Problem Statement

Create 2D Koch Snowflake
Create 3D Koch Snowflake

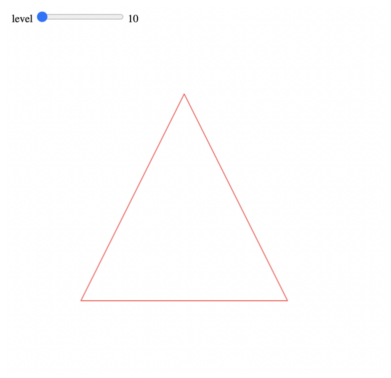
Implementation and Observation

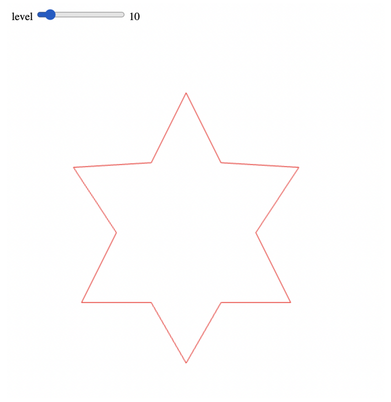
2D snowflake was relatively easy. I used the the midpoint formula to calculate the mid of the given side, and then calculated the height of the resultant smaller triangle by calculating the side of the new triangle, which would be one third of the original triangle. I used the mid and the height to find the next point. These points are added recursively to the array buffer, as they need to be continuos for the LINE_LOOP function to draw lines to form a closed shape. I incorporated slider to switch the level of division.

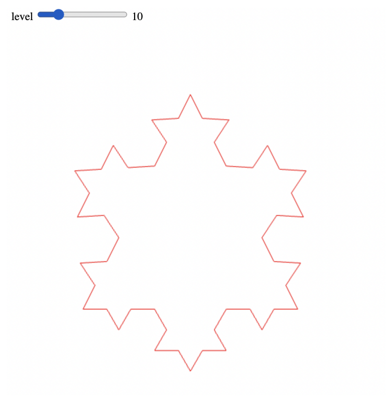
3D snowflake was difficult to implement as it is difficult to calculate the fourth point of the tetrahedron. I started with calculating the centroid, and then total height of the tetrahedron by using the formula $\sqrt{(2/3)} \times side$. Then, if we take the cross product of any two vectors of the triangle, we get the normal vector which is perpendicular to the traiangle surface. We then translate the centroid along the unit normal vector. We still have two directions of the normal. To choose the appropriate direction, I calculate distance between the supposed fourth point and the opposite vertex of the face we are considering. Larger distance is what we want for the tetrahedron to pop out. This way I select the direction, and place the point. I keep on adding the new faces to the fringe, and once the fringe is empty, I reduce the level of division (depth) and then continue the fringe.

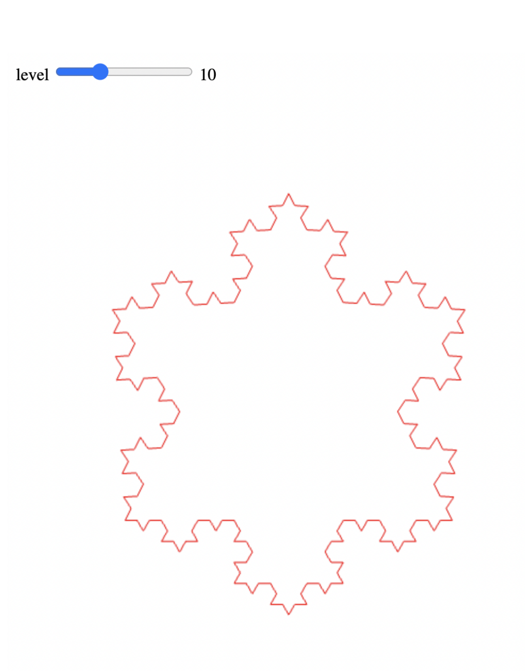
Result

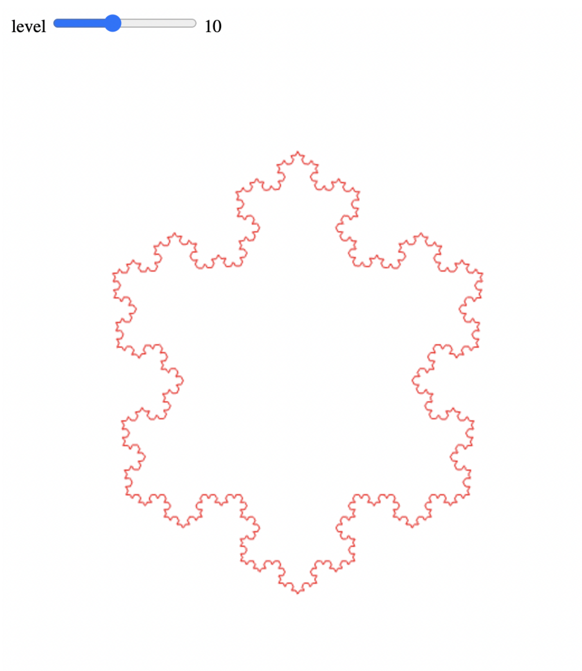
2D Koch Snowflake



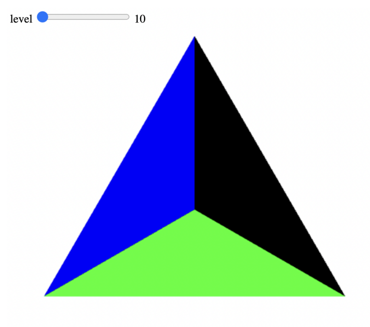


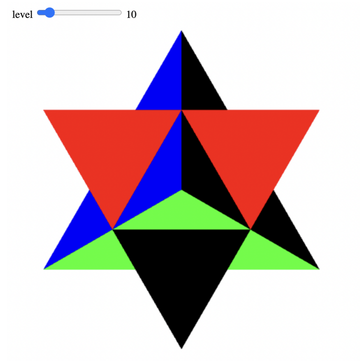


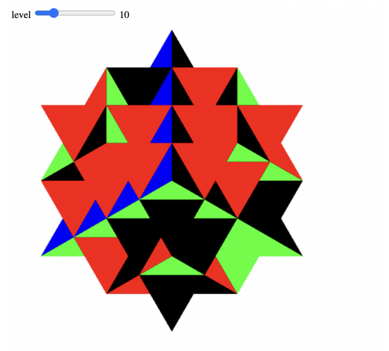


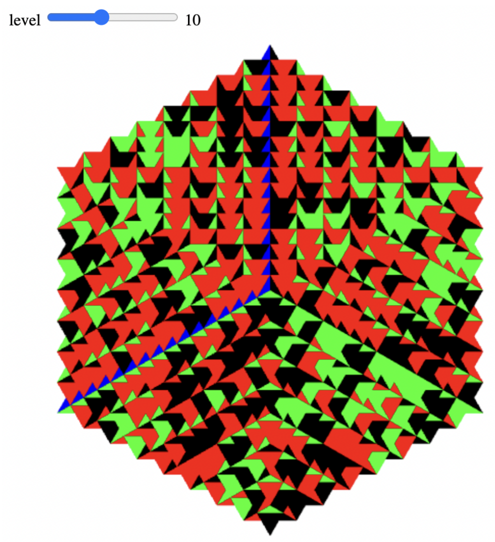


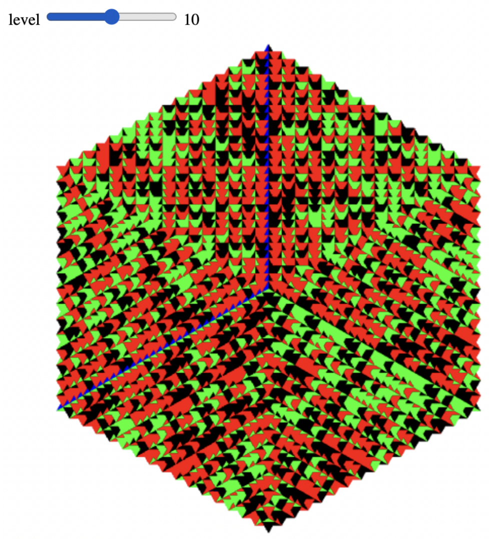
3D Koch Snowflake

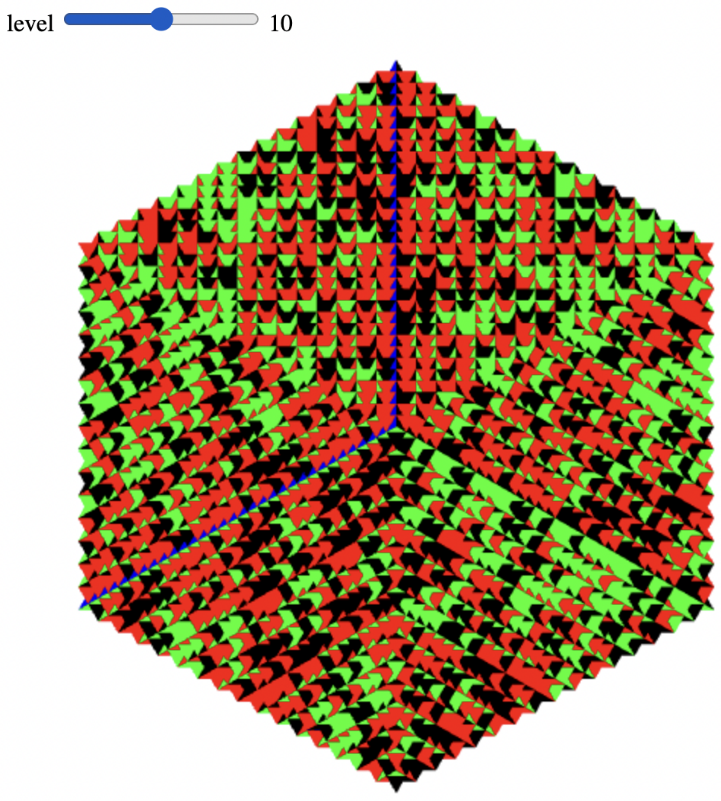







level
10


level
10


level
10