Conrad Horton

CISC203 June 2016

Java FX Lab

20160717

1. What happens if you do not add a GUI component to a container?

   The component is not organized properly.

2. What happens if you forget to register an event handler for a GUI component?

   Nothing happens when an action is performed.

3. What happens when adding a component to a BorderLayout if you do not specify the region in which the component should be placed?

   Places and sizes in the center.

4. What happens when more than one component is added to a particular region in a BorderLayout?

   Border layout tries to fill the space with the components. This can look very bad.

5. What happens at execution time if an attempt is made to add a component to a container, but that component has not yet been instantiated?

   The program will continue to run, but the component won't be there.

6. How is an anonymous inner class different from other inner classes?

   Cannot access non-final properties. The class as a whole is only associated to one object instance.

Part II

1. What does the GUI look like in the following application?

```java
// ProgrammingOutput.java
import java.awt.FlowLayout;
import java.awt.GridLayout;
import javax.swing.JButton;
import javax.swing.JCheckBox;
import javax.swing.JFrame;
import javax.swing.JLabel;
import javax.swing.JPanel;
import javax.swing.JTextField;

public class ProgrammingOutput extends JFrame
{
   private JButton cancelJButton;
   private JButton okJButton;
   private JTextField inputJTextField;
   private JLabel nameJLabel;
   private JCheckBox firstNameJCheckBox;
   private JCheckBox lastNameJCheckBox;
   private JPanel checkJPanel;
   private JPanel buttonJPanel;

   // constructor sets up GUI
   public ProgrammingOutput()
   {
      super( "Input Name" );

      // build nameJPanel
      nameJLabel = new JLabel( "Type your name" );
      inputJTextField = new JTextField( 20 );
      setLayout( new FlowLayout() );
      add( nameJLabel );
      add( inputJTextField );

   } // end ProgrammingOutput constructor
} // end class ProgrammingOutput
```

```java
// ProgrammingOutputTest.java
import java.awt.FlowLayout;
import javax.swing.JFrame;

public class ProgrammingOutputTest
{

   public static void main( String args[] )
   {
      ProgrammingOutput application = new ProgrammingOutput();
      application.setDefaultCloseOperation( JFrame.EXIT_ON_CLOSE );
      application.setSize( 400, 150 );
      application.setVisible( true );
   } // end main
} // end class ProgrammingOutputTest
```

A rectangle with a label and a text input.

2. What does the GUI look like after adding the following code segment is added at the end class ProgrammingOuput's constructor in Programming Output Exercise 1?

```
// build buttonJPanel
okJButton = new JButton( "Ok" );
cancelJButton = new JButton( "Cancel" );
buttonJPanel = new JPanel();
buttonJPanel.setLayout( new GridLayout( 1, 2 ) );
buttonJPanel.add( okJButton );
buttonJPanel.add( cancelJButton );
add( buttonJPanel );

```

An okay and cancel button has been added at the bottom horizontally.

3. What does the GUI from Programming Output Exercises 1-2 look like if the following line of code is in- serted after line 10 ofProgrammingOutputTest.java

```
application.setLayout( new FlowLayout( FlowLayout.LEFT, 10, 5 ) );
```

Everything was aligned left.

4. What does the GUI from Programming Output Exercises 1-3 look like if the following line of code is replaces the line of code added to ProgrammingOutputTest.java in Programming Output Exercise 4?

```
application.setLayout( new FlowLayout( FlowLayout.RIGHT, 10, 5 ) );
```

Everything is aligned right.