ArrayList API

Java ArrayLists are similar to Java's built-in arrays, but are actually more similar to Python's lists, because ArrayLists may grow and shrink in size at any time, whereas the size of a built-in array, once it is first created, can never change.

All the elements in a single ArrayList must be of the same data type. For example, you can have an ArrayList of Integers, an ArrayList of Strings, etc. A single ArrayList cannot hold two different data types at once (at least not without some other stuff we haven't learned yet).

The data type an ArrayList holds **must** be a reference type (a class). ArrayLists cannot directly hold primitive types like ints. However, Java provides the "wrapper" classes Integer, Double, Boolean, Character, etc, that can be used in their places, and Java (most of the time) handles the conversion between a primitive type and its corresponding wrapper class seamlessly.

When do I use a built-in array versus an ArrayList? If you know ahead of time the exact number of elements that will be stored in the array, you can use a built-in array. If you do not, you should use an ArrayList. That said, many people use ArrayLists in both situations. The one place you are more likely to see built-in arrays is when using multidimensional arrays, because otherwise you have use ArrayLists of ArrayLists, and that gets messy quickly.

Constructors:

- `ArrayList<`*`Type`*`> variable = new ArrayList<`*`Type`*`>();   // these constructors`
- `ArrayList<`*`Type`*`> variable = new ArrayList<>();       // mean the same thing`

Common instance methods of the ArrayList class: (Below, "*Type*" stands for the data type the ArrayList holds.)

- `void add(`*`Type`*` element)`
  Adds the element to the end of the ArrayList.
- `void add(int index, `*`Type`*` element)`
  Adds the element to the specified index of the ArrayList, sliding all the other elements to the right.
- `void clear()`
  Clears the ArrayList.
- `boolean contains(`*`Type`*` element)`
  Returns true if this ArrayList contains the given element.
- *`Type`*` get(int index)`
  Returns the element of the ArrayList at the given index.
- `int indexOf(`*`Type`*` element)`
  Returns the first index, searching left to right, where the given element is found, or -1 if not found.
- `boolean isEmpty()`
  Returns true if the ArrayList is empty.
- `int lastIndexOf(`*`Type`*` element)`
  Same as indexOf, but searches right to left.
- *`Type`*` remove(int index)`
  Removes the element at the given index and returns it.
- `boolean remove(`*`Type`*` element)`
  Removes the leftmost matching element from this ArrayList if it exists, sliding the other elements to the left.
  Returns true if a removal happened, and false otherwise.
- `Type set(int index, `*`Type`*` element)`
  Sets the value at the given index to the given element.
- `int size()`
  Returns the number of elements in this ArrayList.