Lab #3
CSE110 - Arizona State University

Topics

- Using methods from the `String` class - Section 2.5
- Using conditional (`if`, `if-else`) statements - Chapter 3

Coding Guidelines

- Give identifiers semantic meaning and make them easy to read (examples numStudents, grossPay, etc).
- Keep identifiers to a reasonably short length.
- Use upper case for constants. Use title case (first letter is upper case) for classes. Use lower case with uppercase word separators for all other identifiers (variables, methods, objects).
- Use tabs or spaces to indent code within blocks (code surrounded by braces). This includes classes, methods, and code associated with ifs, switches and loops. Be consistent with the number of spaces or tabs that you use to indent.
- Use white space to make your program more readable.
- Use comments after the ending brace of classes, methods, and blocks to identify to which block it belongs.

Assignment/Lab Documentation

At the beginning of each programming assignment you must have a comment block with the following information:

```
/*-------------------------------------------------------------------------
// AUTHOR: your name
// FILENAME: title of the source file
// SPECIFICATION: description of the program
// FOR: CSE 110- Lab #3
// TIME SPENT: how long it took you to complete the assignment
//-----------------------------------------------------------------------*/
```

Getting Started

Create a class called `Lab3`. Use the same setup for setting up your class and main method as you did for the previous assignments. Be sure to name your file `Lab3.java`.

Part 1: String Comparison

In your main method, write a segment of code which will **read in two strings** and **output whether the strings are equal or not**. For example, if both `String`s have the value `"hello"`, the following should be printed:

`The strings are the same.`

If the strings have different values, for example the first `String` had `"hello"` and the second `String` had `"world"`, then the following should be printed:

`The strings are not the same.`

The strings can contain spaces in them, so be sure to use the `nextLine()` method of the `Scanner` class to read them in.

Hint: See pg. 90 for information on comparing strings using the `String` class's `equals` method.

Question for thought (You do not have to submit your answer): What happens if instead of using the `equals` method, you use `==` to compare two `String`s? Change your code and input the same string and see what the output is.

Part 2: String Length Comparison

After part 1, write a segment of code which will determine which `String` has the longer length or if they are the same. For example, if the user entered `"hello"` and `"world"` in the first part, the output would be:

`The strings have the same length.`

If the user entered `"Java is fun"` and `I like to code`, then the output should be:

`"I like to code" is longer than "Java is fun"`

Hints: One possibility is to use an `if-else if-else` structure given in the following pseudocode:

```
}
test if string 1 is longer than string 2
    print the appropriate message
otherwise check if string 2 is longer than string 1
    print the appropriate message
otherwise
    print they have the same length
```

The `length()` method will give you the length of a `String`. The escape sequence `\"` can be used in a `String` to print the quotation mark.

Note

Labs are not graded by a program, so you do not need to spend a large amount of time making the output match perfectly with the sample below. Do, however, make sure your output is reasonable. The goal here is for you to demonstrate that you understand the underlying concepts.

Sample Output

Below is an example of what your output should roughly look like when this lab is completed. All text in bold represents user input.

Sample Run 1:
Enter a string: **hello**
Enter another string: **hello**
The strings are the same.
The strings have the same length.

Sample Run 2:
Enter a string: **I like programming**
Enter another string: **Java is fun**
The strings are not the same
"I like programming" is longer than "Java is fun".

Submission

Submit your `Lab3.java` file to the Submission Server. Go to the Submission Server site located on the course website, login, then click on Lab Submissions in the left frame. Choose Lab3 from the dropdown box, click on the browse button and find where you saved your `Lab3.java` file (and not the `Lab3.class` file) on your computer. Upload the file to the site and then click on the Submit button.

Your file will be submitted and a screen will show up displaying if your program compiled and what your output is when run on some sample input.

You should then check to make sure that the actual file submitted properly and is readable to the grader. To do so click on Grades in the frame on the left of the page and then click on the 0 underneath Lab3. You will again see that your program compiled and the sample output, but you should scroll down to the bottom of the screen and make sure your file is readable as well.

Important Note: You may resubmit as many times as you like until the deadline, but we will only mark your last submission.