CSSE 220 – Object-Oriented Software Development
Rose-Hulman Institute of Technology

Worksheet 11

Name (Print):_____ Section:_____

1. _____: describes how the coordinate system is changed before drawing.
2. Which of the following does a `Graphics2D` transform control? (Check all that apply)
   - ☐ Movement of the origin
   - ☐ The color of shapes
   - ☐ Rotation of the coordinate system
   - ☐ Scaling (zoom in / zoom out)
   - ☐ Stroke thickness
3. True / False `Graphics2D` keeps its current transform state until it is explicitly changed or restored.
4. Complete the code below to:
   - move the origin 50 pixels to the right and 100 pixels down
   - rotate the coordinate system by 45 degrees

```
Graphics2D g2 = (Graphics2D) g;

// TODO: move the origin  ____________________________________

// TODO: rotate the coordinates  ______________________________

// TODO: unrotate  ___________________________________

// TODO: untranslate  ________________________________
```

5. Fill in the missing method calls:

```
// 1. copy the current state

AffineTransform old = g2._________________();
// ...  translate / rotate change

// 2. change back to original state

g2._________________(old);
```

6. _____: a programming technique where a method calls itself to solve a problem.

7. Name the two essential parts of recursion:

   (a) _____

   (b) _____

8. In one or two sentences, explain why every recursive method must have a base case:

   _____

9. Identify which part is a base case and which is a recursive case:

```
factorial(1) = 1                          ____________ case
factorial(n) = n * factorial(n - 1)       ____________ case
```

10. Complete the recursive method below so that it draws a stack of diagonal lines. The method should:

    - stop when `n == 0`
    - draw one line
    - move the origin down
    - call itself with a smaller value of `n`
    - restore the transform before returning

```
void drawLineStack(Graphics2D g2, int n) {

    // TODO: base case
    ----------------------------------

    // TODO: save the current transform
    AffineTransform old = ----------------------------------;

    // TODO: draw one line
    g2.drawLine(0, 0, 50, 50);

    // TODO: move the origin down
    g2.--------------------------------;

    // TODO: recursive call
    ---------------------------------;

    // TODO: restore the transform
    g2.--------------------------------;
}
```

11. **Before You Leave**

    Write one question you still have about today's topic.