Colecciones Simples: ArrayList

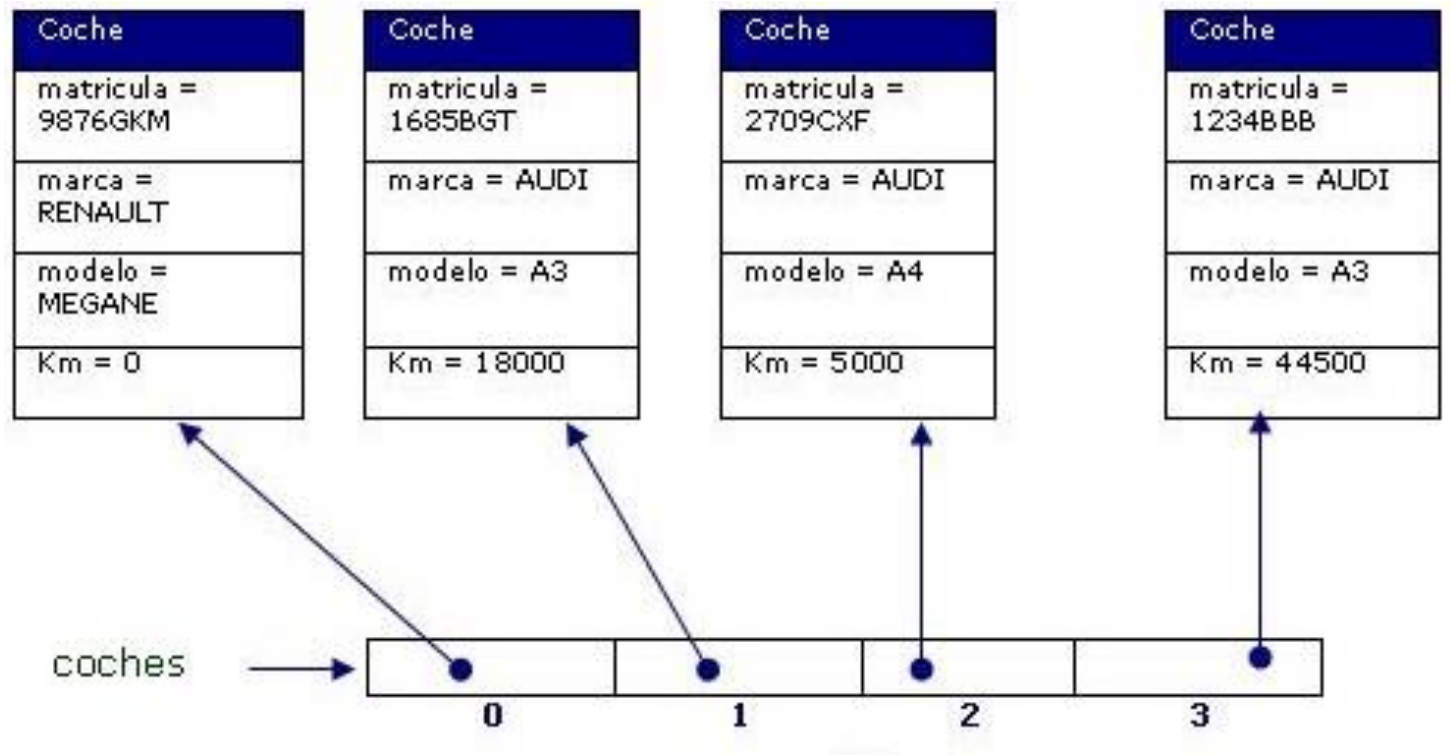
¿Cuál es la diferencia con un arreglo?

```
String[] example = new String[5]
```

```
ArrayList<String> example = new ArrayList<String>();
```

Colecciones Simples: ArrayList

- Tamaño dinámico.
- **Orden:** Dado por el índice de la lista.
- Permite elementos nulos.
- Puede incluir elementos repetidos



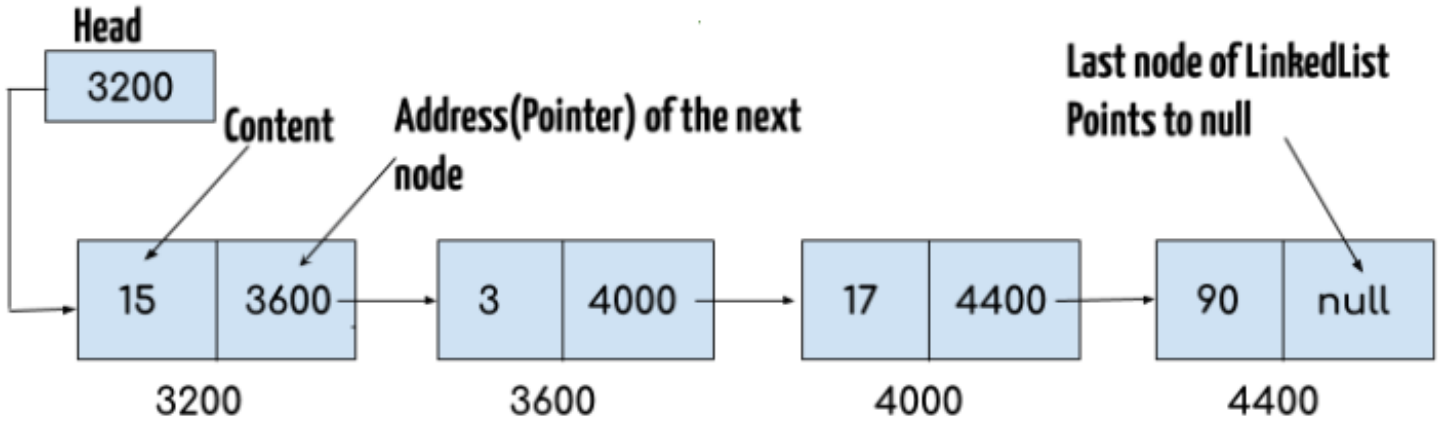
Ejemplo tomado de: http://puntocomnoesunlenguaje.blogspot.com/2013/02/arraylist-de-objetos-en-java.html

Colecciones Simples: LinkedList

- Tamaño dinámico.
- Lista doblemente enlazada.
- **Orden:** De inserción.
- **Comportamiento:** Cola
- Permite elementos nulos.
- Puede incluir elementos repetidos.
- Cada element es un nodo.
- Operaciones sobre el primer y ultimo nodo.

Cada nodo almacena:

- El contenido
- Dirección / Referencia al nodo siguiente



Tomado de: https://beginnersbook.com/2013/12/linkedlist-in-java-with-example/

Colecciones Simples: HashSet

- **Orden:** No tiene. Se basa en un Hashcode
- Permite elementos nulos.
- No se permiten valores repetidos.

Colecciones Simples: TreeSet

- **Orden:** Orden Natural o Comparador
- **NO** Permite elementos nulos.
- No se permiten valores repetidos.

Orden Natural y Comparator

El orden natural dependerá del tipo de dato los elementos.

Ejemplo: Si los elementos son de tipo String -> Código ASCII

```
TreeSet<String> al=new TreeSet<String>();
al.add("Ravi");
al.add("Vijay");
al.add("Ravi");
al.add("Ajay");
```

Orden Natural y Comparator

Comparator

- Interfaz
- 2 métodos: *compare / equals*
- Permite dar un criterio para ordenar elementos.

```
class Student{
int rollno;
String name;
int age;
Student(int rollno,String name,int age){
this.rollno=rollno;
this.name=name;
this.age=age;
}
}
```

Orden Natural y Comparator

```
class Student{
int rollno;
String name;
int age;
Student(int rollno,String name,int age){
this.rollno=rollno;
this.name=name;
this.age=age;
}
}
```

```
import java.util.*;
class AgeComparator implements Comparator{
public int compare(Object o1,Object o2){
Student s1=(Student)o1;
Student s2=(Student)o2;

if(s1.age==s2.age)
return 0;
else if(s1.age>s2.age)
return 1;
else
return -1;
}
}
```

¿Si lo queremos ordenar por nombre?

Orden Natural y Comparator

```
import java.util.*;
class NameComparator implements Comparator{
public int compare(Object o1,Object o2){
Student s1=(Student)o1;
Student s2=(Student)o2;

return s1.name.compareTo(s2.name);
}
}
```

Colecciones con Clave: HashMap

- **Orden:** No tiene.
- Valores basados en una llave.
- No se permiten **llaves** repetidas.
- Puede tener una llave nula.
- Puede tener valores nulos.

Colecciones con Clave: TreeMap

- **Orden:** Orden Natural o Comparador
- Valores basados en una llave.
- No se permiten **llaves** repetidas.
- **No** puede tener una llave nula.
- Puede tener valores nulos.