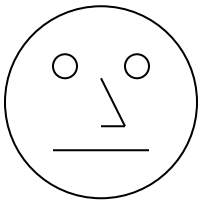
**CMSC 113: Computer Science I**
**Midterm Project: Pong**
**due on Gradescope by the beginning of class on February 27, 2018**

The Pong Project is broken into three parts. It is best to proceed by doing one part at a time. Get everything about Part I working before going on to Part II, for example.

**Collaboration Policy:** You may choose to work with a partner for this project. If you work with a partner, let me know as soon as you make your decision. If you choose not to work with a partner, you must follow the guideline "If you're talking in Java, you've gone too far." Remember, I take a breach of this policy seriously – if you're uncertain, ask before making any assumptions.

**Part I.** Write an applet that shows a ball bouncing back and forth horizontally across the screen. Assume the applet is 200 pixels wide by 200 pixels tall. Use a compound object class to represent the ball. Your compound object must have a method named `update`, which moves the ball one step forward and handles bouncing. All of the logic around bouncing must be in the `Ball` compound object. Recall that compound objects work best if the drawn elements within the object are centered at or near the hotspot (the origin in the local coordinates of the object).

**Extra challenge**: Make the ball speed up after every bounce. To ensure the ball speeds up in a smooth manner, use `double` fields to store the speed and location of the ball. In the online example, the ball's speed after a bounce is 1.05 times its speed before the bounce.



A lot of people have trouble getting this to work. They think that when you detect that the ball has gone past the right edge, make it move left. But that doesn't work! The ball does move left one pixel, but then moves right again. The trick is to have a field to control the ball's direction.

**Part II.** Modify your applet to allow the ball to bounce up and down as well as left and right. This means that your ball will always be traveling at a diagonal. Add two paddles to your applet and make it so that they always track the ball. No part of either paddle should ever go off the screen. When the ball hits a paddle, it should bounce off. Note that the ball never reaches the top or bottom of the applet. For clarification, please see the example online.

Use a class `Paddle` and have two fields of that class in your applet (in other words, have two `private Paddles` in your `GraphicsProgram` file). Your `Paddle` class should have a `moveTo` method that takes an *x*-coordinate to move to as a parameter. The method should check to make sure the paddle will stay on the screen and then move. All the logic around keeping the paddle on the screen must be in the `Paddle` class.

**Extra challenge**: Make it so that the paddles have a maximum speed. Normally, the paddles are always underneath or above the ball; in the Extra Challenge version, the paddles will always try their hardest to keep up. If the ball is moving too fast, the paddle would be unable to catch it.

**Part III.** Add the following features:

- The lower paddle should no longer follow the ball. It should instead respond to key presses. When the user presses and holds the left arrow, it should move left. When the user presses and holds the right arrow, it should move right. The upper paddle should continue to track the ball. There should be no pause between when the user presses an arrow and when the paddle starts moving. To implement this, you will add a `moveBy` method to your `Paddle` class. This method takes a parameter saying how much the paddle should move by; negative numbers move the paddle left, and positive numbers move the paddle right. The paddle must still stay on the screen at all times.
- If the ball passes the player's paddle (the bottom one) without striking it, the game of Pong should end and the applet should say whether the computer or the player won. Note that under the current implementation, it is impossible to win (because the computer's paddle always tracks the ball). You will have to add a parameter to the `Ball`'s `update` method to do this. The ball will use that parameter to determine whether or not it has hit the user's paddle.
- When the applet loads, the game should not start immediately. The game will start on a mouse click.

While I have made specific suggestions about code design above, make sure that, as much as possible, ball-related code goes in the `Ball` class and paddle-related code goes in the `Paddle` class.

**Extra challenge**: Allow the ball to travel in an arbitrary direction (not just at a 45 degree angle). Have the ball bounce off paddles so that, as examples, a bounce very near the left edge of the paddle bounces hard to the left, a bounce near the center is near vertical, and a bounce ¾ of the way to the right side yields a 45 degree angle to the right.

When you have completed coding this project, make a *reflections.txt* file (following the instructions on the Warmup assignment), answering these questions:

1. Did you complete all parts of the assignment?
2. If not, which parts were you unable to complete and why?
3. How long did this assignment take you?
4. What was the most challenging part?
5. Do you have any other questions or experiences to share?

Export your project (following the instructions in the Warmup assignment) and post on Gradescope.