Lab: `MyArrayList<E>` Implementation

AP Computer Science with Data Structures

Background: A `toString` Method

A toString() method returns a String representation of the object that contains the toString() method. It is defined in class Object and its behavior is to return a String containing the memory address where the Object is stored. Typically, the author of a class overrides the toString() method. The toString() method for an object is called automatically whenever a String representation of the object is needed.

You may wish to override toString() in your class. As an example, here is the code for creating a String representation of an array named values of type Object. Notice that it relies on the polymorphic behavior of Java objects.

```java
public String toString()
{
    if (size == 0)
        return "[]";

    String s = "[";
    for (int i = 0; i < size - 1; i++)
        s += values[i] + ",";
    return s + values[size - 1] + "]";
}
```

Exercise: A Smart Array

Create a Java Project to hold your MyArrayList code. Within that project, create a package called myArrayList (notice the capitalization, it is significant.) Your implementation of MyArrayList goes inside the myArrayList package. Testers and other code **does not** go inside the myArrayList package.

Create a class called MyArrayList<E> inside the myArrayList package. Your MyArrayList<E> must implement MyList<E> and Iterable<E>.

Document fully and implement the following methods using your design and the supplied MyList<E>, and MyListIterator<E> interface code. The iterator and

`listIterator` methods should return null. Also implement any other methods you included in your design.

All methods that are not helper methods should be declared as public, which is the default for `Interface` definitions.

```
Interface MyList<E>
```
- `int size()` //returns the current `size` of the list
- `boolean add(E obj)` //appends `obj` to the end of the list; //returns true
- `void add(int index, E obj)` //inserts `obj` at position //`index(0≤index≤size)`, moving elements //at position `index` and higher to the //right (adds one to their index) and //adjusts `size`
- `E get(int index)` // returns the element at position `index`
- `E set(int index, E obj)` //replaces the element at position `index` with `obj`; //returns the element formerly at the specified position
- `E remove(int index)` //removes element from position `index`, moving //elements at position `index+1` and higher to the left //(subtracts one from their index) and adjusts the size; //returns the element formerly at the specified position.
- `Iterator<E> iterator()`
- `MyListIterator<E> listIterator()`

When you have finished, test your design by writing a test program that implements your test plan. Note that you will need to import your `myArrayList` code by using the import statement: `import myArrayList.*;`. You may optionally run the supplied tester (which your teacher will surely run to test your code).

Exercise: Putting it in a Jar

Once you have finished testing your `MyArrayList<E>`, construct a Java Jar library as follows:

1. Create a directory that will hold the jar library and all of the files that will go into the library. Note that the library will grow throughout the course.
2. Create a sub-directory called 'myArrayList'.
3. Copy the source code and the .class files from your project into the myArrayList sub-directory. Do not include any testing code or anything that has a `main` method.
4. Launch the command (terminal) window and navigate to the directory that will hold your jar file (**not** the sub-directory).
5. Run the java jar utility using the following command:
   `jar –cvf MyLib.jar .`

6. Verify that the jar file was created.

To test your library, create a new Java project, and import only the tester. Make sure that your jar file is included as an external jar file or library. Make sure the tester still runs. **Make sure your jar file is named `MyLib.jar`.**

What to Turn In

Turn in your jar file and your `MyArraytList.java` file to Athena2. If you wrote your own tester, submit your tester as well.