Lab #5 – LO8 Methods

Before beginning any of the programs, create a new project and call it Lab5. You will create all of your programs inside this project. Create a single package to contain all of your programs, call it Lab5. For each question, create a new java class named Program## where ## is the question number. You will only need to add a few lines to your main method to show that your method for each works. This code is left to you to determine for each question. As a separate exercise, if you want, you can try to test your code with JUnit.

1. Write a Java method to find the smallest number among three numbers.

- create a method with the follow method signature parts:
modifiers: public static
return type: int
name: smallest
parameters: num1, num2, num3 -> all int

- create a variable result of type int
- add a condition which compares num1 to num2 to find the smallest of these
- add a nested condition to both the initial IF and its ELSE to compare num1 or num2 to num3 to get the smallest and store this in result
- return result

2. Write a Java method to compute the average of three numbers.

- create a method with the follow method signature parts:
modifiers: public static
return type: double
name: average
parameters: x, y, z -> all double

- set up the return to contain an equation averaging x, y and z so that your method body is a single line of code.

3. Write a Java method to display the middle character of a string.
Note: a) If the length of the string is even there will be two middle characters.
b) If the length of the string is odd there will be one middle character.

- create a method with the follow method signature parts:
modifiers: public static
return type: String
name: middle
parameters: String inStr

- Add two local int variables: position and length. You'll use substring to do this so all you need to know is where to start and how many characters.
- Add a condition to check if the length of the inputted string is odd or even.
- If its odd, set position to the inputted string length divided by two. Set length to 1
- If its even, set position to the inputted string length divided by two minus one. Set length to 2
- return a substring of the inputted string using position and position+length as parameters

4. Write a Java method to count all vowels in a string.

- create a method with the follow method signature parts:
modifiers: public static
return type: int
name: countVowels
parameters: String inString

- create a local int variable called count and initialize it to 0
- set up a for loop to iterate through the string, one character at a time
- check each character with a switch. if a character(charAt(i)) is a vowel, increment count
- remember, you can handle multiple cases with a single line in a switch if you need to do the same thing for multiple cases
- after the loop is done, return count

5. Write a Java method to count all words in a string.

- create a method with the follow method signature parts:
modifiers: public static
return type: int
name: countWords
parameters: String inString

- create a local int variable count and initialize it to 0
- set up a condition that checks that neither the first or last characters of the string is a space.
- if the condition is true, set up a for loop to iterate through the string, one character at a time
- inside the loop, check if the current char(charAt(i)) is a space and if it is, increment count
- immediately after the loop completes and within the outer IF, increment count
- once all processing is done, return count