ArrayList Objects

`ArrayList` is a generic class for representing a list of objects. It provides methods for adding, inserting, removing, getting, and setting objects. Internally, the objects are stored using an array.

Manager:		Recorder:	
Presenter:		Reflector:	

Content Learning Objectives

After completing this activity, students should be able to:

- Summarize the differences between arrays and ArrayLists.
- Describe how objects are stored in an ArrayList internally.
- Write code that manipulates an ArrayList using for loops.

Process Skill Goals

During the activity, students should make progress toward:

- Reading API documentation to learn how to use a class. (Information Processing)

Facilitation Notes

Key questions: #5, #11, #17

Source files: *Model1.java*, *Model3.java*

Model 1 Example Code

The following examples are found in *Model1.java*. Open this file on your computer, and run the program. Record the output of each example in the space below.

```
1 int[] nums;
2 nums = new int[3];
3
4 nums[0] = 74;
5 nums[1] = 11;
6 nums[2] = 21;
7
8 System.out.println(nums.length);
9 System.out.println(nums);
```

The output is:

```
3
[I@7440e464
```

```
1 ArrayList<Integer> nums;
2 nums = new ArrayList<Integer>();
3
4 nums.add(74);
5 nums.add(11);
6 nums.add(21);
7
8 System.out.println(nums.size());
9 System.out.println(nums);
```

The output is:

```
3
[74, 11, 21]
```

Questions (20 min) Start time:

1. Compare the examples line by line, and summarize the differences.

a) Line 1:

Arrays are declared with square brackets; ArrayLists are declared with angle brackets. The array example uses the primitive type int, but the ArrayList example uses Integer.

b) Line 2:

When creating the array, you have to specify the length in brackets. When creating the ArrayList, you need parentheses after the <>'s.

c) Lines 3–6:

Arrays use square brackets to index a specific element. ArrayLists use the add method, and no indexes are required.

d) Line 8:

Arrays have a length attribute. ArrayLists have a size method.

2. What is the main difference in the output of these two examples?

Printing the array just displays its memory address. Printing the ArrayList shows the actual contents. (The reason why is because ArrayList has a toString method.)

3. What happens if you add the following code after Line 6 in the array example? Verify your answer by editing *Model1.java* and running the program.

```
nums[3] = 59;
```

```
Exception in thread "main" java.lang.ArrayIndexOutOfBoundsException:
Index 3 out of bounds for length 3
```

4. In *Model1.java*, comment out the line you just added in the previous question. Then add the following line to the ArrayList example. What is the resulting output?

```
nums.add(59);
```

```
4
[74, 11, 21, 59]
```

5. Based on your previous answer, what ability do ArrayLists have that arrays do not?

Their size can change, i.e., they can "grow" as you add new elements.
(They also provide a toString method, so you can print them directly.)

6. Add the following line to the ArrayList example. What is the result?

```
nums[0] = 33;
```

Compiler error:
The type of the expression must be an array type but it resolved to ArrayList<Integer>.

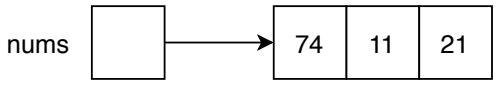
7. In the `ArrayList` example, is `nums` an *array* or an *object*? Justify your answer.

It's an object, because it has methods like add and size. You can't use it like an array, as shown in the compiler error.

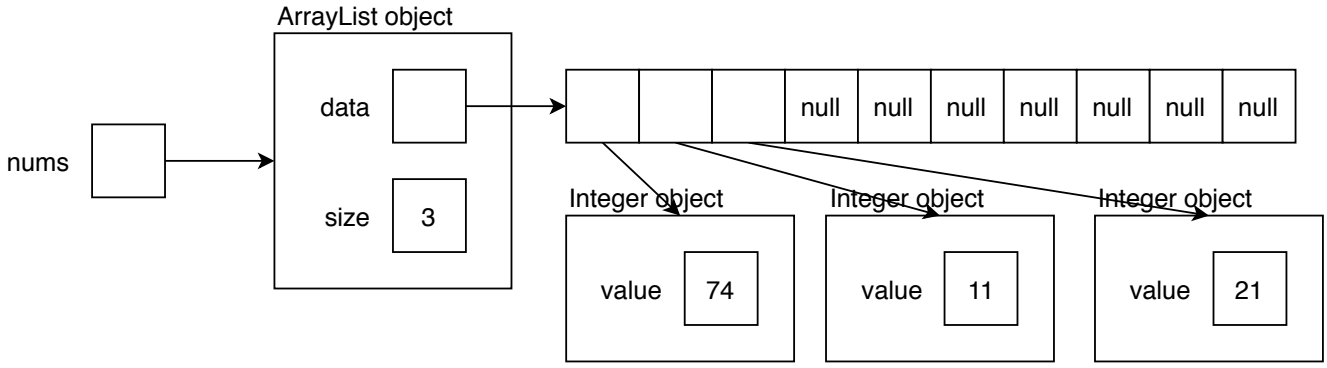
Model 2 Memory Diagrams

The following diagrams are based on the example code from Model 1.

Array of ints:



ArrayList of Integers:



Questions (10 min) Start time:

8. What is the `length` of the array inside of the ArrayList? 10

9. How are the contents of the `data` array different from the array of ints?

The `data` array is storing references to `Integer` objects, each of which wraps an `int` value. The array of ints simply stores the values directly.

10. What happens when a fourth element is added to the ArrayList?

(1) An Integer object is created, (2) it's placed in the next available position of the array, and (3) the size attribute is incremented.

11. Explain how an ArrayList can "grow" when adding new elements.

New elements can be stored in available slots at the end of the array. If the data array itself is full, then a larger array can be created.

12. Why do ArrayLists require so much more memory than arrays?

There is a lot of overhead. Each individual value needs to be wrapped inside of an object. And the array itself will likely have unused elements at the end.

Model 3 ArrayList Methods

Open the ArrayList documentation to answer questions about the following methods. The data type "`E`" refers to the type of elements in the ArrayList (e.g., `Integer`).

Return Type	Method	Description
`boolean`	`add(E e)`	Appends the specified element to the end of this list.
`void`	`add(int index, E element)`	Inserts the specified element at the specified position in this list.
`E`	`get(int index)`	Returns the element at the specified position in this list.
`E`	`set(int index, E element)`	Replaces the element at the specified position in this list with the specified element.
`int`	`size()`	Returns the number of elements in this list.

Questions (15 min) Start time:

13. What value does the `add` method return?

`true` (as specified by `Collection.add(E)`)

14. What value does the `set` method return?

the element previously at the specified position

15. What happens to existing elements when adding an element at a specified index?

Shifts the element currently at that position (if any) and any subsequent elements to the right (adds one to their indices).

16. What are the contents of `nums` after running the following code?

```
ArrayList<Integer> nums;
nums = new ArrayList<Integer>();
nums.add(74);
nums.add(21);
nums.add(0, 11);
nums.set(1, 59);
```

[11, 59, 21]

17. The following program, found in *Model3.java*, uses an array of ints. Rewrite the program to use an ArrayList instead.

```
import java.util.Arrays;

public class Model3 {

    public static void main(String[] args) {
        final int N = 4;
        int[] nums = new int[N];
        for (int i = 0; i < N; i++) {
            nums[i] = i + 1;
        }
        for (int i = 0; i < N; i++) {
            nums[i] *= 5;
        }
        System.out.println(Arrays.toString(nums));
    }

}
```

```
import java.util.ArrayList;

public class Model3ans {

    public static void main(String[] args) {
        final int N = 4;
        ArrayList<Integer> nums = new ArrayList<Integer>();
        for (int i = 0; i < N; i++) {
            nums.add(i + 1);
        }
        for (int i = 0; i < N; i++) {
            nums.set(i, nums.get(i) * 5);
        }
        System.out.println(nums);
    }

}
```