Chaos Theory With Applications to Art & Programming

Sierra Milosh, Nadine Lambert, Sergei Miles, Lauren Murray

Mathscapes

Interested in combining math, art, and programming

Found a book that seemed to include all of the above

End goal was initially a “mathscape”

Sophisticated graphics, highly skilled programmers, and supercomputers

StdDraw Package

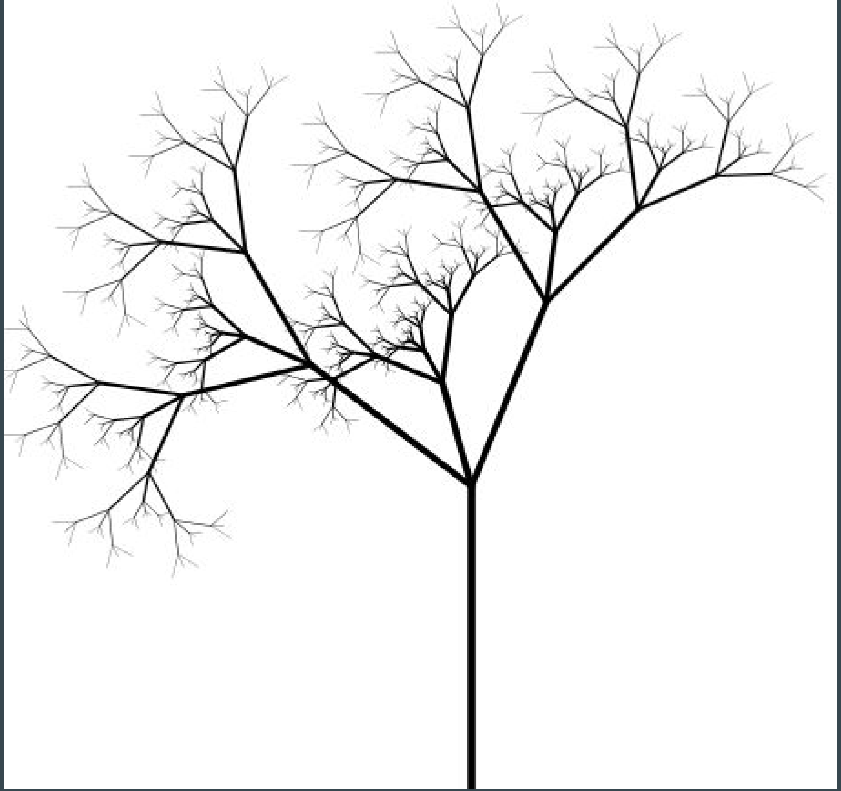
Basic Shapes (circles, ellipses, squares, etc)

Lines (requires two points)

Colors

Points

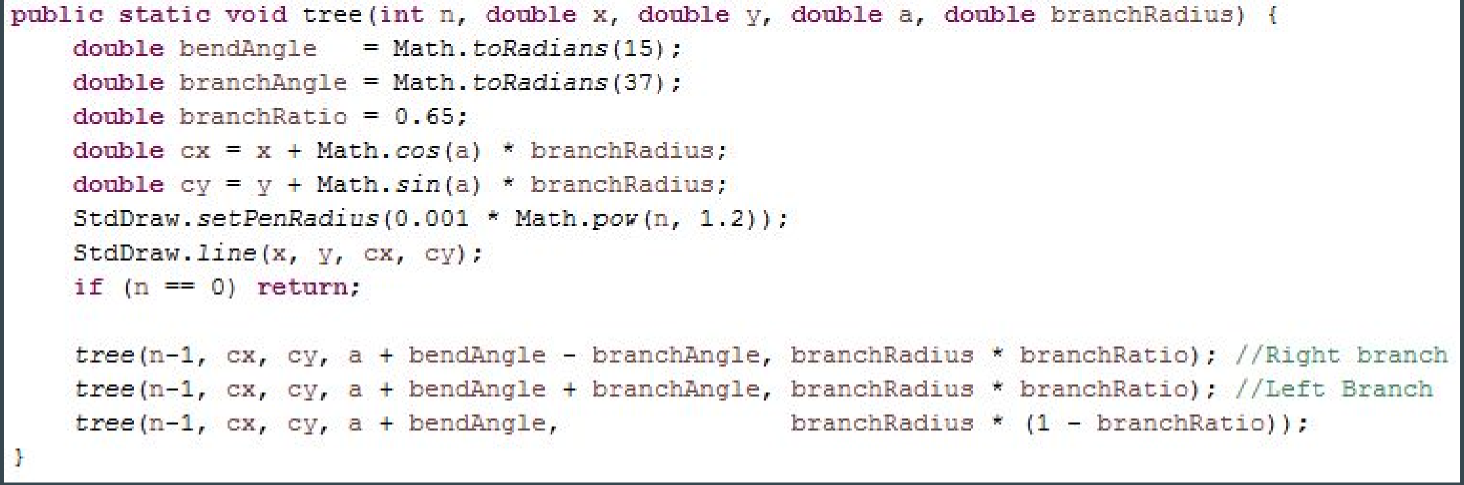
Recursion



"Plant structures illustrate what fractal geometers call *self-similarity* . A branch of a tree looks like a small replica of the tree itself and even the vein structure of a leaf often looks like a two dimensional picture of the tree it came from." [1]

- Recursion is useful when you can reduce a problem into a smaller version of itself.

Recursion Cont.



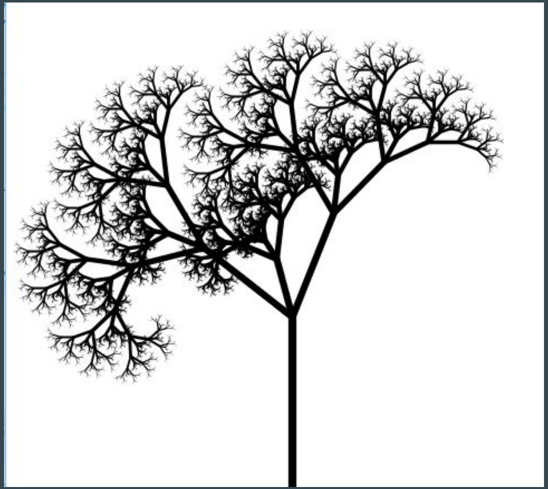
```java
public static void tree(int n, double x, double y, double a, double branchRadius) {
    double bendAngle   = Math.toRadians(15);
    double branchAngle = Math.toRadians(37);
    double branchRatio = 0.65;
    double cx = x + Math.cos(a) * branchRadius;
    double cy = y + Math.sin(a) * branchRadius;
    StdDraw.setPenRadius(0.001 * Math.pow(n, 1.2));
    StdDraw.line(x, y, cx, cy);
    if (n == 0) return;

    tree(n-1, cx, cy, a + bendAngle - branchAngle, branchRadius * branchRatio); //Right branch
    tree(n-1, cx, cy, a + bendAngle + branchAngle, branchRadius * branchRatio); //Left Branch
    tree(n-1, cx, cy, a + bendAngle,               branchRadius * (1 - branchRatio));
}
```

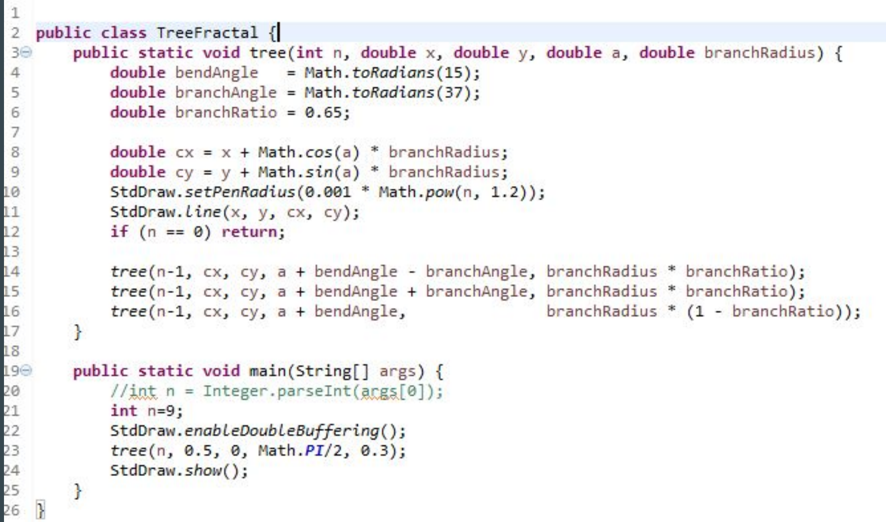
- Various ways to define a base case

Recursive Tree from Princeton

Tree

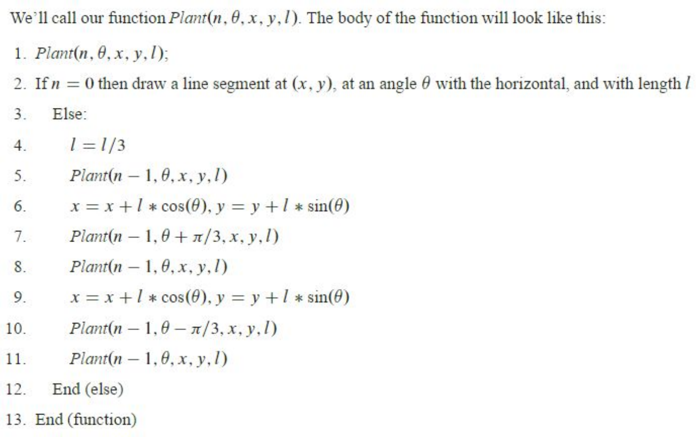
- First attempt at coding so needed an example to get started
- The code is given an “n” and start position





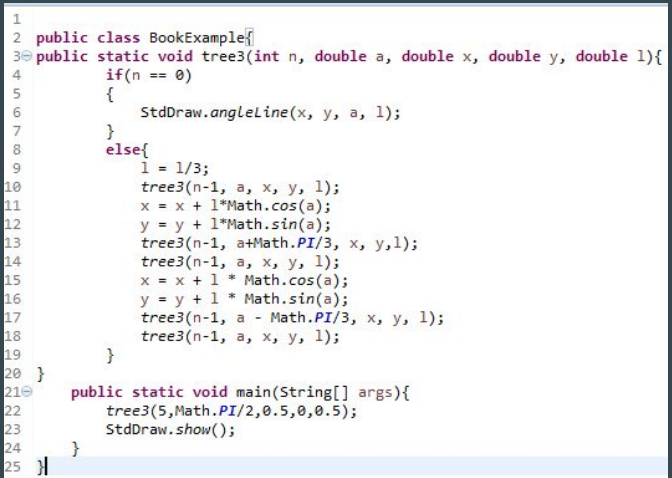
```java

public class TreeFractal {
    public static void tree(int n, double x, double y, double a, double branchRadius) {
        double bendAngle   = Math.toRadians(15);
        double branchAngle = Math.toRadians(37);
        double branchRatio = 0.65;

        double cx = x + Math.cos(a) * branchRadius;
        double cy = y + Math.sin(a) * branchRadius;
        StdDraw.setPenRadius(0.001 * Math.pow(n, 1.2));
        StdDraw.line(x, y, cx, cy);
        if (n == 0) return;

        tree(n-1, cx, cy, a + bendAngle - branchAngle, branchRadius * branchRatio);
        tree(n-1, cx, cy, a + bendAngle + branchAngle, branchRadius * branchRatio);
        tree(n-1, cx, cy, a + bendAngle,               branchRadius * (1 - branchRatio));
    }

    public static void main(String[] args) {
        //int n = Integer.parseInt(args[0]);
        int n=9;
        StdDraw.enableDoubleBuffering();
        tree(n, 0.5, 0, Math.PI/2, 0.3);
        StdDraw.show();
    }
}
```

Applied to Tree in Textbook

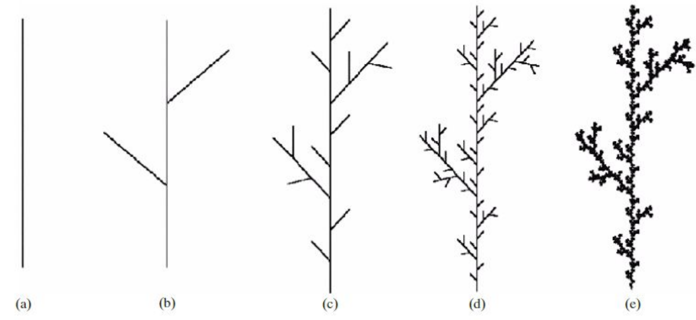


We'll call our function $Plant(n, \theta, x, y, l)$. The body of the function will look like this:

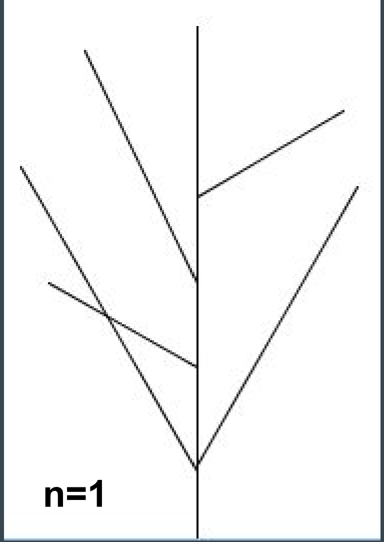
```
1.  $Plant(n, \theta, x, y, l)$;
2.  If $n = 0$ then draw a line segment at $(x, y)$, at an angle $\theta$ with the horizontal, and with length $l$
3.    Else:
4.      $l = l/3$
5.      $Plant(n-1, \theta, x, y, l)$
6.      $x = x + l * \cos(\theta), y = y + l * \sin(\theta)$
7.      $Plant(n-1, \theta + \pi/3, x, y, l)$
8.      $Plant(n-1, \theta, x, y, l)$
9.      $x = x + l * \cos(\theta), y = y + l * \sin(\theta)$
10.     $Plant(n-1, \theta - \pi/3, x, y, l)$
11.     $Plant(n-1, \theta, x, y, l)$
12.   End (else)
13. End (function)
```

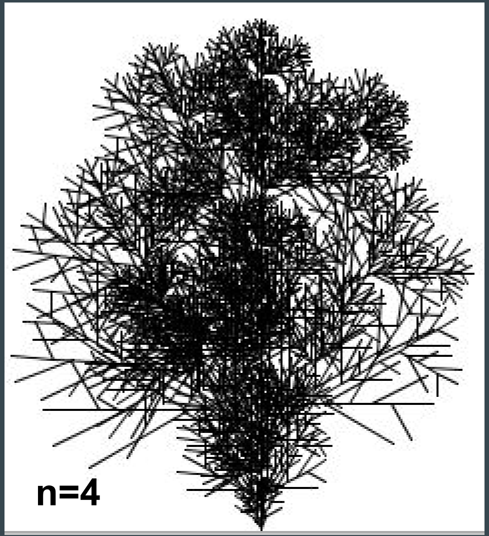


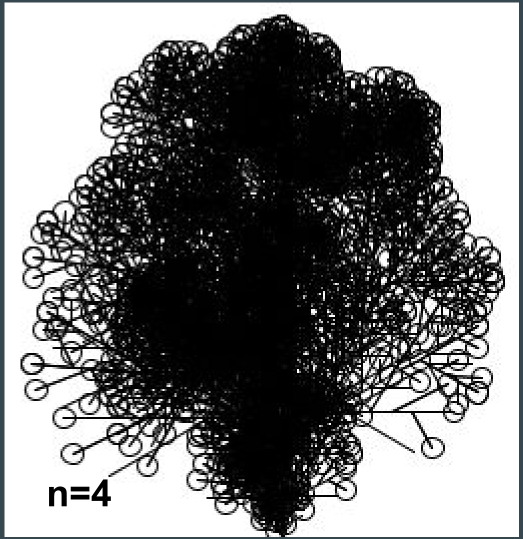
```java

public class BookExample{
public static void tree3(int n, double a, double x, double y, double l){
        if(n == 0)
        {
            StdDraw.angleLine(x, y, a, l);
        }
        else{
            l = l/3;
            tree3(n-1, a, x, y, l);
            x = x + l*Math.cos(a);
            y = y + l*Math.sin(a);
            tree3(n-1, a+Math.PI/3, x, y,l);
            tree3(n-1, a, x, y, l);
            x = x + l * Math.cos(a);
            y = y + l * Math.sin(a);
            tree3(n-1, a - Math.PI/3, x, y, l);
            tree3(n-1, a, x, y, l);
        }
}
    public static void main(String[] args){
        tree3(5,Math.PI/2,0.5,0,0.5);
        StdDraw.show();
    }
}
```



Modified Textbook Tree







Shortcomings

Two team members had never used Java

Turns out programming is difficult and frustrating! (Recursion, anyone?)

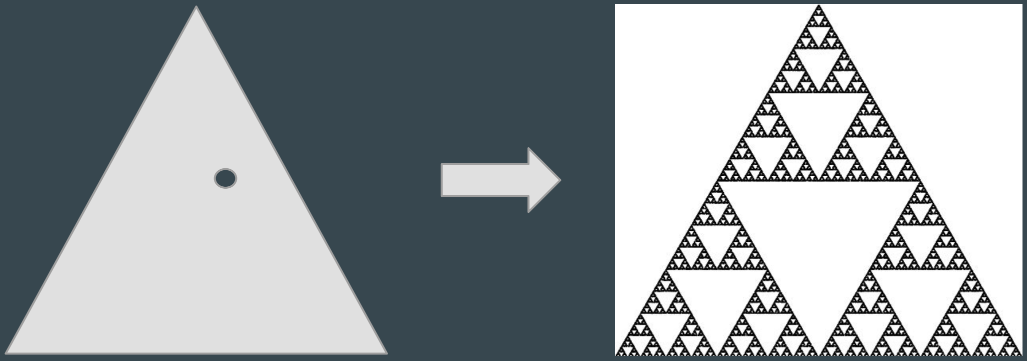
Expected heavier math, more emphasis on algorithms and what they were doing

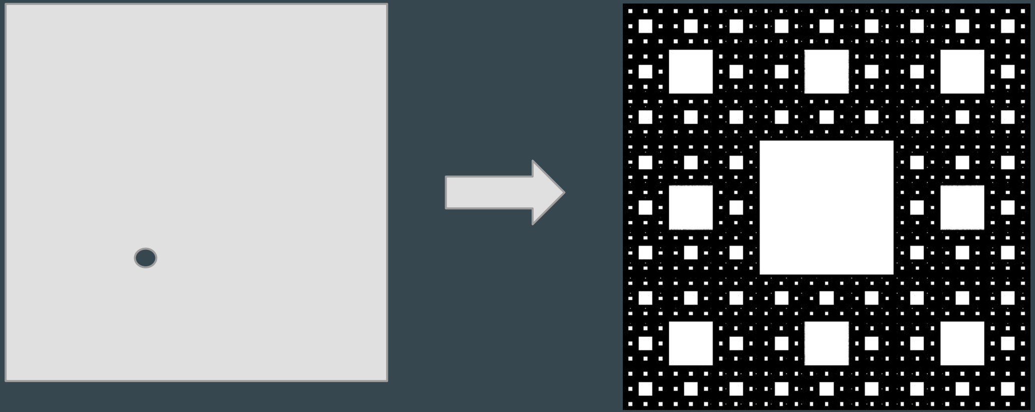
Only working with StdDraw (limited shapes, no angles)

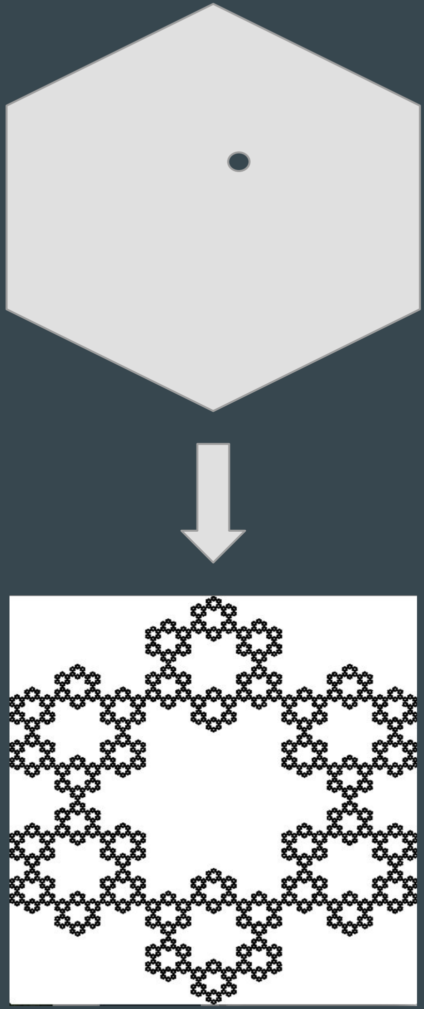
Often no pseudo code to work with

Took a long time to do simple things (flowers)

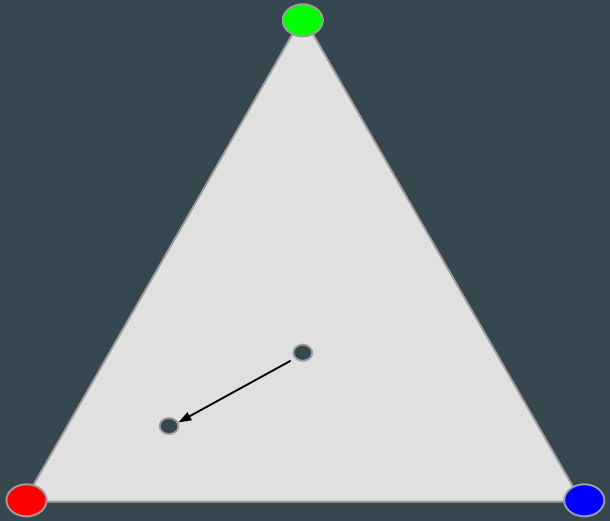
Chaos Games

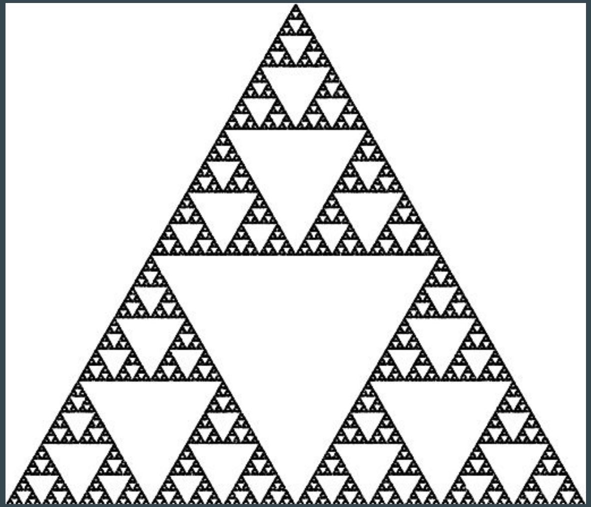




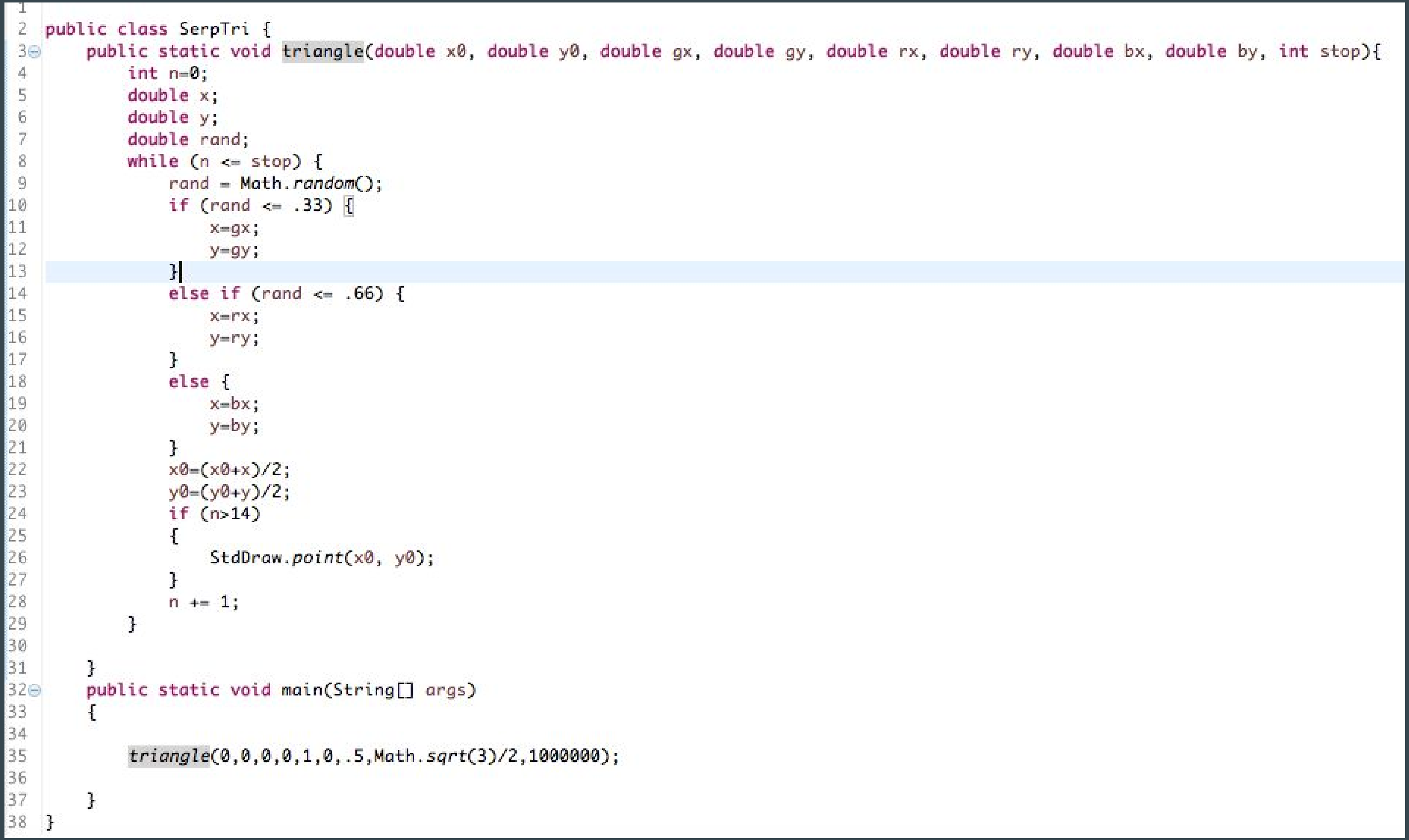


Sierpinski Triangle



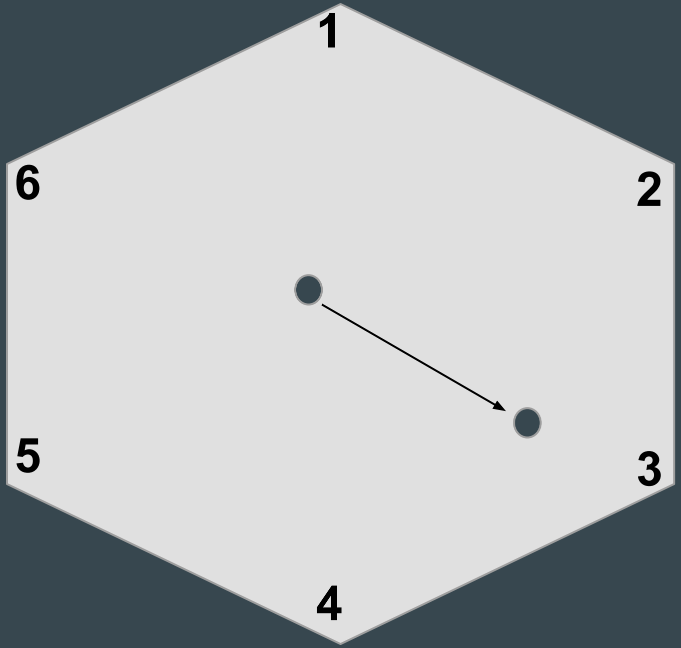


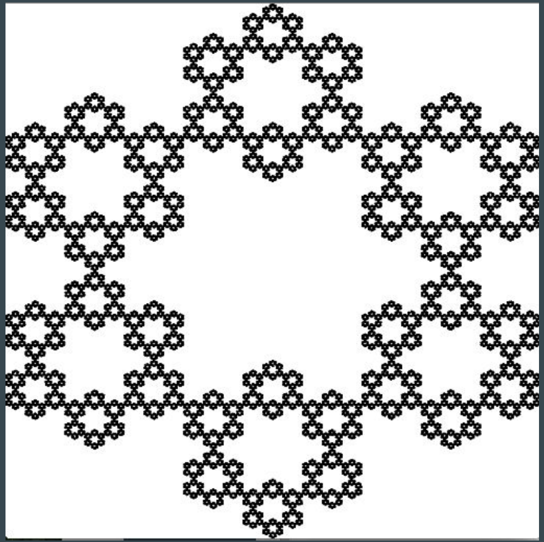
Code for Chaos Games



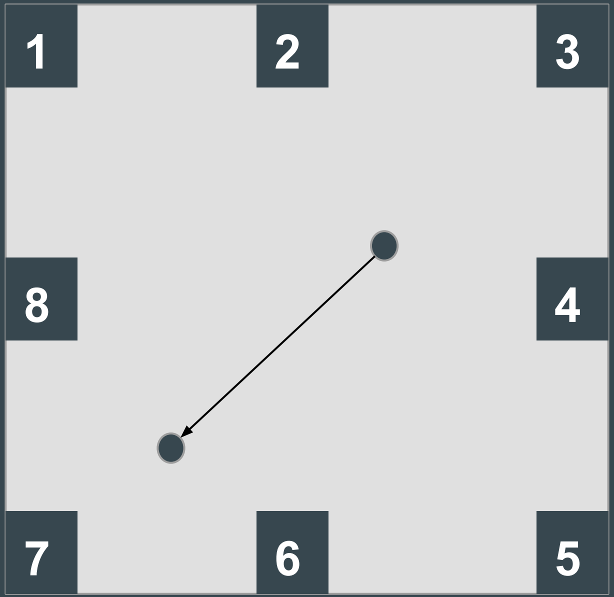
```java

public class SerpTri {
    public static void triangle(double x0, double y0, double gx, double gy, double rx, double ry, double bx, double by, int stop){
        int n=0;
        double x;
        double y;
        double rand;
        while (n <= stop) {
            rand = Math.random();
            if (rand <= .33) {
                x=gx;
                y=gy;
            }
            else if (rand <= .66) {
                x=rx;
                y=ry;
            }
            else {
                x=bx;
                y=by;
            }
            x0=(x0+x)/2;
            y0=(y0+y)/2;
            if (n>14)
            {
                StdDraw.point(x0, y0);
            }
            n += 1;
        }

    }
    public static void main(String[] args)
    {

        triangle(0,0,0,0,1,0,.5,Math.sqrt(3)/2,1000000);

    }
}
```

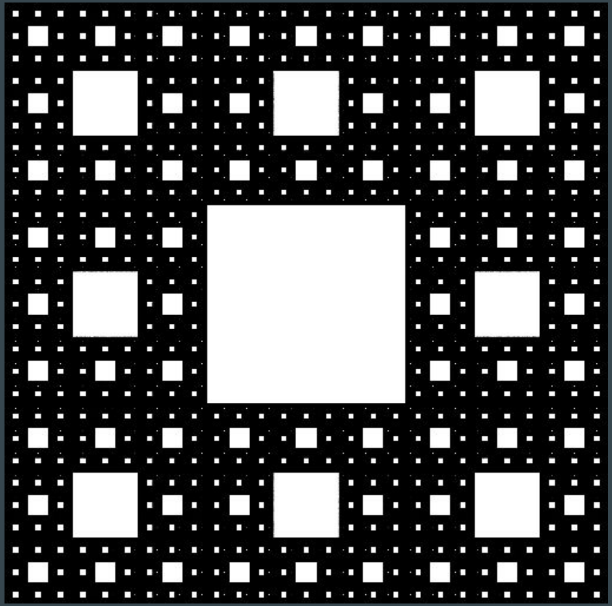
Sierpinski Hexagon





Sierpinski Carpet





Conclusions and Further work

Conclusions:

- Recursion is not intuitive

- Algorithms can be and are used in modeling plant growth

Further Work:

- Investigate other algorithms

- Switch to a different language

References

Scott, J., Gulick, D., & Mathematical Association of, A. (2010). The Beauty of Fractals : Six Different Views. Washington, D.C.: Mathematical Association of America.

Sedgewick, Robert, and Kevin Wayne. "Tree.java." *Princeton University*. N.p., n.d. Web. 14 Nov. 2016.

Thank you

Any questions?