Searching and Sorting In An Integer List

File IntegerList.java contains a Java class representing a list of integers. The following public methods are provided:

* IntegerList(int size) -- creates a new list of *size* elements. Elements are initialized to 0.
* void randomFill() -- fills the list with random integers between 1 and 100, inclusive.
* void print() -- prints the array elements and indices
* int search(int target) -- looks for value *target* in the list using a sequential search algorithm. Returns the index where it first appears if it is found, -1 otherwise.
* void selectionSort() -- sorts the lists into ascending order using the selection sort algorithm.

File IntegerListTest.java contains a Java program that provides menu-driven testing for the IntegerList class. Copy both files to your directory, and compile and run IntegerListTest to see how it works. For example, create a list, print it, and search for an element in the list. Does it return the correct index? Now look for an element that is not in the list. Now sort the list and print it to verify that it is in sorted order.

Modify the code in these files as follows:

1. **Add a method *void sortDecreasing()*** to the IntegerList class that sorts the list into decreasing (instead of increasing) order. Use the selection sort algorithm, but modify it to sort the other way. Be sure you change the variable names so they make sense!
   Add an option to the menu in IntegerListTest to test your new method.

2. **Add a method *void replaceFirst(int oldVal, int newVal)*** to the IntegerList class that replaces the first occurrence of oldVal in the list with newVal. If oldVal does not appear in the list, it should do nothing (but it's not an error). If oldVal appears multiple times, only the first occurrence should be replaced. Note that you already have a search method to find oldVal in the list; use it!
   Add an option to the menu in IntegerListTest to test your new method.

3. **Add a method *void replaceAll(int oldVal, int newVal)*** to the IntegerList class that replaces all occurrences of oldVal in the list with newVal. If oldVal does not appear in the list, it should do nothing (but it's not an error). Does it still make sense to use the search method like you did for *replaceFirst*, should you do your own searching here, or should you repeatedly use another method? Think about this.
   Add an option to the menu in IntegerListTest to test your new method.

Extra Credit: Make one or more of the following changes to IntegerList along with appropriate menu options/changes in IntegerListTest to test each method.

- Add a Binary Search Method (be sure the data is sorted first).
- Add an Insertion Sort Method (be sure the data is random first, so to know it works).
- Change the code to use a java.util.ArrayList<E> instead of an array.