Lab Workbook 7

Software Development HDSWD
Characters and Strings
22/06/12

Jonathan Meaney

For these problems you will be building up a single class with useful methods that can be performed on Strings. The name of this class is StringUtilities. Create a class called StringUtilities and a main class to test it out. The StringUtilities class has no data members or constructors, we're only interested in the methods it can perform. Create an object of the StringUtilities class to run the methods in the main class. You can use the following Strings as a starting point. Try change the String to see different outputs from the methods.

```
String sentence = "The quick brown fox jumps over the lazy dog";

String palindrome = "never odd or even";
```

You can create an object of the StringUtilities class like any other class:

```
StringUtilities utility = new StringUtilities();
```

Then you can run any methods available to the object and if it is returning something store it in an appropriate variable.

String and StringBuffer Documentation

```
http://docs.oracle.com/javase/1.4.2/docs/api/java/lang/String.html
http://docs.oracle.com/javase/1.4.2/docs/api/java/lang/StringBuffer.html
```

Problem 1:

Create a method in the StringUtilities class called toASCII that accepts a character as a parameter. The method calculates its ASCII code and returns it. Use a variable to store what is returned and print it out.

Problem 2:

Create a method in the StringUtilities class called maxWord that accepts a String and figures out which word is longest in the String. The method should return the longest word. Use a variable to store what is returned and print it out.

Problem 3:

Create a method in the StringUtilities class called minWord that accepts a String and figures out which word is the shortest in the String. Use a variable to store what is returned and print it out.

Problem 4:

Create a method in the StringUtilities class called reverseSentence that accepts a String as a parameter. The method should reverse the sentence completely, for example, How are you? becomes ?uoy era woH. Hint the StringBuffer class has a reverse method. Return the reversed String from the method. Use a variable to store what is returned and print it out.

Problem 5:

Create a method in the StringUtilities class called analyzeVowels that accepts a String as a parameter. The method should analyse the String and display the count of individual vowels in the sentence. i.e.

```
Vowel counts for the sentence
Mary had a little lamb.
# of 'a' : 4
# of 'e' : 1
# of 'i' : 1
# of 'o' : 0
# of 'u' : 0
```

Problem 6:

Create a method in the StringUtilities class called checkPalindrome. This method accepts a String as a parameter. A palindrome is a String that reads the same forward and backward, for example, noon and madam. Ignore the case of the letter. So, for example, maDaM, MadAm, and mAdaM are all palindromes. Maybe use toLowerCase() to make all letters in the String lower case to begin with and also remove any whitespaces. Return true or false depending on whether the word is a palindrome or not.