

In-class Exercises 05

*University of Wisconsin - La Crosse**Date: March 4*

1. Name one advantage `ArrayList` gives us as programmers over using an array.

Solution: The programmer no longer needs to manually grow/shrink the array

The programmer no longer needs to manage the array for basic functions (e.g., add at an index)

2. Describe the calculation that allows us to jump directly to any index in an array, skipping over intermediate indexes.

Solution: starting position of the array + (data type size * index)

3. Describe what problem we encounter when setting up an array list, such that we cannot store any type of data in our array list. In what piece of code is this problem made manifest?

Solution: The declaration and instantiation of the data array

4. Consider the below beginnings of an implementation of the `ArrayList` class. Implement a `public` method that has a return type of `ArrayList<E>` called `split` which takes in a given index and splits the list at that index, removing and returning the sublist created by starting at the index through the end of the list (somewhat similar to `substring(int)`). The new sublist must have at least one element in it; an empty sublist means an invalid index was given. Your method should modify the size attribute as appropriate for the current list, and should throw an `IndexOutOfBoundsException` if required. For example, consider an `ArrayList` storing the values `[1, 2, 3, null]`. Calling `split` at index 3 would be invalid, at index 2 would result in the original list looking like `[1, 2, null, null]` and returning `[3]`, and at index 0 would result in the original list looking like `[null, null, null, null]` and returning `[1, 2, 3]`. Note that the portrayed sublists returned might have additional `null` values depending on how the list grows. **Bonus:** how can you write this code such that the new array list never needs to grow?

```

1 public class ArrayList<E> {
2
3     private static int DEFAULT_CAPACITY = 10;
4     private Object data[];
5     private int size;
6
7     public ArrayList(int index) { ... }
8
9     public void add(E e) { ... }
10
11    public boolean add(int index, E e) { ... }
12
13    public E remove(int index) { ... }
14
15 }
```

Solution:

```

public ArrayList<E> split(int index) {

    if (index < 0 || index >= size) {
        throw new IndexOutOfBoundsException();
    }

    ArrayList<E> toReturn = new ArrayList<>(size-index);

    for (int i = index; i < size; i++) {
        toReturn.add(data[i]);
        //could also call remove at an index
        data[i] = null;
        size--;
    }

    return toReturn;
}
```