# *Programming I (420-B10-HR)*

# *Lab 5 – String Processing*

Date assigned: Tuesday, September 22, 2015

Date due: **Tuesday, September 22, 2015**

**Learning Objectives**

Upon successful completion of this lab exercise, the student will be able to:

1. determine the location of a character or a string within a string using the **indexOf()** and **lastIndexOf()** method;
2. extract a character from a string using the **charAt()** method;
3. extract a substring from a string using the **substring()** method;
4. convert strings to upper or lowercase using the **toUpperCase()** and **toLowerCase()** methods;
5. replace text with the **replace()**, **replaceAll()** and **replaceFirst()** methods;
6. use the escape character in an output statement.

**Methods Used:**

***String class methods:***

**char charAt ( int** *index* **)**

**int indexOf ( int** *character* [***,* int** *startingIndex*] **)**

**int indexOf ( String** *string* [***, int*** *startingIndex*] **)**

**int lastIndexOf ( int** *character* [***,* int** *startingIndex*] **)**

**int lastIndexOf ( String** *string* [***, int*** *startingIndex*] **)**

**String replace(char** *oldChar***, char** *newChar***)**

**String replaceAll(String** *oldString***, String** *newString***)**

**String replaceFirst(String** *oldString***, String** *newString***)**

**String substring ( int** *startingIndex* [***, int*** *endingIndex*] **)**

**String toLowerCase()**

**String toUpperCase()**

**To be handed in:**

1. A copy of your ***username*\_B10\_L05\_Strings** folder should be zipped and uploaded to **Moodle**.
2. The **Lab 5 - Review Quiz** should be completed in **Moodle**.

***Note:*** Five (5) per cent will be given for organization. In order to obtain the marks:

* All the files and folders must be correctly named.
* All the Java classes must be correctly formatted using the Format option of Eclipse.
* The completed lab folder must be zipped and uploaded to **Moodle**.

**To Start:**

1. Download and unzip the **B10\_L05\_Strings** folder from **Moodle** to your **H:\420-B10\Labs** folder. Rename it to ***username*\_B10\_L05\_Strings**.
2. Start **Eclipse**. Use your **H:\420-B10\Labs** as the workspace.
3. Create a **New Java Project** called ***username*\_B10\_L05\_Strings**.

# The indexOf() and lastIndexOf() methods

***Purpose:*** Learn to use the **indexOf()** and **lastIndexOf()** methods to find things in a string.

***To Do:***

## Create a new class in the **strings** package called **IndexMethods**. It should contain a **main()** method and should generate comments. The description of the program should be: "This program illustrates the **String.indexOf()** and **String.lastIndexOf()** methods."

## Add lines in the **main()** method to:

### declare a **String** object called **sentence**;

### declare a **Scanner** object called **keyboard**;

### prompt the user for a sentence;

### read the user input into **sentence** with the following statement:

sentence = keyboard.nextLine();

**Find the first and last occurrences of a character:**

## Add the following lines:

**int j = sentence.indexOf(' ');**

**int k = sentence.lastIndexOf(' ');**

**System.out.println("The first space is at position " + j);**

**System.out.println("The last space is at position " + k);**

**int m = sentence.indexOf('^');**

**int n = sentence.lastIndexOf('^');**

**System.out.println("The first ^ is at position " + m);**

**System.out.println("The last ^ is at position " + n);**

## Run **IndexMethods**. Try it a couple of times using different sentences. What do the **indexOf()** and **lastIndexOf()** methods return when the character is not in the string? -1

## Delete the statements to find and print the first and last instances of '^'.

**Finding subsequent occurrences of a character:**

## Add the following lines in the **main()** method before the closing brace ( } ):

**j = sentence.indexOf(' ', j+1);**

**k = sentence.lastIndexOf(' ', k-1);**

**System.out.println("The second space is at position "**

**+ j);**

**System.out.println("The second last space is at position "**

**+ k);**

## Run the program now.

***Explanation:***

**indexOf()**

The format of the **indexOf()** method is:

**indexOf(character, startingIndex)**

where **character** is the character you are looking for and **startingIndex** is the position in the string to start looking.

If **startingIndex** is omitted, it starts searching at the beginning of the string.

For example, given:

**String s = "Mississippi";**

then

To find the position of the first letter '**i*'*** in the string ***s***:

**int j = s.indexOf('i');**  j = 1 (*Remember that the first position is index 0*)

To find the position of the first letter '**i*'*** in the string ***s*** **after** position 2:

**int j = s.indexOf('i',2);**  **j = 4**

**lastIndexOf()**

**lastIndexOf()**works like **indexOf()** except that it searches from right to left instead of from left to right.

For example, given the String s above,

To find the position of the last letter '**i*'*** in the string ***s***:

**int j = s.lastIndexOf('i');**  j = 10

To find the position of the letter 'i' in the **string s** **before** position 9:

**int j = s.lastIndexOf('i',9);**  j = 7

## Modify the programto list the positions of the third space in the string and the third last space in the string.

## Test your changes.

***On your own:***

## You can search for a string in a string as well as for a character. Add code to the programto display the first and last two locations of "in" in the string starting at the beginning. Test your changes on the string "*The rain in Spain stays mainly in the plain*." The new lines in your output should look like the following (be sure to include the "):

**"in" is at position 6**

**The last "in" is at position 41**

**The second "in" is at position 9**

**The second last "in" is at position 31**

## Change your code to look for " in " instead of "in". What's the difference?

There’s only 2 “ in”, so the first and second last have the same value as do the second and the last.

# The charAt() method

***Purpose:*** Learn to extract a character from a string using the **charAt()** method

***To Do:***

## Open the **SubstringMethods** class. Right now it just reads a sentence from the user.

## Declare a character variable called **ch**.

## Add the following statements after reading the sentence:

**ch = sentence.charAt(0);**

**System.out.println("The first character in the sentence is '" + ch + "'.");**

## Add statements to extract and print the last character in the sentence.

# The substring() method

***Purpose:*** Learn to extract a substring from a string using the **substring()** method

***To Do:***

## Add the following lines after reading the sentence:

**int startIndex;**

**System.out.print("Enter an integer less than "**

**+ sentence.length()+ "> ");**

**startIndex = keyboard.nextInt();**

**String sub = sentence.substring(startIndex); // extract the**

**// substring that starts at the index read in**

**System.out.println("The substring starting at position "**

**+ startIndex + " is \"" + sub + "\"");**

## Run the program. Try it with different numbers.

***Explanation***:

The format of the **substring()** method is:

**substring(startIndex, endIndex)**

where **startIndex** is the position in the string to start extracting and **endIndex** is one past the position to stop extracting.

If **endIndex** is omitted, all the remaining characters in the string are extracted.

For example, given:

**String s = "Mississippi";**

then

To extract the substring beginning at position 1:

**String sub = s.substring(1);**  sub = "ississippi"

To extract the substring beginning at position 6:

**String sub = s.substring(6);**  sub = "sippi"

To extract the substring beginning at position 1 and ending at position 5:

**String sub = s.substring(1,6);**  sub = "issis"

## Try entering a number that is larger than the **String** length. What happens?

Returns an error and gives -1

***Explanation***:

The **java.lang.StringIndexOutOfBoundsException** occurs when an invalid value is given for the index. The index must be between 0 and the String length – 1.

## Add the following statement:

sub = sentence.substring(1, 3);

System.*out*.println("substring(1, 3) is \"" + sub + "\"");

## Run the program.

## Add statements to extract the substring in indexes 3, 4, 5, 6 and 7. (e.g. if the string is "A cold winter's day..", it will extract "old w")

## Test your changes.

## Add a statement to extract 3 characters from the input string:

sub = sentence.substring(startIndex, startIndex + 3);

System.*out*.println("The 3 characters starting at position "

+ startIndex + " are \"" + sub + "\"");

## Run the program.

***On your own:***

## Add code to the program to retrieve the first word in the sentence and display it. (***Hint***: Start at position 0 and end at the position of the first space.)

## Run your program using the sentence "Today is a beautiful fall day." The output should look like:

Please enter a sentence: Today is a beautiful fall day.

Enter an integer less than 32> 23

The substring starting at position 23 is "ll day."

substring(1, 3) is "od"

substring(3, 8) is "ay is"

The 3 characters starting at position 23 is "ll "

The first word is Today

## Run the program a few more times to test it on different sentences.

# Using the Escape character

**Purpose**: To learn to use the escape character (\) to output special characters.

* create a String object
* display the number of characters in a String object
* concatenate one String object to another String object.

**To Do**:

## Open **Singer.java**. Run **Singer**. What is displayed?

Old McDonald had a farm. Eeyi! Eeyi! O!

And on this farm he had a duck. Eeyi! Eeyi! O!

With a quack, quack here and a quack, quack there! Here a quack, there a quack. everywhere a quack !

## Add the following line after the second println():

System.out.println("It will now sing "Old McDonald had a farm"");

## What is the error message?

Multiple markers at the line, syntax error

Change the line to:

System.out.println("It will now sing \"Old McDonald had a farm\"");

Save **Singer** and run it again. What is the effect of the \ ?

Ignores the command following it that would normally run.

## Now we want to display one line before the **println()** we just added and two extra lines after the greeting. The \n escape sequence causes the output to go to a new line. Modify the line to:

System.out.println("\nIt will now sing \"Old McDonald had a farm\"");

## Run **Singer** again. What happens?

It begins on a new line

## Modify the **println()** statement to display two blank lines after displaying the message.

# Other useful String methods

***Purpose:*** Learn to replace a character or string in a **String** and to convert a string to upper or lower case.

***To Do:***

## Create a new class called **JavaPsychiatrist**. It should include a main() method and Javadoc comments.

## Add the following lines in the **main()** method:

Scanner keyboard = **new** Scanner(System.*in*);

System.*out*.println("Tell me how you feel. "

+ " Begin your sentence with \"I am\" and finish with a period");

String mood = keyboard.nextLine();

## Now we want to prepare a response. First we create a new String object called response. Then we convert the input sentence to lowercase to keep things calm. Use the **toLowerCase()** method to do this as follows:

String response = mood.toLowerCase();

## Add a **println()** statement to display **response**. Try running the program a few times using input lines such as:

*I am happy.*

*I AM MAD.*

*I AM what I am.*

## Now, we want to echo the user's response back to him/her phrasing it as a question to elicit more information – as a psychiatrist would! First we're going to replace the first person with second person by replacing the phrase "*i am*" with "*So, you are*". Use the replaceFirst() method to do that as follows:

response = response.replaceFirst("i am", "So, you are");

## Now turn it into a question by replacing the period at the end with a question mark using the replace method.

response = response.replace('.', '?');

## Add a **println()** statement to display **response** followed by the phrase "*Tell me more.*"

## Run the program again. Try it with different sentences.

***On your own:***

## Add a prompt that displays the following message:

Tell me something that you love. Begin your sentence with "I love" and finish with a period.

## Read the user's answer into a **String** called **text**.

## Change **text** to lowercase and assign the result to **response**.

## Replace "i" with "You" in **response**.

## Replace the period at the end of **response** with an exclamation mark.

## Put the last word in uppercase. To do this:

### Find the position of the last space and assign it to an integer called **lastWordBeginsAt**.

### Use the **substring()** method to extract the last word and assign it to a new String object called **lastWord**. (Hint the starting index is **lastWordBeginsAt**.)

### Use the **substring()** method to extract the first part of the sentence and assign it to a new String object called **firstPart**. (Hint the ending index is **lastWordBeginsAt**.)

### Use the **toUpperCase()** method to convert **lastWord** to uppercase.

### Assign the concatenation of **firstPart** and **lastWord** to **response**.

### Print **response**.

## Test your program. The program output should look similar to:

Tell me how you feel. Begin your sentence with "I am" and finish with a period

I am happy.

So, you are happy? Tell me more.

Tell me something that you love. Begin your sentence with "I love" and finish with a period.

I love Java.

You love JAVA!

# Review Exercise

## Complete the **Moodle** **Lab 5 - Review Quiz**.