# Lab 2 — Interfaces and Polymorphism

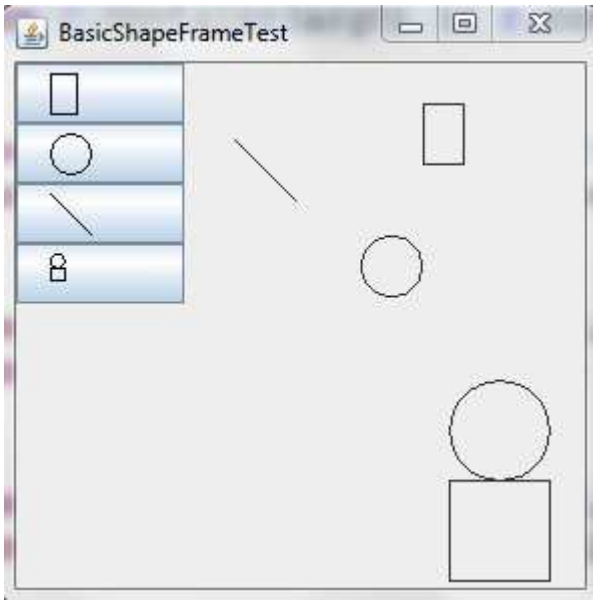
CS 205

Due Oct. 6 at 11:55 pm

Lab objectives:

- Implement interfaces to achieve polymorphism.

You will modify a simple program which displays shapes in a panel. The user selects a shape from a list of buttons and then clicks on the panel to add that shape. You will add another button which will contain a compound shape of your choosing. This which will require you to implement the Shape and PathIterator interfaces for your compound shape.




1. Pairs for this lab:
   - Alex and Jay
   - Justin and Nico
   - Sam, Surbhi, and Drew
   - Anthony and Conor
   - John and Jordan
   - Alison and Weston
2. In pair-programming teams, download lab2Starter.zip, import the project into Eclipse and try it out. Spend some time looking through the code so that you understand what is happening.

3. Your job is to add another button with some compound shape without modifying any of the files except for the BasicShapeFrameTest which will add your button. Your compound shape should consist of at least two *different* basic shapes. I created a compound shape made up of a circle and a square, as you can see above. The addShape method used in BasicShapeFrameTest takes a Shape parameter. Thus, you need to produce an object that implements the Shape interface.

   You will need to create two classes to implement the Shape interface and to implement the PathIterator interface. Your classes will most likely contain instance variables containing the shape objects that make up the compound shape. You will make extensive use of the Shape and PathIterator methods already defined for you within these shapes. These methods will do the bulk of the work for you.

   Most of the methods required by the Shape interface are simple to implement. The only moderately challenging ones involve PathIterator. For implementing PathIterator, you will want to make extensive use of the same PathIterator methods of your shapes that make up the compound shape. For example, for the currentSegment method you will return a currentSegment of one of the shapes making up the compound shape. When all the segments of one shape are completed (determined by the isDone method), then you can return the currentSegment from the next shape.

4. Add Javadoc comments to record the members of your team in the BasicShapeFrameTest file and also to document the code that you add to the project. Export your lab into a ZIP archive and submit it in GoucherLearn.