```java
public interface ListInterface<T>
{
    public void add(T newEntry);
    /*Adds a new entry to end of the list.
    Entries currently in the list are unaffected.
    The list's size is increase by 1.

    @param newEntry The object to be added as a new list entry
     */

    public void add(int newPosition, T newEntry);
    /*Adds a new entry to a specific position within the list.
    Entries currently at position are moved up by one.
    List size increased by 1.

    @param newPosition An int that specifies the index position of
    new entry.

    @param newEntry The object to be added as a new list entry

    @throws IndexOutOfBoundsException if either new position < 1
    or newPosition > getLength() + 1
     */

    public T remove(int position);
    /* Removes entry at given position from the list.
    Entries originally at higher positions all moved down by one
    and list size shrinks by 1.

    @param newPosition An int that specifies the index position of
    entry to be removed.

    @return Removed entry at given position

    @throws IndexOutOfBoundsException if either position to be removed
      < 1 or position to be removed > getLength()
     */

    public void clear();
    /* Removes all entries from the list

     */

    public T replace(int position, T newEntry);
    /* Replaces the entry at the given position in this list

    @param newPosition An int that specifies the index position of
    entry to be removed.

    @return Removed entry at given position

     @throws IndexOutOfBoundsException if either position to be replaced
      < 1 or position to be replaced > getLength()
     */

    public T getEntry(int position);
    /* Retrieved the entry at the given position in this list

    @param newPosition An int that specifies the index position of
    entry to be retrieved.

    @return Removed entry at given position

     @throws IndexOutOfBoundsException if either position to be retrieved
      < 1 or position to be replaced > getLength()
     */

    public T[] toArray();
    /* Retrieves all entries in the list in the order which they appear.

    @return A newly allocated array of all the entries in the list.
    If the list is empty, returned array is empty.
     */

    public int getLength();
    /*
    @return the integer length of listArray
     */

    public boolean isEmpty();
    /* Checks whether list is empty.

    @return true if getLength == 0, otherwise false
     */
}

```