Similarities between ArrayList<E> and LinkedList<E>

Previously we saw [differences between ArrayList<E> and LinkedList<E>.](http://data-structure-learning.blogspot.com/2015/05/difference-between-arraylist-and_28.html) Click [here to read when to use ArrayList<E> and when to use LinkedList<E>](http://data-structure-learning.blogspot.com/2015/05/when-to-use-arraylist-and-linkedlist.html)

1. Insertion Order?
   1. The [List<E> interface](http://data-structure-learning.blogspot.com/2015/05/java-collections-part-5list-interface.html)’s add(E e) method which defines a contract that specified elements must be appended to the list. Hence insertion order is preserved.
2. Cloning?
   1. clone() operations of ArrayList<E> as well as LinkedList<E> returns the shallow copy of elements. This means elements are not itself copied or backup.
3. Synchronization?
   1. ArrayList<E> and LinkedList<E> both of them are non-synchronized collection. They can be synchronized by using Collections.synchronizedList(..) method of Collections class. All methods are synchronized except [iterator()](http://data-structure-learning.blogspot.com/2015/05/java-collections-part-8-iterator.html), [listIterator()](http://data-structure-learning.blogspot.com/2015/05/java-collections-part-9-listiterator.html) and listIterator(int index).
4. Iterators?
   1. The iterators used in ArrayList<E> and LinkedList<E> are [fail fast](http://data-structure-learning.blogspot.com/2015/05/fail-fast-iterator.html). Fail fast iterators throws [ConcurrentModificationException](http://data-structure-learning.blogspot.com/2015/05/concurrentmodificationexception.html).
5. Implementations of?
   1. Both ArrayList<E> and LinkedList<E> are implementations of [List<E> interface](http://data-structure-learning.blogspot.com/2015/05/java-collections-part-5list-interface.html).