The purpose of this lab is to set up a Graphical User Interface.

Part 1 – Get a simple screen to appear

Using the code from lectures create a screen class (that extends JFrame) to get a simple screen running that contains one button.

To "run" or use the screen, it's the usual step for using any java class: Use a main method. From the main method, instantiate your screen class.

Change these things and rerun your screen to see the impact:
- The setSize() parameters;
- comment out the setVisible(true);

Part 2 – Add some more GUI components

Add the following components to it:
Another button,
A label ,
+ another GUI components of your choice

See a list of all the GUI component classes (the ones beginning with "J") in the Javax.Swing package from the Java API to give you ideas and let you know the class names to use
https://docs.oracle.com/javase/8/docs/api/javax/swing/package-summary.html

Run your screen again to make the new components appear.

Part 3 – Implement "event programming" to make the GUI responsive

Your screen doesn't DO anything until you add event programming.
Add functionality so that when the first button is clicked, it displays a pop up method
(Pop up code is: `JOptionPane.showMessageDialog(this, "MyFirst event!")`):

3 steps to implementing event programming:

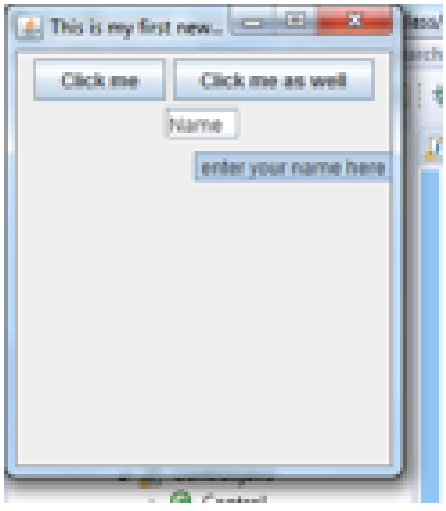
- implement the Listener interface (see notes) – for buttons, this is `ActionListener`
- Attach the listener to the GUI component that will be clickable;
- Put the response behaviour into the event handler method (`ActionPerformed` is the method name for ActionListener)

Part 4 – Make more components clickable

Make the second button clickable as well. Add the ActionListener to the second button – so that both buttons are being "listened" to.

They share the same event handler method (ActionPerformed). Clicking the first button should display one message and the second button a different message. *Your event handler methods, ActoinPerformed, will need to figure out which one was clicked.*

If you don't have one already, add a JTextField as shown in the diagram below, with default text "Name" in it. Include a tool tip that appears as the cursor goes on to the JTextField. JTextField has a method that allows you to add a tooltiptext as show in the left hand diagram.



Now add "event programming" to the JTextField. Attach a listener to it that activates when you enter in text to the JTextField (this event is captured by the same listener as buttons, `ActionListener`), and get it to pop up a display message as shown on right hand side with the name you entered (hint: look at "getText" method of the JTextField class).

Part 5 – capture "mouse" events

"MouseListener" is used to capture mouse events (e.g. mouse pressed, mouse clicked etc). Don't forget you can implement more than one interface in a class.

Add the following functionality, as shown in the picture:

Add a panel (JPanel class) in to your screen.
Create a JLabel with textin it, and add that label to your panel as shown below.
Set the background colour of the panel to red:

Add mouse event detection so that:

When the mouse enters the panel, it pops up a message "Mouse entered the panel";
When the mouse exits the panel, it pops up a message "Mouse left the panel";

When you click on the panel, it pops up a message to say clicked. See if you can get it to distinguish between “right” and “left” mouse clicks.