# CS 273 Laboratory 3: Introduction to Graphics

This laboratory begins by giving you practice fixing compile-time errors in some simple graphics-oriented programs. It then continues by having you create some of your own graphics programs.

## Preliminaries

* Load the lab3 project in BlueJ. This lab contains seven Java classes. Each class contains a separate Java program. Once a program is compiled, you can run it by right-clicking on the class and selecting the main() method.

## Laboratory

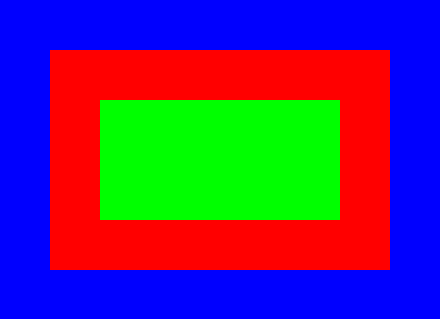
### Part 1: Fix some compile-time errors.

The classes E1, E2, and E3 each contain Java code that fails to compile because each has one or more source-code errors. Your job is to fix the errors so that the programs compile. If you fix the errors in a "straightforward" manner, you should also be able to run each program and have something happen.

**checkpoint 1 (20 points): Have your lab instructor or assistant verify that you can run all three programs.**

### 

### Part 2: Three concentric boxes*.*



Open the Boxes class. Modify the paint() method so that it displays three *concentric*, *filled-in* boxes (see the example to the right). The inner box should be green; the middle box should be red; the outer box should be blue. **All boxes should be drawn on a white background.**

You can use whatever size boxes you’d like. Note that the order in which these boxes are drawn is important. Each box that is drawn will cover previously drawn boxes.

Set a breakpoint in the debugger so that you can see the boxes being drawn one at a time.

**checkpoint 2 (20 points): Run your program in front of your lab instructor or assistant. Use the debugger to show them the boxes being drawn one at a time.**

### Part 3: Draw a large, green triangle.

Open the Triangle class. To draw a triangle, Java’s Graphics class has a method called drawPolygon() which is designed to be used with Java’s Polygon class. Read about the Polygon class in your textbook or on the Internet (<https://docs.oracle.com/javase/8/docs/api/java/awt/Polygon.html>); note that the drawPolygon() method is described in the documentation for the Graphics class (https://docs.oracle.com/javase/8/docs/api/java/awt/Graphics.html). Modify the paint() method in the Triangle.java file so that a large, green, filled-in triangle appears in a window on a **yellow** background.

To do this, start by creating a Polygon object and add points to it.

**checkpoint 3 (20 points): Show your code and output to your lab instructor or assistant.**

### 

### Part 4: Create a program that draws a house.

Open the House class. Modify the paint() method in the .java file so that it displays a house. Your house should have a roof, a door, at least one visible window and at least one additional feature of your choice (e.g., a lawn, trees, a chimney, blue sky, a mailbox, a doorknob, etc.). You can also "enhance" the window so that it has multiple panes, shutters, etc. Choose colors that you find visually pleasing. For ideas, you might want to review the colors in the *Graphics in Java* PDF included in the lab.

It might be difficult for someone reading your code to tell which lines of code are drawing which parts of the house. Help them out by giving helpful names to all Polygon objects, and leaving a brief comment for each other bit of code where it’s not obvious what it’s drawing.

**checkpoint 4 (20 points): Show your code and output to your lab instructor or assistant. Show how you made your code readable via variable names and comments.**

### Part 5: Create a program that draws a jack-o-lantern

Open the Jackolantern class. Modify the paint() method in the .java file so that it displays a reasonably nice-looking jack‑o‑lantern. The jack-o-lantern’s expression should clearly indicate a strong emotion (fear, joy, disgust, etc.). The jack-o-lantern must have a stem. Its body must consist of at least a couple ridges (it cannot be a single oval or circle).

Use helpful variable names and comments to make it clear what each part of your code is doing.

**checkpoint 5 (20 points): Show your code and output to your lab instructor or assistant.** **Show how you made your code readable via variable names and comments.**

### 

### Extra Credit: Add a swimming pool to your yard

Add a swimming pool to the yard in front of the house. The pool should have a diving board and at least two swimmers. These features should be clearly identifiable.

Use helpful variable names and comments to make it clear what each part of your code is doing.

**checkpoint EC1 (5 points): Show your code and output to your lab instructor or assistant.** **Show how you made your code readable via variable names and comments.**