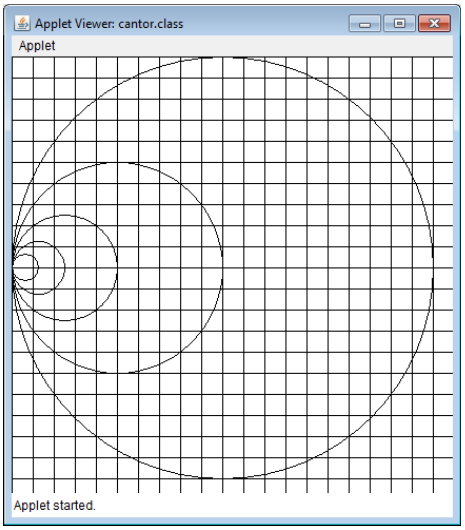
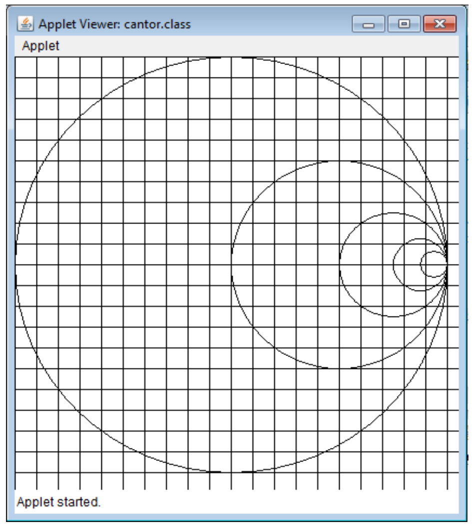
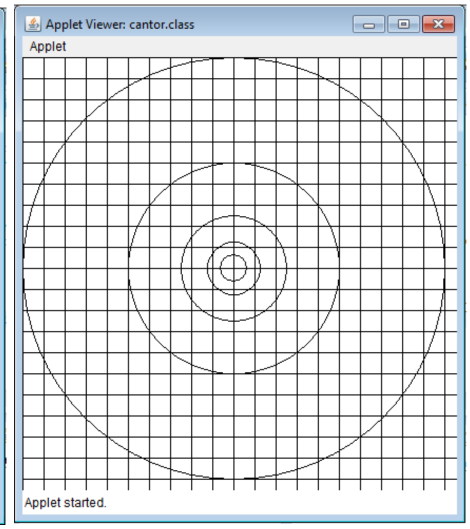
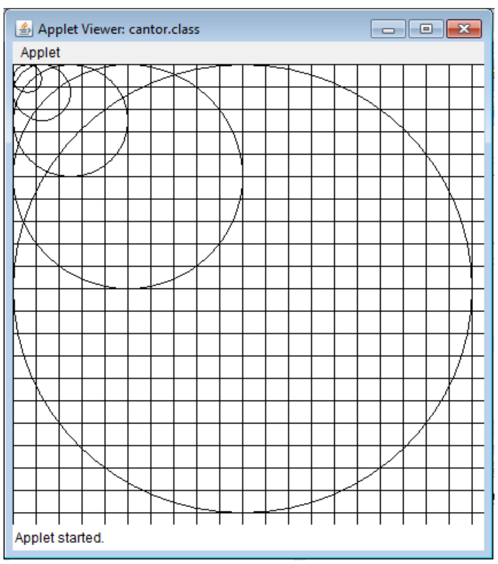
Cantor’s Cheese

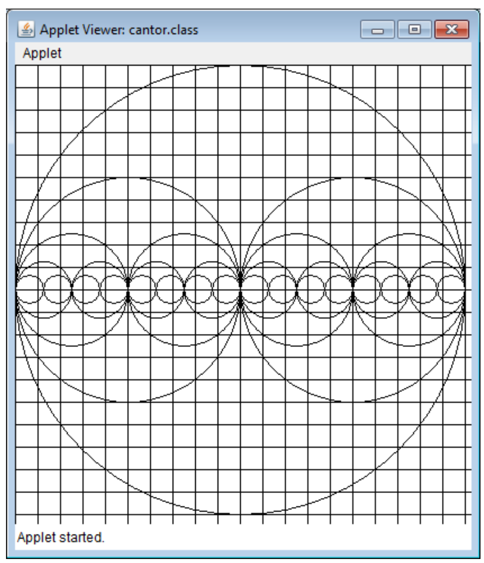
- Take one of the pictures from the Cantor’s cheese worksheet and code it.





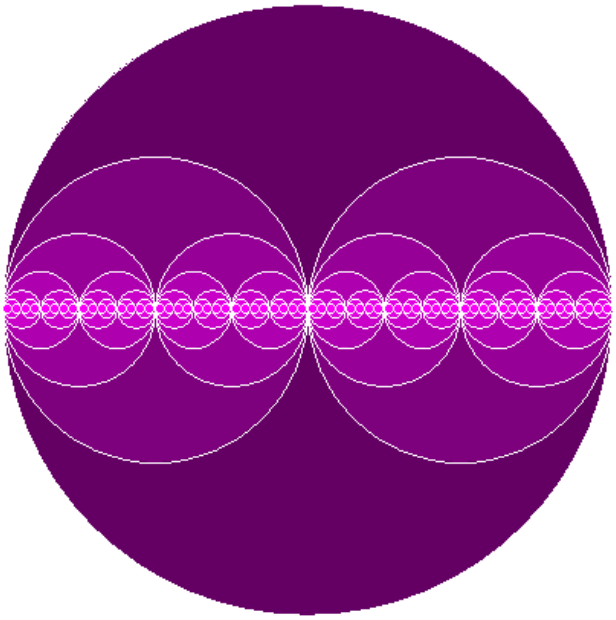


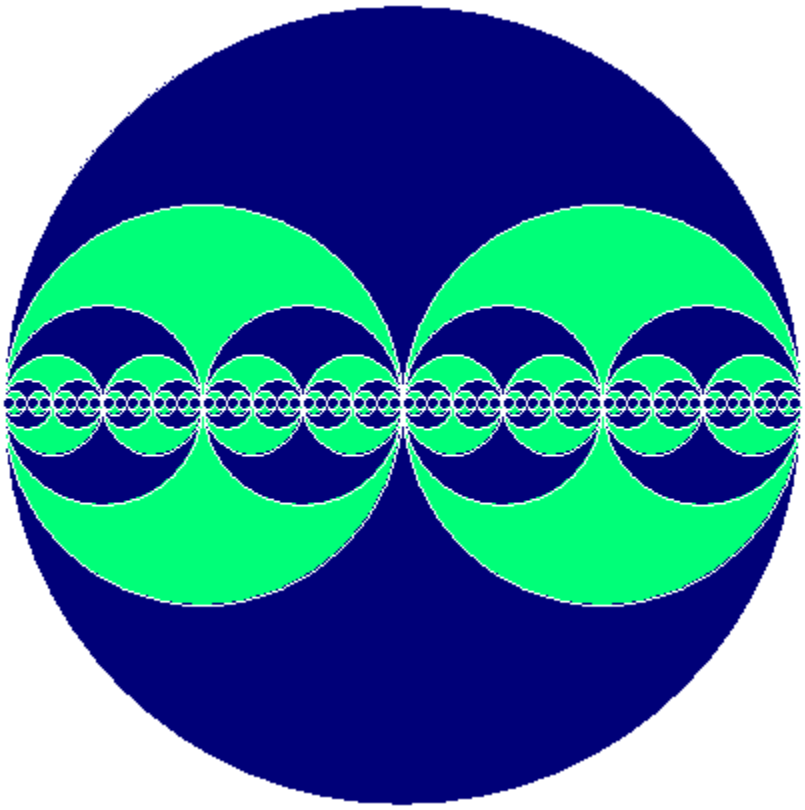




Colour in Cantor's Cheese

Fill the shapes and add colour. Two examples follow.



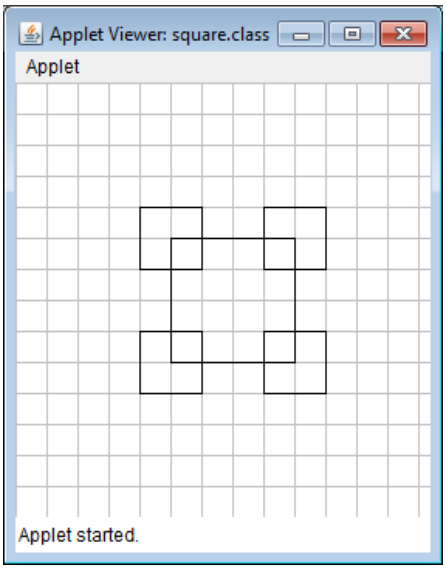


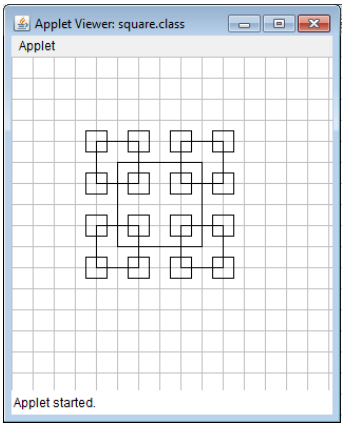
Simple Fractal

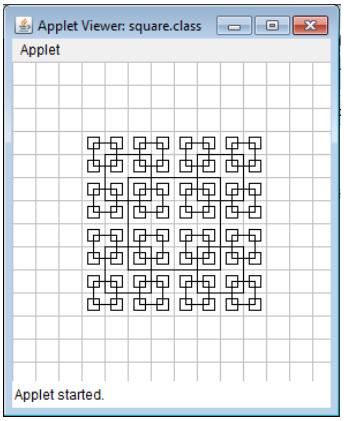
Pick **a** fractal. Draw it. (Obviously, you can do more if you want)

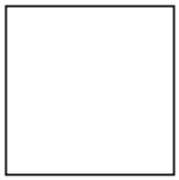
1. Square fractal

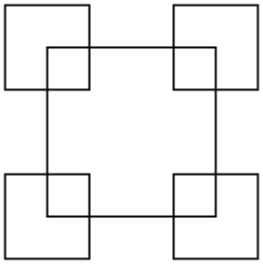
Given a procedure which draws a square centered at x,y of size 2r, then:

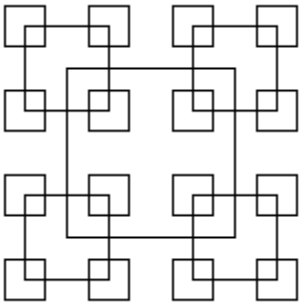




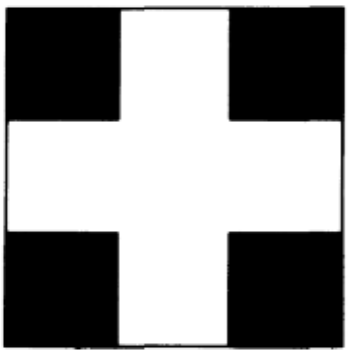




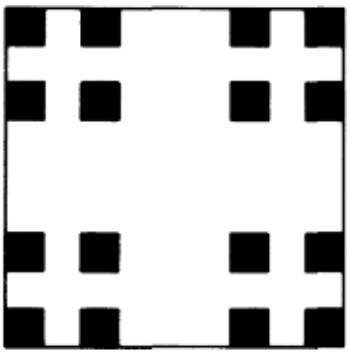




2. Cantor's Dust

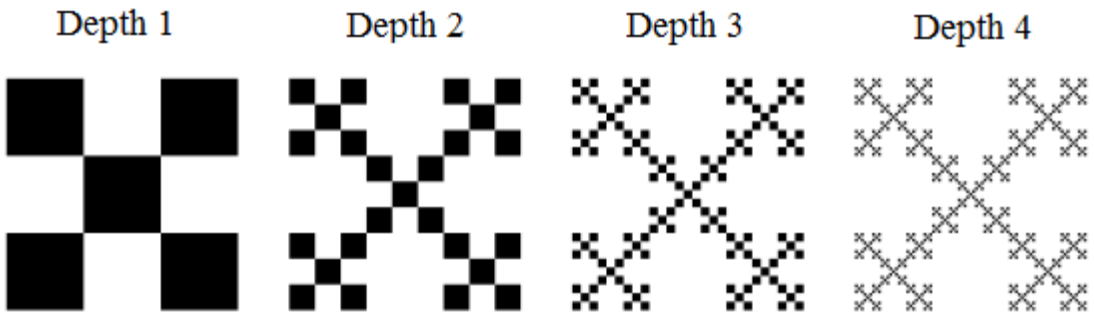


(A)

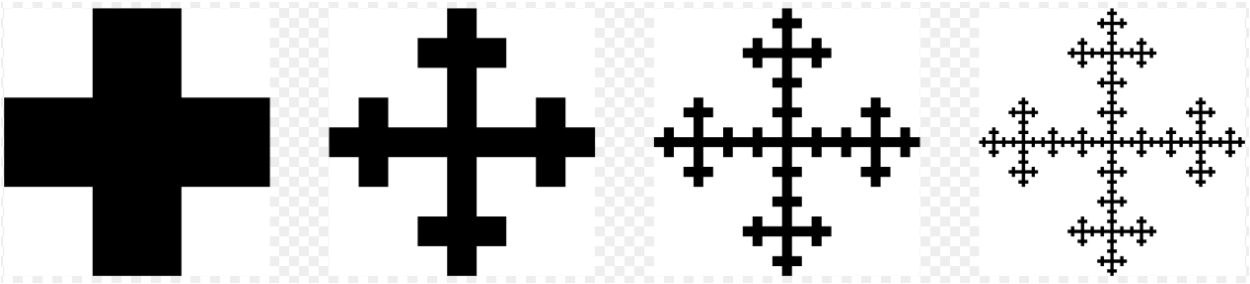


(B)

3. Vicsek



4. Quadric Cross



5. Cantor set

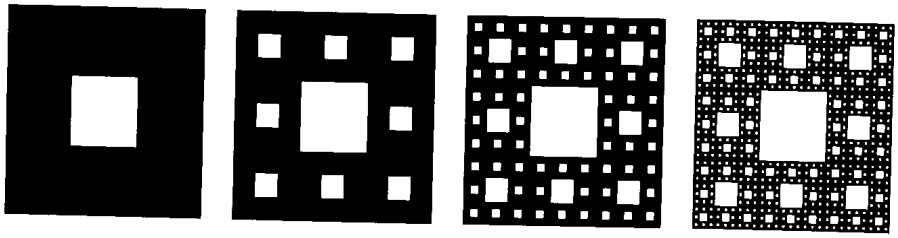
Draw the Cantor set: draw a line in white then erase the middle third of the line (draw in black).

Recursively do that to the first and last thirds of the line, for a given number of levels.

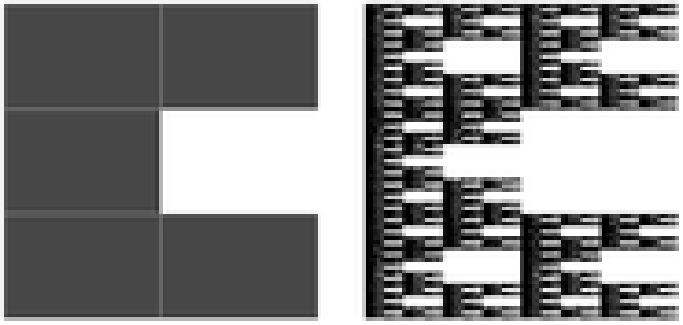
6. Sierpinski's triangle

```
if level > 1
       compute the 3 midpoints
       call Sierpinski again 3 times using the 3 new triangles (omit the central one)
else
       plot the triangle
```

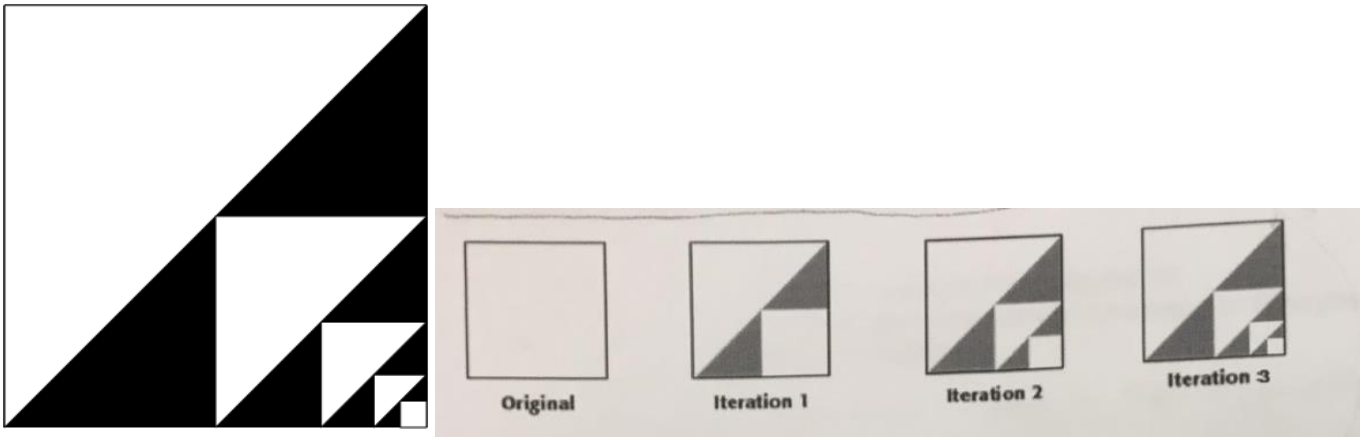
7. Sierpinski's carpet



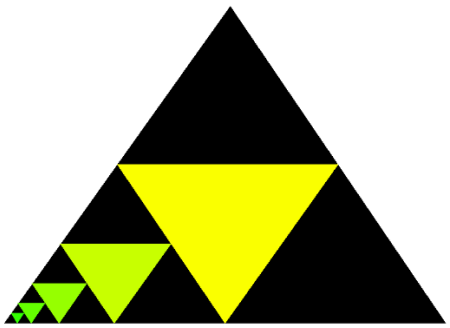
8. Self-Affine



9. Another Square Fractal



Starter Code #1:



```
import java.applet.*; import java.awt.*;

public class exampleTri extends Applet
{
    public void paint (Graphics g)
    {
        int x[] = {487, 859, 97};
        int y[] = {44, 591, 591};
        g.fillPolygon (x, y, 3);
        g.setColor (Color.white);
        tri (487, 44, 859, 591, 97, 591, 5);
    }


    public void tri (int ax, int ay, int bx, int by, int cx, int cy, int i)
    {
        Graphics g = getGraphics ();
        if (i < 1)
            return;
        else
        {
            int x2[] = { (ax + bx) / 2, (cx + bx) / 2, (ax + cx) / 2};
            int y2[] = { (ay + by) / 2, (cy + by) / 2, (ay + cy) / 2};
            g.setColor (new Color (i * 50, 255, 0));
            g.fillPolygon (x2, y2, 3);
            tri (x2 [2], y2 [2], x2 [1], y2 [1], cx, cy, i - 1);
        }
    }
}
```