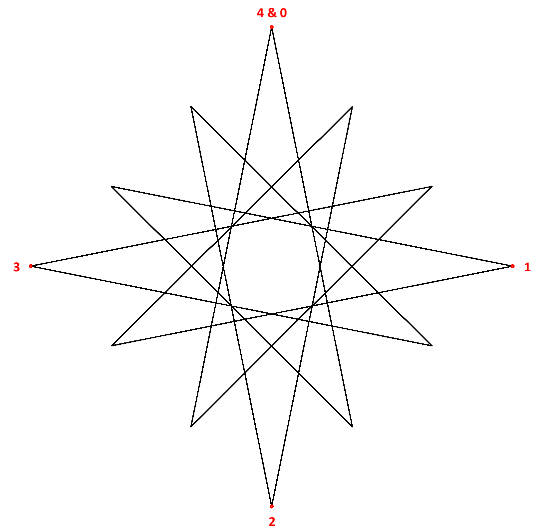
Where the 1st Cycle ends tells us how the Image is Filled In

Finding *E*. If the first cycle ends at the top, the image is complete. In all other circumstances, there will be multiple cycles to complete the circuit (or image). There are ***C*** lines per cycle, and each line is created by counting out ***P*** subdivisions on the vertex frame. But note that a polygonal vertex occurs every ***S*** subdivisions on the vertex frame and, by construction, this point is ***J*** polygonal vertices away from the 0, the top of the parent polygon. Putting all of this together, we see that *the endpoint of the first cycle*, ***E***, is:



E = MOD($\frac{C*P}{S}$**J*, ***n***) where MOD is the remainder upon division by ***n***.

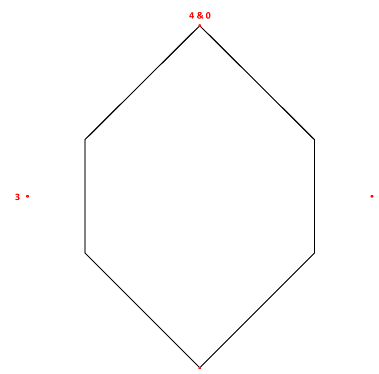
The image will appear to add polygonal vertices to the image in a ↻ (clockwise) direction if ***E*** < ***n***/2 *because* the 2nd endpoint will be at 2***E*** < ***n***.

If ***E*** > ***n***/2, the polygonal vertices appear to be added to the image in a ↺ (counterclockwise) direction *because* MOD(2***E***, ***n***) < ***E***.

An Example: Consider two ways of drawing the image to the right. Quite clearly, ***S*** = 3 and ***n*** = 4. ***J*** can be either 1 or 3, let ***J*** = 1. The first cycle must start at the top so the end of the first segment in the cycle is the last subdivision before vertex 2 or the first subdivision after vertex 2 on the vertex frame. Simple counting suggests that ***P*** = 5 or ***P*** = 7 and ***C*** = 3.

P = 5: Then ***E*** = MOD(3*5/3, 4) = 1 and polygonal vertices are added in a ↻ direction from 1 to 2 to 3 to 4&0.

P = 7: Then ***E*** = MOD(3*7/3, 4) = 3 and polygonal vertices are added in a ↺ direction from 3 to 2 to 1 to 4&0.



Finally, if ***E*** = ***n***/2 the image will have two and only two cycles which are mirror images of one another. This occurs because the second cycle ends at the top of the parent polygon: 0 = MOD(2***E***, ***n***) if ***E*** = ***n***/2. The hexagonal image to the right has the same ***S***, ***n***, and ***J*** as above but now ***P*** = 2 or ***P*** = 10.

P = 2: Then ***E*** = MOD(3*2/3, 4) = 2 and the 1st cycle is the right half of the image.

P = 10: Then ***E*** = MOD(3*10/3, 4) = 2 and the 1st cycle is the left half of the image.

In both instances, the second cycle completes the image.

How *E* tells us how the image is filled in. The size of ***E*** relative to the number of cycles in the image, ***M***, tells us how many times around the parent polygon vertices the drawing must lay down cycles to fill in the image. For example, this image has ***E*** = 3 with 29 cycles and ***n*** = 29 so it takes *3x-around* to fill in vertices 1 and 2.

Click *Toggle Drawing*: https://www.playingwithpolygons.com?vertex=29&subdivisions=12&points=224&jumps=13 .

By contrast, this ***E*** = 4 image is a *2x-around image* since the number of cycles, ***M*** = 15, so only half the vertices are points on the image: https://www.playingwithpolygons.com?vertex=30&subdivisions=3&points=58&jumps=13 .

More generally, the number of times around the parent polygon vertices one must travel to fill in the image, ***T***, is given by an equation that depends on whether those additions are counter-clockwise (***E*** > ***n***/2) or clockwise (***E*** < ***n***/2):

Times around to fill in the image, ***T*** = IF(***E*** > ***n***/2, (***n***-***E***)/(***n***/***M***), ***E***/(**n**/***M***)).

1x-around images. If ***T*** = 1, then the image is created in a series 1 used-vertex changes around the parent polygon. This may involve a large number of rotations about the vertex frame, but the cycle moves one used-vertex at a time. Additionally, if ***n*** = ***M***, the images is drawn 1 vertex at a time. The way such images are drawn can be quite dramatic:

↻ because ***E*** = 1: https://www.playingwithpolygons.com?vertex=30&subdivisions=11&points=127&jumps=13

↺ because ***E*** = ***n***-1: https://www.playingwithpolygons.com?vertex=30&subdivisions=12&points=233&jumps=13