Absolutely! Here's a **comprehensive list of all the methods** available in Java's `ArrayList` class, formatted as a table for clarity.

> ✅ This includes **all methods** inherited from `List`, `Collection`, `Iterable`, `Cloneable`, and `Serializable` interfaces as well as those defined in `ArrayList` itself.

✅ Complete List of `ArrayList` Methods

Method Signature	Description
`boolean add(E e)`	Appends the specified element to the end.
`void add(int index, E element)`	Inserts element at the specified index.
`boolean addAll(Collection<? extends E> c)`	Appends all elements from the specified collection.
`boolean addAll(int index, Collection<? extends E> c)`	Inserts all elements at the specified index.
`void clear()`	Removes all elements from the list.
`Object clone()`	Returns a shallow copy of this `ArrayList`.
`boolean contains(Object o)`	Checks if the list contains the specified element.

`void ensureCapacity(int minCapacity)`	Increases the capacity if necessary.
`E get(int index)`	Returns element at the specified index.
`int indexOf(Object o)`	Returns the first index of the specified element, or -1.
`int lastIndexOf(Object o)`	Returns the last index of the specified element, or -1.
`E remove(int index)`	Removes and returns element at the specified index.
`boolean remove(Object o)`	Removes the first occurrence of the specified element.
`boolean removeAll(Collection<?> c)`	Removes all elements found in the specified collection.
`boolean retainAll(Collection<?> c)`	Retains only the elements found in the specified collection.
`E set(int index, E element)`	Replaces element at index with specified element.
`int size()`	Returns number of elements in the list.

`boolean isEmpty()`	Returns `true` if the list contains no elements.
`Object[] toArray()`	Returns an array containing all elements.
`<T> T[] toArray(T[] a)`	Returns an array containing elements of the specified type.
`boolean containsAll(Collection<?> c)`	Checks if list contains all elements of specified collection.
`Iterator<E> iterator()`	Returns an iterator over the elements.
`ListIterator<E> listIterator()`	Returns a list iterator.
`ListIterator<E> listIterator(int index)`	Returns a list iterator starting from specified index.
`List<E> subList(int fromIndex, int toIndex)`	Returns a view of the portion of this list between indexes.
`void trimToSize()`	Trims capacity to current size.
`Spliterator<E> spliterator()`	Creates a spliterator for this list.

`void forEach(Consumer<? super E> action)`	Performs the given action for each element.
`boolean removeIf(Predicate<? super E> filter)`	Removes elements that satisfy the filter condition.
`void replaceAll(UnaryOperator<E> operator)`	Replaces each element using the operator.
`void sort(Comparator<? super E> c)`	Sorts the list using the comparator.
`boolean equals(Object o)`	Compares the list with another object for equality.
`int hashCode()`	Returns hash code for the list.
`String toString()`	Returns a string representation of the list.

Would you like code examples for each method or maybe a downloadable notebook for practice?