**Addition Challenge – Circle Pack Fractal**

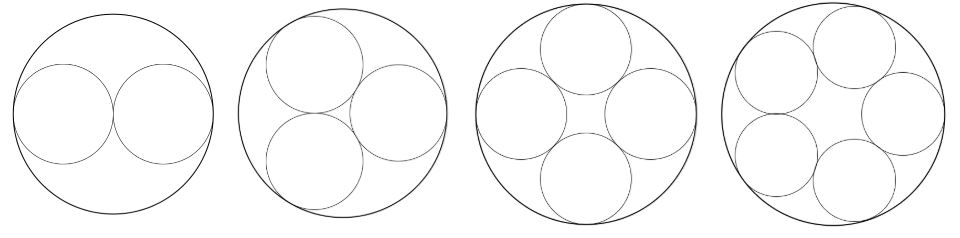
If you've finished your **Create-a-Fractal** project a little sooner than the rest of the class, here is a fractal problem to push your geometry and computer science skills a little further.

**The Task**

Create a fractal with the following production rules:

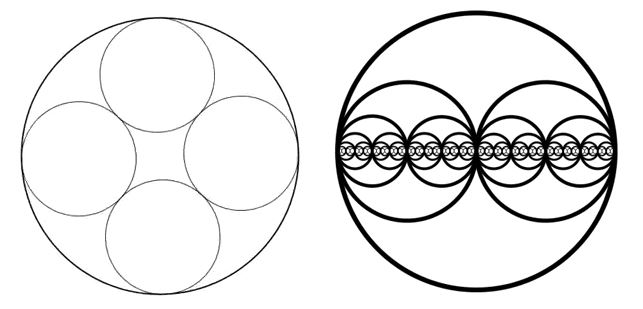
1. Create a Circle
2. For a given number (2 or more):
   * Find that many centrepoints inside the circle AND
   * determine the size SO THAT
   * that many circles can be recursively created (these circles should fit perfectly within the circle)

For example, the following would be results for different given numbers of circles:



Your sketch should be **interactive** so that:

* The number recursion depth can be controlled by the user
* The number of circles to fit in each circle can be controlled by the user

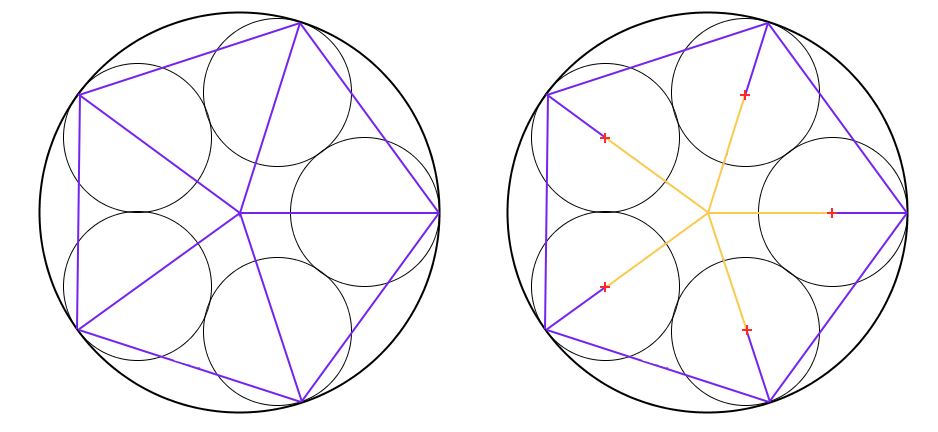


**Breaking the problem down:**

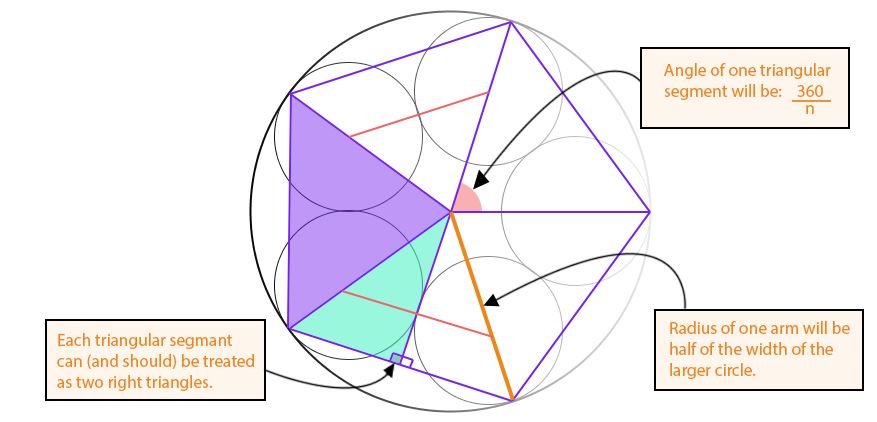
The primary challenge in solving this problem is determining two pieces of information for each smaller circle that is to be recursively drawn:

* Where the circle's centrepoint is (with relationship to the larger circle)
* The size that the smaller circles need to be to fit perfectly

One possible way to attack this problem is to consider an **inscribed regular polygon**, composed of **triangular segments**. Each of the edges extending from the center pass through each of the centerpoints we need to find! The unknown quantity that needs to be determined is how far along those edges the centrepoints lie.



 There are a few **known quantities**, which will come in useful:



Using trigonometry, you need to establish a relationship between the **angle in the right triangle** and **radius** of the larger circle and the **distance along the highlighted edge** and **radius** of the smaller circles,

**Polar Coordinates to Cartesian Coordinates**

The work done above finds the locations of interest using **Polar Coordinates,** where the locations are expressed using an **angle**and a **distance.** This is a natural system for the problem above, but for implementing the solution in code recursively, translating from **Polar Coordinates** to **Cartesian Coordinates (x,y)** will simplify the process.

This translation is not difficult; a tutorial outlining how it works can be seen [here.](https://processing.org/examples/polartocartesian.html)

**Make it Beautiful**

Once the base fractal has been expressed, experiment with Stroke and Fill Colors, Stroke Weight, Background Color, and Transparency to create a stunning final product.