**CSCI 3302 Programming Assignment 03 (100 Points)**

**Due: Oct 13, 8:00 AM**

GITHUB Link: [Program 03](https://classroom.github.com/a/rbZT91nd)

Objectives:

* Demonstrate basic competency in using Java arrays.
* Manage dynamic array storage.
* Implement an array-based ArrayList.

Assignment Assistance:

* This homework assignment is due at the beginning of the class period on the due date.
* This assignment is restricted to individual effort.
* Any resource used (other than Dr. Becnel or the course text) must be documented in the code (as comments) detailing the source and describing exactly what was learned and how that information was used. Submissions will be severely penalized if copied in part or in whole from any source.
* If you need help, visit your instructor during his posted office hours. If your schedule cannot accommodate any of these times, then email your instructor to schedule a different time.

Problem Description:

1. Java’s ArrayList is used to store a dynamically sized collection of elements. Design and implement (using an array) an ADT MyArrayList that stores items of type String and supports the following operations:

+MyArrayList()

// constructor that creates an empty list

+isEmpty() : boolean

// Determines whether the list is empty

+size() : integer

// Returns the number of items stored in the list

+add(in item : String) : void

// Appends the specified element to the list

+remove(in index : integer) : String

// Removes the element at the specified index from

// the list. The value stored at that index is

// returned.

+remove(in item : String) : void

// Removes the first occurrence of the specified

// element from the list, if it is present.

+get(in index : integer) : String

// Returns the element at the specified index from

// the list.

+removeAll() : void

// Removes all the items in the list.

+toString() : String

// String representation of list

+equals(o : Object) : boolean

// determine if two lists are equal

1. Your implementation should contain two class attributes. The first, called lstArray, is an array of String. The second, called size, is of type int. The lstArray class attribute should be created on the call to the constructor and initially have a size of two (2). The constructor should also initialize the size attribute to zero (0).
2. On each call to add, check to see if lstArray is full. If adding the new element would cause the array to overflow, then
   1. create a new larger array that is twice the size of the original array;
   2. copy the contents of the original array to the new array;
   3. add the new element to the new array; and
   4. reassign lstArray to reference the new array.
3. On each call to remove, check to see if lstArray is less than or equal to a quarter full. If removing the specified element would cause the array to be less than 25% full, then
   1. remove the selected element from the original array;
   2. create a new smaller array that is half the size of the original array;
   3. copy the contents of the original array to the new array; and
   4. reassign lstArray to reference the new array.
4. Your implementation should reset the size of lstArray back to size 2 if all the elements are removed (i.e. removeAll() is called).
5. To make sure that your implementation is working correctly, you need to create a MyArrayListException class as an extension of a RuntimeException. This exception needs to be thrown if any attempt is made to access an illegal index within the lstArray attribute. Create a file called MyArrayListException.java containing the following code:

**public** **class** MyArrayListException **extends** RuntimeException {

**public** MyArrayListException(**String** s) {

**super**(s);

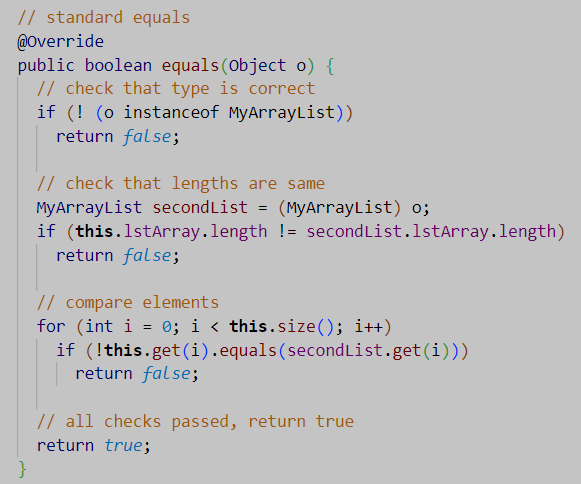
}

}

1. Include a toString() method which returns a String representation of the MyArrayList. For, example if the list contains the element “Hello”, “Data”, “Structures”, then the method would return the String

“[Hello, Data, Structures]”

1. Include the following equals method



1. You may write any private helper methods, as needed.
2. Your program should work in the GitHub codespace (Linux environment) and locally (Windows environment).
3. The files MyArrayListException.java and MyArrayList.java n should NOT contain a main method or any extraneous testing code. If you wish to include non-working code for insight into your thought process, make sure to contain it within comment blocks and ensure that submission successfully compiles. You can include test files in your repository, however, they will not be considered when grading.

Submission:

* Review the Evaluation below to ensure you have met all the requirements.
* Commit MyArrayListException.java and MyArrayList.java to GitHub. Upload a backup copy to D2L.

**Evaluation**

* 1. Project is late or not submitted at all. -100
  2. Project does not run/compile. -50
  3. Project compiles with warnings. -30
  4. Project does not correctly implement the interface. -30
  5. Calculations/output are for requirements incorrect. -10 each
  6. Code is not well organized or properly indented. -5
  7. Code is inadequately commented for readability. -5
  8. Code does not contain the student’s name, course section, -5

and date of submission.

* 1. Code is not submitted to the Github -15