Problem 1

public class SimpleGUI extends JFrame {
    private boolean isYellow; // Is the circle yellow or black?
    public SimpleGUI() {
        isYellow = true;
        JPanel circle = new CirclePanel();
        this.add(circle);
        this.setVisible(true);
    }
}

class CirclePanel extends JPanel {
    protected void paintComponent(Graphics g) {
        if (isYellow) g.setColor(Color.yellow);
        else g.setColor(Color.black);
        g.fillOval(0, 0, 100, 100);
    }
}

class CircleListener implements MouseListener {
    public void mouseClicked(MouseEvent e) {
        System.out.println("Clicking the circle!");
    }
}
}
a) Right now the code does not compile because of a problem with the CircleListener class. What is the problem?

```
class CircleListener implements MouseListener {
    public void mouseClicked( MouseEvent e ) {
        System.out.println( "Clicking the circle!" );
    }
}
```

CircleListener does not implement all the required methods of MouseListener interface. Fixed:

```
class CircleListener implements MouseListener {
    public void mouseClicked( MouseEvent e ) {
        System.out.println( "Clicking the circle!" );
    }
    public void mouseExited(MouseEvent e){};
    public void mouseEntered(MouseEvent e){};
    public void mouseReleased(MouseEvent e){};
    public void mousePressed(MouseEvent e){};
}
```
b) Assume that the issue from question 1 is fixed and the code compiles. The code runs and displays a yellow circle, but when the user clicks on it, no message is printed. Add code to the SimpleGUI constructor above to make the message “Clicking the circle” print when the user clicks on the yellow circle in the GUI.

```java
public SimpleGUI() {
    isYellow = true;
    JPanel circle = new CirclePanel();
    this.add(circle);
    this.setVisible(true);
}

Add an instance of a CircleListener as MouseListener of the circle object.

```java
public SimpleGUI() {
    isYellow = true;
    JPanel circle = new CirclePanel();
    circle.addMouseListener(new CircleListener());
    this.add(circle);
    this.setVisible(true);
}
```
c) Below, write a new the mouseClicked method (to replace the one above) so that the circle changes from yellow to black when it is clicked, and then from black to yellow the next time it is clicked, etc so that each time the user clicks the circle it either changes from yellow to black or from black to yellow.

Change the variable isYellow to the opposite value;

public void mouseClicked( MouseEvent e ) {
    System.out.println( "Clicking the circle!" );
    isYellow = !isYellow;
    repaint();
}
EXAM 4 - Solutions

Problem 2

Series: \[ mySequence(i) = 1 + \frac{1}{2} + \frac{1}{3} + \frac{1}{4} + \cdots + \frac{1}{i} \]

a. Your friend tries to convince you that the base case should be the following:
\[
\text{if ( mySequence(i) == 1 ) return 1;}
\]
But you know better. Explain why your friend is wrong. Make sure to include in your explanation what would happen if you used this as your base case in your method?

Base case should not have a recursive call, it should just stop execution and return the value 1. If we leave the call to mySequence in the base case and without modifying i, then the execution will never terminate and it will consume the memory very fast.
b) Below, fill in the code to complete the recursive method mySequence as defined above.

```java
public double mySequence(int i) {
    if (i == 1)
        return 1;
    else
        return 1/((double)i) + mySequence(i-1);
}
```
PSA 7 - Solution

We gave you this...

```
LetterGenerator: JApplet

-factory

canvas: JPanel

GraphicLetterFactory
```
Add ActionListener
canvas.removeAll();
textField.setText("");
repaint();

Add KeyListener
keyPressed(KeyEvent e){
    GraphicLetter letter = factory.getRandomCharacterIgnoreCase(key);
canvas.add(letter);
}

GraphicLetterFactory

GraphicLetter_cs8sae1 GraphicLetter_cs8mae2

LetterGenerator:JApplet

canvas: JPanel

JPanel

JTextField JButton

Add ActionListener
Canvas
TextField
Button
PSA 7 - Solution

LET SEE THE CODE!!!
Base cases:
- Both strings are empty, return true;
- One one is empty, return false;
- First character of each string are different, return false;

Recursive call:
- Call match again without the first letter.

```java
public boolean match( String s1, String s2 )
{
    if ( s1.length() == 0 && s2.length() == 0 )
        return true;
    else if ( s1.length() == 0 || s2.length() == 0 )
        return false;
    else if ( s1.charAt( 0 ) != s2.charAt( 0 ) )
        return false;
    else
        return match( s1.substring( 1 ), s2.substring( 1 ) );
}
```
Recursion

Base case:
- Input is 0, then do nothing or just print a new line.

Recursive call:
- Print the input mod 10 (the remainder)
- Call printReverseDigits with the quotient of input / 10

```java
public void printReverseDigits( int input )
{
    if ( input == 0 )
        System.out.println();
    else {
        System.out.print( input % 10 );
        printReverseDigits( input / 10 );
    }
}
```