Name _____ Period _____

Advanced String Methods

Your Tasks (Mark these off as you go)

- ☐ Practice with advanced string methods
- ☐ Write algorithms with advanced string methods
- ☐ Receive credit for this lab guide

☐ Practice with advanced string methods

We have explored several String methods in previous lessons including concatenation, length(), substring(), etc. In this lab will you explore some of the more advanced String methods. For all the examples below we will assume that "s" is a String as follows,

```
String s = "\t\tTake a Hike!\t\t";
```

Recall that the indices of the individual characters of this String are as follows,

\t	\t	T	a	k	e		a		H	i	k	e	!	\t	\t
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15

Notice that the tabs (\t) also count as characters. In fact, the ASCII value for the tab is 9.

public int compareTo(Object myObj)

The *compareTo* method accepts any Object, here we will specify a String object. The general syntax for usage of compareTo is shown below,

Code	Output	Explanation

`String s = "\t\tTake a Hike!\t\t";`
`int j = s.compareTo("Idaho");`
`System.out.println(j);`
	-64	The ASCII code for \t is 9 and the ASCII code for I is 73. 9 – 73 = -64 is printed.

`String s = "\t\tTake a Hike!\t\t";`
`int j = ("Idaho").compareTo(s);`
`System.out.println(j);`
	64	The ASCII code for I is 73 and the ASCII code for \t is 9. 73 – 9 = 64.

`String s = "\t\tTake a Hike!\t\t";`
`int j = ("idaho").compareTo(s);`
`System.out.println(j);`
	96	The ASCII code for i is 105 and the ASCII code for \t is 9. 105 – 9 = 96.

public int indexOf()

This method comes in 6 flavors (each is described below). For each case, all return negative one (-1) if the search is unsuccessful.

Code	Output	Explanation

`String s = "\t\tTake a Hike!\t\t";`
`int j = s.indexOf("ake");`
`System.out.println(j);`
	3	The search begins at the beginning of the String. Where (\t) has an index of 0. The String "ake" is located at index 3.

`String s = "\t\tTake a Hike!\t\t";`
`int j = s.indexOf("ake", 7);`
`System.out.println(j);`
	-1	The search begins at index 7. The String "ake" does not exist after index 7 and -1 is returned.

`String s = "\t\tTake a Hike!\t\t";`
`int j = s.indexOf('H');`
`System.out.println(j);`
	9	The position of the character H is at index 9.

`String s = "\t\tTake a Hike!\t\t";`
`int j = s.indexOf(97);`
`System.out.println(j);`
	3	97 corresponds to the character a which is at index 3.

`String s = "\t\tTake a Hike!\t\t";`
`int j = s.indexOf('e', 4);`
`System.out.println(j);`
	5	The search begins at index 4. The character e is at index 5.

`String s = "\t\tTake a Hike!\t\t";`
`int j = s.indexOf(101, 5);`
`System.out.println(j);`
	5	101 corresponds to the character e. The search starts at index 5. e is at index 5.

public char charAt(int indx)

Code	Output	Explanation

`String s = "\t\tTake a Hike!\t\t";`
`char myChar = s.charAt(5);`
`System.out.println(myChar);`
	e	The char at index 5 is e.

public String trim()

Code	Output	Explanation
`String s = "\t\tTake a Hike!\t\t";` `System.out.println("X" + s.trim() + "X");`	XTake a Hike!X	The white space at the beginning and end of the string is trimmed of. X is concatenated at the beginning and end.

Public boolean contains(String ss)

Code	Output	Explanation

`String s = "\t\tTake a Hike!\t\t";`
`boolean b = s.contains("a H");`
`System.out.println(b);`
	true	The String a H is in the String s. true is printed.

Refer to the code below to predict the output to the following. You will confirm your predictions as part of Code Challenge 1.

```
String s = "\t\tTake a Hike!\t\t";
String i = "ike";
String t = "ake";
int j = 6, z = 99;
```

Code	Output
`int k = s.indexOf(i);` `System.out.println(k);`	
`int k = s.indexOf('c');` `System.out.println(k);`	
`char p = s.charAt(7);` `System.out.println(p);`	
`int k = s.indexOf(z);` `System.out.println(k);`	
`int k = s.indexOf('e', j);` `System.out.println(k);`	
`char p = s.charAt(z - 90);` `System.out.println(p);`	
`int k = s.indexOf(t, 5);` `System.out.println(k);`	
`int k = s.indexOf(z + 2, 4);` `System.out.println(k);`	
`boolean k = s.contains(t);` `System.out.println(k);`	
`String str = s.trim();` `System.out.println("+"+str+"");`	
`System.out.println(i.compareTo(t));`	

□ Write algorithms with advanced string methods

The methods of the String class are very powerful. In the next few exercises, you will apply them to write some interesting algorithms. You will implement the algorithms you write as part of your Code Challenges.

Refer to the `compareTo()` method above to write an algorithm that could be used to alphabetize the Strings s1, s2, and s3 as shown below.

Values of Strings s1, s2, and s3 before	Values of s1, s2, and s3 after

`String s1 = "cat";`
`String s2 = "car";`
`String s3 = "dog";`
	`String s1 = "car";`
	`String s2 = "cat";`
	`String s3 = "dog";`

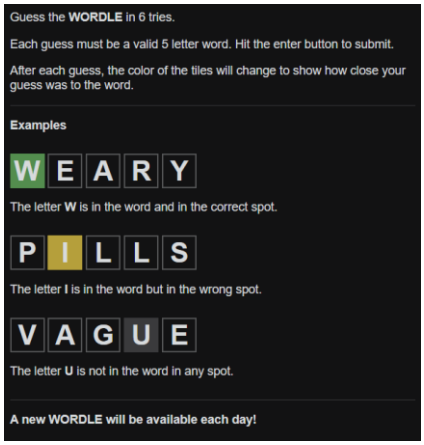
`String s1 = "dog";`
`String s2 = "cat";`
`String s3 = "car";`
	`String s1 = "car";`
	`String s2 = "cat";`
	`String s3 = "dog";`

Write an algorithm that could be used to count the number of times a string occurs in another string. Consider the examples below[1]. This algorithm requires that you incorporate a loop along with the substring() and length() methods.

String to search	String to find	Occurrences
BAAB	AA	1
AAAAA	AA	2
AABABABAA	ABA	2
ABBAABB	ABA	0

[1] Adapted from the 2020 AP Computer Science A Exam

Consider the WORDLE game described below,



The word to guess is defined below,

```
String wordToGuess = "earth";
```

Now consider a user's guess provided by a scanner called `sc`.

```
String guess = sc.next();
```

Write an algorithm that evaluates the first letter of the guess. Your algorithm should provide feedback as follows,

- If the first letter in the guess is not in the word, a "_" should be printed in the first position of `result`.
- If the first letter in the guess is in the word, but not in the correct location, a "*" should be printed in the first position of `result`.
- If the first letter in the guess is in the word and in the correct location, the first letter of the guess should be printed in the first position of `result`.

Below are some examples,

wordToGuess	guess	result
EARTH	BEATS	_
EARTH	EVERY	E
EARTH	ANGRY	*

☐ Receive credit for this lab guide

Submit this portion of the lab to Pluska to receive credit for the lab guide. Once received, your completed code challenges will also be graded and will count towards your final lab grade.