**Unit 2: Using Objects**

**Topic 7 Lab 2: String Engineering!**

| **Name:** |  | | |
| --- | --- | --- | --- |

##### 

| **Time to kick it up a notch and do some "string engineering"! 😵**   1. Open up your U2T7 Partner Challenge Replit. 2. Work with your partner to add the six additional methods described below. These get a little crazy and brain busting! **🤯** You will need to have open [the Java API for the String class](https://docs.oracle.com/javase/8/docs/api/java/lang/String.html) to write the yellOrWhisper method. 3. Test each method using the test cases posted in a separate document in Google Classroom. As software engineers, you will be writing some pretty challenging code and will need to thoroughly test all parts of your code with *many* test cases! | | |
| --- | --- | --- |
| **/\* BELOW ARE THE 6 NEW METHODS TO ADD TO YOUR CustomStringMethods CLASS!** **\*/**  /\*\*Client provides two strings, str1 and str2,and method *prints* a message to the user that states  whether str1 comes *before* str2, comes *after*, or they are the *same* alphabetically.  This method has no return value (void).  **Example:** if str1 is "apple" and str2 is "banana", this method should *print* a message like:  "apple comes BEFORE banana alphabetically"  **Example:** if str1 is "banana" and str2 is "apple", this method should *print* a message like:  "banana comes AFTER apple alphabetically"  **Example:** if str1 and str2 are both "apple", this method should *print* a message like:  "apple and banana are the SAME string!"  \*/  public void alphabetical(String str1, String str2) {    }  /\*\*Client provides myString and the method returns a String that represents mystring  with its halves reversed; for example, for the string: "reverse me!" the method would  return "e me!revers"; strings of **odd length** should have the extra character be a part  of the *second* half when initially halved (and appear in the first half in the returned String).  \*/  public String halvesReversed(String myString) {    }  // The method below will require the use of String methods **toLowerCase()** and **toUpperCase**,  // neither of which are required on the AP Exam but both are very useful Java methods to know.  // Look them up in the [Java API docs](https://docs.oracle.com/javase/8/docs/api/java/lang/String.html) to see how they work!  /\*\*Client provides myString and this method should return a String with all characters in myString  in *uppercase* if the *first letter* of myString is an *uppercase* letter. If the first letter of myString is a  *lowercase* letter, this method should return a String with all characters in myString in lowercase.  You can assume myString will always begin with a letter (and not a number or some other character).  **Example:** If myString is "Hello James!", this method returns the String "HELLO JAMES!"  because the first letter of myString , "H", is an uppercase letter.  **Example:** If myString is "hello James!", this method returns the String "hello james!"  because the first letter of myString, "h", is a lowercase letter.  \*/  public String yellOrWhisper(String myString) {    }  /\*\*Client provides myString and the method returns a new String with the last numToCap characters in  uppercase, if not already; if myString has less than numToCap characters, uppercase the entire  String. *Any punctuation marks at the end should count towards* numToCap.  **Example:** If myString is "hello" and numToCap is 3, this method returns the String "heLLO"  **Example:** If myString is "hello" and numToCap is 6, this method returns the String "HELLO"  **Example:** If myString is "Gigantic" and numToCap is 3, this method returns the String  "GiganTIC"  **Example:** If myString is "Gigantic!!" and numToCap is 3, this method returns the String  "GigantiC!!"  \*/  public String endUp(String myString, int numToCap){    }  /\*\*Client provides myString and removeIdx and method returns a new String with the character  located at removeIdx in myString removed. If removeIdx is outside the bounds of myString,  the method should return myString unchanged.  **Example:** If myString is "Halloween" and removeIdx is 5, this method should the String  "Halloeen"  **Example:** If myString is "Halloween" and removeIdx is 0, this method should the String  "alloween"  **Example:** If myString is "Halloween" and removeIdx is 9 (outside the bounds of myString),  this method should return the String "Halloween" (the original myString unchanged).  \*/  public String removeCharacter(String myString, int removeIdx) {  }  /\*\*Client provides orig, insertText, and searchStr, and the method returns a new String where  insertText has been inserted into orig starting *at* the index where searchStr is *first* found in  orig, "pushing” all characters that come after insertIdx in orig *behind* insertText.  In the event insertText is not found in orig, append insertText onto the end of orig and  return that String.  **Example:** If myString is "ghost", insertText is "BOO!", and searchStr is "o",  this method would return the String "ghBOO!ost" (since in orig, searchStr is found at index 2).  **Example:** If myString is "ghost", insertText is "BOO!", and searchStr is "st",  this method would return the String "ghoBOO!st" (since in orig, searchStr is found at index 3).  **Example:** If myString is "ghost", insertText is "BOO!", and searchStr is "m",  this method would return the String "ghostBOO!" (since searchStr is not found in orig).  \*/  public String insertAt(String orig, String insertText, String searchStr) {    }  }  **TESTING! Test cases for all 6 methods have been provided in a separate Google Doc posted in Google Classroom (or open it** [**here**](https://docs.google.com/document/d/1oonf5oB6BUHhleWr0dtKdGalHd825uy6Jd8WkkGCxJU/edit?usp=sharing)**)** | | |
| **Copy/paste your code below for your 6 (fully tested!) new methods:** | | |
|  | | |

*Sample solutions will be posted later during class.*

**Lab continues on the next page!**

| **Coding Bat!** | | |
| --- | --- | --- |
| For these last problems, you will use [**CodingBat Java**](https://codingbat.com/java) to develop and test your methods, since it has a lot of *built-in* testing for you for each problem. *If you do not already have a CodingBat account*, create one first (use your preferred email) so your progress gets saved. If you already have an account, log in.   1. Go to [**CodingBat**](https://codingbat.com/java) and create an account (or log in if you already have an account); use any email you prefer (school or personal):      1. **Login**, then go to **prefs**:      1. Under **Teacher Share**, enter **mmiller@bths.edu**, then click **share** (this will allow Mr. Miller to see your progress and problem completion status): | | |
| **1. Open up the String-1 problems:**    **2. Find the endsLy problem:**  **Read the problem, then write and test your method on CodingBat's website. Click "Go" to perform automated testing of your method:**  **You have a fully valid solution when ALL AUTOMATED TESTS PASS!** | | |
| Copy and paste your method code that passes ***all***CodingBat tests:  public boolean endsLy(String str) {    } | | |
| **3. Find and solve the conCat problem:** | | |
| Copy and paste your method code that passes ***all***CodingBat tests:  public String conCat(String a, String b) {    } | | |
| **You should notice that you now have these two solved:** | | |
| **4. Find and solve one other string problem of your choosing!** | | |
| Which problem did you choose? |  | |
| Copy and paste your method code that passes ***all***CodingBat tests: | | |

###### Want to see sample solutions for endsLy and conCat? [click here](#_f3rsosx5c1we)

Done!

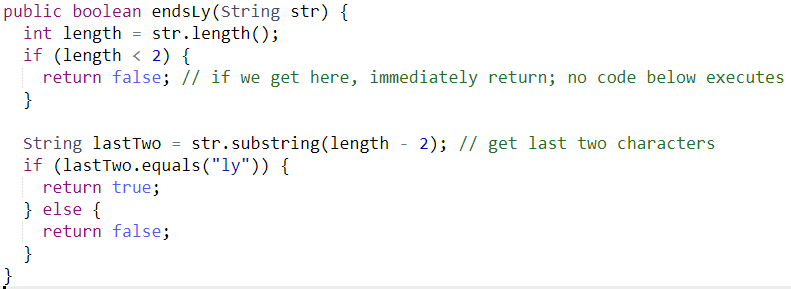
Submit in Google Classroom:



### 

### Sample solutions ([back](#_997sct3s0cqz))

**endsLy**



**conCat**

