GUI Practice

Ch. 16

Create a GUI that has a square following the mouse around, changing color as the mouse moves.

1.) Create the JComponent subclass.

2.) Create a class that implements a MouseMotionListener interface. It is in this class we will do all our drawing.
   (a) We will draw a square that follows around the mouse.
   (b) The color of the square will depend on the mouse's location.
   (c) When we drag the mouse, the JComponent will clear.

3.) Create the Tester file that has the main method with our GUI.

To have some code execute each time a button is pressed you would put the code in the actionPerformed() method of a class that implements ActionListener, let's call it listener, and then

A. Pass listener to a method of the button.

B. Pass the button to a method of listener.

C. Do both A and B.

D. None of the above, you would add the code to a method of a class that extends JButton.

A listener object can be "listening" to multiple components and do different things depending upon which component generates an event. True or False?

A listener object can be "listening" for one and only one type of event (e.g. ActionEvents for clicks on buttons or MouseEvents to respond to mouse motion like drawing but not both). True or False?

Multiple listener objects can be listening and execute code for a single event on one component. For example, two different listeners could respond, each in their own way, to a single button click. True or False?

```java
/*Complete the import statements below that will allow us
to create a GUI.*/

import _____;
import _____;
public class GUITest{
    public static void main(String[] args){
    // GUI code omitted



    }
}
```

```java
/*Create a simple GUI that has one button with text
"Click". Create an action listener object of type Echo,
add that object as an action listener to the button.*/

import java.awt.*;
import javax.swing.*;
public class ButtonTest{
    public static void main(String[] args){
    // Your code goes here




    }
}
```

```
/*Complete the class below such that every time the button
is pressed, the number on the button goes up by one. You
can assume the JButton being shown has an object of this
class as an ActionListener*/
```

```
...
JButton b = new JButton("0");
CounterListener c = new CounterListener(b);
b.addActionListener(c);
...
```

```java
import java.awt.*;

import java.awt.event.*;

public class CounterListener implements ActionListener{

    JButton b;

    int count = 0;

    // Your code goes here

}
```

5