**BRAC University**

**Department of Computer Science and Engineering**

**CSE111 Lab 6**

***For all of the tasks below, assume that you are using a character array to store values. Methods of the String class CANNOT be used for any of the tasks, except for toCharArray ().***

***For all tasks, take String inputs and put them in a character array before you proceed to do what is asked in the tasks.***

**Task 1 (length())**

Take a string as input from the user and print the length of the string (Remember that you cannot use any string class methods other than *toCharArray ()*

**Task 2 (charAt ())**

Take a string as input from the user … then take a position *n* as input. “n” must be a valid index (which is an integer) less than the length of the String (Refer to Task1 on how to find the length). If n is invalid, then print “Invalid index”, otherwise print the character at that index/position.

**Task 3 (startsWith ())**

Take two Strings as input from the user … check if the second String is a prefix of the first String. If yes, print “true” else print “false”. Please ensure your program works for empty string inputs, as in it shouldn’t crash if the first or second or both of the strings are empty: P

*Remember that you cannot use any String class methods here, hence character by character comparison will be required.*

**Task 4 (endsWith () )**

Take two Strings as input from the user … check if the second String is a suffix of the first String. If yes, print “true” else print “false”. Please ensure your program works for empty string inputs, as in it shouldn’t crash if the first or second or both of strings are empty: P

**Task 5 (replaceFirst(char,char))**

Take a String as input from the user. Then take in two more characters, oldChar and newChar (Read in as string and then convert to charArray and the save first index to a char variable). Replace the first occurrence of oldChar with newChar .. You are not permitted to modify the original character array ...: P

**Task 6 (replaceAll(char,char))**

Take a String as input from the user. Then take in two more characters, oldChar and newChar (use the same method as Task 5). Replace all occurrences of oldChar with newChar ..you are not permitted to modify the original character array.

**Task 7 (replaceLast(char,char))**

Take a String as input from the user. Then take in two more characters, oldChar and newChar (use the same method as Task 5 and 6). Replace the last occurrence of oldChar with the newChar… you are not permitted to modify the original character array.

**Task 8 (toLowerCase())**

Take a String as input from the user. Convert all uppercase to lowercase.

**Task 9 (toUpperCase())**

Take a String as input from the user. Convert all lowercase to uppercase.

**Task 10(equals())**

Take two Strings as input; if they are equal print “true”, else print “false”.

**Task 11(equalsIgnoreCase()**

Take two Strings as input; if they are equal regardless of the case, print “true”, else print “false”.

**Task 12(compareTo())**

Take two String as input. Print 0 if the Strings are equal. If the second string is lexicographically before the first one, returns a negative value, else returns a positive value. This task should behave the same as the compareTo() method in the String class, so look at the method definition in java doc for String.

**Task 13(compareToIgnoreCase())**

Take two String as input. Print 0 if the Strings are equal. If the second string is lexicographically before the first one, returns a negative value, else returns a positive value, regardless of the case. This task should behave the same as the compareToIgnoreCase() method in the String class, so look at the method definition in java doc for String.

**Task 14(substring (int))**

Take a String as input, and then an integer. If the integer is within the valid range, then find the substring starting with that integer till the end

**Task 15(substring (int,int))**

Take a String as input, and then two integers. If the integers are within the valid range, then find the substring starting with the character at the position of the first value and ending at the position of the last character.

**Task 16(indexOf(char))**

Take a String as input and then take a character as input (Same method as in Task 5/6). If the character exists in the String, print the location of the first occurrence of the character. If the character is not found, print -1.

**Task 17(lastIndexOf(char))**

Take a String as input and then take a character as input (Same method as in Task 5/6). If the character exists in the String, print the location of the last occurrence of the character. If the character is not found, print -1.

**Task 18(indexOf(char,int))**

Take a String as input and then a character and a number as input. Check the string from location indicated by the integer value, and print the location of the first occurrence of the character. If not found, print -1.

**Task 19(lastIndexOf(char,start))**

Take a String as input and then a character and a number as input. Check the string from location indicated by the integer value, and print the location of the last occurrence of the character. If not found, print -1.

**Task 20(concat())**

Take two strings as input and create a new character array that contains all characters of the first String followed by all characters of the second string. Print the new array and verify.