Lab -12: 1301 Java Strings

Learning Objectives

- Understand key Java language statements and keywords
- Be able to develop confidently with Strings
- Be able to implement specialization using subclasses
- Work with fields and variables

Please Go thru the entire lab and assign a mixture of the assignment along with the final program at the end of this document. I am thinking that this is enough work for this lab and next weeks

Part 1- String Methods

Java comes with a number of classes built in, to complement the basic types (*int*, *boolean*, *double*, etc.). The String class is one of these, and it allows you to create objects for holding strings of text:

```
String name;
name = "Susan";
//Note that String data is always wrapped in double-quotes.
```

The String class also has a few predefined methods which we can use to manipulate and do things with our String data, which you will generally use with a String variable, so that it looks something like this: variable.method(). Bear in mind that method() here is a placeholder, and you will use one of the methods described below. To store the result in a new variable will look something like this:

```
Type variable = string.method()
```

Example: *charAt(index)*

```
String name = "Peter";

char letter = name.charAt(2);
```

Here, the value of the variable letter will be 't'. Notice that the type of the data returned by the method charAt(index) is a *char*.

Other methods:

- length() : gives the number of chars in a String (returns an *int*)
- toLowerCase() : converts to lowercase (returns a *String*)
- toUpperCase() : converts to uppercase (returns a *String*)
- substring(start, end) : extracts the substring from start (an integer) to end (another integer) (returns a *String*)

Be aware that none of these methods affect the value of the variable they are attached to! They return the affected value to you, which you must store in a new variable if you want to do anything else with it. For example, try running the below program and checking the results:

```
String var = "LaB eXeRcIsE!";
String low = var.toLowerCase();
System.out.println(var);
System.out.println(low);
```

Sample Program _Practice

Write a Java program that reads a string from the keyboard, translates it to all uppercase letters and outputs it on the display. If, for instance, the string "HeLLo" is given, the output will be "HELLO".

Solution:

```
import java.util.*;

public class Convertstring {
```

```
  public static void main(String[] args) {
    Scanner s=new Scanner(System.in);

    String a=s.next();
    System.out.println(a.toUpperCase() );
  }

        }
```

Exercise 0:

Write a Java program that reads a string from the keyboard, and outputs the string twice in a row, first all uppercase and next all lowercase. If, for instance, the string "Ciao" is given, the output will be "CIAOciao".

```
import java.util.*;

public class Convertstring {


  public static void main(String[] args) {
    Scanner s=new Scanner(System.in);

    String a=s.next();
   System.out.println(a.toUpperCase()+a.toLowerCase());
  }



}
```

Exercise 1:

a) Write a statement that declares a String variable named city. Initialized it to "San Francisco".
b) Store its length in a variable called stringlen.
c) Store the first char in a variable called firstChar.
d) Store the uppercase version of the string in upperCity.
e) Store in lowerCity the lowercase version of the string.
f) Store just the first part of the String (i.e. "San") in a variable called firstPart.
g) Print everything to the screen (one variable per line!)

String Comparisons

The method contentEquals(otherString) returns true if and only if this String that we're testing represents the same sequence of characters as otherString. It returns a *Boolean* value (true or false), which depends on whether the match succeeds or fails.

```
String str1 = "Test";
boolean equalStrings = str1.contentEquals("test");
```

Attention: do not confuse with compareTo(otherString). This method compares two Strings lexicographically (i.e. it returns an integer giving the distance between the Strings in a dictionary-style ordering). You can still use this method to check if two Strings are equal - just note that the method returns an *int* which will be 0 if and only if the Strings are equal.

Exercise 2:

a) Write a program that declares two String variables:str1 = *"home"* and str2 = *"house"*
b) Compare them and, using if-else statements, print
"The Strings "home" and "house" are equal" if they are the same, and
"The Strings are not the same" if they are different.

Notice the quote-symbols within the Strings - you must use the escape character (a backslash) to print these in your String (e.g. "This is an \"example\" here.").

Submit lab12.java includes three methods: Cat, HalfHalf, FancyMyName

(1)Write a program called Cat

a) Write a program that declares a String and initialize it to "cat".
b) Print your variable and then print the reverse: "tac" (this can be done using one of the methods introduced earlier.)

(2)Write a method called HalfHalf that reads a string from the keyboard, and outputs the string such that the first half is all lowercase and the second half is all uppercase. If, for instance, the string "HelloWorld" is given, the output will be "helloWORLD".

(3)Write a method call FancyMyName which asks you to write a program that tests and using thedifferent methods for working with Strings.

This program will ask the user to enter their first name and their last name, separated by a space.

Read the user's response using Scanner. Seperate the input string up into two strings, one containing the first name and one containing the last name.You can accomplish this by using the indexOf() hint*** find the position of the space, and then using substring() to extract each of the two names. Also output the number of characters in each name, and output the user's initials. The initials are the first letter of the first name together with the first letter of the last name.

A sample run of the program should look something like this:

Please enter your first name and last name, separated by a space?

You entered the name: Louis Henry
Your first name is Louis: has 5 characters
Your last name is Henry: has 5 characters
Your initials are: LH