Lab 01

Instructor: Shahriar Ivan

General instructions:

Create a java application project naming the java file as Lab01_2B_ID where ID is your student ID. The example snippets will use the general class name Lab01_2B without the ID portion. If there are multiple tasks, you don't have to create separate projects for each task. A single project file should contain all the .java files that would be necessary to satisfy all the tasks given here. For this particular task, you only need one .java file inside your project, one public class with the same name as the java file and only one main function inside the public class. And you may declare multiple functions within the public class that you can call inside the main function.

Tasks:

String is a built-in class in Java. You can declare an object of String class as follows:

```
public class Lab01_2B {

    public static void main(String[] args) {

        String str1 = "Test string";      //here we declared an object of
        String class

        System.out.println(str1);

    }

}
```

You can access a particular character within the string object using charAt() method. And the string can be taken as input rather than assigned manually through code. In order to take input, Java has a built-in Scanner class. We need to import it so as to create an object of the Scanner class. Refer to the example below:

```
import java.util.Scanner;

public class Lab01_2B {

    public static void main(String[] args) {

        Scanner sc = new Scanner(System.in);


        System.out.println("Input a string: ");

        String str1 = sc.nextLine();

        int index = 0;

        System.out.println("The whole string is: "+str1);

        System.out.println("The character at position "+index+":
      "+str1.charAt(index));
```

```
    }
}
```

1. Now create a Java program that takes as input two strings of arbitrary length and passes them as arguments to another function that checks whether the given strings are anagrams of each other. Two strings are said to be anagrams of each other if the characters in one string can be rearranged to get the other string. Note that we are not considering case-sensitivity for this check. That is, uppercase letter and lowercase letter are considered the same as long as they are the same letter. For example:
    a. Input: “Listen” and “Silent”
       Output: Given strings are anagrams of each other.
    b. Input: “Tools” and “Loot”
       Output: Given strings are NOT anagrams of each other.
2. Take a new string str1 as input. Create another string str2 from the original string str1 by putting a ‘*’ after every odd position character. Consider the first character to be of position 1 (not position 0 like we address arrays). Print the new string in a separate line.
   Example:
    a. Input: “ABCDEFG”
    b. Output: “A*BC*DE*FG*”