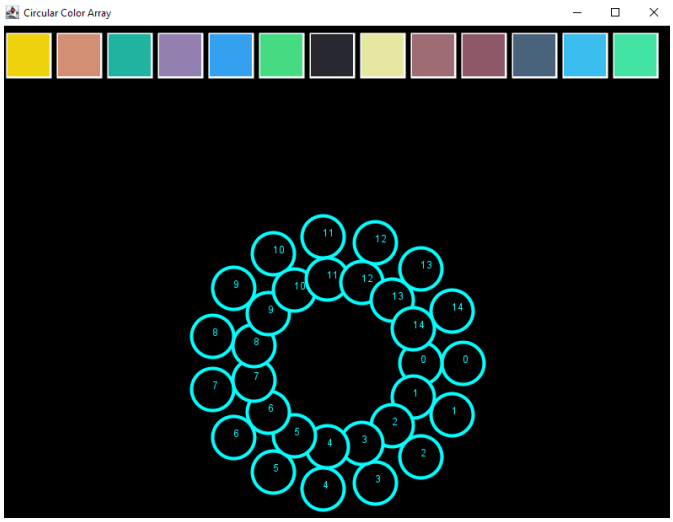
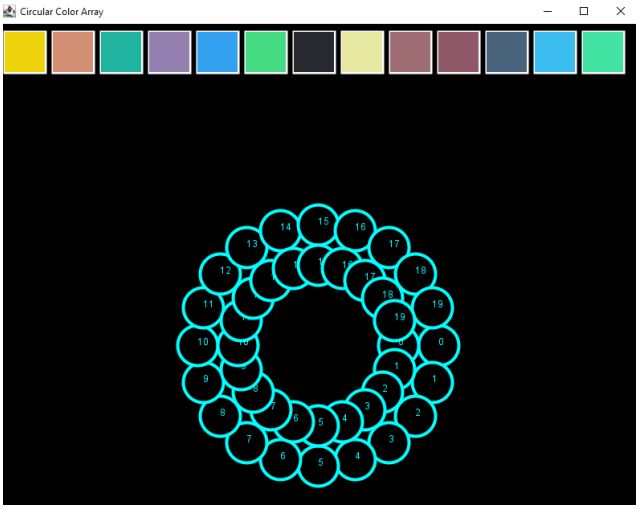
HW1: Circular Arrays

You can start with the HW1Starter, which handles most of the visuals for this assignment.

The top bar has some buttons in random colors. In the center you should have two circles made out of smaller circles. Each of the large circles represents a circular array, and each individual small circles is one item or index. The indexes are numbered. The asdf keys change the number of items that comprise these arrays, and also reset them back to the default state.





(75%) Your task is to turn these into functional circular arrays that represent a series of queues. Clicking on one of the colored buttons at the top should add that color to the inner queue. Once that queue is full, clicking a button will add to the outer queue. Pressing the “space” key should change the background color of the application to the oldest Color in the inner array (hint: you can draw a rectangle that spans the entire panel), and remove that color from the list. If the inner array is empty, pull from the oldest in the outer array instead.

(25%) Keep spawning more circular arrays if the outer one fills up. The new one should wrap directly around the outer array you started with (you can simply add 50 to the outer array’s radius). For full credit your application should do this indefinitely, just keep growing as the new outer layer fills up, so after the radius 200 array fills up, a radius 250 spawns, and so forth. A reset should also reset these arrays, and the number of items they contain should match the others like outer/inner currently do.

To summarize: clicking a button always adds a Color to the innermost available queue, and hitting space always removes the oldest Color in the innermost queue and sets the background to that color.

Hint: the most simple solution utilizes ArrayList...

(Extra credit 5%) The background color is a gradient using the last two colors that have been popped.

(Extra credit 10%) Replace the circles with slices so that each array looks like a pie chart (like on the right). You’ll end up with a bunch of nested pie charts. Hint: this is actually pretty easy, there’s a method on the Graphics2D tool that handles all the heavy lifting for you….

As always, up to 5% for exceptional documentation, especially method and class headers.