ESCUELA IT

Instructor Rafael Benedettelli

Curso Java Profesional

ESCUELA IT

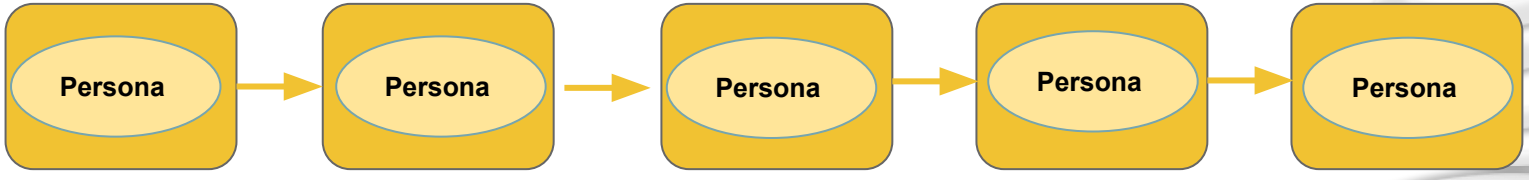
git clone

https://github.com/Escuelalt/Java-Profesional

ESCUELA IT

Colecciones

- Listas - Set
- Mapas
- Colas - Pilas



ESCUELA IT

- Dimensión variable
- Uso de API
- Potencia
- Opciones Avanzadas

Arrays

Colecciones

ESCUELA IT

Wrapper

ArrayList<String>

Object[] arrelgo = new Object[INIT_CAPACITY];

“curso”	“java”	“prof”	“esio”	“nal”	“Escu”	“la”	“IT”	“2020”

ESCUELA IT

```
ArrayList<float> listaDeNumeros = new ArrayList<>();
ArrayList<int> listaDeNumeros = new ArrayList<>();
ArrayList<long> listaDeNumeros = new ArrayList<>();
ArrayList<byte> listaDeNumeros = new ArrayList<>();
ArrayList<boolean> listaDeBooleanos = new ArrayList<>();
ArrayList<char> listaDeLetras = new ArrayList<>();
```

No se pueden trabajar con primitivos

ESCUELA IT

Wrappers

Primitivos		Objetos
byte	→	Byte
short	→	Short
int	→	Integer
long	→	Long
float	→	Float
double	→	Double
boolean	→	Boolean
char	→	Character

ESCUELA IT

Wrappers

```
ArrayList<Byte> listaDeNumeros = new ArrayList<>();
ArrayList<Short> listaDeNumeros = new ArrayList<>();
ArrayList<Integer> listaDeNumeros = new ArrayList<>();
ArrayList<Long> listaDeNumeros = new ArrayList<>();
ArrayList<Float> listaDeNumeros = new ArrayList<>();
ArrayList<Double> listaDeNumeros = new ArrayList<>();
ArrayList<Boolean> listaDeBooleanos = new ArrayList<>();
ArrayList<Chararcter> listaDeLetras = new ArrayList<>();
```

ESCUELA IT

Wrappers

Byte	→	**parseB**yte(“1”);
Short	→	**parseS**hort(“2”);
Integer	→	**parseI**nt(“3”);
Long	→	**parseL**ong(“4”);
Float	→	**parseF**loat(“4.2”);
Double	→	**parseD**ouble(“5.23”);
Boolean	→	**parseB**oolean(“true”);
Character	→	valueOf(‘a’);

ESCUELA IT

Wrappers

```
public Integer suma(Integer x,Integer y){
      return x+y;
}
```

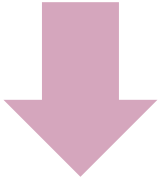
ESCUELA IT

Wrappers

```
int resultado= suma(2,4);
```

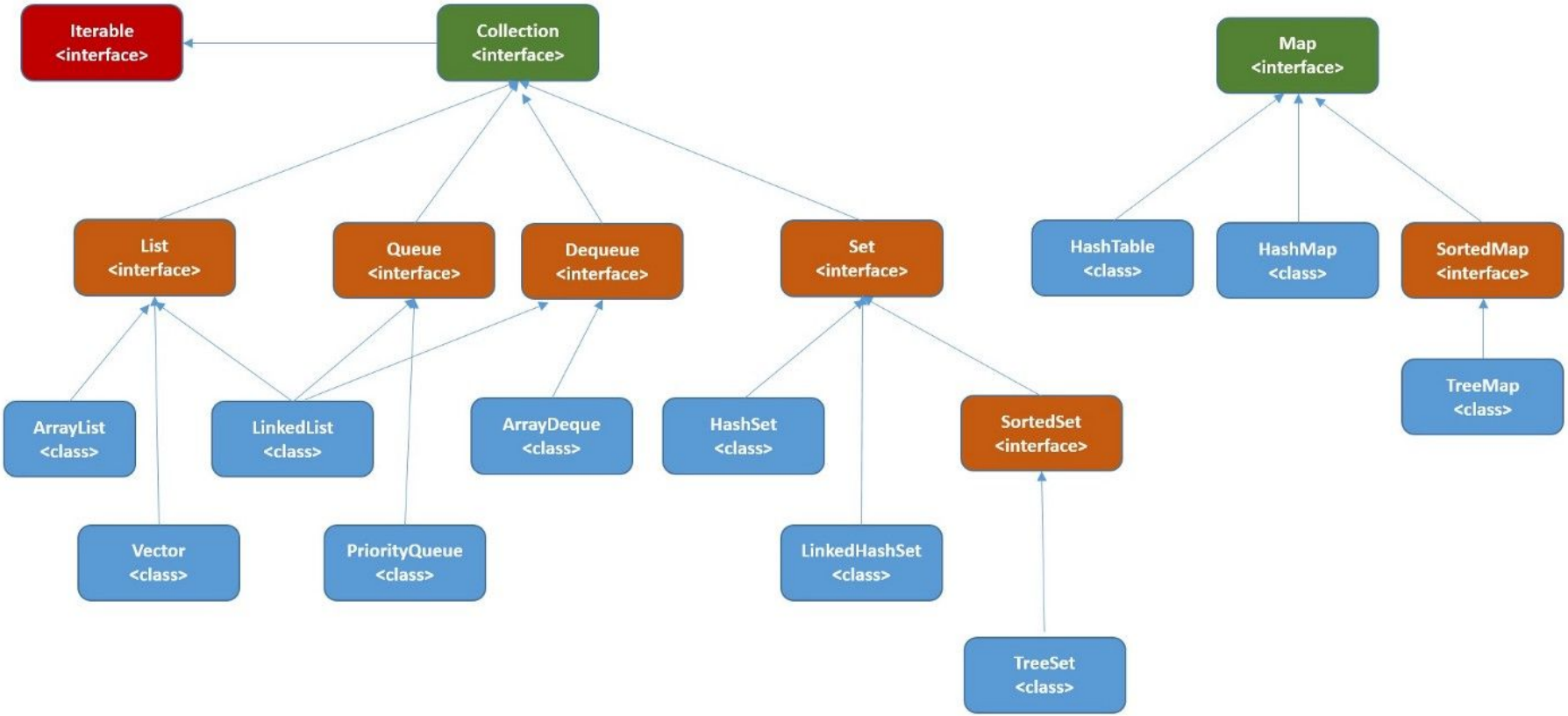
OUTBOXING

BOXING



```
public Integer suma(Integer x,Integer y){
      return x+y;
}
```

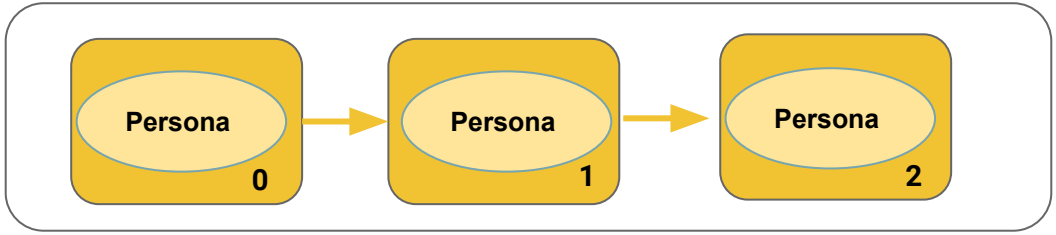
Collection Framework Hierarchy



ESCUELA IT

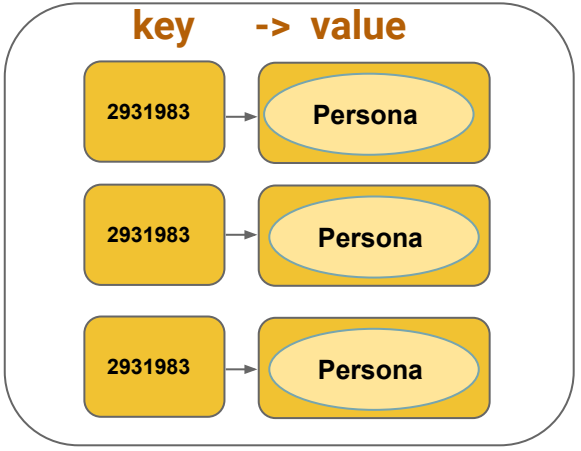
Listas

Colección de datos tipo arreglo



Mapas

Mapa tipo clave-Valor

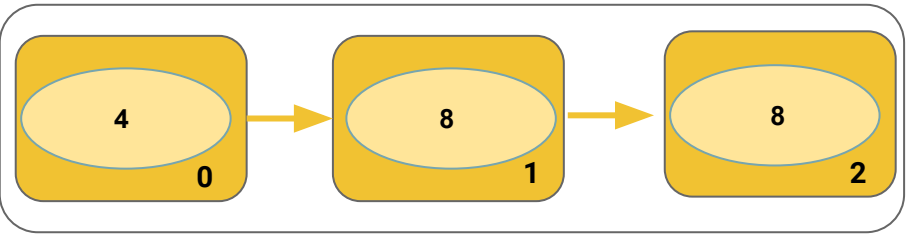


ESCUELA IT

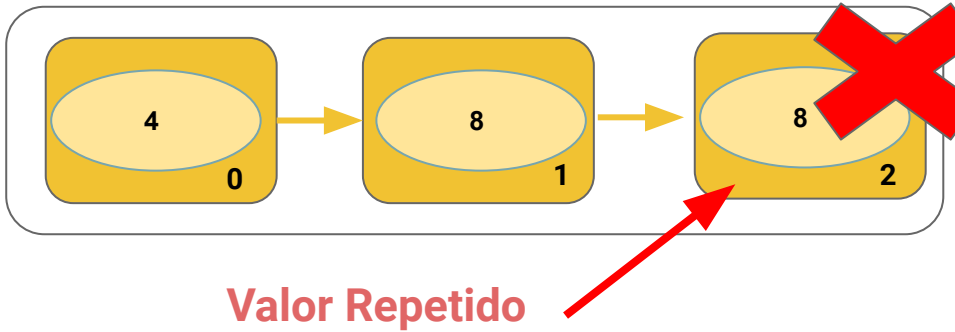
List

Set

Acepta valores repetidos



No Acepta valores repetidos



ESCUELA IT

	List	Set	Map
Básico	ArrayList	HashSet	HashMap
		Inserción única	

ESCUELA IT

Orden natural de los elementos

ESCUELA IT

Colecciones con Orden "Tree"

	List	Set	Map
Básico	ArrayList	HashSet	HashMap
Orden	———————	TreeSet	TreeMap

ESCUELA IT

Ranking de estadios por Aforo

ESCUELA IT

Colas / Pilas

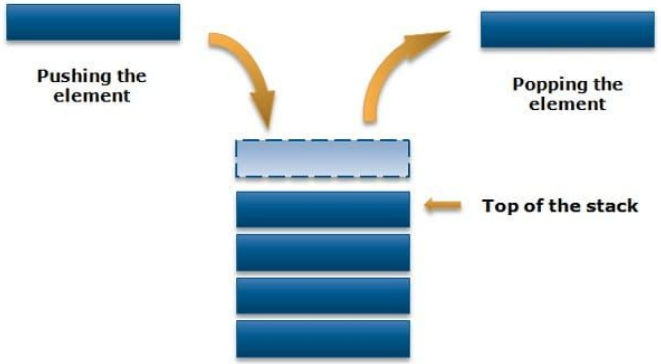
Queue

FIFO: Primero en llegar → Primero en Atender

ESCUELA IT

Colas / Pilas

Stack

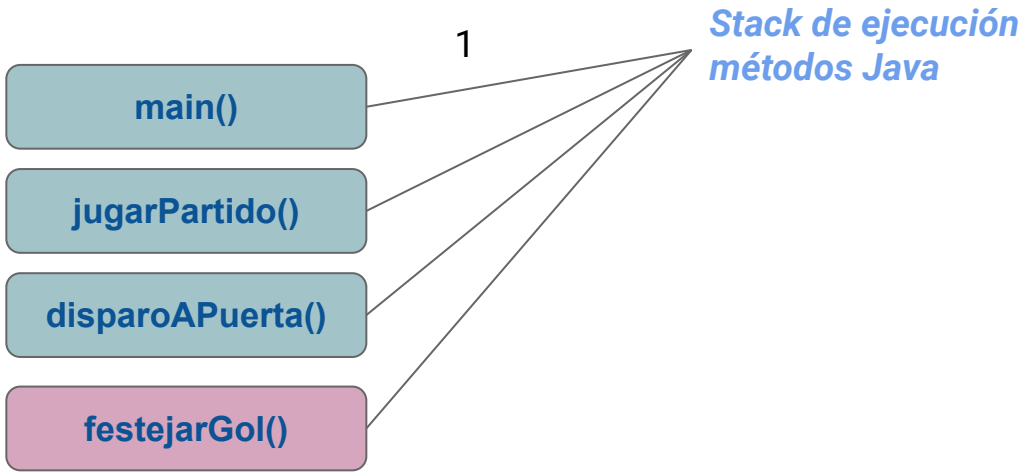


LIFO: Ultimo en llegar
Primero en llegar

Primero en Salir
Ultimo en salir

ESCUELA IT

Colas / Pilas

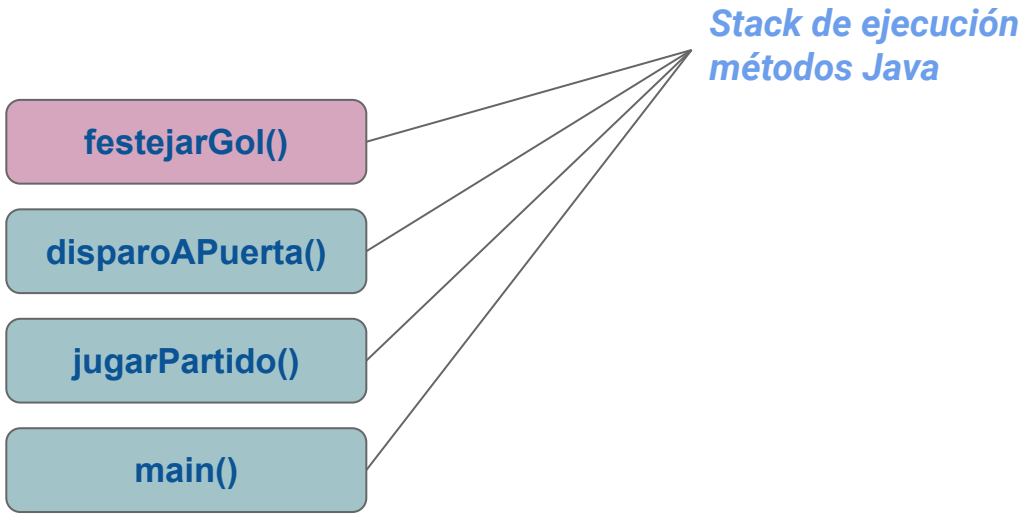


LIFO: Ultimo en llegar
Primero en llegar

Primero en Salir
Ultimo en salir

ESCUELA IT

Colas / Pilas

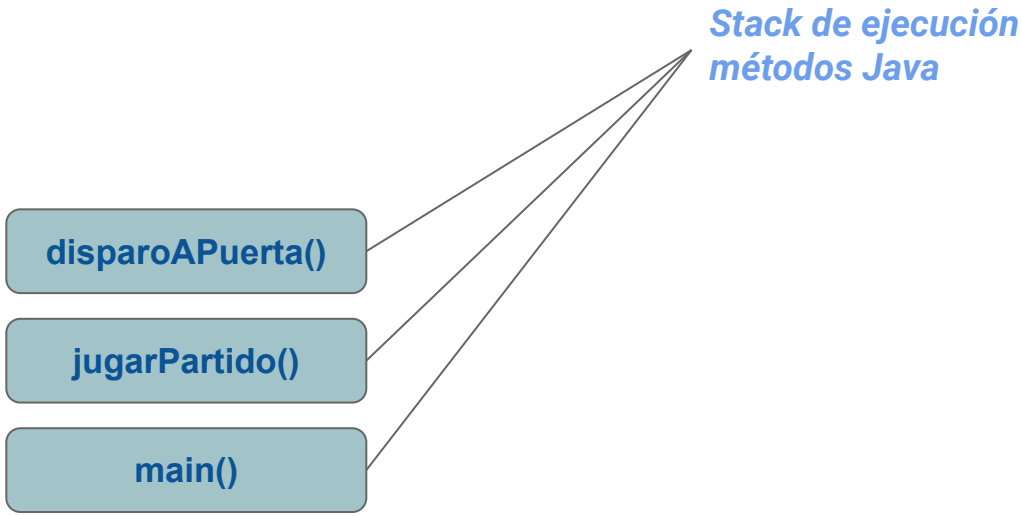


LIFO: Ultimo en llegar
Primero en llegar

Primero en Salir
Ultimo en salir

ESCUELA IT

Colas / Pilas

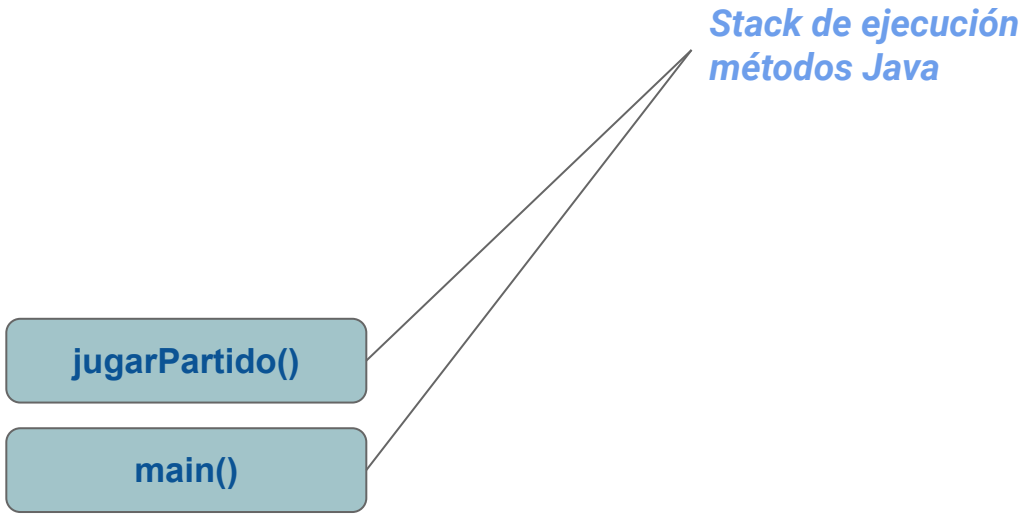


LIFO: Ultimo en llegar
Primero en llegar

Primero en Salir
Ultimo en salir

ESCUELA IT

Colas / Pilas

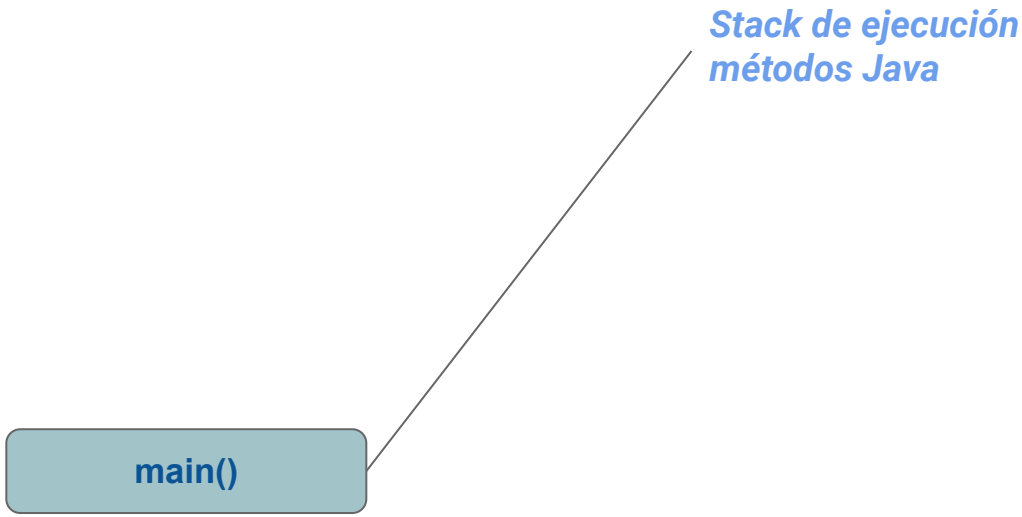


LIFO: Ultimo en llegar
Primero en llegar

Primero en Salir
Ultimo en salir

ESCUELA IT

Colas / Pilas



LIFO: Ultimo en llegar
Primero en llegar

Primero en Salir
Ultimo en salir

ESCUELA IT

Colas / Pilas

Atención de jugadores lesionados

ESCUELA IT

Colecciones con Orden "Tree"

	List	Set	Map
Básico	ArrayList	HashSet	HashMap
Colas/Pilas	LinkedList	LinkedHashSet	LinkedHashMap
Orden	———	TreeSet	TreeMap

ESCUELA IT

Resumen

Hash → **Unicidad**

Linked → **Colas/Pilas (Queue/Stack)**

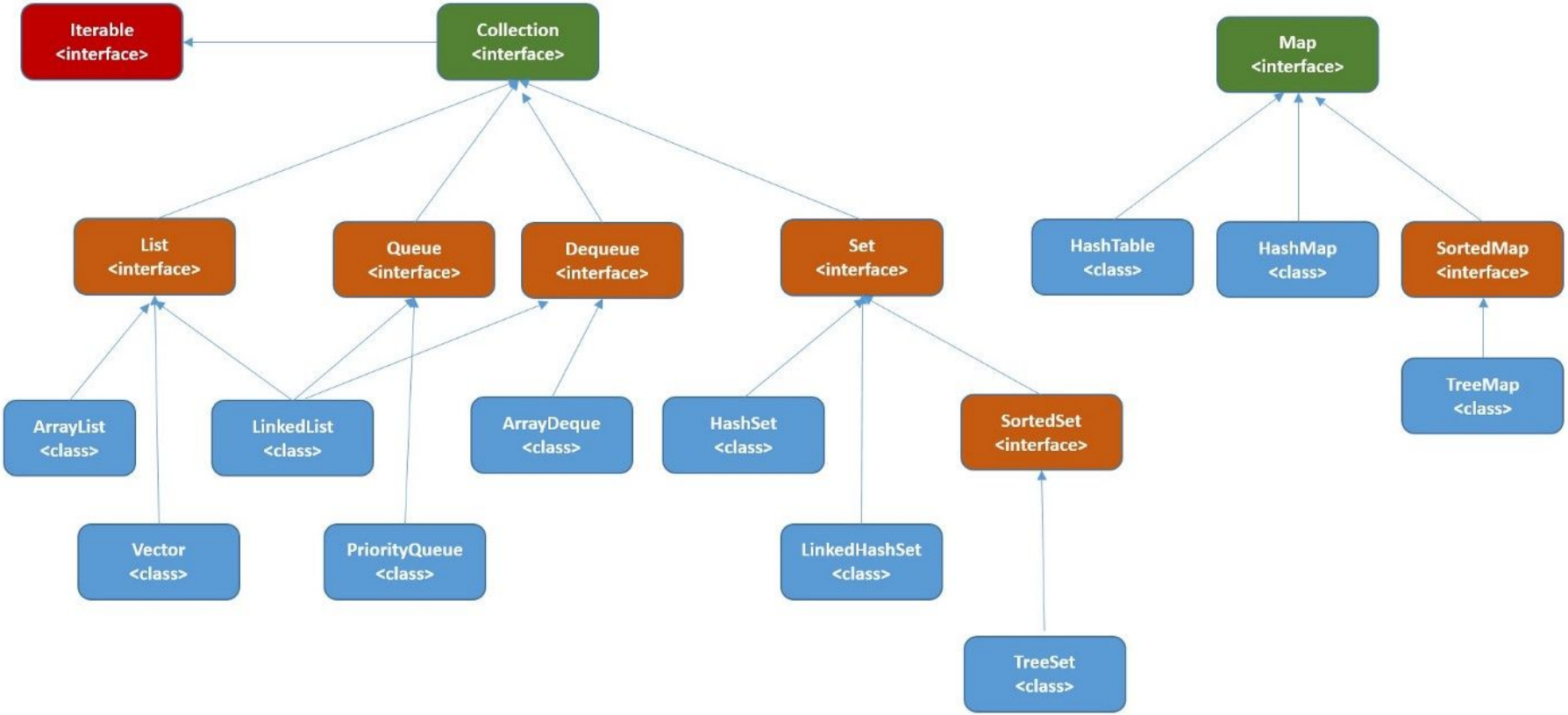
Tree → **Orden Natural**

Map  **Mapa = clave/valor**

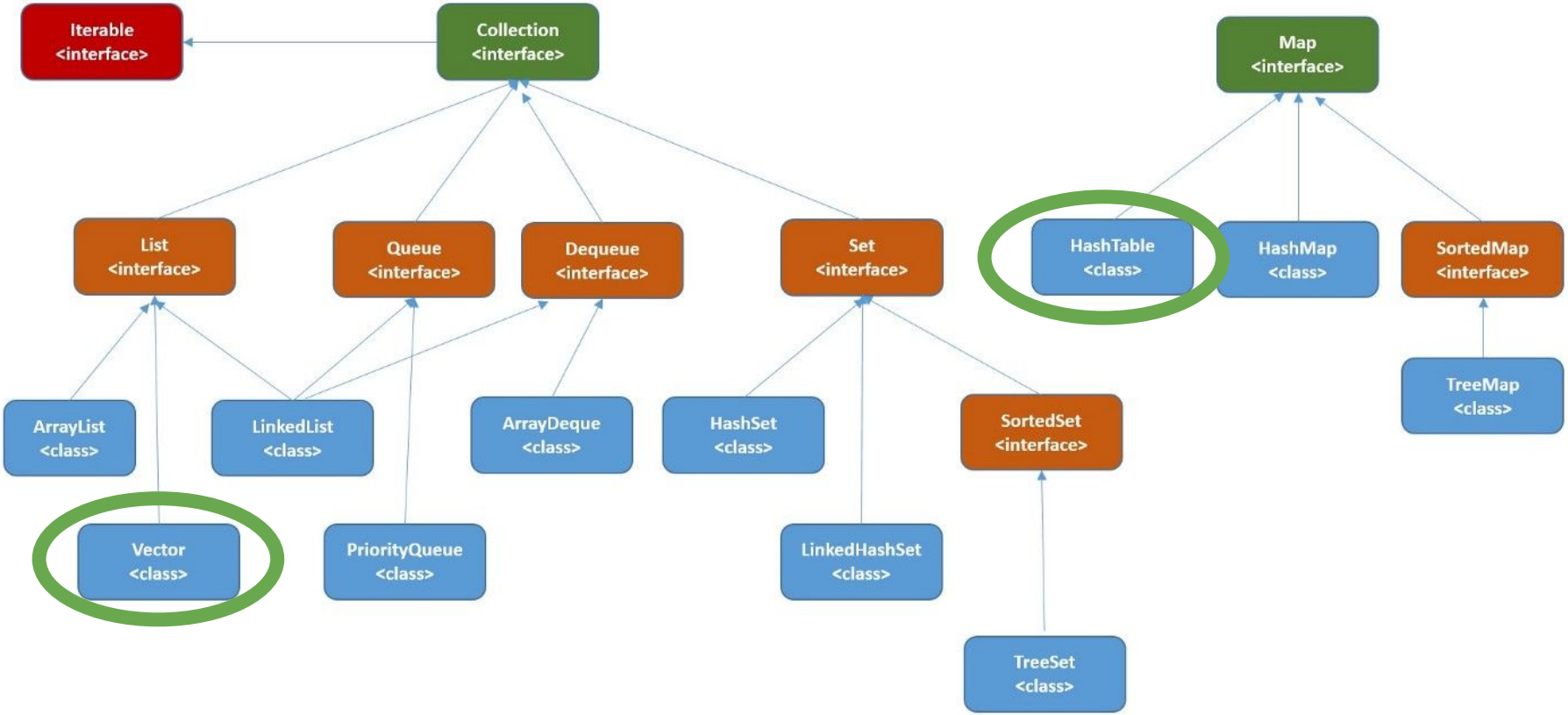
Set  **Conjunto = Unicidad**

List  **Lista = Admite repetidos**

Collection Framework Hierarchy



Collection Framework Hierarchy



Streams

Streams

- Realizar operaciones complejas de la menera mas simple posible
- Sirve para realizar operaciones bien conocidas sobre colecciones
- Mejora la expresion del codigo haciendolo mas sintáctico
- Utiliza DSL, Lambdas y Optional<>

Expresiones Lambdas

```
int suma(int x,int y){
   return x + y;
}
```

```
(x,y) -> x+y;
```

- Identificable
- Reutilizable
- Recurisivdad
- Modificadores de acceso

- Anonima
- Sintetizada
- Descartable
- Currificar