Introduction

We once asked 2110 students in the first few weeks to become familiar with Java Strings, if-statements, etc. This suggestion wasn't taken seriously, and 7 weeks later we found that many were still very shaky with these basic features. We don't want this to happen again, so we give this little assignment.

Please keep track of the time you spend on A2. You will have to tell us the time spent when you submit A2. We will publish the mean, median, and max and give a histogram of the time spent.

Learning objectives

1. Gain familiarity with String methods, if-statements, assignment, functions, and loops.
2. Get used to Eclipse and JUnit testing and with searching the Java API documentation for information.

Collaboration policy

You may do this assignment with one other person. If you are going to work together, then, as soon as possible —and certainly before you submit the assignment —get on the CMS for the course and do what is required to form a group. Both people must do something before the group is formed: one proposes, the other accepts.

If you do this assignment with another person, you must *work together*. Usually, we say that it is against the rules for one person to do some programming on this assignment without the other person sitting nearby and helping. You should take turns "driving" —using the keyboard and mouse. If you are the weaker of two students on a team and you let your teammate do more than their share, you are hurting only yourself. You can't learn without doing.

However, during this pandemic, where you and your partner may not be physically together, we can't require the above. However, you should zoom often and program together, looking at each other's suggestions for a method, say, and agreeing on the best solution. This includes looking at and discussing test cases in the JUnit testing class. In Eclipse, using Saros allows you and your partner to work together on this project.

With the exception of your CMS-registered partner, you may not look at anyone else's code, in any form, or show your code to anyone else, in any form. You may not look at solutions to similar previous assignments in 2110. You may not show or give your code to another person in the class. While you can talk to others, your discussions should not include writing code and copying it down.

Academic Integrity

With the exception of your CMS-registered partner, you may not look at anyone else's code from this semester or a previous semester, in any form, or show your code to anyone else, in any form. You may not show or give your code to another person in the class.

Getting help

If you don't know where to start, if you don't understand testing, if you are lost, etc., please SEE SOMEONE IMMEDIATELY —an instructor, a TA, a consultant, the Piazza for the course. Do not wait.

What to do

File `A2.java` contains seven (7) functions for you to write. Instructions are given at the top of that file and also within the body of each function, as comments. Each function you write has a "// TODO" comment. Please leave the // TODO comments in —don't delete them. Complete the seven functions and submit file `A2.java` on the CMS by the end of the due date given on the CMS.

Your grade will depend basically on the correctness of the seven functions, determined by running test cases on each function. We provide JUnit testing class `A2Test.java`, with all necessary test cases for the first five functions. If these functions pass the give test cases, they will receive full credit.

For the last two functions, `nCat` and `shortest`, we expect *you* to develop your own cases and test them thoroughly. You will not submit your test cases.

If a function is incorrect, you may receive 0 for it.

We reserve the right to deduct points if:

1. Method `areMidsDiff` uses a loop or recursion.
2. Method `exactly1` uses a loop or recursion.
3. Method `areAnagrams` uses a loop or recursion and doesn't follow the instructions.
4. In the really exceptional case, we see something fishy that violates the spirit of the assignment.

Submission

When you know your A2 is correct, fill in the time you spent on A2. Do this by looking at the declaration-assignment of static field `A2.timeSpent` (on about line 49) and changing the expression -1 to the time you used, to the nearest 15 minutes. Read the comment on the field please. Two points may be deducted if this field is not properly filled in.

Then, in the comment at the top of the class, put your netid(s) and tell us what you thought of this assignment.

Finally, submit `A2.java` on the CMS.

CAUTION: Your two .java files must be in package `a2`. If they are not in package `a2`, you will lose 10 points.

Guidelines/Instructions for working with Eclipse and JUnit testing in A2

1. Create a new project `a2` (or any name you want), using menu item: `File -> New -> Java Project`. If asked whether to create `module-info.java`, don't do it.

2. In the Package Explorer, click the right arrow preceding the project name. It turns into a down arrow and you see a directory `src` (for source) with nothing in it

3. Select directory `src` in the Package Explorer pane and use menu item `File -> New -> Package` to create package `a2`. Make sure the name is `a2` and not `a2test`. Don't create package-info.java.

4. Drag downloaded file `A2.java` on top of `src/a2`. A window will pop up, asking whether to link or copy. Select *copy* and click `OK`.

5. Double click file name `A2.java`. It opens in the big pane to the right of the Package Explorer pane. You can now edit that file.

6. Drag downloaded file `A2Test.java` on top of `src/a2` and do as you did on step 3 above.

7. File `A2Test.java` does not compile because JUnit 5 is not available. Here are two ways to add it to the "Build Path":

   7.1. Create a new JUnit test class (menu item `File → New → Junit test case`). Follow directions, and when it asks whether you want Jupiter (or Junit 5), click YES. Then delete the new JUnit class.

   7.2. In the Package Explorer, select the project to which JUnit 5 should be added. Right click and select:
   `Build path → Add Libraries`
   In the window that opens, select JUnit and click button Next. The window should change to one showing JUnit library version JUnit 5. Click button Finish.