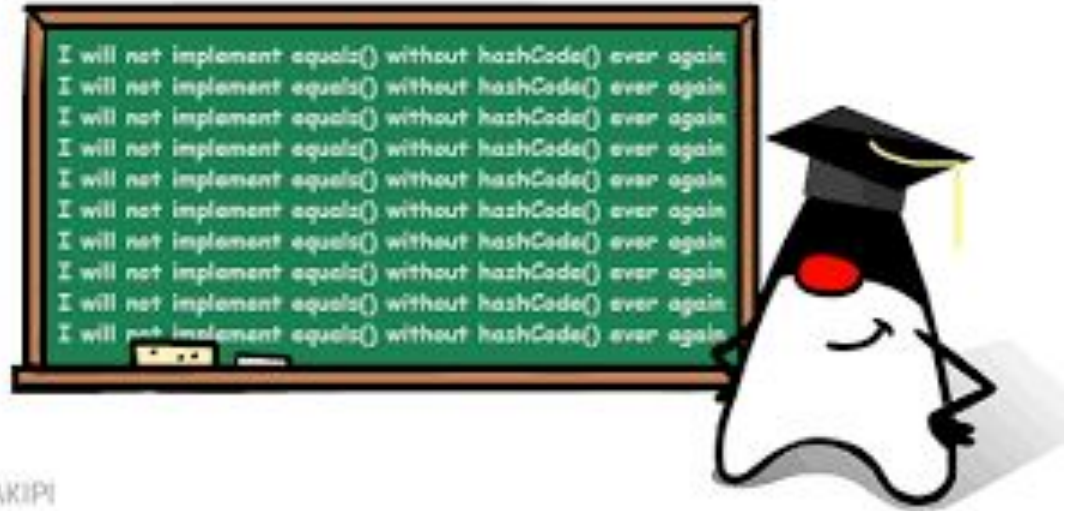
Método equals() (2)

- El método equals() está estrechamente relacionado con la función hashCode()
- Si sobreescribimos equals deberemos de sobrescribir también hashCode.
- hashCode debe cumplir que si dos objetos son iguales, según la función equals, debe de dar el mismo valor para ambos objetos.

Si `a.equals(b)` entonces `a.hashCode() == b.hashCode()`

Agenda

- ~~Collection API~~
- ~~Interfaz Collection~~
- ~~List~~
- ~~ArrayList~~
 - ~~Insertar elemento~~
 - ~~Lectura de un elemento~~
 - ~~Eliminar elemento~~
- ~~Método equals()~~
- **Método hashCode()**
- Método equals() - Ejemplo



Método hashCode()

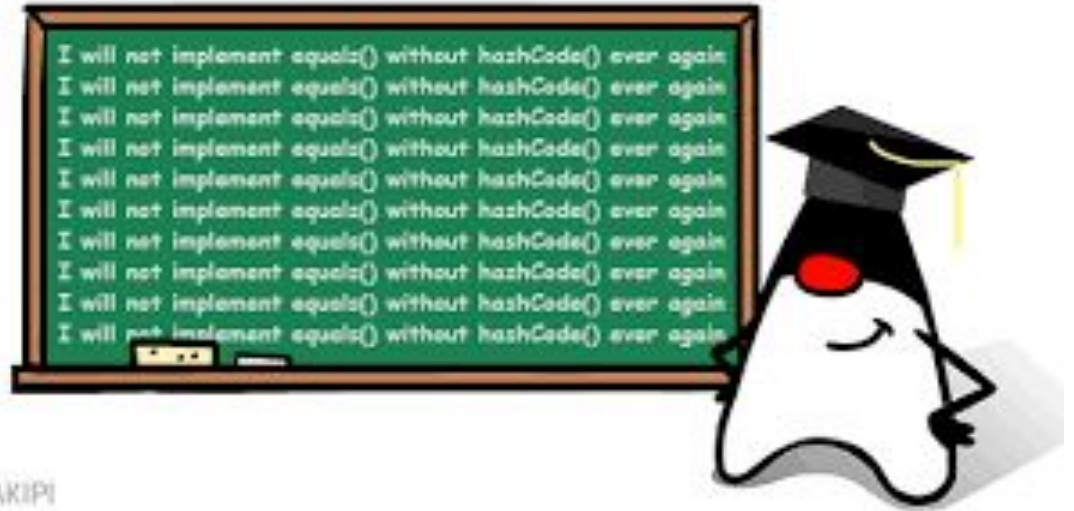
```
public int hashCode()
```

- Este método viene a complementar al método equals y sirve para comparar objetos de una forma más rápida en estructuras Hash ya que únicamente nos devuelve un número entero.

“To be continued...”

Agenda

- ~~Collection API~~
- ~~Interfaz Collection~~
- ~~List~~
- ~~ArrayList~~
 - ~~Insertar elemento~~
 - ~~Lectura de un elemento~~
 - ~~Eliminar elemento~~
- ~~Método equals()~~
- ~~Método hashCode()~~
- **Método equals() - Ejemplo**



TAKIPI

Método equals() - Ejemplo

```
public class Perro extends Animal {  
  
    ...  
    @Override  
    public boolean equals(Object obj) {  
        if (this == obj)  
            return true;  
        if (obj == null)  
            return false;  
        if (getClass() != obj.getClass())  
            return false;  
        Perro otroPerro = (Perro) obj;  
        if (nombre.equals(otroPerro.equals()))  
            return true;  
        return false;  
    }  
}
```

Bibliografía oficial

- <https://docs.oracle.com/javase/tutorial/collections/interfaces/collection.html>
- <https://docs.oracle.com/javase/tutorial/collections/interfaces/list.html>
- <https://docs.oracle.com/javase/8/docs/api/java/util/ArrayList.html>
- <https://docs.oracle.com/javase/tutorial/java/landl/objectclass.html>