Oreo McFlurry
Jing Yi Feng & Emily Ortiz
APCS Pd 6

Description
For our final project, we are creating the classic game of Simon. The game consists of four squares that are green, red, blue, and yellow. A random square will light up and the player has to click on the right color. This repeats and each time the player has to replicate the same sequence of the squares that have lighted up before. Each color square also has a distinct sound that will play when the square is pressed. The game ends when the time runs out (player doesn't press the square fast enough) or if the player presses the wrong sequence.

Implementation
We are working primarily on Processing.

To set up the game, four square shapes are created and assigned the appropriate colors and places. There is a counter in the center showing how many points the player has (starts at 0). There is also an int storing the high score. An int ArrayList is initialized which stores the color pattern.

Numbers:
0 = sq0 = green (top left)
1 = sq1 = red (top right)
2 = sq2 = yellow (bottom left)
3 = sq3 = blue (bottom right)

This configuration →

A random number is generated and added to the ArrayList. Then an instance of the AL iterator is created (itr0). The iterator traverses the list, and for each number in it, the appropriate square lights up. Once that is done, itr1 is created (or reinitialized).

A timer for 1 second (change time if necessary). If this timer reaches 0, the game is over. The timer is reset when the mouse is clicked. Also on every mouse click: check if the square matches the number itr1 is on. If it does, the appropriate music note will be played and the square will "flash" or become a lighter color briefly and return to its original color. itr1 then moves onto the next element. If the click isn't on the right square, the game is over.

Once itr1 doesn't have a next element, the score counter adds one. A random color is added to the AL, itr0 is reinitialized, and the process repeats.

When the game is over: a box pops up saying the player's score, high score, and asks them if they want to play again. To play again, rest everything except the high score.

Extra notes / potential problems / To-do:
If the mouse is clicked before the complete pattern is shown, the click won't impact anything and the pattern continues. If itr0 has a next element, the click won't count.

Make sure the program doesn't break when the player clicks a square before another square is done lighting up and playing sound.

Look into buttons in Processing and see if they're more efficient than square shapes.

Think of how to implement more spring semester topics.