The built-in ArrayList class

Arrays are convenient to use, but the drawback is that you must know the maximum size before using the array… It turns out that in Java there are many other ways to aggregate data, and one of these is with the ArrayList class. An ArrayList works much like an array, but you don't need to know the size in advance – it can grow or shrink as needed.

The ArrayList class is defined in java.util, so to include it you'll need to say "import java.util.ArrayList" at the top of your source file that uses this class. Complete documentation is available on docs.oracle.com/javase/8/docs/api. The most useful ArrayList methods are given in the table below.

Method	Example
default constructor (where *type* is the name of some type)	ArrayList<*type*> a = new ArrayList<*type*>();

add (obj)
(the new object goes to end of list)
	String name = "Bob";
	a.add(name);

get (int)	String thirdName = a.get(2);

contains (obj)	String word = "fire";
	if (a.contains(word)) …

indexOf / lastIndexOf (obj)	int pos = a.indexOf(word);
set (int, obj)	a.set(4, brownCar);
size ( )	for (i = 0; i < a.size(); ++i) …
remove (int)	remove (word);

An ArrayList begins with no elements (its size is 0), and each time we use the add() function, the size increases by 1.

Each element of an ArrayList may be an object, not a primitive value. What if you want to store integers? One way is to store them as String, since it's easy to convert back and forth between int and String. The other way is to store them as type Integer. If you simply type

```
a.add(16);
```

then Java assumes you want 16 to be stored as an Integer, not String. When you want to retrieve this value later, you will probably want to make it an int again. Then you would use Integer's intValue() function, which is analogous to Integer.parseInt() for Strings.

```
int valueFromList = a.get(0).intValue();
```

It would be a good exercise to go back to any program we've done that has used arrays, and change the array references to ArrayList. Let's compare.

Operation	array	ArrayList
declare	String [] words;	ArrayList<String> words;
allocate space	words = new String[100];	words = new ArrayList<String>();
initialize one cell	words[i] = s;	words.add(s); **OR** words.set(i, s);
obtain value from	s = words[i];	s = words.get(i);
take out	/* not possible */	words.remove(s);
search	/* requires a loop */	if (words.contains(s)) …