Practice Exercises: Teaching Block 4

- GUI
- Exception Handling
- File I/O
- Other practice exercise types: “Fill in the gaps”

This set of exercises is in addition to those included directly in lecture slides (and extra reading materials), which you should also attempt.

Question 1

- The following program is supposed to display a button in a frame, but nothing is displayed. What is the problem?


```
import javax.swing.*;

public class Test extends JFrame {
  public Test() {
    getContentPane().add(new JButton("OK"));
  }

  public static void main(String[] args) {
    JFrame frame = new JFrame();
    frame.setSize(100,200);
    frame.setVisible(true);
  }
}
```

`Test frame = new Test();`

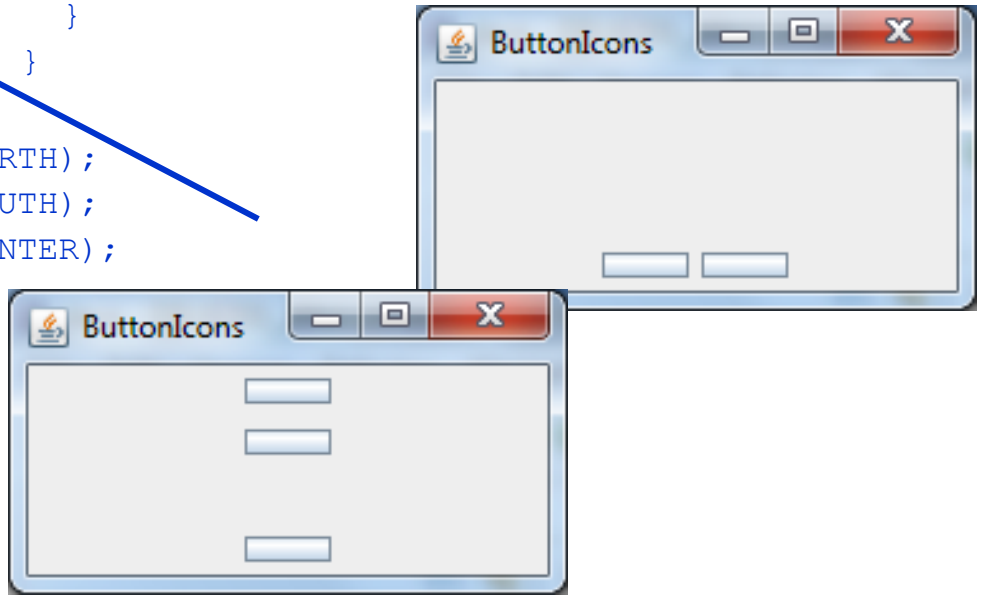
Question 2

- What happens when the code below is run? Will anything be displayed?

```
import java.awt.*;
import javax.swing.*;
public class Test extends JFrame {
  public Test() {
  JButton jbt1 = new JButton();
  JButton jbt2 = new JButton();
  JPanel p1 = new JPanel();
  p1.add(jbt1);
  JPanel p2 = new JPanel();
  p2.add(jbt2);
  JPanel p3 = new JPanel();
  p2.add(jbt1);
  getContentPane().add(p1, BorderLayout.NORTH);
  getContentPane().add(p2, BorderLayout.SOUTH);
  getContentPane().add(p3, BorderLayout.CENTER);
}
```

(code cont.)

```
  public static void main(String[] args) {
    // Create a frame and set its properties.
    JFrame frame = new Test();
    frame.setTitle("ButtonIcons");
    frame.setSize(220,120);
    frame.setDefaultCloseOperation(
                        JFrame.EXIT_ON_CLOSE);
    frame.setVisible(true);
  }
}
```

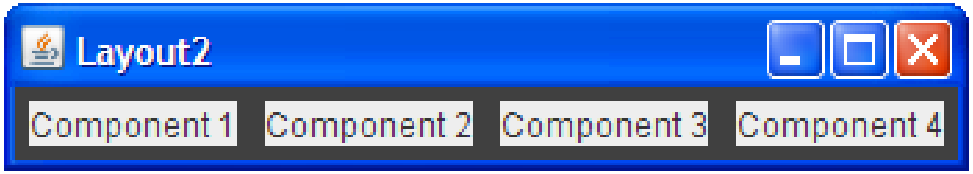


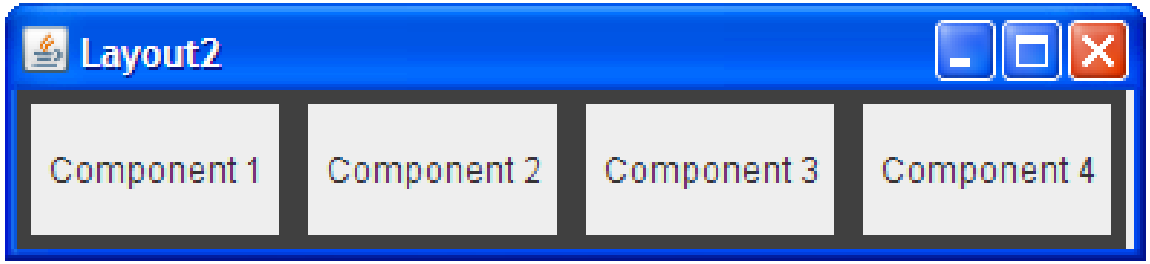
Question 3

- Choose the layout manager(s) most naturally suited for the following layout description, an example of which is given below: “the container has a row of components that should all be displayed at the same size, filling the container’s entire area”.

  a. **`FlowLayout`**
  b. **`GridLayout`**
  c. **`BorderLayout`**
  d. Options *a* and *b*.

Note: You can assume that the container controlled by the layout manager is a **`JPanel`**.



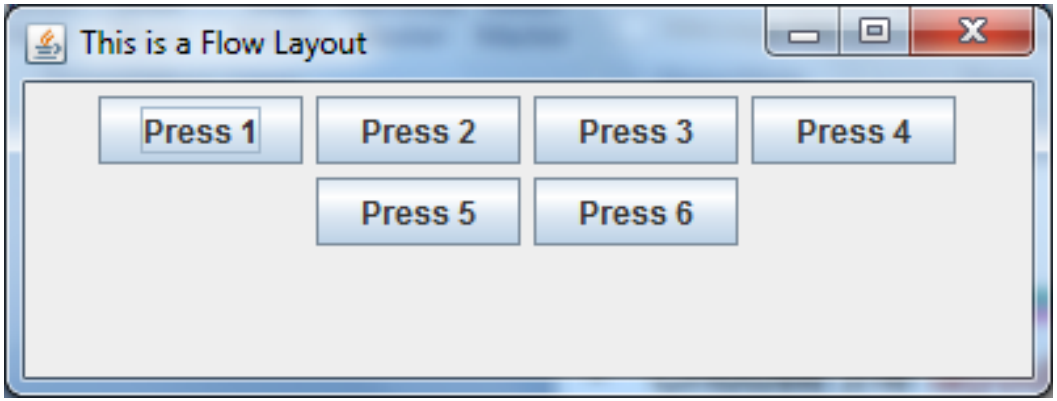


Question 3 (Answer)

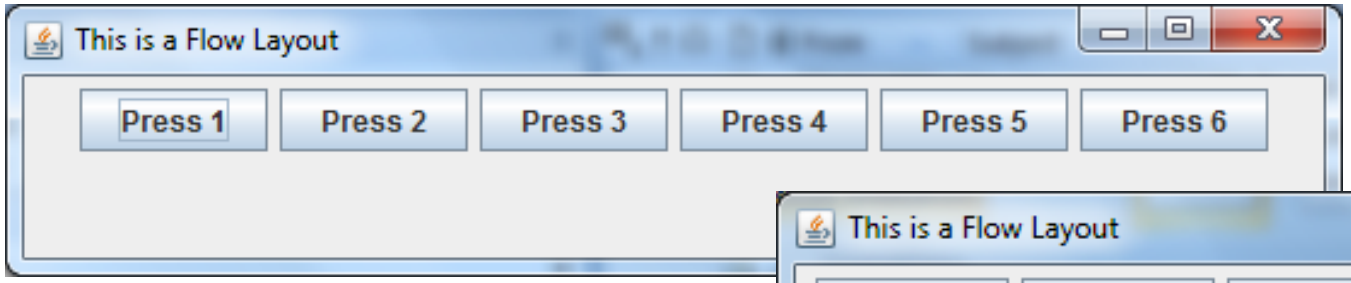
```
JFrame frame = new JFrame("Layout2");
JPanel myPanel = new JPanel(new
GridLayout(1,0));
myPanel.add(createComponent("Component 1"));
myPanel.add(createComponent("Component 2"));
myPanel.add(createComponent("Component 3"));
myPanel.add(createComponent("Component 4"));
frame.setContentPane(myPanel);
```

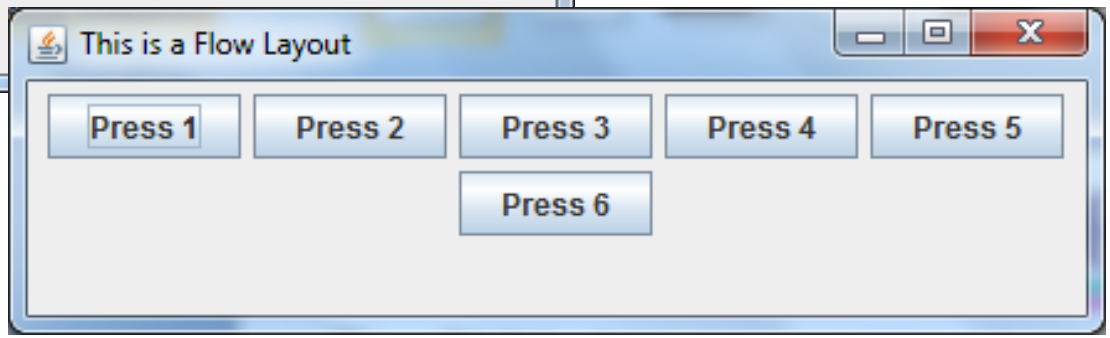
Question 4

- The GUI below uses a **`FlowLayout`** manager to arrange the display of the 6 buttons.
  - Write the Java code that generates this GUI.
  - What would happen to the displayed GUI if it was resized into a bigger window?



Possible outcomes …





Question 4 (Answer)

```
import javax.swing.*;
import java.awt.*;
public class TryFlowLayout {
  public static void main(String[] args) {
    int windowWidth = 400;
    int windowHeight = 150;
    JFrame aWindow = new JFrame("This is a Flow Layout");
    aWindow.setBounds(100, 100, windowWidth, windowHeight);
    aWindow.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    FlowLayout flow = new FlowLayout();
    Container content = aWindow.getContentPane();
    content.setLayout(flow);
    for (int i = 1; i <= 6; i++)
      content.add(new JButton("Press " + i));
    aWindow.setVisible(true);
  }
}
```

Question 5

- Consider a command-line calculator application. Write a program to deal with non-numeric operands, so it displays a message telling the user of the wrong operand type before exiting using an *exception handler*.

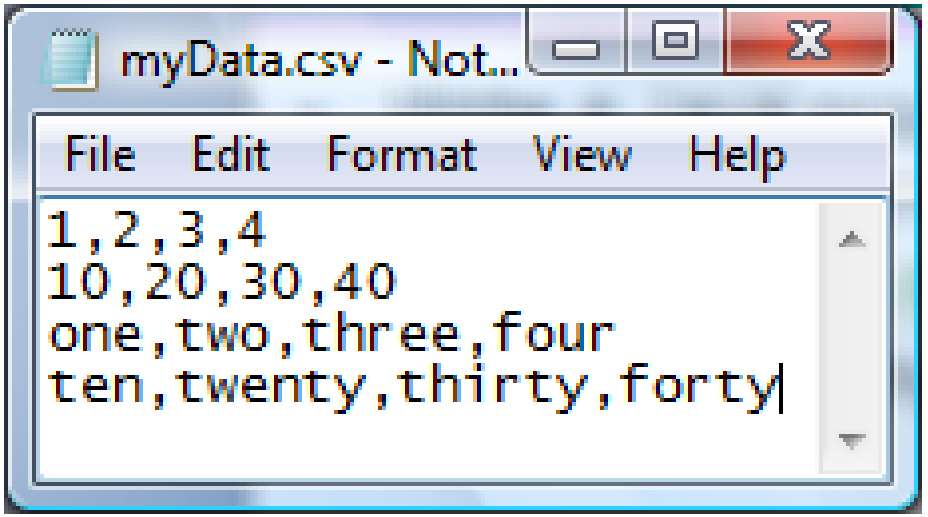
```
public class Calculator {
  public static void main(String[] args) {
    if (args.length != 3) {
      System.out.println("Usage: java Calculator op1 operator op2");
      System.exit(0);
    }
    int result = 0;
    switch (args[1].charAt(0)) {
      case '+': result = Integer.parseInt(args[0]) +
                         Integer.parseInt(args[2]); break;
      case '-': result = Integer.parseInt(args[0]) -
                         Integer.parseInt(args[2]); break;
      // Similarly for the '*' and '/' operations ...
    }
    System.out.println(args[0] + args[1] + args[2] + " = " + result);
  }
}
```

Can you think of other potential problems with this program?

Question 6

- Write a Java program that reads each line of the data in a file called **`myData.csv`** and stores it in an array, after which it *displays the first element of the array onto the console* for each line of the file. Below is the expected output for running the program with the file shown:

```
> java FileProcessor
1st element of line 1 = 1
1st element of line 2 = 10
1st element of line 3 = one
1st element of line 4 = ten
```



What will happen if the file **`myData.csv`** does not exist?

Exercise: Fill in the gaps (part 1/2)

```
public class InvalidAgeException ____(1)_____ {
  public InvalidAgeException() {
    super("Invalid Age Was Entered...");
  }
}
```

(1) The InvalidAgeException class needs to extend the _____ class.
(2) The PersonAge class uses the Scanner class; therefore we are required to _____.

```
import java.io.*; ____(2)_____;
public class PersonAge {
  int age;
  AgeValidity myAge;
  Scanner input;
  public void checkAge() throws InvalidAgeException {
    input = new Scanner(System.in);
    myAge = new AgeValidity();
    System.out.println("Please enter your age : ");
    age = input.nextInt();
    if (!myAge.checkAgeValidity(age))
      ____(3)_____ InvalidAgeException();
  }
}
```

(3) The _____ keyword needs to be used before the InvalidAgeException class name to show the occurrence of the error.

Exercise: Fill in the gaps (part 2/2)

```
public class AgeValidity {
  public AgeValidity() { }
  public boolean checkAgeValidity(int age) {
    return (age > 0 && age < 100);
  }
}
```

(4) In the AgeChecker class, the theAge.CheckAge() statement will raise an error because _____.

```
public class AgeChecker {
  public AgeChecker() { }
  public static void main (String [] args) {
    PersonAge theAge = new PersonAge();
    theAge.checkAge(); (4)
  }
}
```