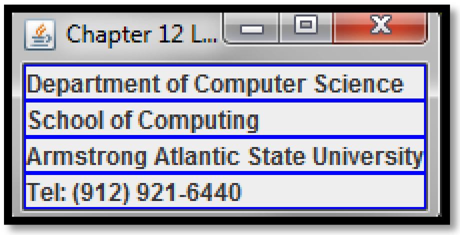
CIT 249: Java II
Chapter 12
Lab1

In this lab we will complete #5 on page 476. When we run the GUI it will display as:



Department of Computer Science
School of Computing
Armstrong Atlantic State University
Tel: (912) 921-6440

1. Open a new document and save the file as Ch12Lab1.java.
2. Our first step is to import the predefined classes that are required by typing:

```
import javax.swing.JFrame;
import javax.swing.JLabel;
import java.awt.GridLayout;
import java.awt.Color;
import javax.swing.border.LineBorder;
```

- Notice that I had multiple import statements for classes in the same package. Instead of JFrame and JLabel I could have combined these import statements together by substituting an asterisk instead. However this would also import a lot of classes we do not require, so having separate import statements is more proficient.

3. Next we type our class header and opening brace:

```
 public class Ch12Lab1 extends JFrame
{
```

- Here we extend the JFrame. This predefine class contains all attributes and methods required to create standalone GUI applications.

4. When you extend the JFrame you normally include a constructor method for constructing the components that are added to the frame. The constructor method always has the same name as the class. Type:

```
public Ch12Lab1()
{
```

5. We construct 4 labels by typing:

```
JLabel jlbl1 = new JLabel("Department of Computer Science");
JLabel jlbl2 = new JLabel("School of Computing");
JLabel jlbl3 = new JLabel("Armstrong Atlantic State University");
JLabel jlbl4 = new JLabel("Tel: (912) 921-6440");
```

- The text within double quotes sets the text that will display for this label. Since it is a String argument the text must be enclosed within double quotes.

6. Next we set the border that will display around the labels by typing:

```
jlbl1.setBorder(new LineBorder(Color.BLUE, 1));
jlbl2.setBorder(new LineBorder(Color.BLUE, 1));
jlbl3.setBorder(new LineBorder(Color.BLUE, 1));
jlbl4.setBorder(new LineBorder(Color.BLUE, 1));
```

- Java is capable of creating many different types of borders using the setBorder() method. The LineBorder class creates a solid border. Here set the color of the border to blue with a size of 1 pixel.

7. Frames by default have a layout of BorderLayout. BorderLayout is divided into coordinates of: NORTH, SOUTH, EAST, WEST and CENTER. The problem with this layout is that only one component can go into a single location. There are many ways to get around this limitation. This in case we will change the frame's layout to GridLayout which is divided into rows and columns. Type:

```
setLayout(new GridLayout(4, 1));
```

- Here we have set the layout to 4 rows and 1 column per row.

8. We next add the labels to the frame using the JFrame add() method, and close the constructor method. Type:

```
        add(jlbl1);
        add(jlbl2);
        add(jlbl3);
        add(jlbl4);
    }
```

- The order of how these are added matters. Components are added to the frame in the order listed. Here we add the jlbl1 label to the first row, jlbl2 to the second row, etc.

9. Since this is a standalone application you must have a main method. Type:

```
    public static void main(String[] args)
    {
        Ch12Lab1 frame = new Ch12Lab1();
        frame.setTitle("Chapter 12 Lab 1");
        frame.pack();
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        frame.setLocationRelativeTo(null); // Center the frame
        frame.setVisible(true);
    }
```

- First you create an instance of the class. The class is actually the frame itself.
- We set the text that will display on the title bar.
- The pack() method packs the frame into the smallest possible size that will display everything.
- The setDefaultCloseOperation() closes the frame when the closing button at the upper right-hand corner is pressed. If this is not included, the program will appear to close, but actually the frame closes but the program is still running.
- The setLocationRelative() method places the frame at the center of the window.
- The setVisible(true) is required for the frame to display. If this line is omitted the frame will not display.

10. End by closing the class.
11. Compile the program and fix any errors if necessary.
12. Run the program.
13. Compress all files into a single zip or rar file and submit.