Com S 227
Spring 2022
Miniassignment 2
50 points

Due Date: Friday, April 1, 11:59 pm (midnight)
5% bonus for submitting 1 day early (by 11:59 pm March 31)

This assignment is to be done on your own. See the Academic Integrity policy in the syllabus, for details.
You will not be able to submit your work unless you have completed the *Academic Dishonesty policy questionnaire* on the Assignments page on Canvas. Please do this right away.

If you need help, see your instructor or one of the TAs. Lots of help is also available through the Piazza discussions.

Note: This is a miniassignment and the grading is completely automated. If you do not submit it correctly and we have to run it by hand, you will receive at most half credit.

Contents

Overview

This is a set of practice problems involving writing arrays. Three of the methods are marked as extra credit. The specchecker has tests for these methods, but it is not currently showing points for them. You can earn up to 15 points in the final grading for completing all three of these methods.

For details and examples see the online Javadoc. There is a skeleton of the class on Canvas. If you use the skeleton code, be sure you put it in a package called `mini2`.

You are allowed to use any Java library classes for this assignment. When you test your program, you may want to print out an array, you can use Arrays.toString method for this purpose, for example:

```
System.out.println(Arrays.toString(arr));
```

and for 2-D arrays:

```
System.out.println(Arrays.deepToString(arr));
```

Advice

Before you write any code for a method, work through the problem with a pencil and paper on a few concrete examples. Make yourself write everything down; in particular, write down things that you need to remember from one step to the next (such as indices, or values from a previous step). Try to explain what you are doing in words. Write your algorithm in pseudocode, etc.

The other key problem-solving strategy is to remember that you don't have to solve the whole problem in your head all at once. Try solving *part* of the problem, or solving a *related, simpler problem*.

Some of the methods build on each other, you are encouraged to call methods of the public API and also create additional private methods to simplify the code.

For the **findRuns** method, the following gives a hint on how you may do it:

set a counter to 1
for each new element
if it matches the previous element
add 1 to the counter
otherwise
if we counted 3 or more elements that are the same there is a run
set the boolean array to true at the indices of the run
reset the counter to 1 and repeat for the next element

You may also need additional logic to handle the case when an array ends with a run.

My code's not working!!

If you are getting errors, a good idea is to go back to a simple concrete example, describe your algorithm in words, and execute the steps by hand.

If your strategy works when you carry out the steps by hand, and you are confident that your algorithm is right but you are still getting errors, you then have a *debugging* problem – at some point you've coded something that isn't producing the result you intend.

In simple cases, you can verify what's happening in the code by temporarily inserting **`println`** statements to check whether variables are getting updated in the way you expect. (Remember to remove the extra **`println`**'s when you're done!)

Ultimately, however, the most powerful way to trace through code is with the debugger, as we are practicing in Lab 6. Learn to use the debugger effectively, and it will be a lifelong friend.

If you have an infinite loop, please refer to "Syllabus" -> "Technical Guides and Resources" -> "Tips for debugging infinite loops" on Canvas for additional tips.

The SpecChecker

A SpecChecker will posted with a number of functional tests. However, when you are debugging, it is usually helpful if you have a simpler test case of your own.

Remember that to call a static method, you prefix it with the class name, not with an object reference. For example, here is simple test case for the **`lastIndexOf`** method:

```
import mini2.ArrayAdventures;
public class SimpleTest
{
  public static void main(String[] args)
  {
    // a sample test of lastIndexOf method
    int[] arr = { 1, -10, 3, -10, 5 };
    int actual = ArrayAdventures.lastIndexOf(arr, -10);
    System.out.println("Expected: 3");
    System.out.println("Actual  : " + actual);
  }
}
```

Since no test code is being turned in, you are welcome to post your tests on Piazza for others to use and comment on.

Style and documentation

Since this is a miniassignment, the grading is automated and in most cases we will not be reading your code. Therefore, there are no specific documentation and style requirements. However, writing a brief descriptive comment for each method will help you clarify what it is you are trying to do. Likewise, brief internal comments can help you keep track of what you are trying to do when you write a tricky line of code.

If you have questions

For questions, please see the Piazza Q&A pages and click on the folder **`miniassignment2`**. If you don't find your question answered, then create a new post with your question. Try to state the question or topic clearly in the title of your post, and attach the tag **`miniassignment2`**. *But remember, do not post any source code for the classes that are to be turned in.* It is fine to post source code for general Java examples that are not being turned in. (In the Piazza editor, use the button labeled "pre" to have Java code formatted the way you typed it.)

If you have a question that absolutely cannot be asked without showing part of your source code, make the post "private" so that only the instructors and TAs can see it. Be sure you have stated a specific question; vague requests of the form "read all my code and tell me what's wrong with it" will generally be ignored.

Of course, the instructors and TAs are always available to help you. See the Office Hours section of the syllabus to find a time that is convenient for you. We do our best to answer every question carefully, short of actually writing your code for you, but it would be unfair for the staff to fully review your assignment in detail before it is turned in.

Any posts from the instructors on Piazza that are labeled "Official Clarification" are considered to be part of the spec, and you may lose points if you ignore them. Such posts will always be placed in the Announcements section of the course page in addition to the Q&A page. (We promise that no official clarifications will be posted within 24 hours of the due date.)

What to turn in

Note: You will need to complete the "Academic Dishonesty policy questionnaire," found on the Homework page on Blackboard, before the submission link will be visible to you.

Please submit, on Canvas, the zip file that is created by the SpecChecker. The file will be named **`SUBMIT_THIS_mini2.zip`**. and it will be located in the directory you selected when you ran the SpecChecker. It should contain one directory, **`mini2`**, which in turn contains one file, **`ArrayAdventures.java`**. Always LOOK in the zip file the file to check.

We strongly recommend that you just submit the zip file created by the specchecker, AFTER CHECKING THAT IT CONTAINS THE CORRECT CODE. ***If you mess something up and we have to run your code manually, you will receive at most half the points.***

Submit the zip file to Canvas using the Miniassignment2 submission link and verify that your submission was successful. If you are not sure how to do this, see the document "Assignment Submission HOWTO" which can be found in the Piazza pinned messages under "Syllabus, office hours, useful links."

We strongly recommend that you submit the zip file as created by the specchecker. If necessary for some reason, you can create a zip file yourself. The zip file must contain the directory **mini2**, which in turn should contain the file

`ArrayAdventures.java`. You can accomplish this by zipping up the **src** directory of your project. The file must be a zip file, so be sure you are using the Windows or Mac zip utility, and not a third-party installation of WinRAR, 7-zip, or Winzip.