Implementations of Set interface don’t allow duplicate elements. We can remove the duplicate elements in an integer array by adding the individual elements into different implementations of Set interface – HashSet, LinkedHashSet and TreeSet

**Method 1:**

Hashset has the fastest performance when compared to the other implementations of the Set interface. It removes duplicates but it doesn’t maintain the insertion order. It internally uses HashMap to store its elements.

The time complexity of basic methods to add or remove elements is O(1)

We can see HashSet in use cases where we need NOT maintain the insertion order

**Method 2:**

LinkedHashSet has slightly less performance than Hashset because it maintains a doubly linked list to maintain the insertion order

The time complexity of basic methods to add or remove elements is O(1)

We can see HashSet in use cases where we need to maintain the insertion order

It internally uses LinkedHashMap to store its elements

**Method 3:**

TreeSet sorts the elements in its natural ordering. It also comes with a comparator using which we can provide our own sorting logic for custom objects

Treeset has various methods like first(), last(), headset(), tailSet() which we can use to manipulate the ordered collection

The time complexity of basic methods to add or remove elements is O(log(n))

It internally uses TreeMap to store its elements