Programming Assignment 5
Stringy Stuff

The purpose of this assignment is to gain experience in text processing. The two objects commonly used to interact textually between programs and users are labels and text fields. A label is an object that displays text. The `java.awt.Label` class is normally used, but since these labels use a small font, we will also use a class called `BigLabel` that has all the same methods but merely uses a larger font. The methods of the `BigLabel` class are shown in the class diagram on the next page. Excepting for the constructor, they are identical to `java.awt.Label`.

Label classes, like `java.awt.Label` and `BigLabel`, provide a way for the program to display text to the user, namely the `setText` method. In order for the user to provide information to an executing program, a different kind of object is needed. One such class is `javax.swing.JTextField`. An object of type `JTextField` can be thought of as a colorless label, with one important inclusion -- the `getText` method. The `getText` retrieves whatever String is displayed in the text field at the time of call. That string was typically entered by the user but could also have been the result of some `setText` instruction.

A third class that is often used by programs processing textual input is `java.util.Scanner`. A `Scanner` object is the preferred way to translate from text into any primitive type. For this assignment we need only translate from String to `int`. The way to use a `Scanner` is shown in the following example:

```java
// assume the myString variable stores text in the form of some valid integer literal
Scanner stringToIntConverter = new Scanner(myString);
int myInt = stringToIntConverter.nextInt();
```

The first instruction above instantiates a scanner object, passing it the text (`myString`) that needs to be converted. The second instruction calls the `nextInt` method to accomplish the conversion. It is important that `myString` actually contains a string that represents a valid integer, and it won't work to call `nextInt` a second time to convert the same text. (BTW: There are other methods, `nextDouble`, `nextByte`, and so forth, to convert into other types of primitives. These methods are not required for this assignment.)

**10-point assignment**

Initially

As the program begins execution, it produces the `ThreeButtonFrame` shown to the right. The `Top String`, `Bottom String`, `Position:`, and `Length:` text are each created using `java.awt.Label` objects. The `Result:` and the green rectangular region are created as `BigLabel` objects. Four `JTextField` objects are needed to create the white blocks.

It is assumed that the user will type text into the four text fields and then strike one of the buttons.

**LEFT**

Each time the LEFT button is clicked the text in both of the top two text fields is changed as follows. The top textfield is rearranged by moving its first character to the end. (For example, if the user had entered the string `abcd` it would become `bcda`.) The second text field has its last character moved to the front. (For example, the string `abcd` it would become `dabc`.) Your code may assume only that the content of the two longer text fields are non-empty when LEFT is clicked.

**MID**

Each time the MID button is clicked the green `BigLabel` displays the following: all of the text from the top text field with all of the text from the second text field inserted. The location of the insertion is determined by the position text field. (The value of the position text field specifies the number of top string characters preceding the point of insertion.) Your code may assume that only that the number in the position field is from zero through the length of the top text field string. An example of the result after clicking MID is shown to the left.
For two additional points implement the RIGHT button as described below.

RIGHT
Each time the RIGHT button is clicked the green BigLabel displays the following: the text from the top text field with a contiguous substring deleted. The beginning position of the deletion is given by the number in the position field and the length of the deleted substring is given by the length subfield. The example to the right illustrates the result after the RIGHT button has been clicked for a particular case.

Your code may assume only that the position field and length field represent valid integers within the range and possible length to be deleted (from zero through the top string length).

to submit your solution...
Email your solution as an attachment that is a zip version of the project file to riley@cs.uwlax.

Due date: Mar. 4, 2016