**Lab 19 – Name That Celebrity**

Open BlueJ, and create a new BlueJ project titled **Lab19-NameThatCelebrity** in your CS\LABS folder.

Create a new class and **type** in our code skeleton:

//Name:

import java.util.\*;

public class PracticeProblems

{

public static void main(String[] args)

{

Scanner console = new Scanner(System.in);

}

}

Strings, unlike the other data types we’ve worked with (int, double, and boolean), are **objects** in Java. Objects are different than primitive data types – objects are references to a chunk of data, rather than a small piece of data (like an int, which is just a number).

Further, you can’t subtract a character from a String - Strings are **immutable** (they can’t be changed after they’re created). To get around this, there are built-in methods in the String class we can use that will perform some common operations. More information is available on our [website.](http://bit.ly/mrbunnCS)

**Before each problem, insert a COMMENT with the problem number.**

1. Get a value for a String variable called s from the keyboard, and (using a String class method) print the number of characters in s. **This (and all following problems) should work for any length of String** (use a method call, rather than hard-coding the answer).
2. Get a new value of s from the keyboard and (using a String class method) print a version of s without the first letter.
3. Get a new value of s from the keyboard and print a version without the last letter.
4. Get the value of two String variables, wordOneand wordTwo, from the keyboard. Print a new String in the form of short + long + short, with the shorter of the two Strings on the outside. Example:

wordOne = "hi", wordTwo = "bye" >>> hibyehi

1. Get a new value for s from the keyboard. Print out the first half of the String. You can’t assume that **s** will always been the same length!
2. Get two words, a and b, from the keyboard. Print a concatenation of the two values, omitting the first character from each. Example:

a = "something", b = "pluto" >>> omethingluto

1. (Riddle) What is black when you buy it, red when you use it, and gray when you throw it away?
2. Get a new value for a from the keyboard, and print “Yummy!” to the console if ahas the String “salsa” at the beginning (starting at index 0). **You can't compare Strings with the == operator! Check the powerpoints for more info.**
3. Get a new value for b from the keyboard, and print a version of b without the first and last characters. **This should be accomplished with one substring method call.**
4. Ask the user to type the word “fish” and print “Lame!” if what the user typed does NOT equal “fish”.
5. Get the value of a String variable oddString from the keyboard. It should be a String with an odd number of characters. Print out the middle 3 characters in oddString.

//remember – the length() method will return the number of characters in the String



**Celebrity guessing game app**

Create a new class and **type** in our code skeleton:

//Name:

import java.util.\*;

public class NameThatCelebrity

{

public static void main(String[] args)

{

Scanner console = new Scanner(System.in);

}

}

In this program, you’ll create a game where the user has to guess a celebrity’s name, given only a portion of the letters in the name. First, create a String variable containing a celebrity name of your choosing (e.g. String celeb = "George Clooney"). **Using substring methods**, remove some of the letters to make determining the full name more difficult. For example, George Clooney could become: rge oney.

Begin the game by printing the instructions, then give the player (the user) the “clue” (e.g. rge oney) and read in their guess. Compare their guess to the correct answer (use the equals() method to compare the *guess* to the *celeb* String we initialized earlier).

**Your game should have three 'difficulty levels' for our user to choose from** (prompt them to select a difficulty level, for example easy/medium/hard (should be a String)). An easier level would give the user more letters as a clue - a harder level would give fewer letters.

**Important Note:**  Changing the game difficulty does NOT change the celebrity; it just changes how many letters you give them in the clue. Your clue based on substrings of the celebrity name. Easier settings will reveal longer substrings than the had setting.

The program should have a loop that allows them to keep guessing. If they guess incorrectly 3 times, give them a hint (e.g. “former president”, or “pop music singer”). After the hint, they get 1 more try. If they guess incorrectly a fourth time, they lose the game (and you should tell them “**You lose! The answer was \_\_\_\_**”).

Click [here](https://youtu.be/2fVBeXs2Qq4) to watch a sample run of the program. Model your output formatted after this video.