

## ISTE-121

### Lab 03: GUI, Inner classes and Binary IO

#### Part 1: Inner Classes (6 Points)

##### Overview

Read all of Problem 1 before starting, hints are given throughout the problem statement. See the Execution Examples, below, to see what the GUI should look like.

##### Fields and Behaviors:

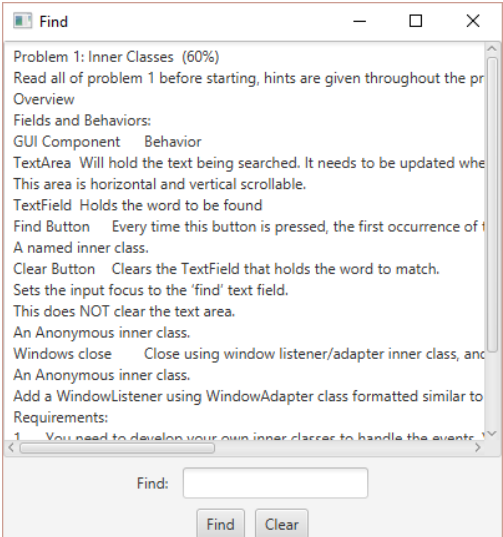
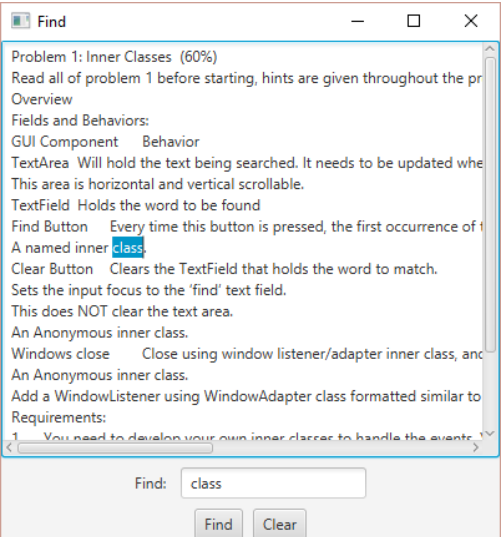
GUI Component	Behavior
TextArea	Will hold the text being searched. It needs to be updated when a word is found. This area is horizontally and vertically scrollable.
TextField	Holds the word to be found.
Find Button	Every time this button is pressed, the <u>first</u> occurrence of the "find" word in the TextArea will be located and highlighted. <b>A named inner class.</b>
Clear Button	Clears the TextField that holds the word to match. Sets the input focus to the 'find' text field. This does NOT clear the text area. <b>An Anonymous inner class.</b>
Windows close	Close using window listener/adaptor inner class, and print a "Thank you for using finder" message. <b>An Anonymous inner class.</b> Add a WindowListener using WindowAdapter class formatted similar to an ActionListener.

##### Requirements:

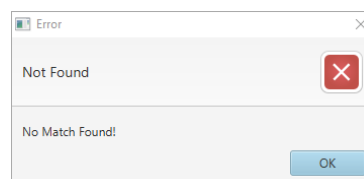
1. You need to develop your own inner classes to handle the events. You must handle the window closing and button press events.
2. The search is case sensitive. If the case of the letters in the word in the TextArea does not match exactly the word in the TextField, skip over it.
3. You only need to locate the **first** occurrence of a string of text. Use ONE of the string methods for this 'find' (look up the indexOf method of the String class).
4. The Find operation is able to find text such as in this sentence, "this sentence" should be found.

**NOTE:** To get text into the text area, simply copy text from another window and paste it in.

**Execution example:** (screens shrunk)

<p>Loaded the text area with text from this assignment. Notice the word “Classes” in the first line.</p> 	<p>Entered “find” and pressed the find button. Notice it skipped “Classes” to get to “class”.</p>  <p>The word “class” is highlighted.</p>
--	--

5. In the event there is no match, your inner class should create an Alert stating the lack of a match, as shown here.



**JavaDoc research you will need to do:**

1. Look at the String class for string matching methods.
2. Look at TextField, TextArea, and TextInputControl for useful methods to *select text*.
3. If you like, you can *set the selected text color* from gray to another **Color**, such as red or blue. But this requires the use of CSS, which you may or may not know about, so this is not required. If you would like to do this, consider the code

```
scene.getStylesheets().add(  
    this.getClass().  
        getResource("FindHiLight.css").toExternalForm()  
);
```
4. For a TextArea to indicate selected text, the TextArea may require focus. Read the javadocs for the **requestFocus()** method.

To what class does requestFocus() belong?

**Place your answer as a comment at the bottom of your Part 1 code (submitted below).**

**Submit your code (the .java file(s) only) to the Lab03 Assignment folder when Part 1 is working correctly.**

**Did you know? (Extra practice / learning, after Part 2 is done)**

A TextField can use setOnAction. Set the OnAction for the 'find' TextField that will call the same code as the find button. Do not duplicate your code.

## Part 2 - Byte I/O (4 points)

### Objectives:

Learn about binary file I/O reading and writing.

### Description:

Write a program (no GUI required), `CalcAverages.java`, that reads a binary file `ClassList.dat`, calculating the grades and prints the results. The file layout is as follows:

Data type	Data item
UTF	Student Name
integer	Student number
double	Grade in class 1
double	Grade in class 2
double	Grade in class 3
double	Grade in class 4

### Requirements:

Read the binary data file, not knowing how many records are in the file. Compute the average of each student's grades, and print the average as the last column of the data read.

There are three lines of output shown. However, the data file contains more data than this. Do not program for only three people. The program must handle **all** the data, and terminate without errors.

Use the `printf` or `String.format` method for formatting. Make sure your headings and column output matches exactly.

### Partial sample of the output:

Name	ID	Grade1	Grade2	Grade3	Grade4	Avg.
Josephyn Brown	192837873	98.5	88.9	92.4	91.1	92.7
Fred Brown	982883456	72.5	8.4	82.4	91.1	63.6
George Foreman	678374875	72.5	78.4	81.1	75.6	76.9

**Submit your code (the .java file(s) only) to the Lab03 Assignment folder when Part 2 is working correctly.**