1. Collection Interface

The Collection interface is the interface that is implemented by all the classes in the collection framework. It declares the methods that every collection will have. In other words, we can say that the Collection interface builds the foundation on which the collection framework depends.

Methods of the Collection Interface

This interface contains various methods which can be directly used by all the collections which implement this interface. They are:

`Method`	Description
`boolean add(E e)`	Ensures that this collection contains the specified element (optional operation).
`boolean addAll(Collection <? extends E> c)`	Adds all of the elements in the specified collection to this collection (optional operation).
`void clear()`	Removes all of the elements from this collection (optional operation).

`boolean contains(Object o)`	Returns true if this collection contains the specified element.
`boolean containsAll(Collection<?> c)`	Returns true if this collection contains all of the elements in the specified collection.
`boolean equals(Object o)`	Compares the specified object with this collection for equality.
`int hashCode()`	Returns the hash code value for this collection.
`boolean isEmpty()`	Returns true if this collection contains no elements.
`Iterator<E> iterator()`	Returns an iterator over the elements in this collection.
`default Stream<E> parallelStream()`	Returns a possibly parallel Stream with this collection as its source.

`boolean remove(Object o)`	Removes a single instance of this collection's specified element if it is present (optional operation).
`boolean removeAll(Collection c)`	Removes all of this collection's elements that are also contained in the specified collection (optional operation).
`default boolean removeIf(Predicate<? super E> filter)`	This method is used to remove all the elements of this collection that satisfy the given predicate.
`boolean retainAll(Collection<?> c)`	Retains only the elements in this collection that are contained in the specified collection (optional operation).
`int size()`	Returns the number of elements in this collection.
`default Spliterator<E> spliterator()`	Creates a Spliterator over the elements in this collection.

`default Stream<E> stream()`	This method is used to return a sequential Stream with this collection as its source.
`Object[] toArray()`	Returns an array containing all of the elements in this collection.