1. **¿Qué es una relación involutiva en java?**

Una relación involutiva en Java es una relación en la que un objeto está asociado consigo mismo.

1. **De un ejemplo de relación involutiva.**

Un ejemplo de relación involutiva podría ser una clase Persona que tiene una relación consigo misma para representar la relación "padre de". Es decir, una persona puede ser el padre de otra persona, y al mismo tiempo, puede ser padre de sí misma.

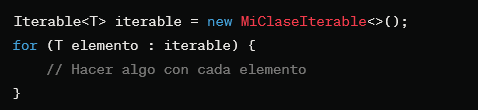
1. **¿Qué métodos comunes se encuentran en la interfaz Collection y qué funcionalidades proporcionan?**

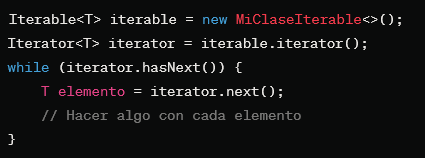
* **add(E e)**: Agrega un elemento a la colección si es posible. Por ejemplo, en una lista, este método añadirá un elemento al final de la lista.
* **remove(Object o)**: Elimina la primera ocurrencia del elemento especificado de la colección, si está presente.
* **contains(Object o)**: Devuelve true si la colección contiene el elemento especificado.
* **isEmpty()**: Devuelve true si la colección no contiene elementos.
* **size()**: Devuelve el número de elementos en la colección.
* **clear()**: Elimina todos los elementos de la colección.
* **iterator()**: Devuelve un iterador sobre los elementos de la colección. Permite recorrer la colección de forma secuencial.
* **toArray()**: Devuelve un array que contiene todos los elementos de la colección.
* **addAll(Collection<? extends E> c)**: Agrega todos los elementos de la colección especificada a esta colección.
* **removeAll(Collection<?> c)**: Elimina de esta colección todos los elementos que están contenidos en la colección especificada.
* **retainAll(Collection<?> c)**: Retiene solo los elementos de esta colección que están contenidos en la colección especificada.
* **containsAll(Collection<?> c)**: Devuelve true si esta colección contiene todos los elementos de la colección especificada.

1. **Explica el propósito de la interfaz Iterable en Java y cómo se utiliza.**

Proporciona una manera estándar de hacer que los objetos sean iterables, lo que facilita la iteración a través de sus elementos utilizando bucles for-each o iteradores explícitos.

Define un método **iterator**, que devuelve un objeto **iterator** que se utiliza para recorrer los elementos del objeto.





1. **¿Qué ventajas ofrece el uso de la interfaz Iterable en comparación con simplemente iterar sobreuna colección utilizando un bucle for estándar?**

El uso de la interfaz Iterable proporciona una **abstracción** más clara, una **mayor flexibilidad** en la implementación, compatibilidad con el **bucle for-each**, mejor **encapsulación** y la posibilidad de **reutilizar** el código de iteración.