FlexArray.java files

Design two different classes named `FlexArrayPrimitive` and `FlexArrayRectangle` to model arrays that have some convenient methods. The `FlexArrayPrimitive` will contain a "backbone" array of `ints`; The `FlexArrayRectangle` will contain a "backbone" array of `Rectangle` objects. Use the Rectangle class provided in the assignment. The descriptions below apply to both FlexArrays.

To be used along with these specs is an informational document named "Array Element Manipulations". This document will be helpful when implementing the methods asked for below.

Coding:

Your class should contain the following features:

Private instance fields: (no additional fields should be created)

`private int mySize` represents the logical size of the array. (The quantity of active elements)

`private myArray`: an array that is appropriate to each FlexArray object type

Constructors:

a no args constructor should initialize `myArray` to hold 20 elements

a constructor that takes an integer, representing the quantity of elements `myArray` should contain

Methods: (you may add other methods as you see appropriate, but you must implement the ones shown below)

`public int getLength()`

returns value of `mySize`

`public boolean isEmpty()`

returns true if `mySize` is zero, false otherwise

`public void append(int data)`

or `public void append(Rectangle data)`

adds `data` to the end of the current list

```
//pre-condition:  index >= 0
public void insert(int index, int data)
```

or `public void insert(int index, Rectangle data)`

- inserts `data` into the list at the index value `index`
- the order of existing elements remains the same
- Example:
  - if the `FlexArrayPrimitive` contains `[4, 6, -8, 2, 6, garbage . . .]`
  - after the call of `insert(3, 5)`, the array will look like `[4, 6, -8, 5, 2, 6, garbage . . .]`
- any value cannot be placed at an index higher than `mySize`. To ensure this, if `index > mySize`, simply add data to the end of the active elements.

```
//pre-condition:  index >= 0
public int discard(int index)
```

or `public Rectangle discard(int index)`

- the value of the element at `index` is returned
- removes the element at the index value `index`
- the order of remaining elements remains the same

- if `index >= mySize`, return `-999` for the integer array, `null` for the `Rectangle` array
- Example:
  if the `FlexArrayPrimitive` contains `[4, 6, -8, 2, 6, garbage . . .]`
  after the call of `discard(1)`, the array will look like `[4, -8, 2, 6, garbage . . .]` and a 6 will be returned

`public String toString()`
output for the values for each `FlexArray` must show the list as demonstrated below.

The contents of a `FlexArrayPrimitive` should be listed in order from the beginning to the end of the active elements, all on one line with one comma and a space between each element, with hard brackets showing the beginning and the end of the list. For example, a `FlexArrayPrimitive` object filled with only 4 elements should look like:

```
[43, 6, 298, 7]
```

It is okay for a long list to wrap to the next line.

Similarly, the `FlexArrayRectangle` should contain hard brackets at its start and end, but should list each Rectangle on a new line like:

```
[Rectangle, width = 0 length = 1
Rectangle, width = 4 length = 2
Rectangle, width = 8 length = 3
Rectangle, width = 12 length = 4]
```

`private` `void resize()`

This private method will change the physical size of the array when needed. Make a local array that is larger than `myArray`'s current length. Copy its elements into the new, larger array. You may decide how much to enlarge the array. Outcome: `myArray` is larger than before the call to this method and contains the same values in the new array from [0, `mySize` – 1] as before. This method is private because the driver will never need to see the inner workings of how the FlexArray enlarges as needed. The FlexArray objects will control it internally.

What to turn in:

Only these two files will be turned in. All of the methods in your files will be tested with my driver. All identifiers and parameter lists will be exactly as written above.