2021年度(第12回)電気学会東京支部
群馬支所・栃木支所 合同研究発表会
ETG-22-40, ETT-22-40
2022/03/02

フラクタルを用いた集積回路での小チップ面積での抵抗実現法

Resistor Implementation Algorithm with Small Chip Area based on Fractals

ムハマド ハキブ ビン ハミド
Muhammad Hakib bin Hamid

桑名 杏奈　小林 春夫

(群馬大学)

Outline

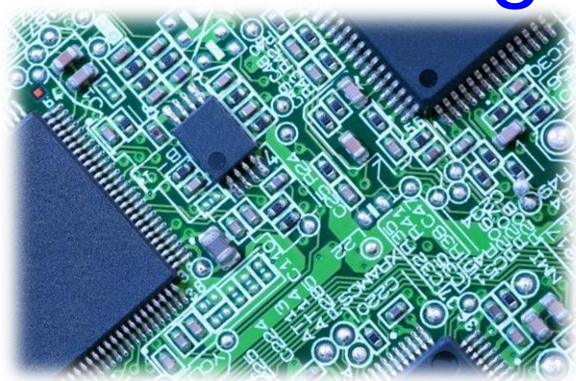
- Introduction
- Well-known fractal
    - -Sirpienski gasket
    - -Koch Curve
    - -Barnsley Fern
- Fractal Properties
- Program & Algorithm
- New Fractal
- Discussion
- Conclusion

Outline

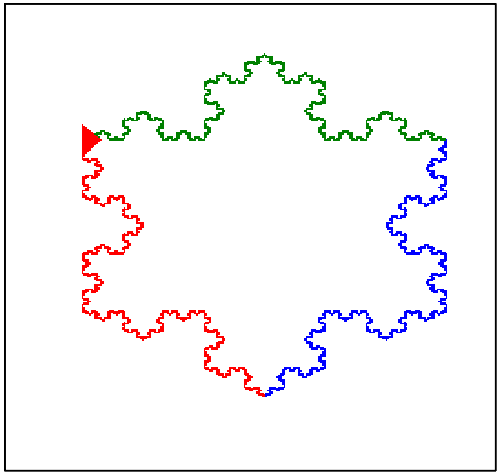
- Introduction
- Well-known fractal
  - -Sirpienski gasket
  - -Koch Curve
  - -Barnsley Fern
- Fractal Properties
- Program & Algorithm
- New Fractal
- Discussion
- Conclusion

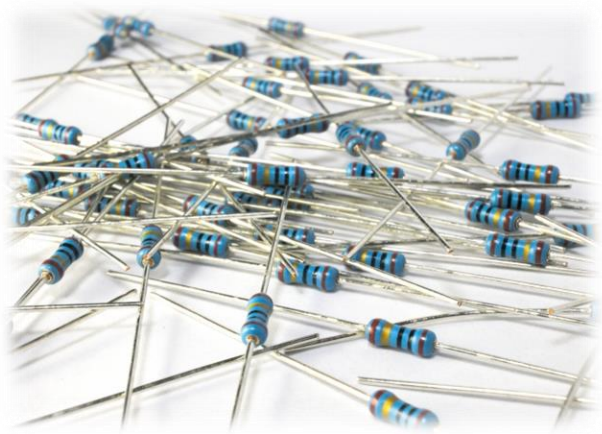
In Kobayashi Lab.

Applying Mathematics to Circuit Design



I applied Fractal to Resistor

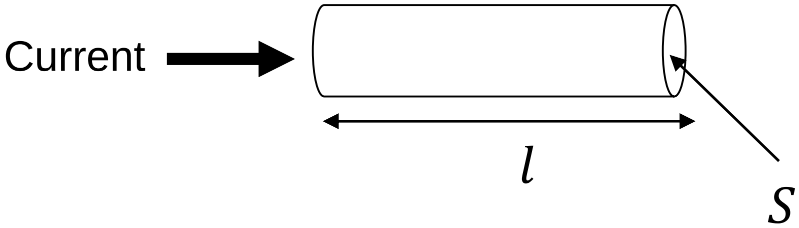




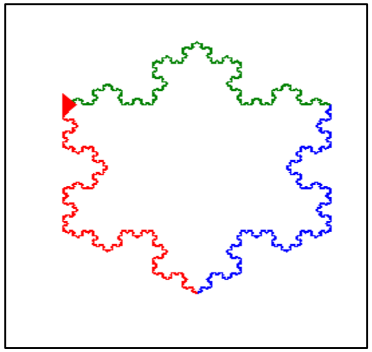
Introduction

Resistor : $r = \rho \frac{l}{S}$ [Ω]

ρ: Electrical resistivity [Ω·m]
l: Length of object [m]
S: Cross-sectional area[m^2]



Fractal:
Limited area,
Long line.



Fractal Resistor:
Limited area, large resistor

Outline

- Introduction
- Well-known fractal
  - -Sirpienski gasket
  - -Koch Curve
  - -Barnsley Fern
- Fractal Properties
- Program & Algorithm
- New Fractal
- Discussion
- Conclusion

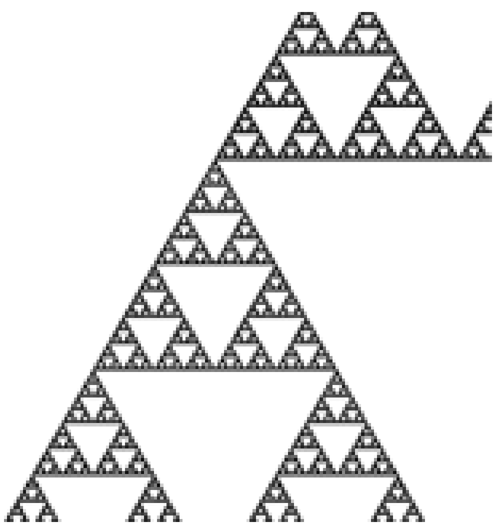
Fractal

What is fractal?

A fractal is a never ending pattern that repeats itself at different scales.

Fractals are extremely complex, sometimes infinitely complex - meaning you can zoom in and find the same shapes forever.

Fractals can also be used
by repeatedly calculating
a simple equation over and over.



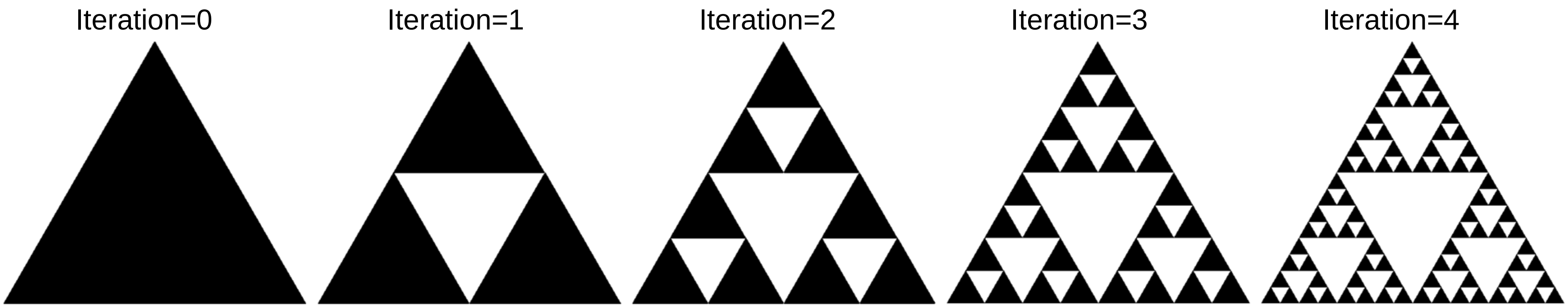
画像出典:Wikipedia

Sierpinski gasket

Also known as Sierpinski Triangle

Purely geometric fractals can be made by repeating a simple process.

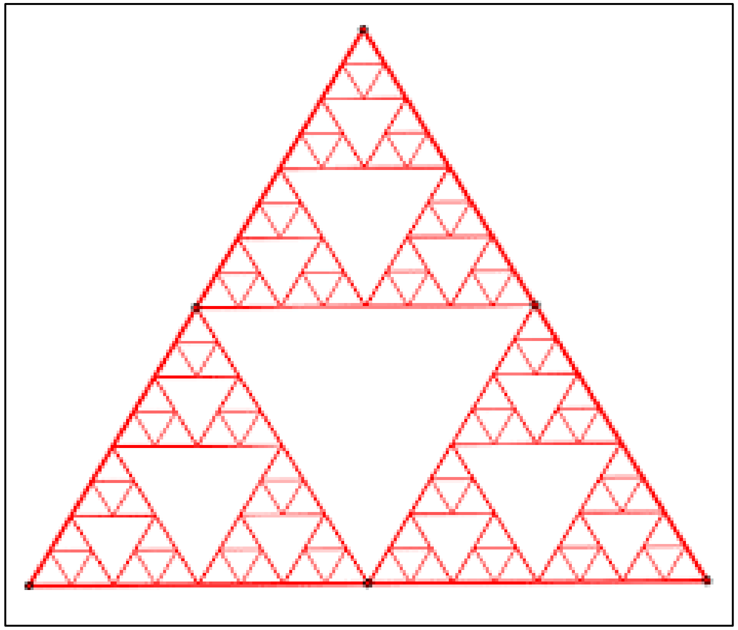
The Sierpinski Triangle is made by repeatedly removing the middle triangle from the prior generation.



画像出典：Wikipedia

Sierpinski gasket

The midpoints of the line segments of the largest triangle is connected resulting smaller triangles. This pattern is then repeated for the smaller triangles, and essentially has infinitely many possible iterations.



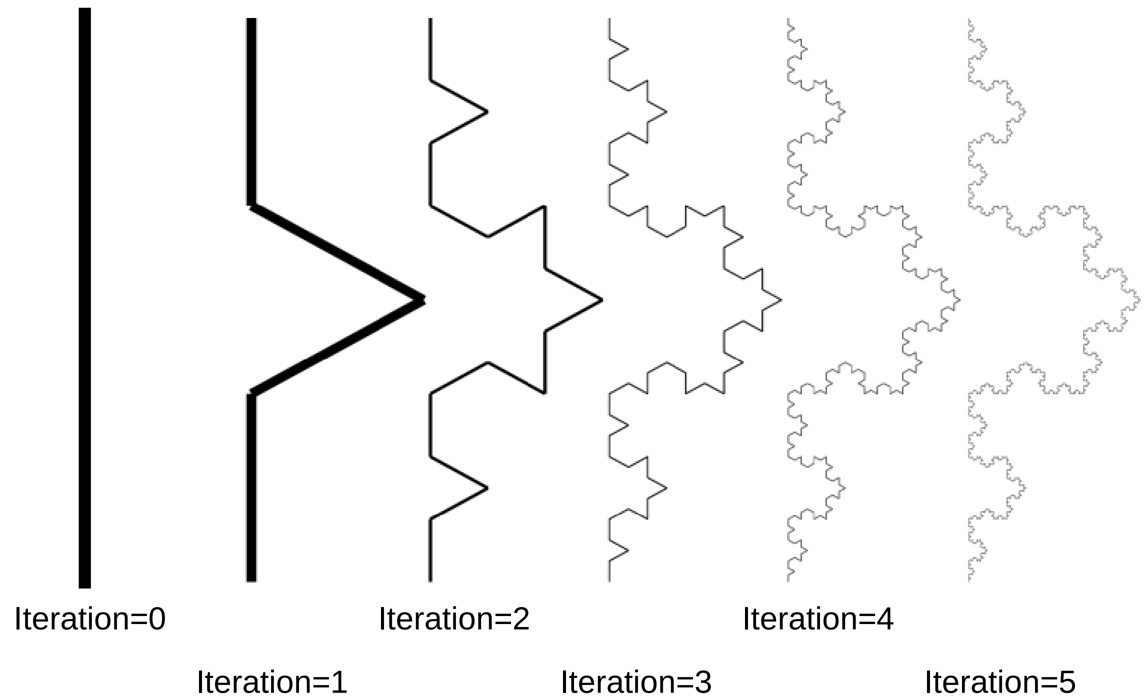
Outline

- Introduction
- Well-known fractal
  - -Sirpienski gasket
  - -Koch Curve
  - -Barnsley Fern
- Fractal Properties
- Program & Algorithm
- New Fractal
- Discussion
- Conclusion

Koch curve

The Koch Curve is made by repeatedly replacing each segment of a generator shape with a smaller copy of the generator.

At each step, the total length of the curve gets longer approaching infinity. The length of the curve increases the more closely you measure it.

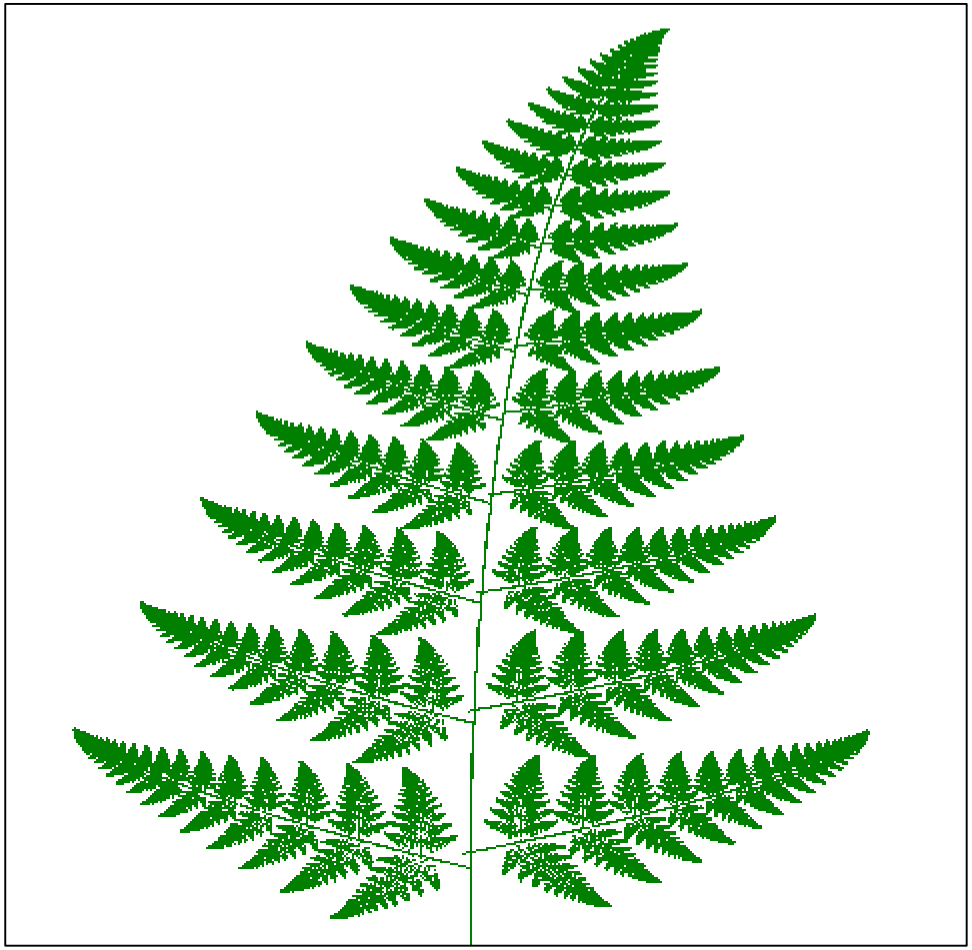


Outline

- Introduction
- Well-known fractal
  - -Sirpienski gasket
  - -Koch Curve
  - -Barnsley Fern
- Fractal Properties
- Program & Algorithm
- New Fractal
- Discussion
- Conclusion

Barnsley fern

The Barnsley Fern is a fractal named after the British mathematician Michael Barnsley.



Barnsley fern

The Barnsley fern shows how graphically beautiful structures can be built from repetitive uses of mathematical formulas.

The complexity of creating the Barnsley fern model, together with the fact that the number of iterations required could be tens of thousands, makes it extremely hard to plot by hand.

While it is not impossible, it is much easier, and often preferred, to use a computer instead.

Fractals in the real world

Fractals in the real world' often break down when examined closely enough

Fractal shapes exist throughout the human body, in lungs, blood vessels, and neurons

In cosmology, fractal distributions of galaxies have been detected over relatively small scales.
Other uses include antennae design and image analysis using multifractals.

画像出典:Wikipedia

Outline

Koch Curve properties

LENGTH OF A SIDE (length)

If we begin with an equilateral triangle with side length 1, then the length of a side is

$$S_n = \frac{S_{n-1}}{3} = \frac{1}{3^n}$$

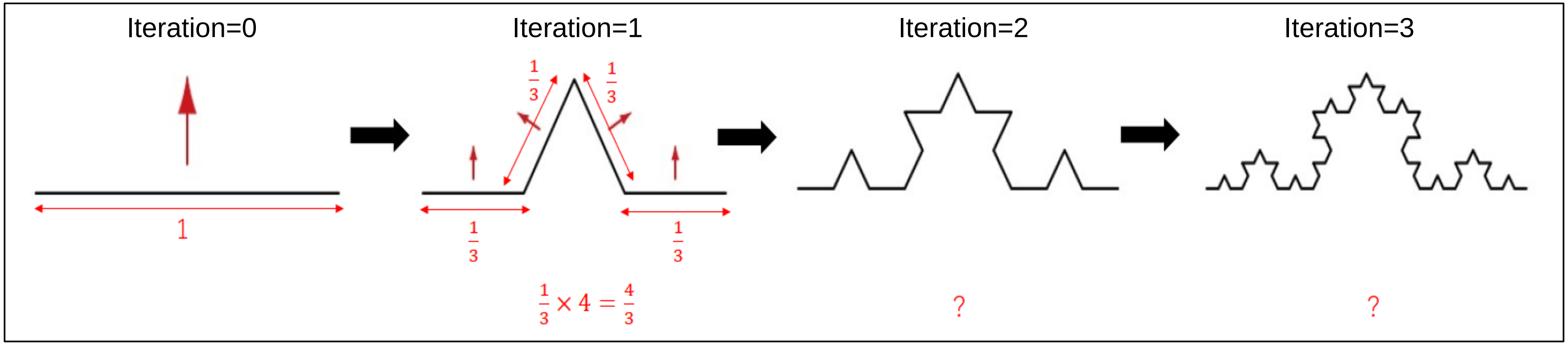
For iterations 0 to 3, length = 1, 1/3, 1/9 and 1/27.

NUMBER OF SIDES (n)

For each iteration, one side of the figure from the previous stage becomes four sides in the following stage. Since we begin with three sides, the formula for the number of sides in the Koch Curve is

$$N_n = N_{n-1} \cdot 4 = 3 \cdot 4^n$$

For iterations 0, 1, 2 and 3, the number of sides are 3, 12, 48 and 192, respectively.

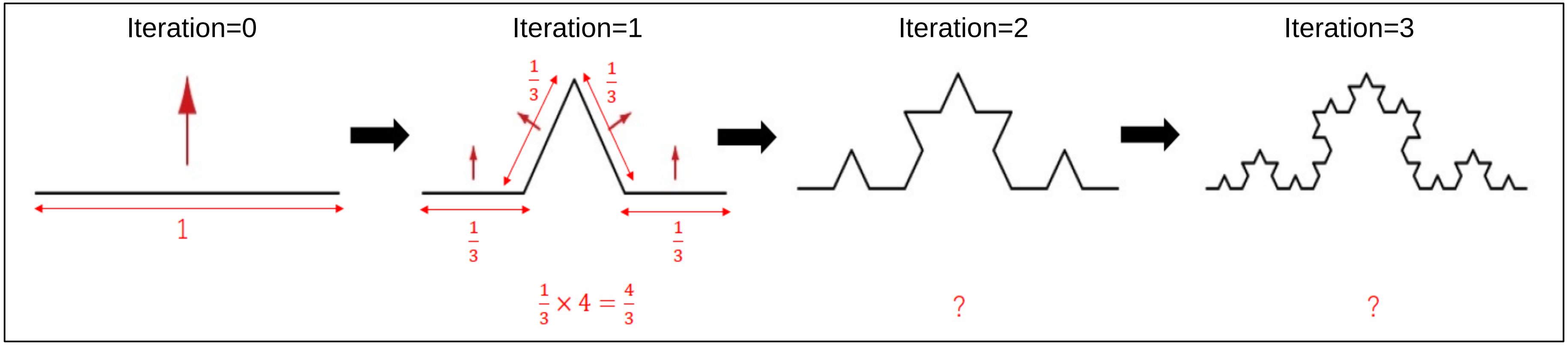


Koch Curve properties

PERIMETER (p)
Since all the sides in every iteration of the Koch Curve is the same the perimeter is simply the number of sides multiplied by the length of a side

$$P_n = N_n \cdot S_n = 3 \cdot 1 \cdot \left(\frac{4}{3}\right)^n$$

for the first 4 iterations (0 to 3) the perimeter is $1, \frac{4}{3}, \frac{16}{9}, \frac{64}{27}$.



Koch Curve properties

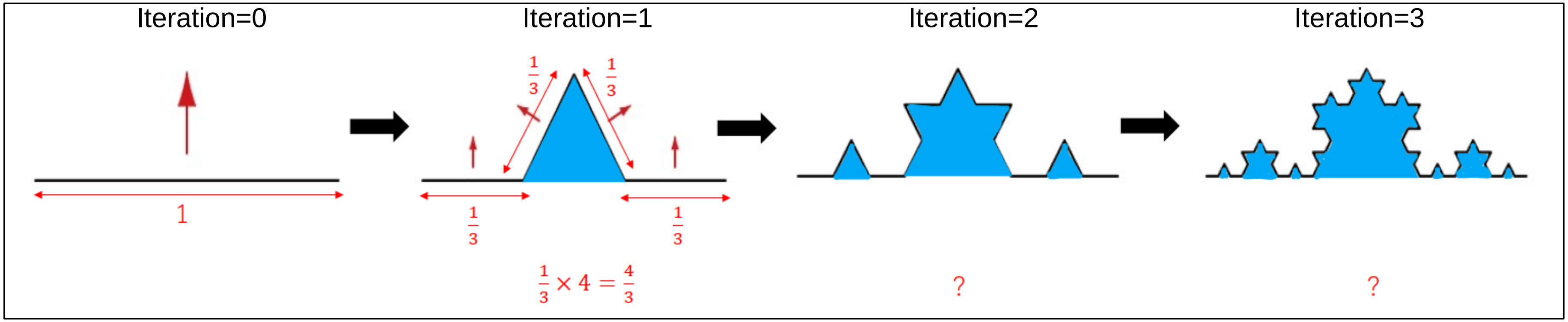
Area of the Koch Curve

In each iteration a new triangle is added on each side of the previous iteration, so the number of new triangles added in iteration *n* is:

$$T_n = T_{n-1} = 3 \cdot 4^{n-1} = \frac{3}{4} \cdot 4^n$$

The area of each new triangle added in an iteration is 1/9 of the area of each triangle added in the previous iteration, so the area of each triangle added in iteration *n* is:

$$a_n = \frac{a_{n-1}}{9} = \frac{a_0}{9^n}$$



Koch Curve properties

where a_0 is the area of the original triangle. The total new area added in iteration *n* is therefore:

$$b_n = T_n \cdot a_n = \frac{3}{4} \cdot \left(\frac{4}{9}\right)^n \cdot a_0$$

The total area of the curve after *n* iterations is:

$$A_n = a_0 + \sum_{k=1}^{n} b_k = a_0 \left(1 + \frac{3}{4} \sum_{k=1}^{n} \left(\frac{4}{9}\right)^k \right) = a_0 \left(1 + \frac{1}{3} \sum_{k=0}^{n-1} \left(\frac{4}{9}\right)^k \right)$$

$$A_n = a_0 \left(1 + \frac{3}{5}\left(1 - \left(\frac{4}{9}\right)^n\right)\right) = \frac{a_0}{5}\left(8 - 3\left(\frac{4}{9}\right)^n\right)$$

If the length of one side of the first triangle s is 1, $A_0 = a_0 = \frac{\sqrt{3}}{4}$

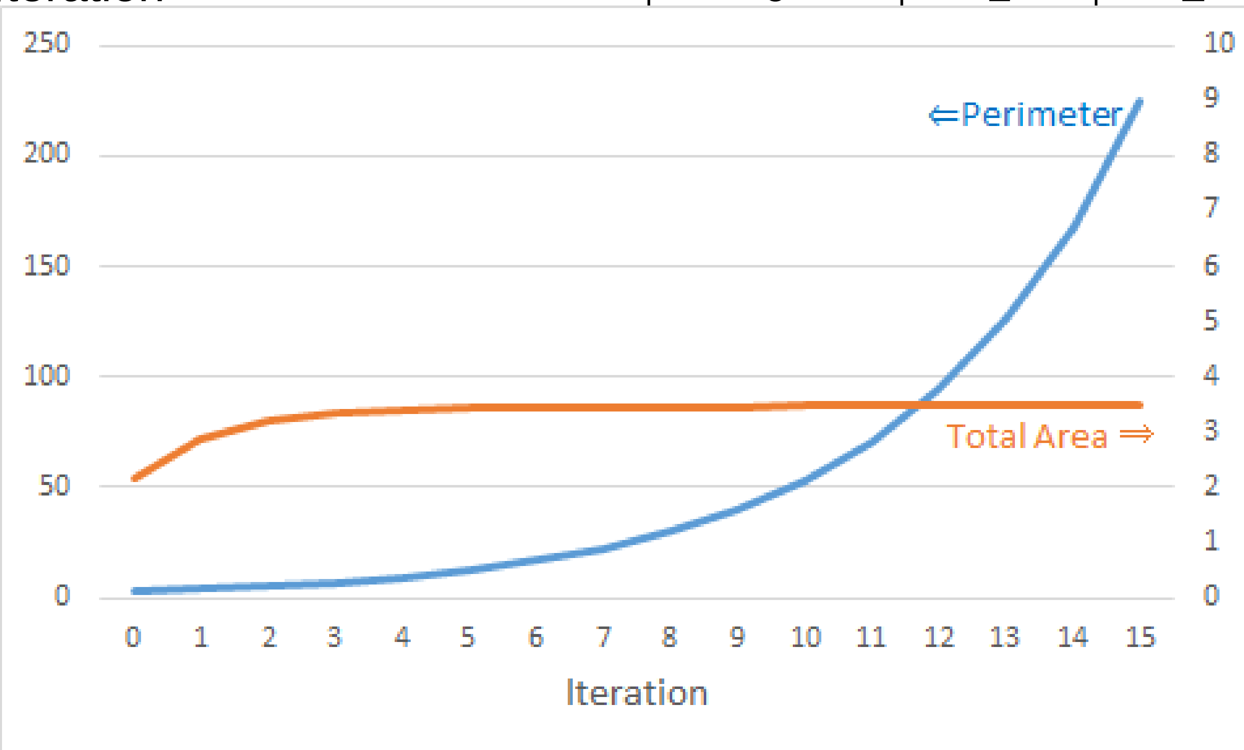
$$A_0 = \frac{\sqrt{3}}{4}$$

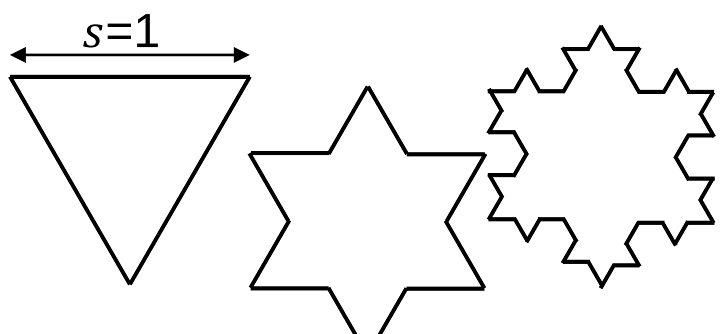
Then, $A_n = \frac{1}{5}\frac{\sqrt{3}}{4}\left(8 - 3\left(\frac{4}{9}\right)^n\right)$

For iterations 0, 1, 2 and 3, the number of sides are $\frac{\sqrt{3}}{4}$, $\frac{\sqrt{3}}{3}$, $\frac{10\sqrt{3}}{27}$ and $\frac{94\sqrt{3}}{243}$, respectively.

Koch Snowflake properties

Iteration	0	1	2	n
				$\frac{1}{3^n}$
				$3 \cdot 4^n$
				$3 \cdot 1 \cdot \left(\frac{4}{3}\right)^n$
				$\frac{3}{4} \cdot 4^n$
				$\frac{a_0}{9^n}$
	4	3	27	$\frac{\sqrt{3}}{4}\left(8 - 3\left(\frac{4}{9}\right)^n\right)$



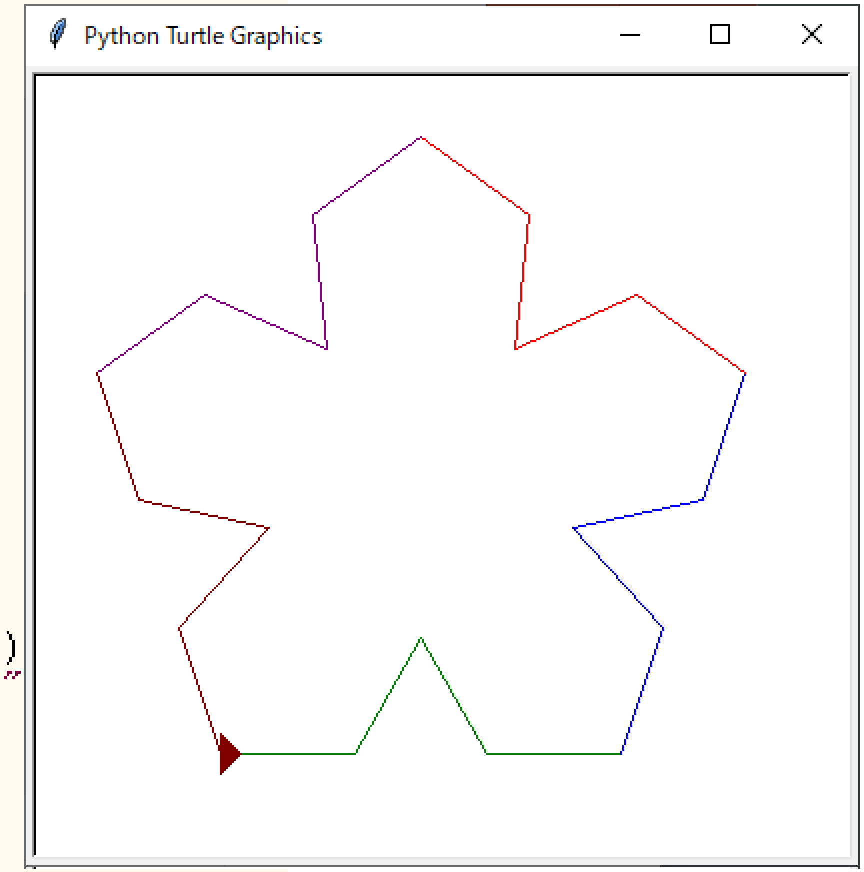


Outline

- Introduction
- Well-known fractal
  - -Sirpienski gasket
  - -Koch Curve
  - -Barnsley Fern
- Fractal Properties
- Program & Algorithm
- New Fractal
- Discussion
- Conclusion

Python Program

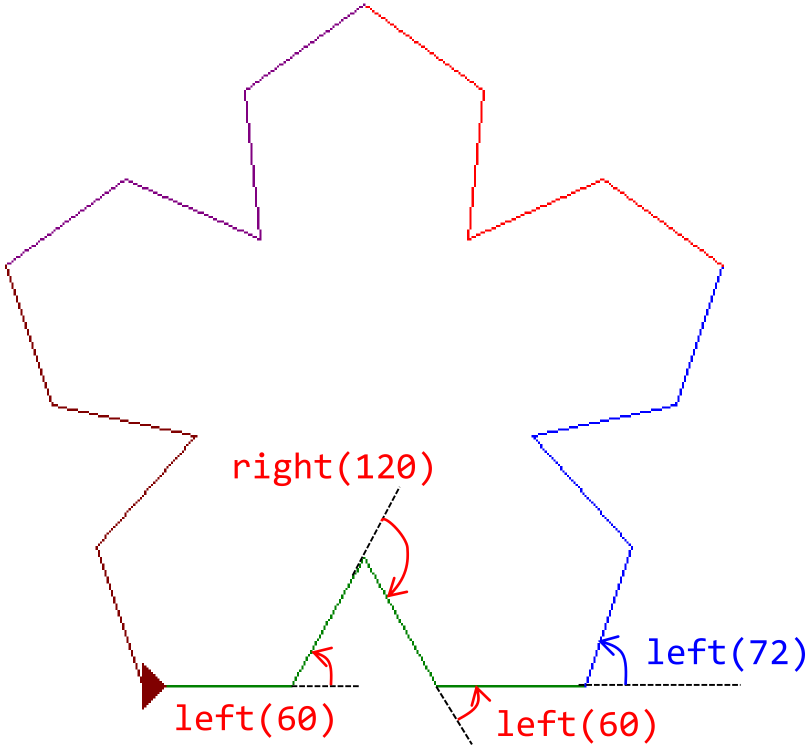
```python
from turtle import*

shape("arrow")
speed(0)

def snowflake_side(length, levels):
    if levels == 0:
        forward(length)
        return

    length /=3.0
    snowflake_side(length, levels -1)
    left(60)
    snowflake_side(length, levels -1)
    right(120)
    snowflake_side(length, levels -1)
    left(60)
    snowflake_side(length, levels -1)

def create_snowflake(sides, length, iteration)
    colors = ["green", "blue", "red", "purple"
    for i in range(sides):
        color(colors[i])
        snowflake_side(length, iteration)
        left(360 / sides)


create_snowflake(5, 200, 1)
mainloop()
```



https://docs.python.org/ja/3/library/turtle.html

Algorithm (1)

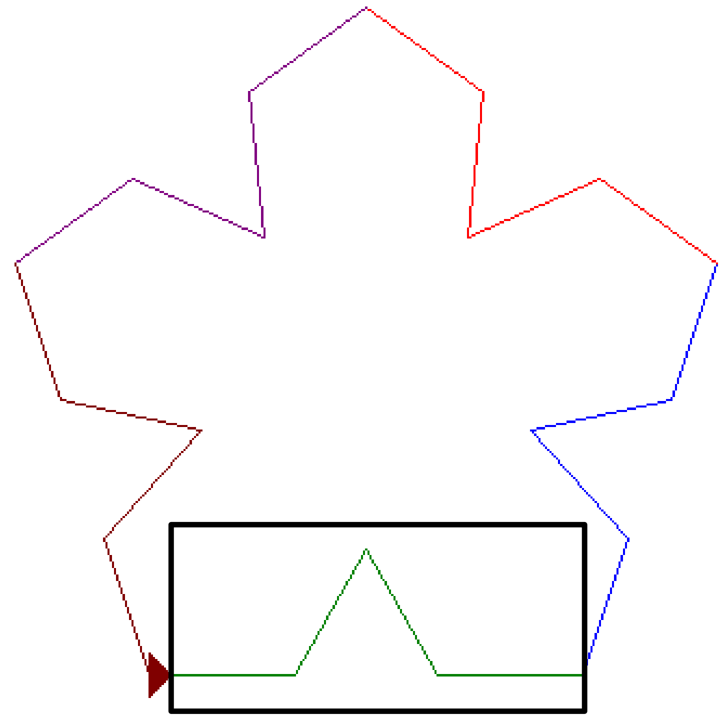
```
from turtle import*
shape("arrow")
speed(0)
def snowflake_side(length, levels):
        if levels == 0:
                forward(length)
                return

        length /=3.0
        snowflake_side(length, levels -1)
        left(60)
        snowflake_side(length, levels -1)
        right(120)
        snowflake_side(length, levels -1)
        left(60)
        snowflake_side(length, levels -1)

def create_snowflake(sides, length, iteration):
        colors = ["green", "blue", "red", "purple", "maroon"]
        for i in range(sides):
                color(colors[i])
                snowflake_side(length, iteration)
                left(360 / sides)

create_snowflake(5, 200, 1)
mainloop()
```



Algorithm (2-1)

```
from turtle import*
shape("arrow")
speed(0)
def snowflake_side(length, levels):
        if levels == 0:
                forward(length)
                return

        length /=3.0
        snowflake_side(length, levels -1)
        left(60)
        snowflake_side(length, levels -1)
        right(120)
        snowflake_side(length, levels -1)
        left(60)
        snowflake_side(length, levels -1)

def create_snowflake(sides, length, iteration):
        colors = ["green", "blue", "red", "purple", "maroon"]
        for i in range(sides):
                color(colors[i])
                snowflake_side(length, iteration)
                left(360 / sides)

create_snowflake(5, 200, 1)
mainloop()
```

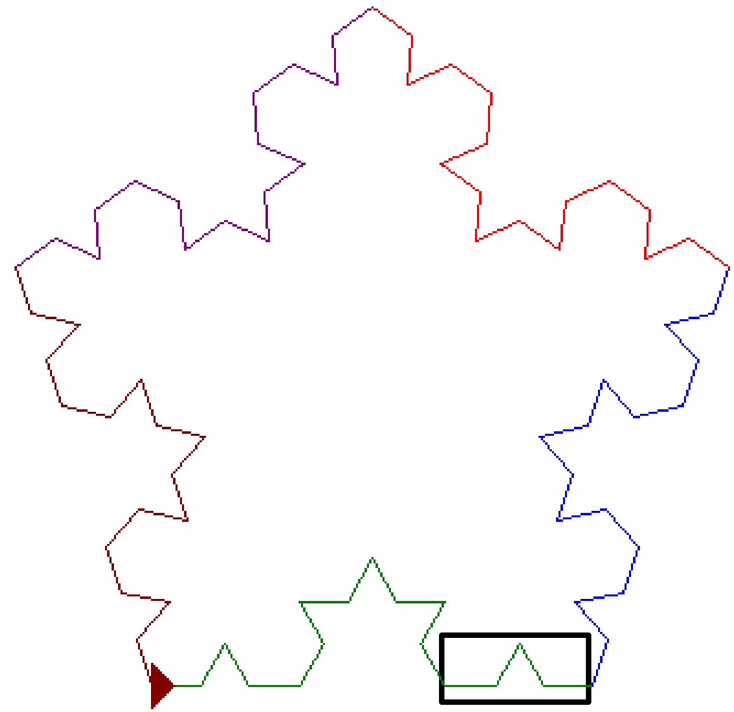
Recursion
再帰



Algorithm (2-2)

```
from turtle import*
shape("arrow")
speed(0)
def snowflake_side(length, levels):
        if levels == 0:
                forward(length)
                return

        length /=3.0
        snowflake_side(length, levels -1)
        left(60)
        snowflake_side(length, levels -1)
        right(120)
        snowflake_side(length, levels -1)
        left(60)
        snowflake_side(length, levels -1)

def create_snowflake(sides, length, iteration):
        colors = ["green", "blue", "red", "purple", "maroon"]
        for i in range(sides):
                color(colors[i])
                snowflake_side(length, iteration)
                left(360 / sides)

create_snowflake(5, 200, 2)
mainloop()
```

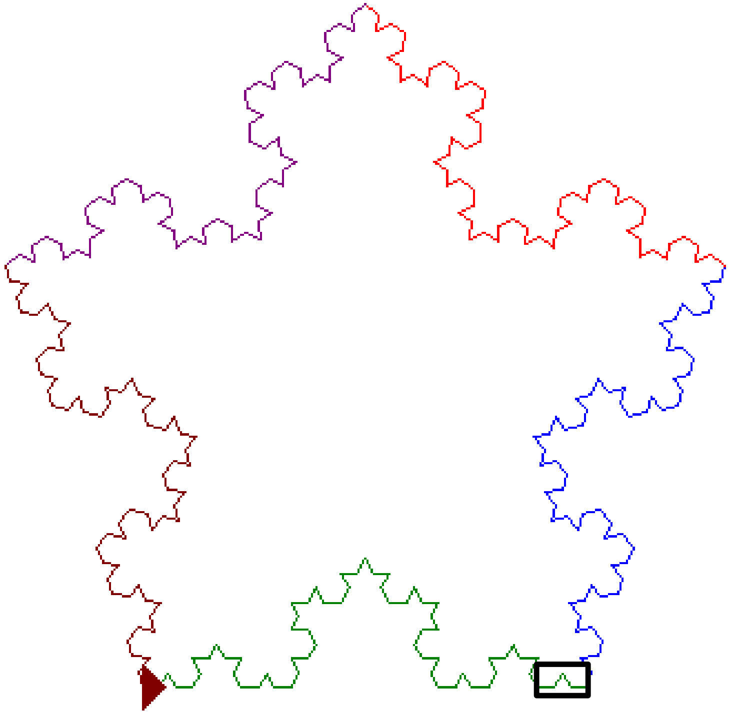
Recursion
再帰



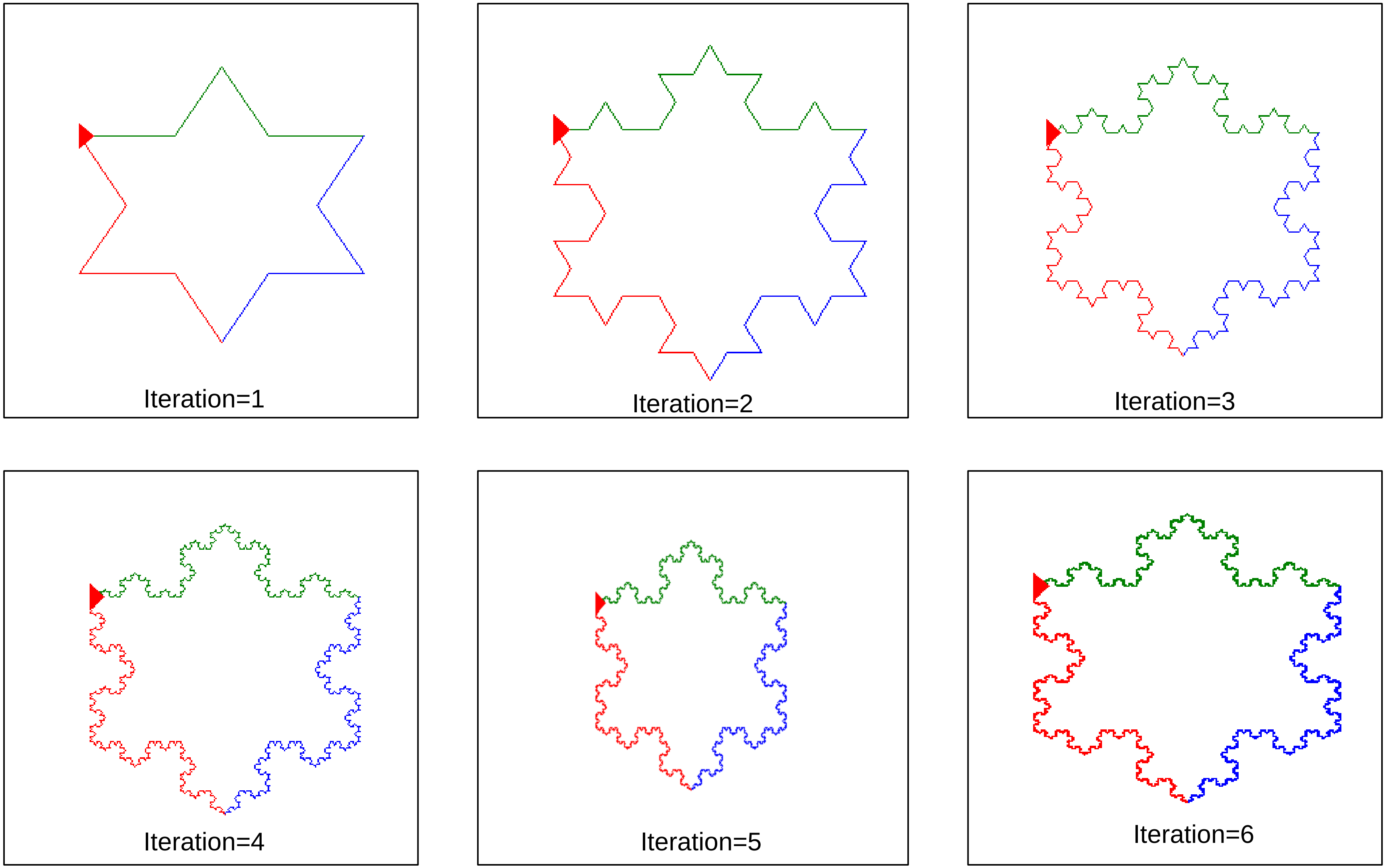
Algorithm (2-3)

```
from turtle import*
shape("arrow")
speed(0)
def snowflake_side(length, levels):
        if levels == 0:
                forward(length)
                return

        length /=3.0
        snowflake_side(length, levels -1)
        left(60)
        snowflake_side(length, levels -1)
        right(120)
        snowflake_side(length, levels -1)
        left(60)
        snowflake_side(length, levels -1)

def create_snowflake(sides, length, iteration):
        colors = ["green", "blue", "red", "purple", "maroon"]
        for i in range(sides):
                color(colors[i])
                snowflake_side(length, iteration)
                left(360 / sides)

create_snowflake(5, 200, 3)
mainloop()
```

Recursion
再帰



Python Animation No.1

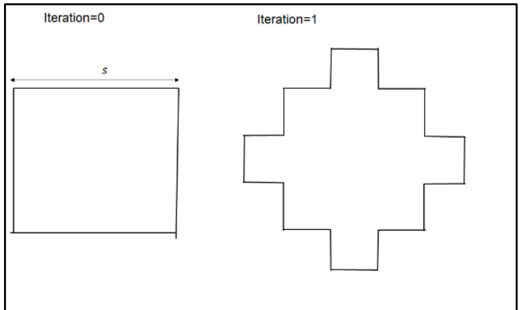


Outline

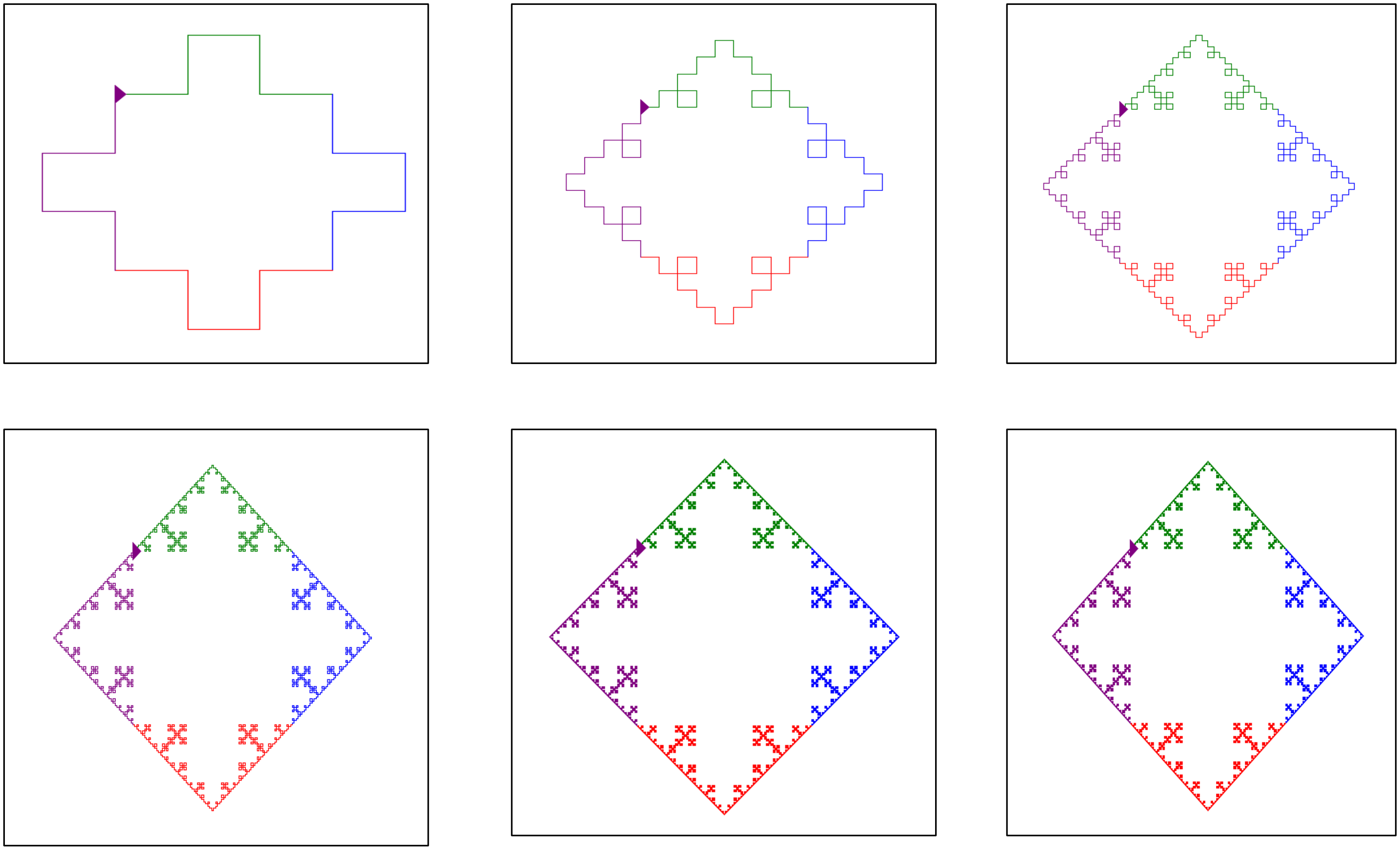
- Introduction
- Well-known fractal
  - -Sirpienski gasket
  - -Koch Curve
  - -Barnsley Fern
- Fractal Properties
- Program & Algorithm
- New Fractal
- Discussion
- Conclusion

Square Fractal properties

Iteration	0	1	2	n
Length of a Side S_n	1	1/3	1/9	$\frac{1}{3^n}$
Number of Sides N_n	4	20	100	$4 \cdot 5^n$
Perimeter P_n	4	$\frac{20}{3}$	$\frac{100}{9}$	$4 \cdot \left(\frac{5}{3}\right)^n$
Number of New Squares T_n	0	4	20	$4 \cdot 5^{n-1}$ (when n≥ 1)
Area of Each New Square a_n	$a_0(=1)$	$\frac{a_0}{9}$	$\frac{a_0}{81}$	$\frac{a_0}{9^n}$
Total Area A_n	1	$\frac{13}{9}$	$\frac{137}{81}$	$2 - \left(\frac{5}{9}\right)^n$

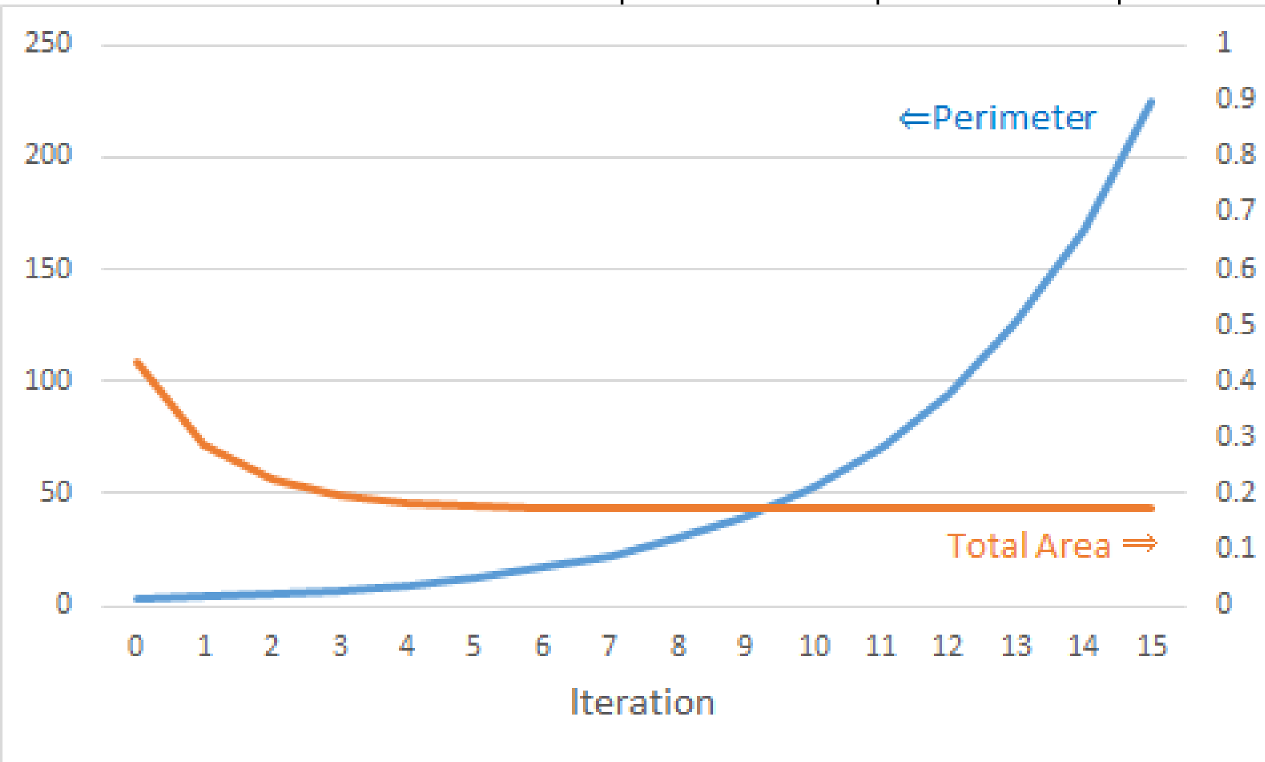


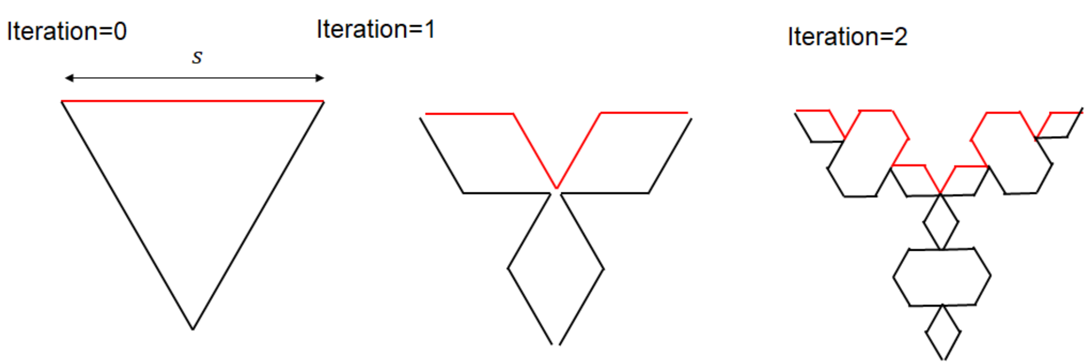
Python Animation No.2



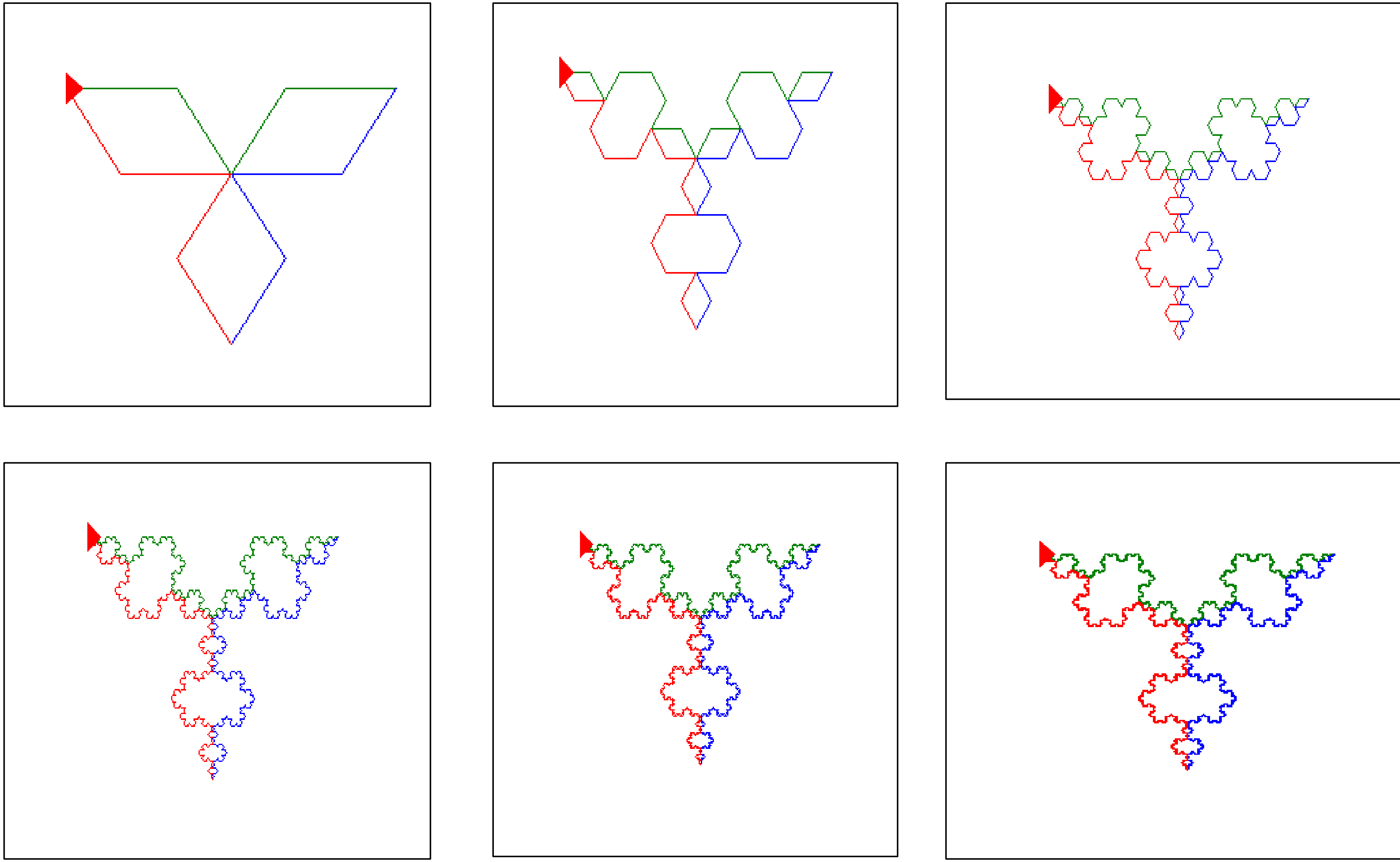
Reverse Triangle Fractal Properties

				n
Ite				n
Le				$\frac{1}{3^n}$
N				$3 \cdot 4^n$
P				$3 \cdot 1 \cdot \left(\frac{4}{3}\right)^n$
N				$3 \cdot 4^{n-1}$ (when n≥ 1)
A				$\frac{a_0}{9^n}$
To				$\frac{\sqrt{3}}{4}\frac{1}{5}\left(2+3\left(\frac{4}{9}\right)^n\right)$



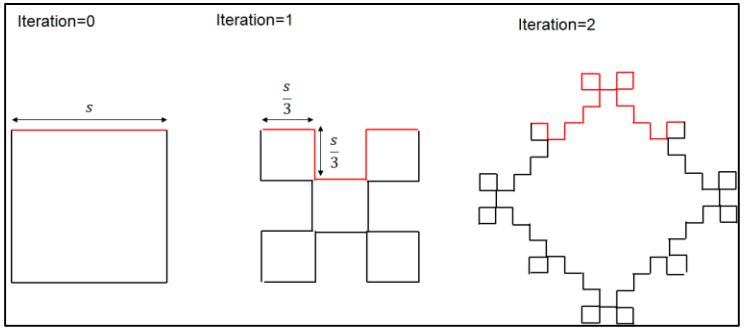


Python Animation No.3

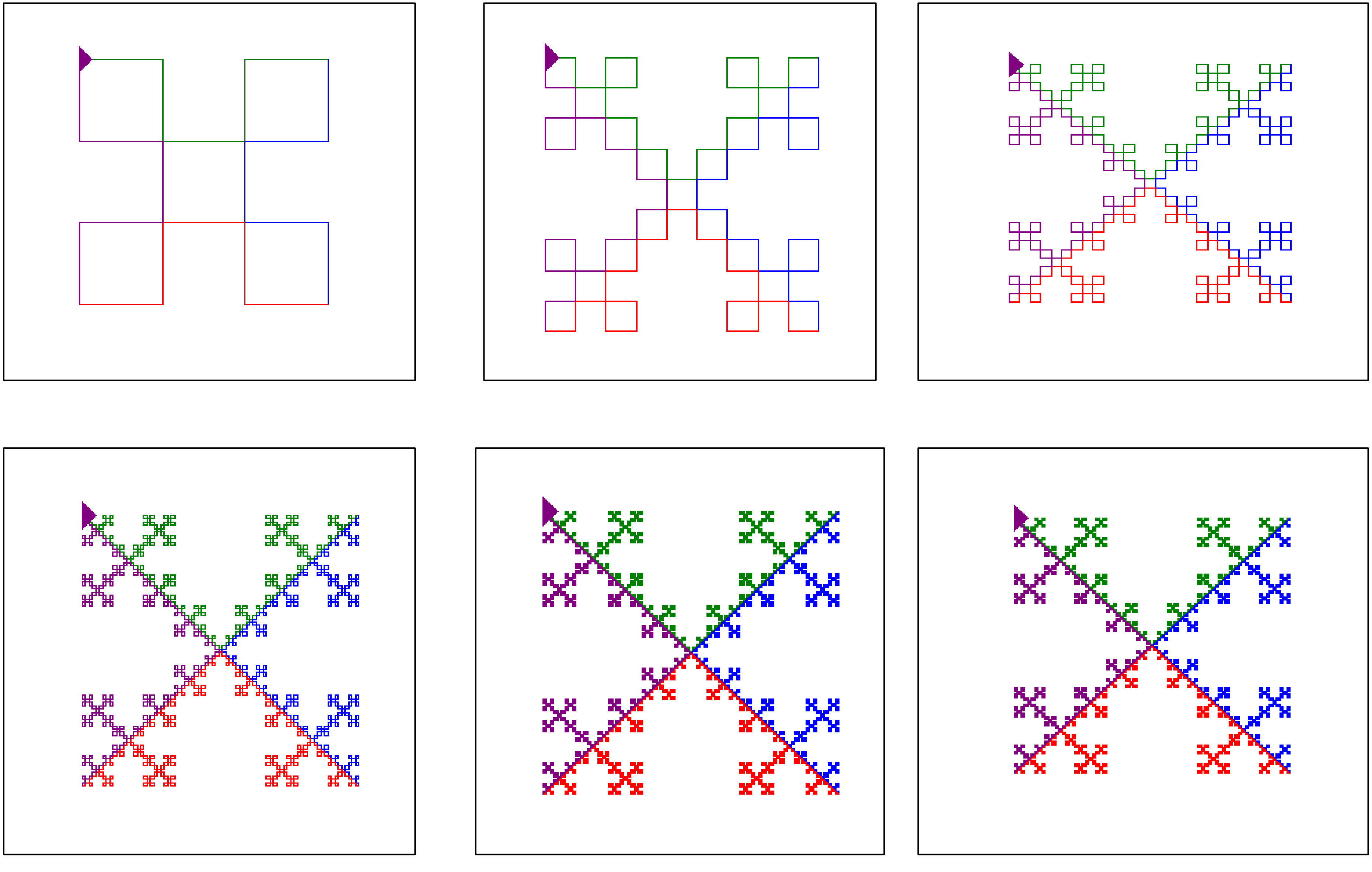


Reverse Square Fractal Properties

Iteration	0	1	2	n
Length of a Side S_n	1	$\frac{1}{3}$	$\frac{1}{9}$	$\frac{1}{3^n}$
Number of Sides N_n	4	20	100	$4 \cdot 5^n$
Perimeter P_n	4	$\frac{20}{3}$	$\frac{100}{9}$	$4 \cdot \left(\frac{5}{3}\right)^n$
Number of New Squares T_n	0	4	20	$4 \cdot 5^{n-1}$ (when n$\geq$ 1)
Area of Each New Square a_n	$a_0 = 1$	$\frac{a_0}{9}$	$\frac{a_0}{81}$	$\frac{a_0}{9^n}$
Total Area A_n	1	$\frac{5}{9}$	$\frac{25}{81}$	$\left(\frac{5}{9}\right)^n$



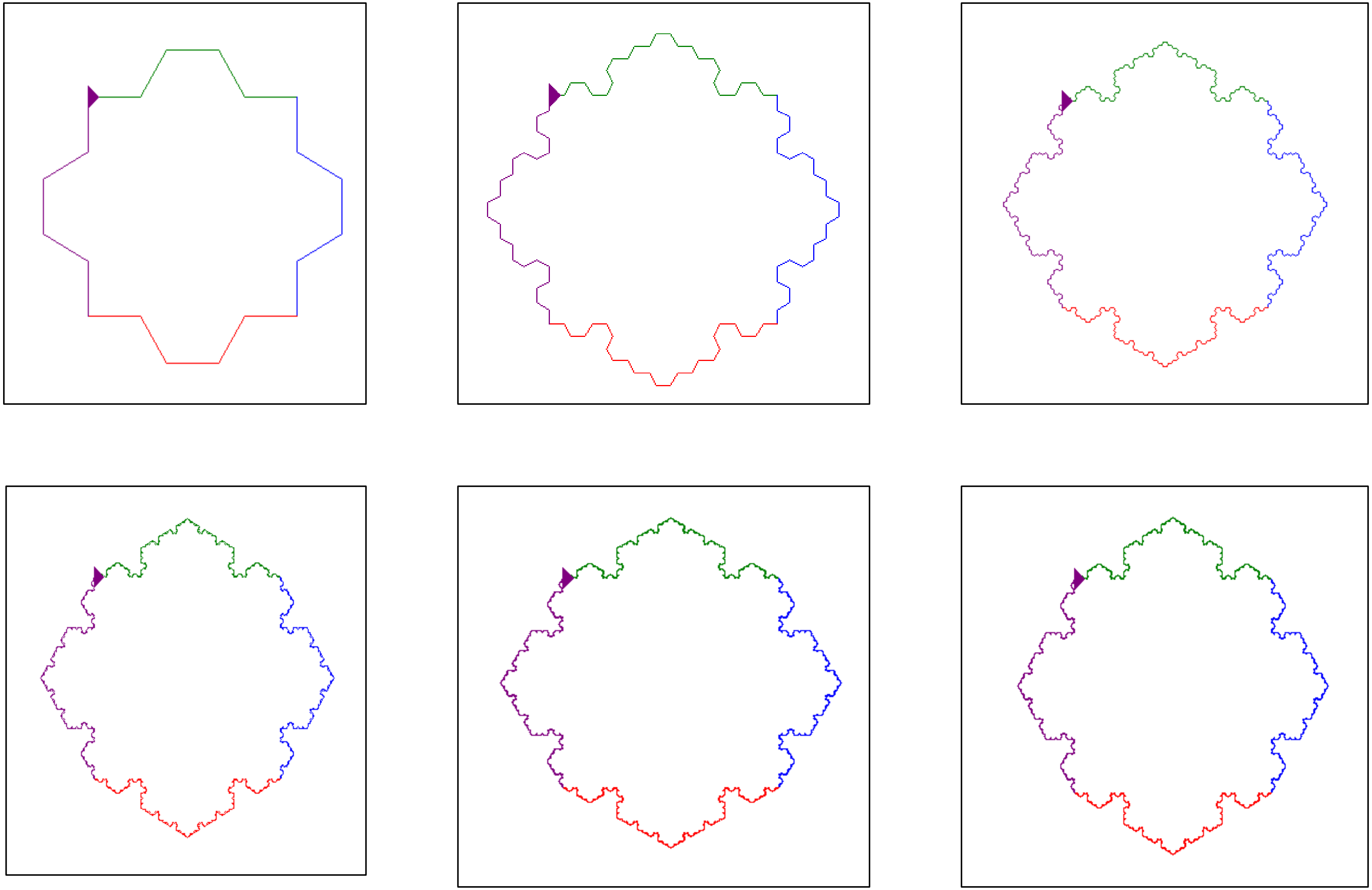
Python Animation No.4



Square-Half-Hexagon Fractal Properties

Iteration	0	1	2	n
Length of a Side S_n	1	1/4	1/16	$\frac{s}{4^n}$
Number of Sides N_n	4	20	100	$4 \cdot 5^n$
Perimeter P_n	4	5	$\frac{25}{4}$	$4 \cdot \left(\frac{5}{4}\right)^n$
Number of New Half-Hexagon T_n	0	4	20	$4 \cdot 5^{(n-1)}$ when n≥ 1
Area of Each New Half-hexagon a_n	1	$\left(\frac{1}{4}\right)^2 \cdot \frac{3\sqrt{3}}{4}$	$\left(\frac{1}{16}\right)^2 \cdot \frac{3\sqrt{3}}{4}$	$\left(\frac{1}{4^n}\right)^2 \cdot \frac{3\sqrt{3}}{4}$
Total Area A_n	1	$1 + \frac{3\sqrt{3}}{16}$	$1 + \frac{63\sqrt{3}}{256}$	$1 + \frac{3\sqrt{3}}{55}\left\{1 - \left(\frac{5}{16}\right)^n\right\}$

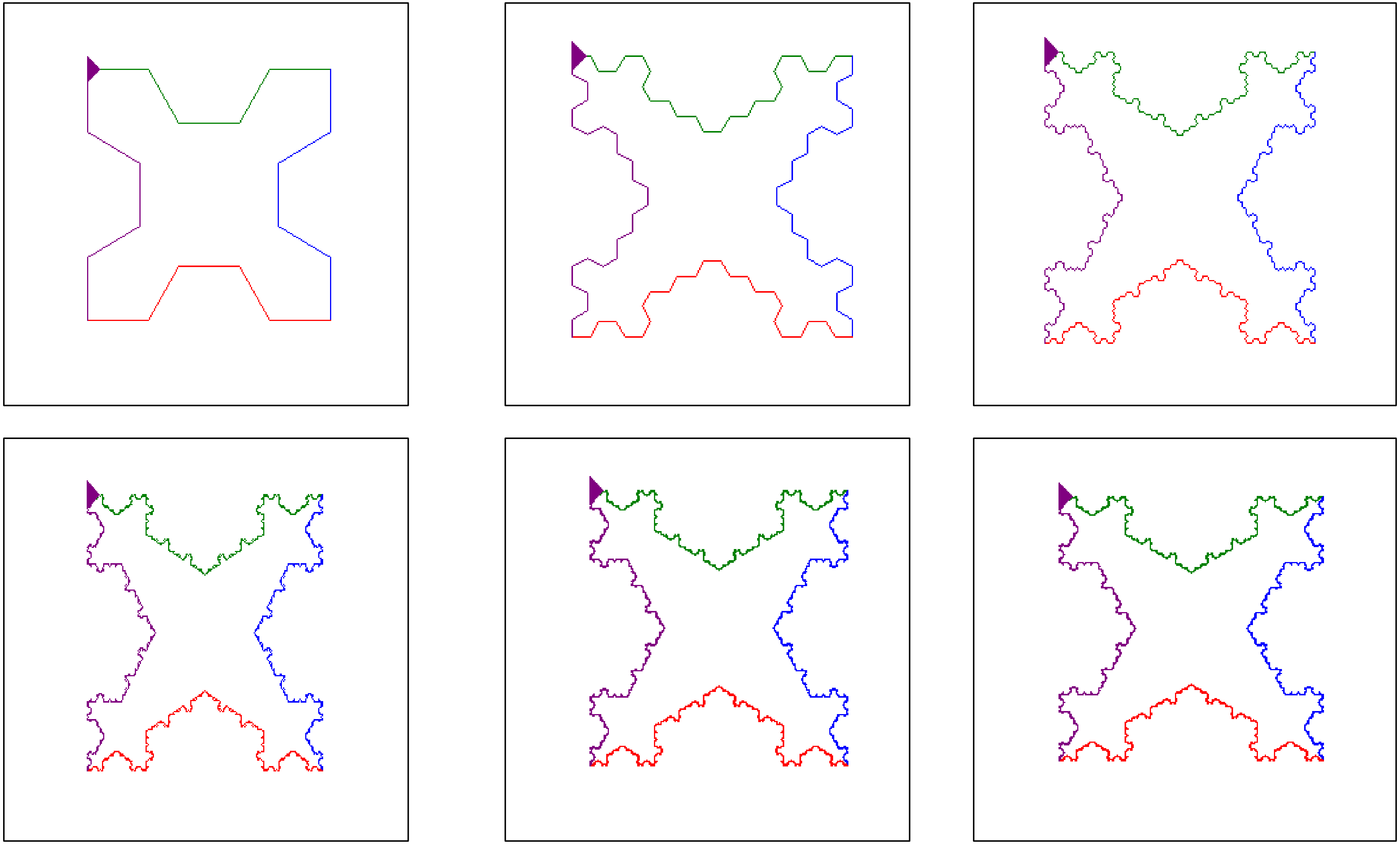
Python Animation No.5



Reverse Square-Half-Hexagon Fractal Properties

Iteration	0	1	2	n
Length of a Side S_n	1	1/4	1/16	$\frac{s}{4^n}$
Number of Sides N_n	4	20	100	$4 \cdot 5^n$
Perimeter P_n	4	5	$\frac{25}{4}$	$4 \cdot \left(\frac{5}{4}\right)^n$
Number of New Half-Hexagon T_n	0	4	20	$4 \cdot 5^{(n-1)}$ when n≥ 1
Area of Each New Half-hexagon a_n	1	$\left(\frac{1}{4}\right)^2 \cdot \frac{3\sqrt{3}}{4}$	$\left(\frac{1}{16}\right)^2 \cdot \frac{3\sqrt{3}}{4}$	$\left(\frac{1}{4^n}\right)^2 \cdot \frac{3\sqrt{3}}{4}$
Total Area A_n	1	$1 - \frac{3\sqrt{3}}{16}$	$1 - \frac{63\sqrt{3}}{256}$	$1 - \frac{3\sqrt{3}}{55}\left\{1 - \left(\frac{5}{16}\right)^n\right\}$

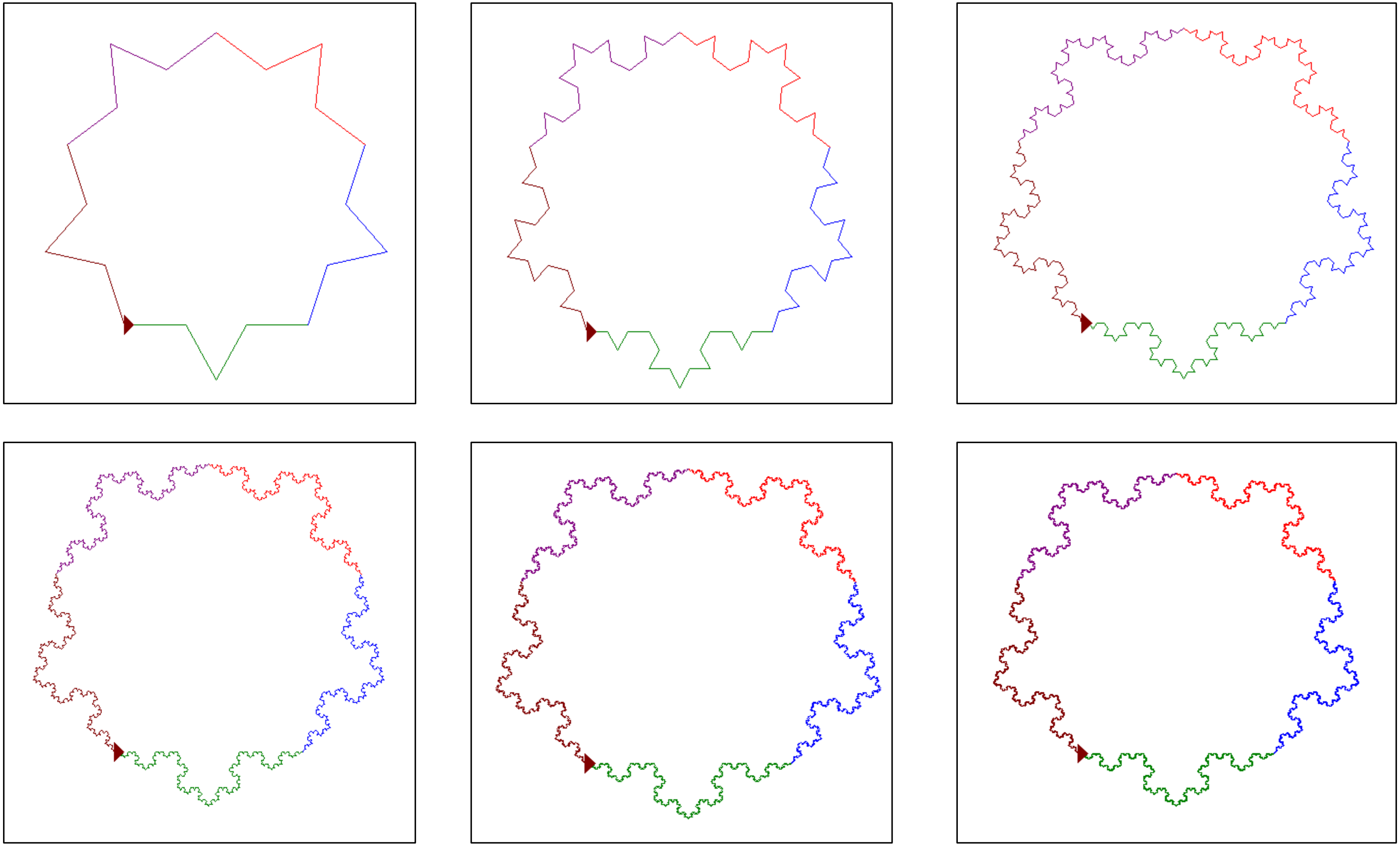
Python Animation No.6



Pentagon-Triangle Fractal Properties

Iteration	0	1	2	n
Length of a Side S_n	1	$\frac{1}{3}$	$\frac{1}{9}$	$\frac{1}{3^n}$
Number of Sides N_n	5	20	80	$5 \cdot 4^n$
Perimeter P_n	5	$\frac{20}{3}$	$\frac{80}{9}$	$5 \cdot \left(\frac{4}{3}\right)^n$
Number of New Triangles T_n	0	5	20	$5 \cdot 4^{(n-1)}$ (n≥ 1)
Area of Each New Triangles a_n	$\frac{5}{8}\sqrt{10+2\sqrt{5}}$	$\frac{1}{9}$	$\frac{1}{81}$	$\frac{1}{9^n}$ (n≥ 1)
Total Area A_n	$\frac{5}{8}\sqrt{10+2\sqrt{5}}$	$\frac{5}{8}\sqrt{10+2\sqrt{5}} + \frac{5}{9}$	$\frac{5}{8}\sqrt{10+2\sqrt{5}} + \frac{65}{81}$	$\frac{5}{8}\sqrt{10+2\sqrt{5}} + \left\{1-\left(\frac{4}{9}\right)^n\right\}$

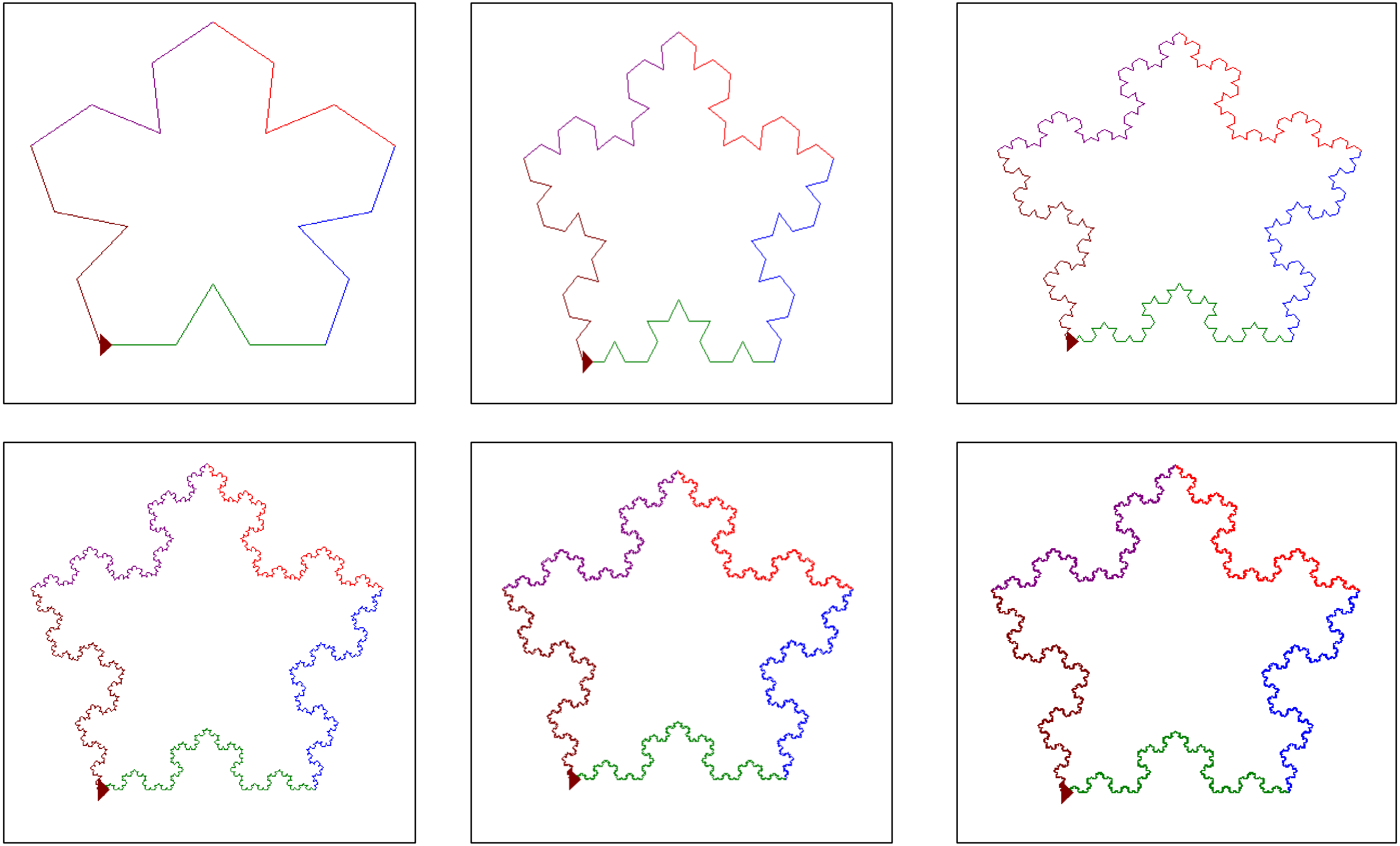
Python Animation No.7



Reverse Pentagon-Triangle Fractal Properties

Iteration	0	1	2	n
Length of a Side S_n	1	$\frac{1}{3}$	$\frac{1}{9}$	$\frac{1}{3^n}$
Number of Sides N_n	5	20	80	$5 \cdot 4^n$
Perimeter P_n	5	$\frac{20}{3}$	$\frac{80}{9}$	$5 \cdot \left(\frac{4}{3}\right)^n$
Number of New Triangles T_n	0	5	20	$5 \cdot 4^{(n-1)}$ (n≥ 1)
Area of Each New Triangles a_n	$\frac{5}{8}\sqrt{10+2\sqrt{5}}$	$\frac{1}{9}$	$\frac{1}{81}$	$\frac{1}{9^n}$ (n≥ 1)
Total Area A_n	$\frac{5}{8}\sqrt{10+2\sqrt{5}}$	$\frac{5}{8}\sqrt{10+2\sqrt{5}} - \frac{5}{9}$	$\frac{5}{8}\sqrt{10+2\sqrt{5}} - \frac{65}{81}$	$\frac{5}{8}\sqrt{10+2\sqrt{5}} - \left\{1 - \left(\frac{4}{9}\right)^n\right\}$

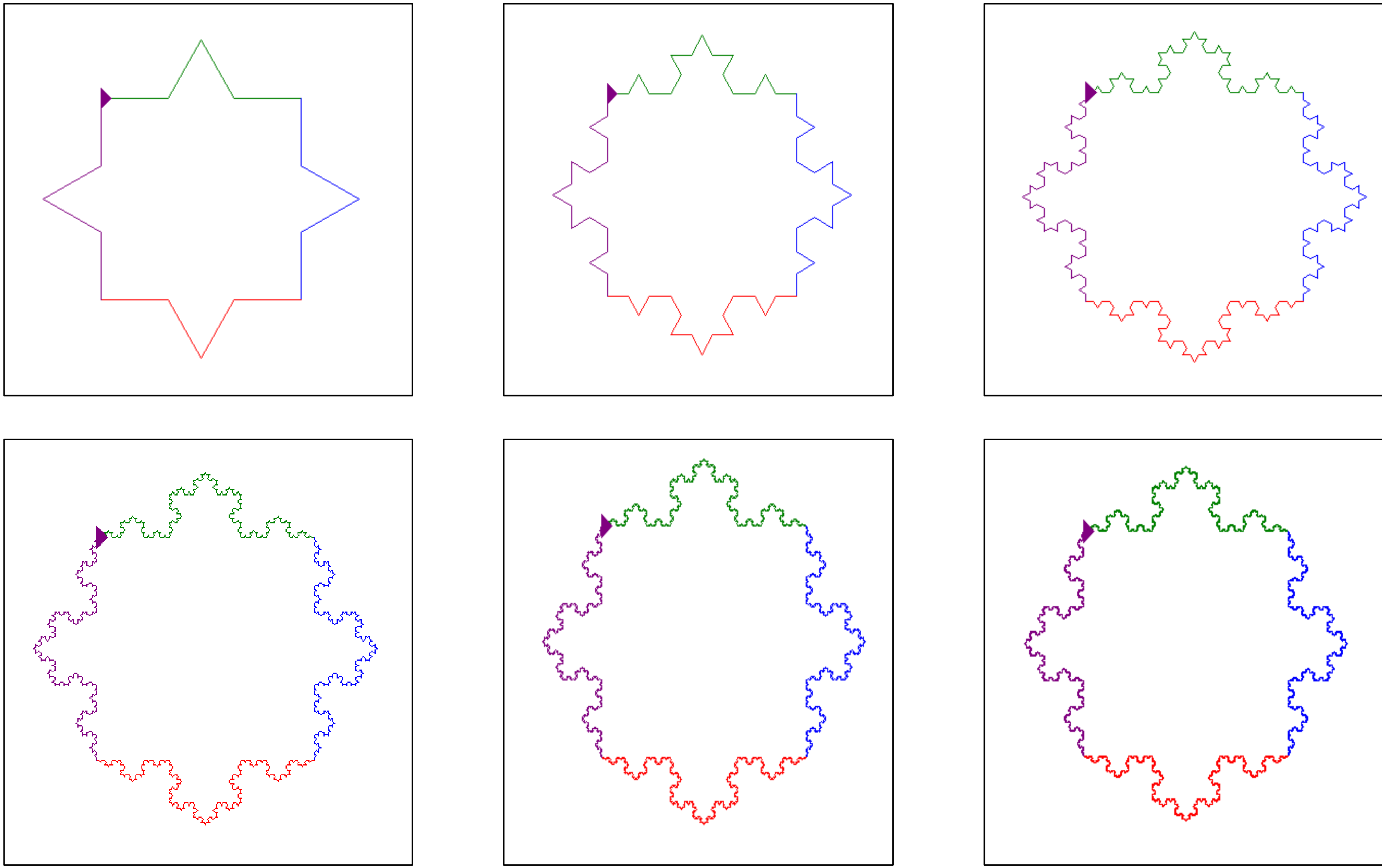
Python Animation No.8



Square-Triangle Fractal Properties

Iteration	0	1	2	n
Length of a Side S_n	1	$\frac{1}{3}$	$\frac{1}{9}$	$\frac{1}{3^n}$
Number of Sides N_n	4	16	64	$4 \cdot 4^n$
Perimeter P_n	4	$\frac{16}{3}$	$\frac{64}{9}$	$4 \cdot \left(\frac{4}{3}\right)^n$
Number of New Triangles T_n	0	4	16	$4 \cdot 4^{n-1}$ (when n≥ 1)
Area of Each New Triangles a_n	1	$\frac{\sqrt{3}}{4} \cdot \frac{1}{9}$	$\frac{\sqrt{3}}{4} \cdot \frac{1}{81}$	$\frac{\sqrt{3}}{4} \cdot \left(\frac{1}{9}\right)^n$
Total Area A_n	1	$1 + \frac{\sqrt{3}}{36}$	$1 + \frac{5\sqrt{3}}{162}$	$1 + \frac{\sqrt{3}}{32}\left\{1 - \left(\frac{1}{9}\right)^n\right\}$

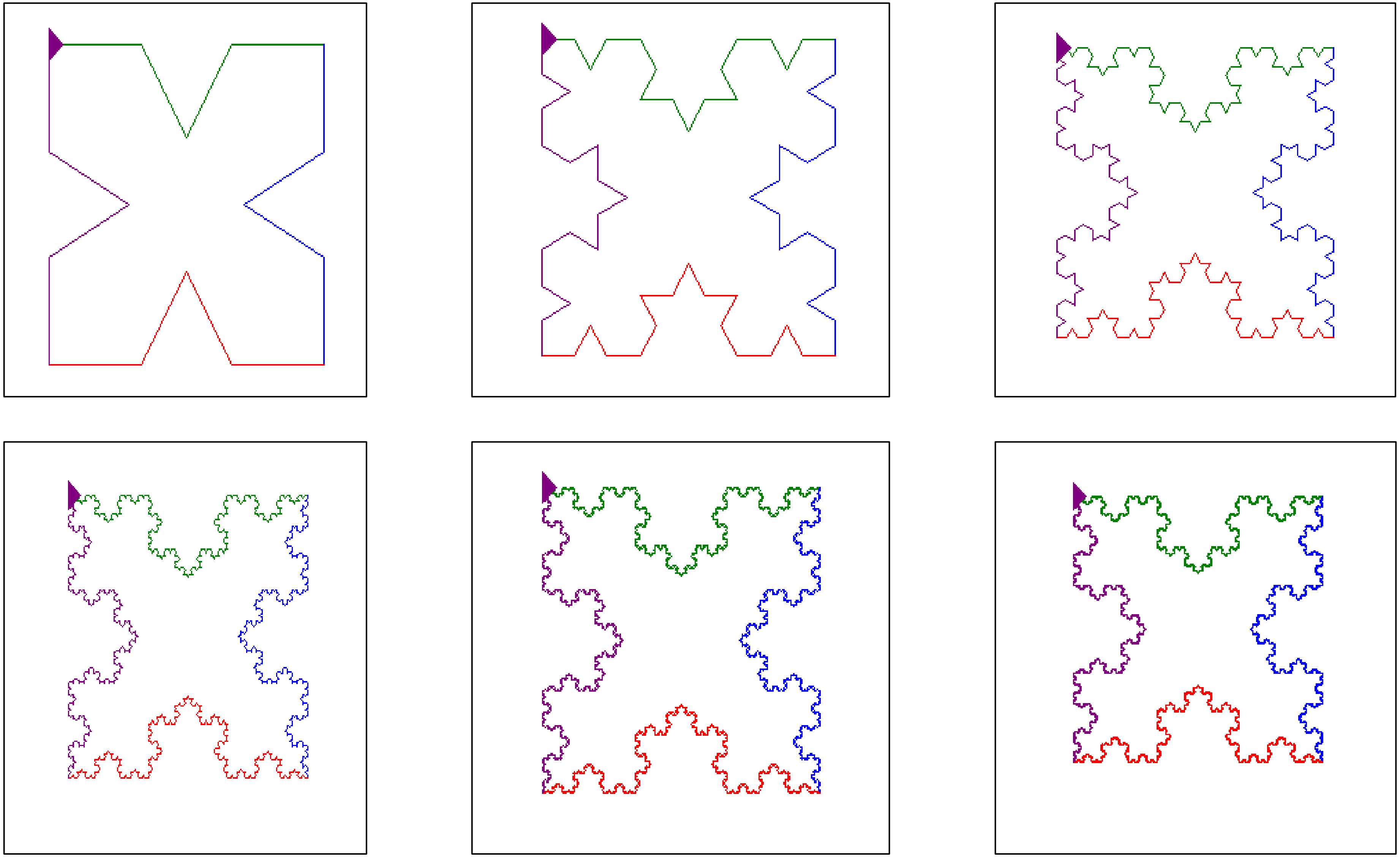
Python Animation No.9



Reverse Square-Triangle Fractal Properties

Iteration	0	1	2	n
Length of a Side S_n	1	$\frac{1}{3}$	$\frac{1}{9}$	$\frac{1}{3^n}$
Number of Sides N_n	4	16	64	$4 \cdot 4^n$
Perimeter P_n	4	$\frac{16}{3}$	$\frac{64}{9}$	$4 \cdot \left(\frac{4}{3}\right)^n$
Number of New Triangles T_n	0	4	16	$4 \cdot 4^{n-1}$ (when n≥ 1)
Area of Each New Triangles a_n	1	$\frac{\sqrt{3}}{4} \cdot \frac{1}{9}$	$\frac{\sqrt{3}}{4} \cdot \frac{1}{81}$	$\frac{\sqrt{3}}{4} \cdot \left(\frac{1}{9}\right)^n$
Total Area A_n	1	$1 - \frac{\sqrt{3}}{36}$	$1 - \frac{5\sqrt{3}}{162}$	$1 - \frac{\sqrt{3}}{32}\left\{1 - \left(\frac{1}{9}\right)^n\right\}$

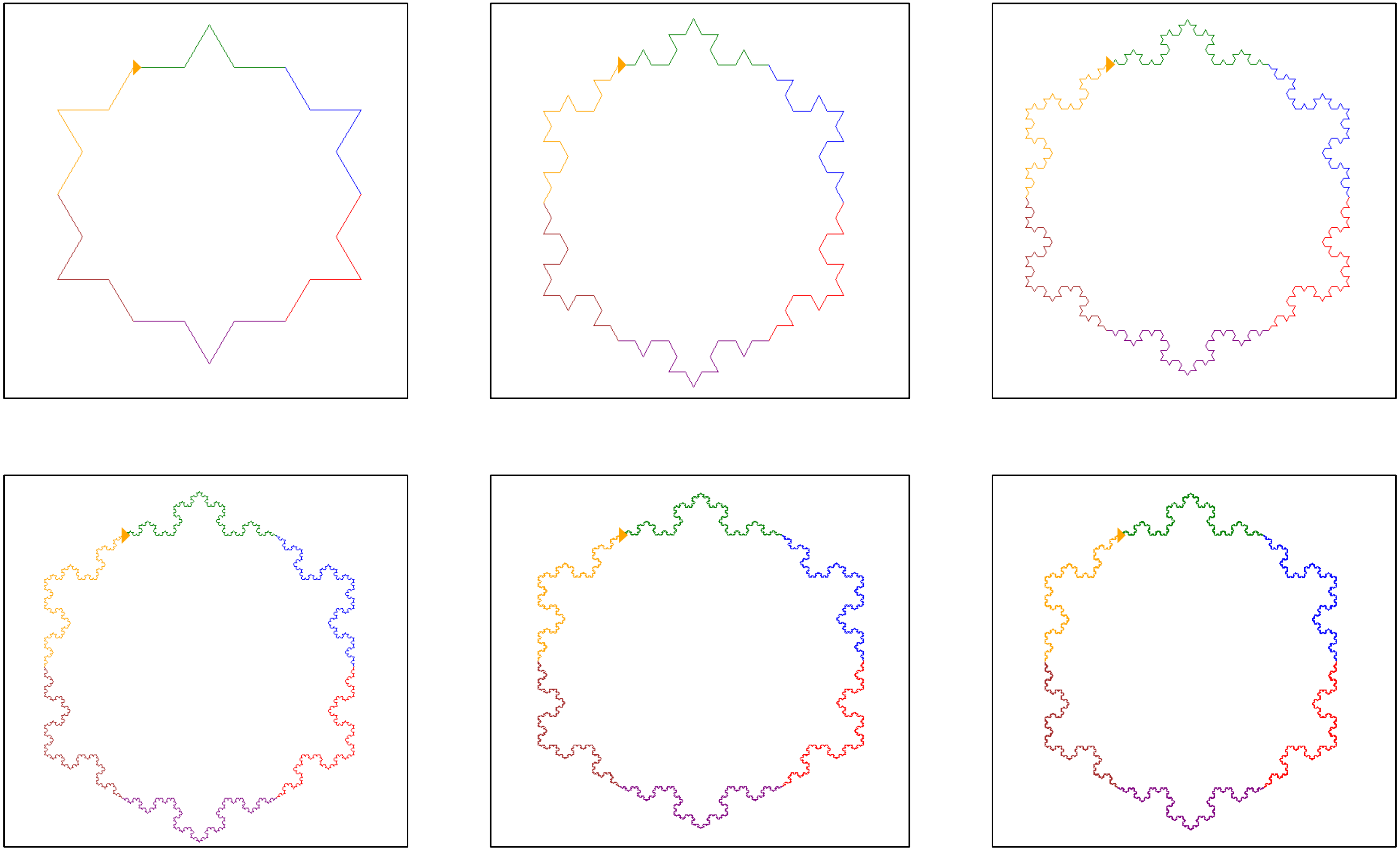
Python Animation No.10



Hexagon-Triangle Fractal Properties

Iteration	0	1	2	n
Length of a Side S_n	1	$\frac{1}{3}$	$\frac{1}{9}$	$\frac{1}{3^n}$
Number of Sides N_n	6	24	96	$6 \cdot 4^n$
Perimeter P_n	6	8	$\frac{32}{3}$	$6 \cdot \left(\frac{4}{3}\right)^n$
Number of New Triangles T_n	0	6	24	$6 \cdot 4^{n-1}$ (when n≥ 1)
Area of Each New Triangles a_n	$\frac{3\sqrt{3}}{2}$	$\frac{\sqrt{3}}{4} \cdot \frac{1}{9}$	$\frac{\sqrt{3}}{4} \cdot \frac{1}{81}$	$\frac{\sqrt{3}}{4} \cdot \left(\frac{1}{9}\right)^n$
Total Area A_n	$\frac{3\sqrt{3}}{2}$	$\frac{55\sqrt{3}}{36}$	$\frac{124\sqrt{3}}{81}$	$\frac{3\sqrt{3}}{2} + \frac{\sqrt{3}}{32}\left\{1 - \left(\frac{1}{9}\right)^n\right\}$

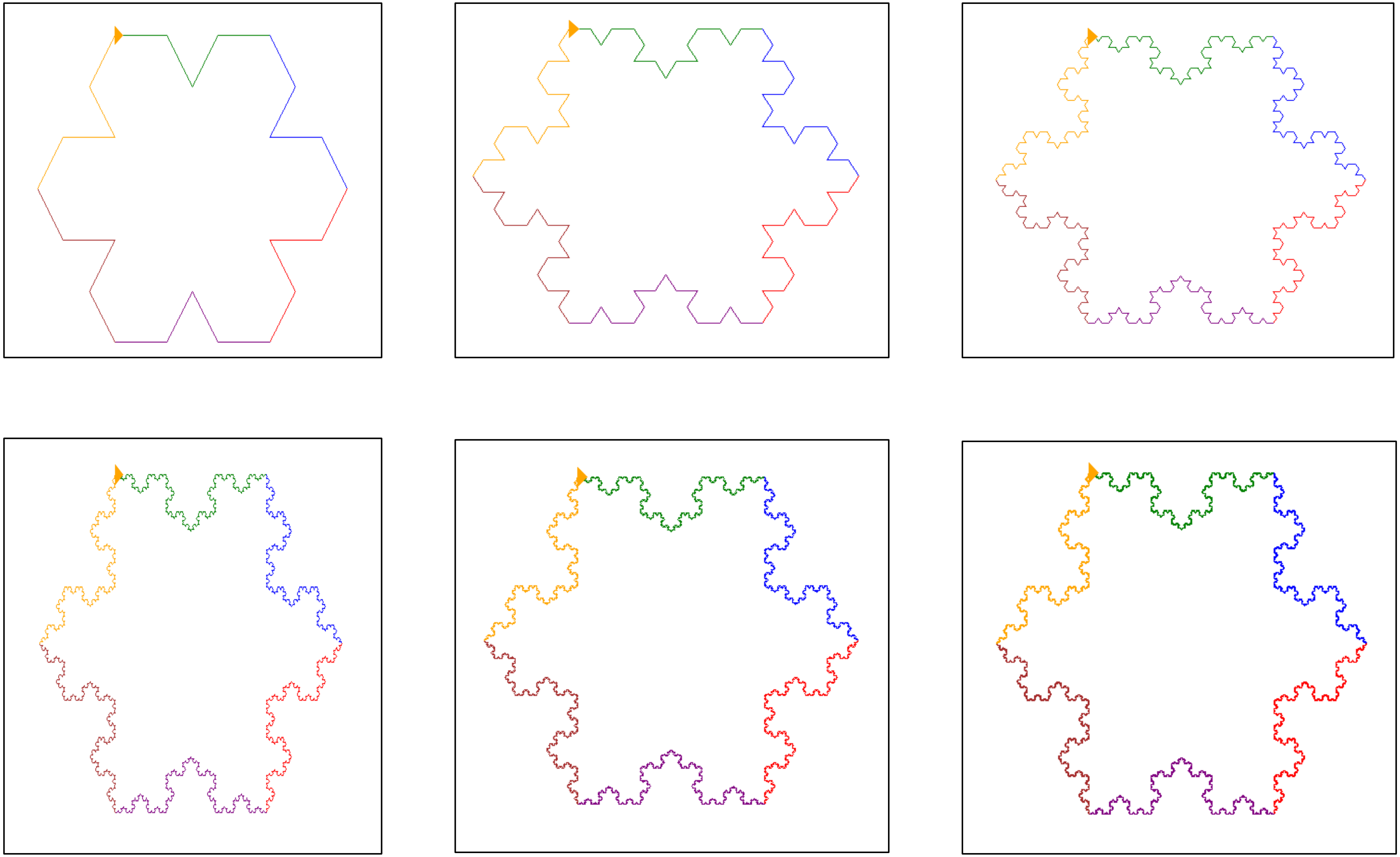
Python Animation No.11



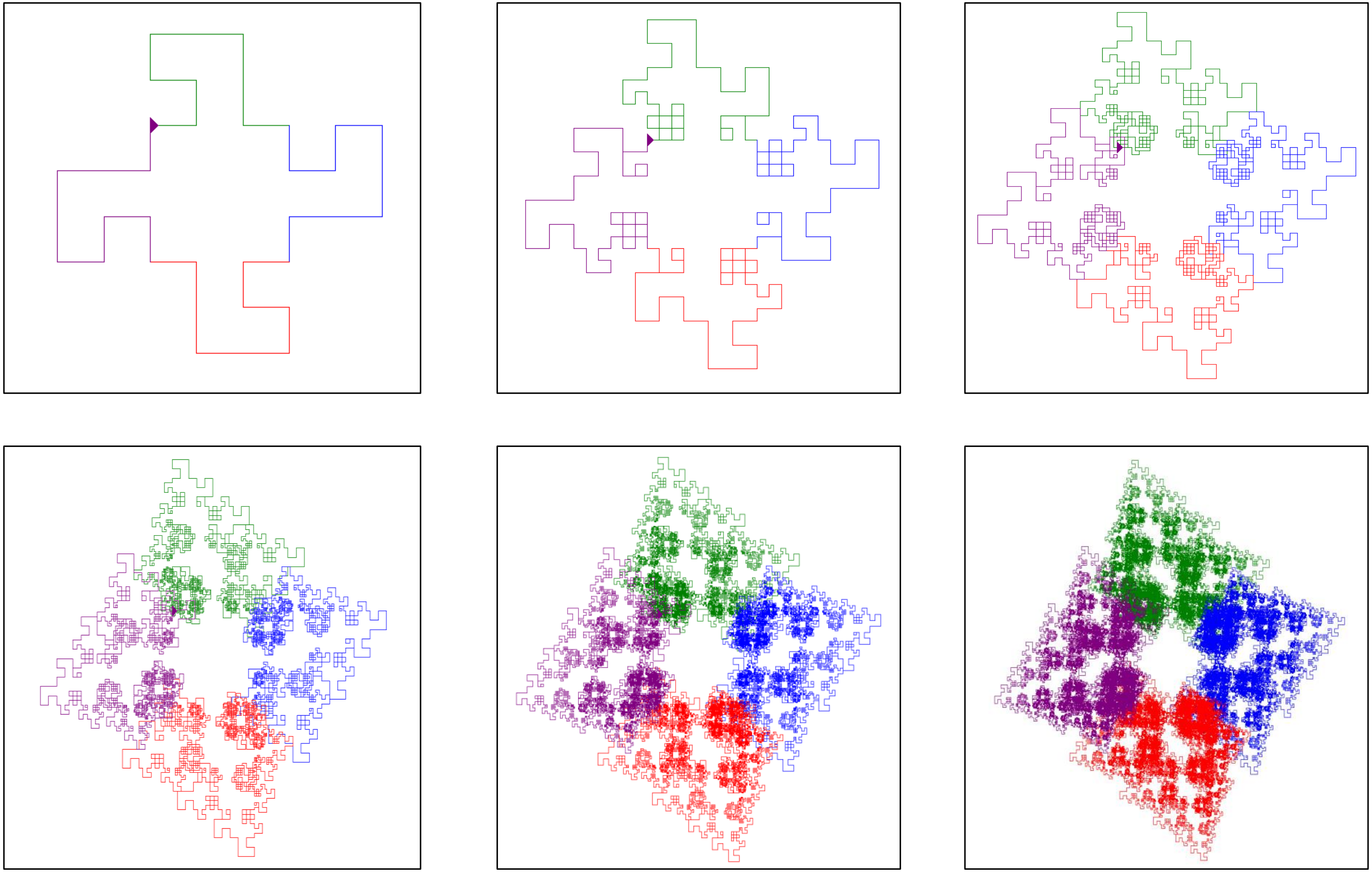
Reverse Hexagon-Triangle Fractal Properties

Iteration	0	1	2	n
Length of a Side S_n	1	$\frac{1}{3}$	$\frac{1}{9}$	$\frac{1}{3^n}$
Number of Sides N_n	6	24	96	$6 \cdot 4^n$
Perimeter P_n	6	8	$\frac{32}{3}$	$6 \cdot \left(\frac{4}{3}\right)^n$
Number of New Triangles T_n	0	6	24	$6 \cdot 4^{n-1}$ (when n≥ 1)
Area of Each New Triangles a_n	$\frac{3\sqrt{3}}{2}$	$\frac{\sqrt{3}}{4} \cdot \frac{1}{9}$	$\frac{\sqrt{3}}{4} \cdot \frac{1}{81}$	$\frac{\sqrt{3}}{4} \cdot \left(\frac{1}{9}\right)^n$
Total Area A_n	$\frac{3\sqrt{3}}{2}$	$\frac{53\sqrt{3}}{36}$	$\frac{119\sqrt{3}}{81}$	$\frac{3\sqrt{3}}{2} - \frac{\sqrt{3}}{32}\left\{1 - \left(\frac{1}{9}\right)^n\right\}$

Python Animation No.12



Python Animation No.13

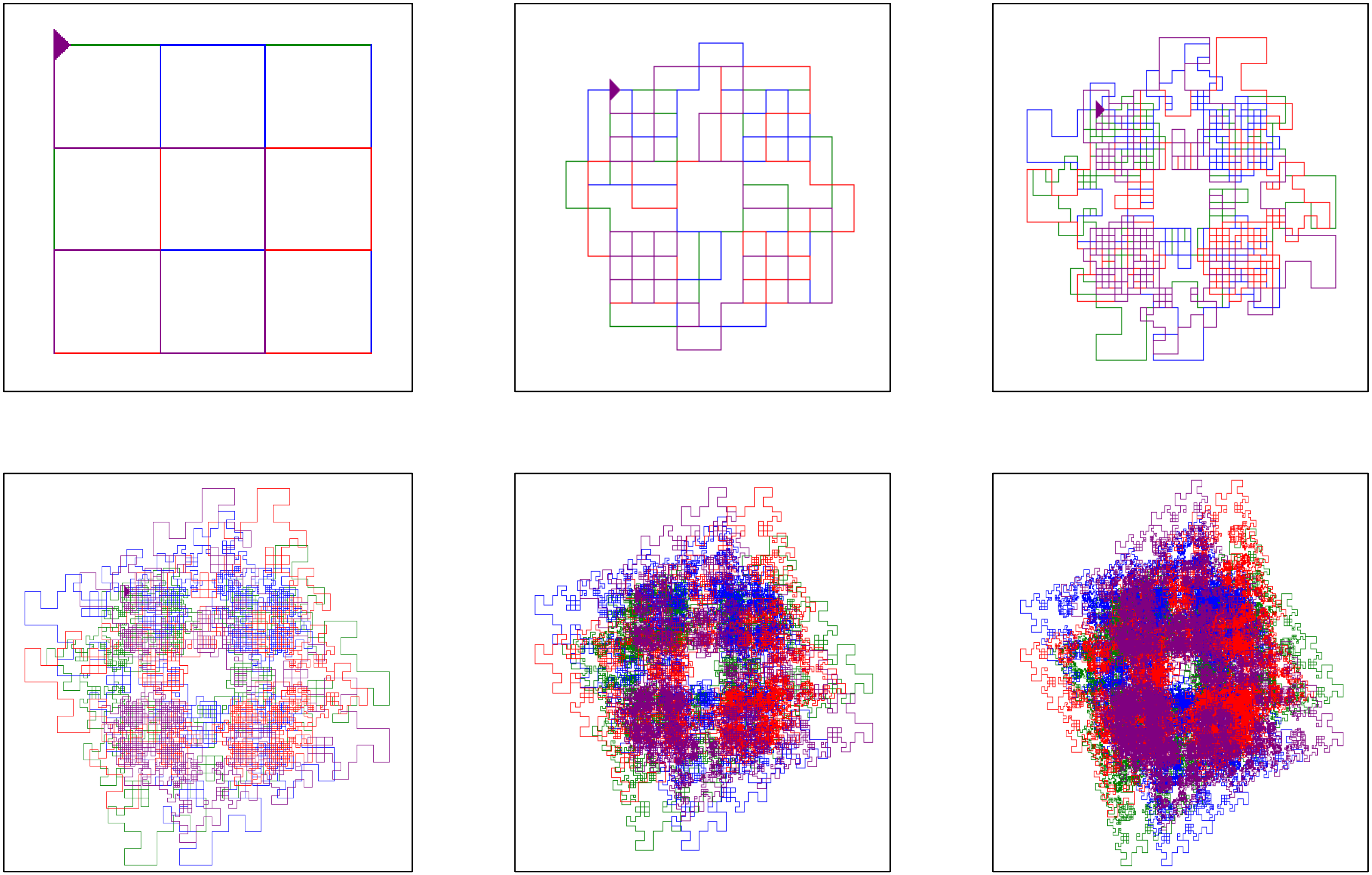


Square-L-Shaped Fractal Properties

Iteration	0	1	2	n
Length of a Side S_n				
Number of Sides N_n				
Perimeter P_n	4	12	36	$4 \cdot 3^n$
Number of New L-Shapes T_n	0	4	20	$4 \cdot 5^{n-1}$ (when n≥ 1)
Area of Each New L-Shapes a_n	1			
Total Area A_n	1	$\frac{7}{3}$	$\frac{115}{27}$	$\left\{7 - 6\left(\frac{2}{3}\right)^n\right\}^2$

Because of the complexity of the shape,
it cannot be formulated using the same procedure as other fractals.
When iteration is n, total area is an approximation.

Python Animation No.14



Reverse Square-L-Shaped Fractal Properties

Iteration	0	1	2	n
Length of a Side S_n				
Number of Sides N_n				
Perimeter P_n	4	12	36	$4 \cdot 3^n$
Number of New L-Shapes T_n	0	4	20	$4 \cdot 5^{n-1}$ (when n≥ 1)
Area of Each New L-Shapes a_n	1			
Total Area A_n	1	1	$\frac{49}{9}$	$\left\{5 - 6\left(\frac{2}{3}\right)^n\right\}^2$

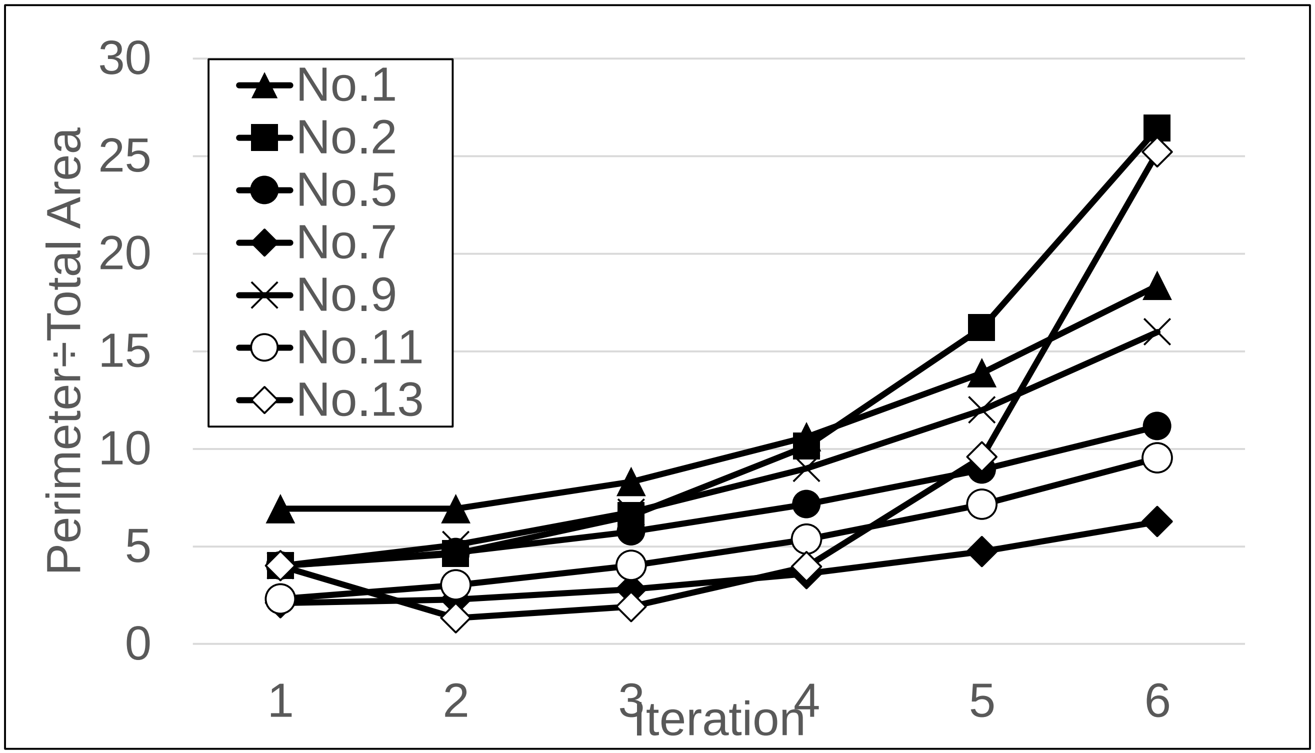
Because of the complexity of the shape,
it cannot be formulated using the same procedure as other fractals.
When iteration is n, total area is an approximation.

Outline

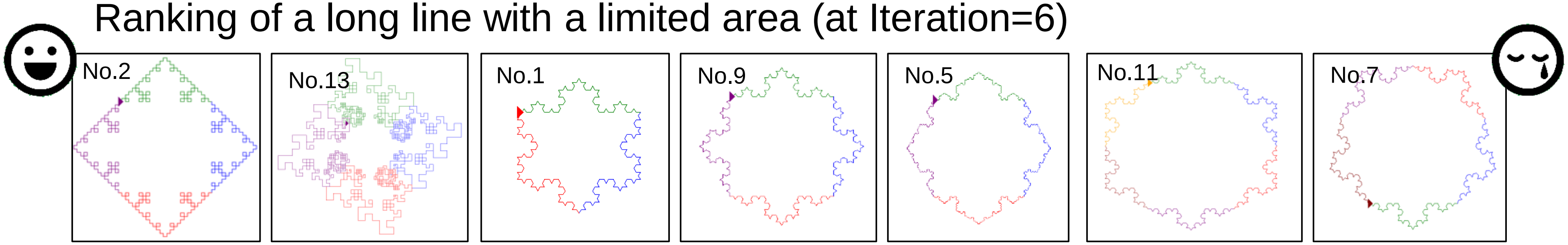
- Introduction
- Well-known fractal
  - -Sirpienski gasket
  - -Koch Curve
  - -Barnsley Fern
- Fractal Properties
- Program & Algorithm
- New Fractal
- **Discussion**
- Conclusion

Discussion

Fractal: Limited area, Long line.
Fractal Resistor: Limited area, large resistor



Ranking of a long line with a limited area (at Iteration=6)



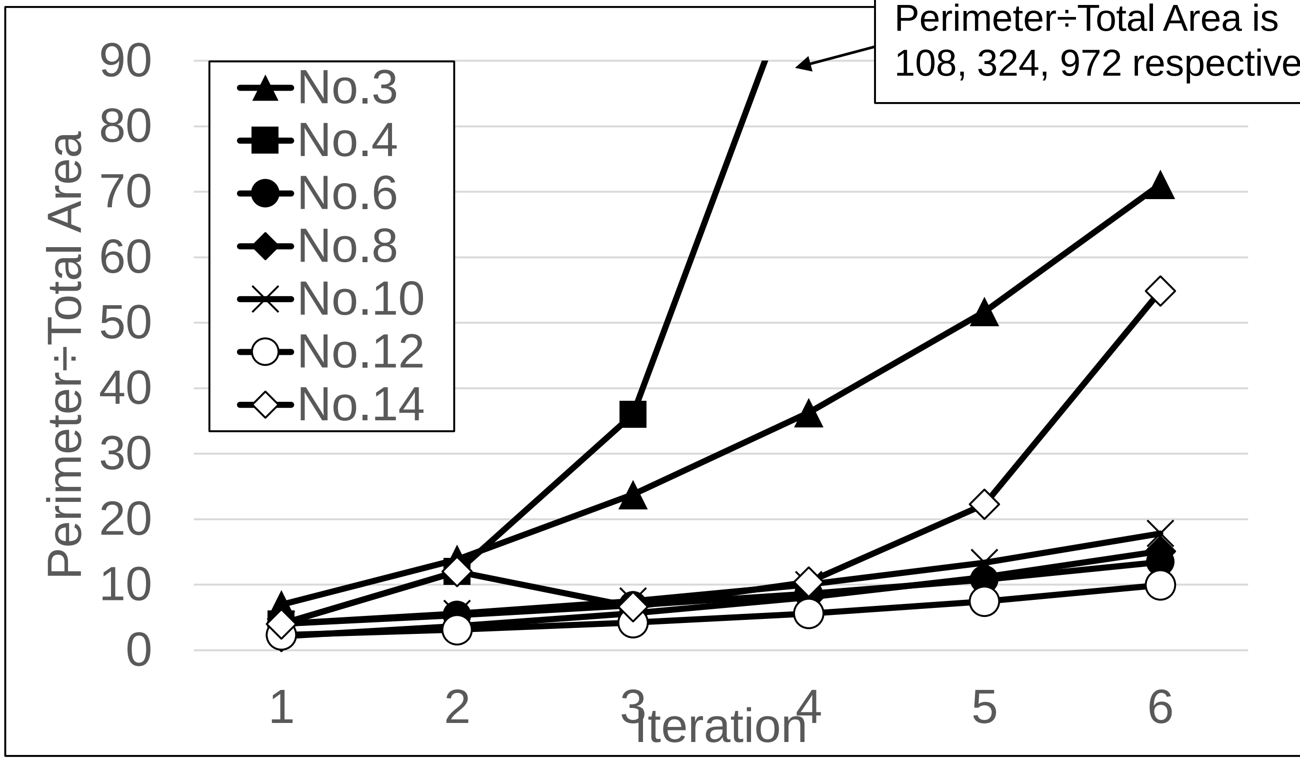
Discussion

Fractal: Limited area, Long line.

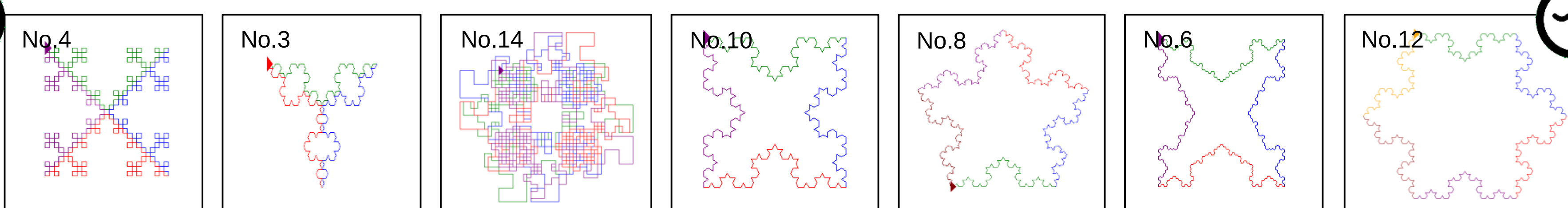
Fractal Resistor: Limited area, large resistor

Reverse Fractals

When Iteration is 4, 5, 6
Perimeter÷Total Area is
108, 324, 972 respectivelly.



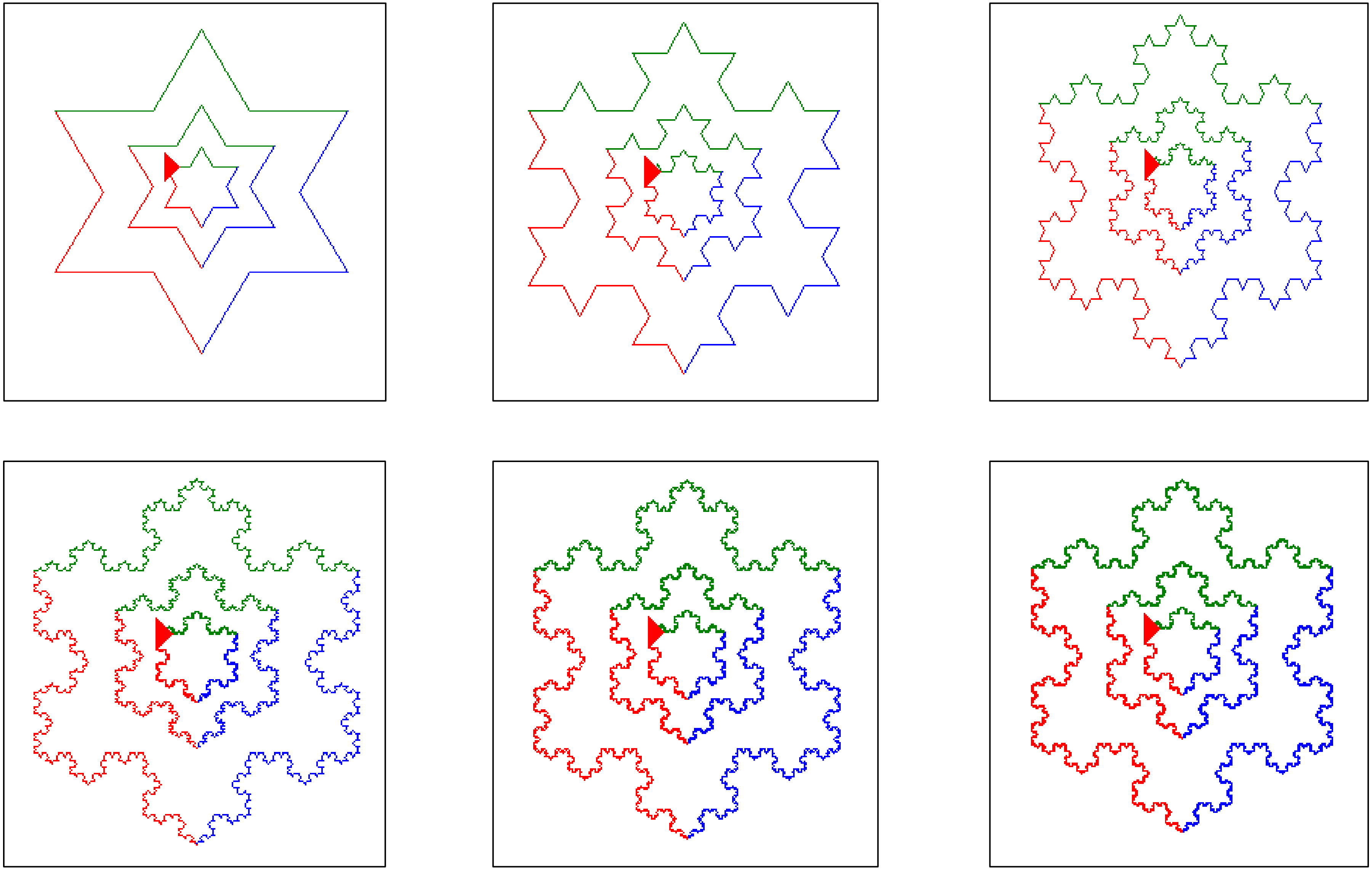
Ranking of a long line with a limited area (at Iteration=6)



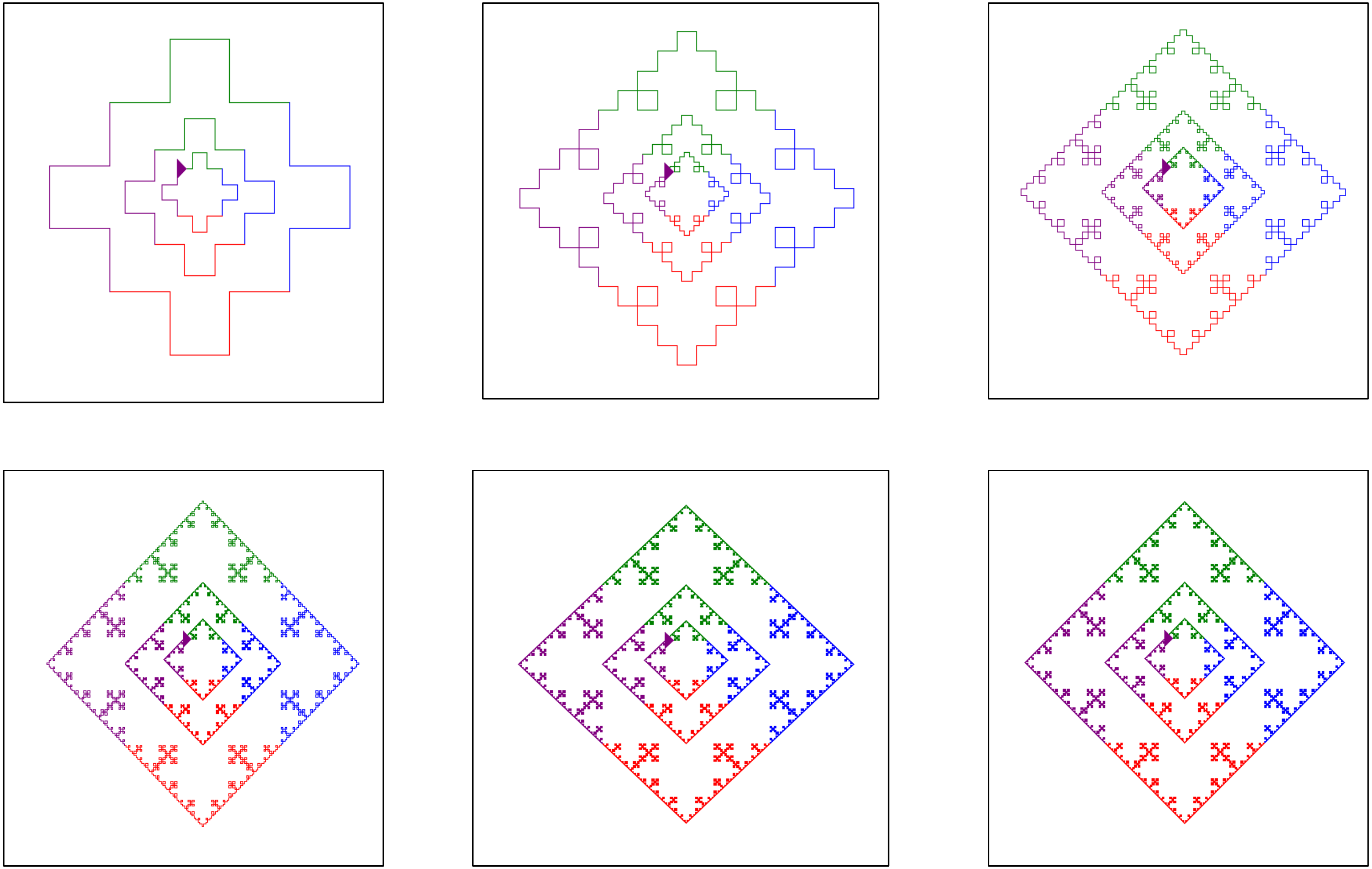
Appendix (Overlapping Fractals)

- Python Animation of overlapping fractals that contain area inside.
- Total area is the same as the original fractal
- For perimeter of original shape is *P* then perimeter of overlapping image is $\frac{9}{4} \times P$ (when $i = n$)
- Python Animation No.1 as an example
- The total area is the same as the original Python Animation No.1
- The perimeter although is different from the original animation.
- The perimeter of the original No.1 is $4 \cdot \left(\frac{5}{3}\right)^n$ (when $i = n$) but the new perimeter for the overlapping fractals No.1 is $\frac{9}{4}\left\{4 \cdot \left(\frac{5}{3}\right)^n\right\}$ (when $i = n$) .

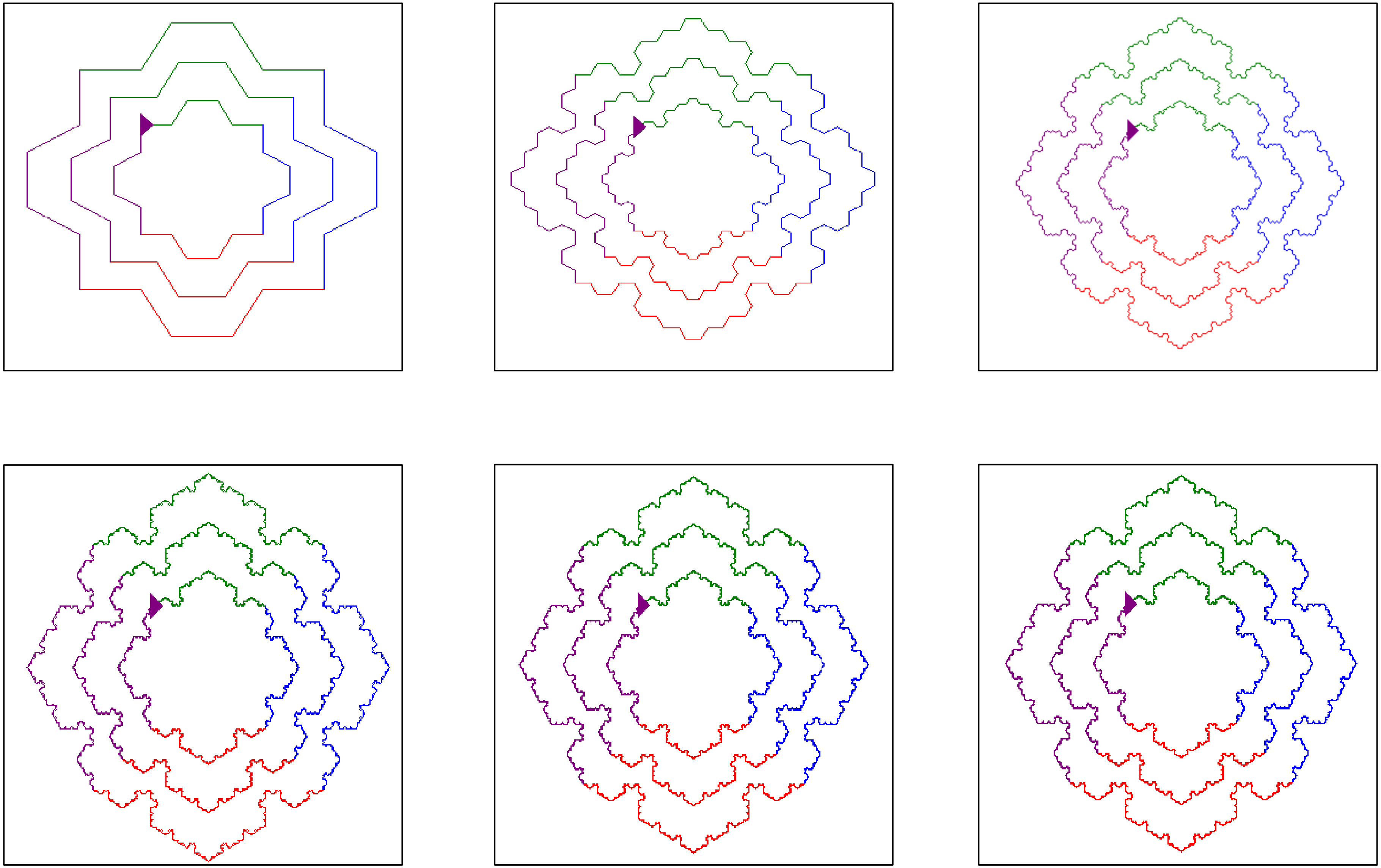
Python Animation (overlap) No.1



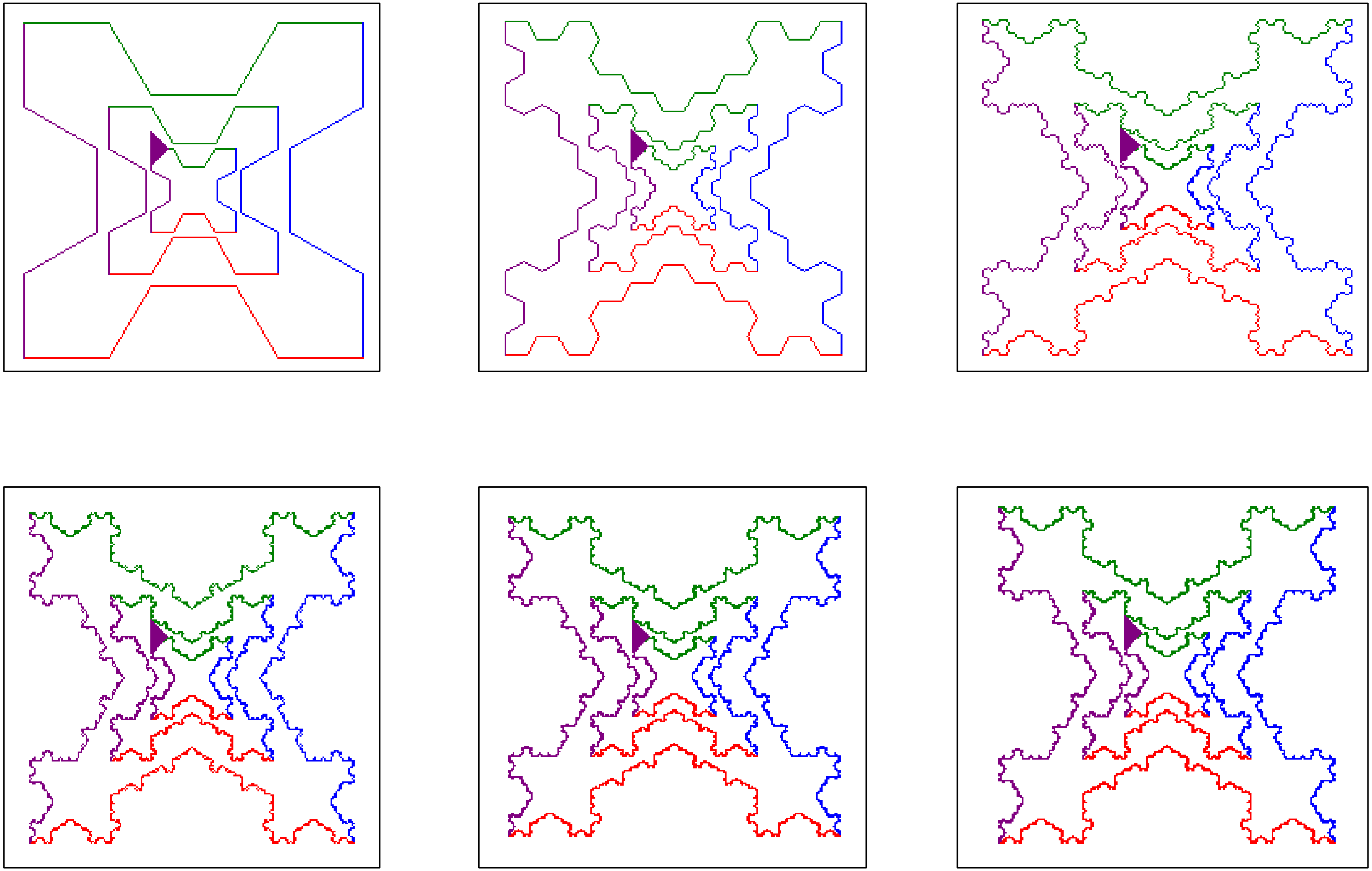
Python Animation (overlap) No.2



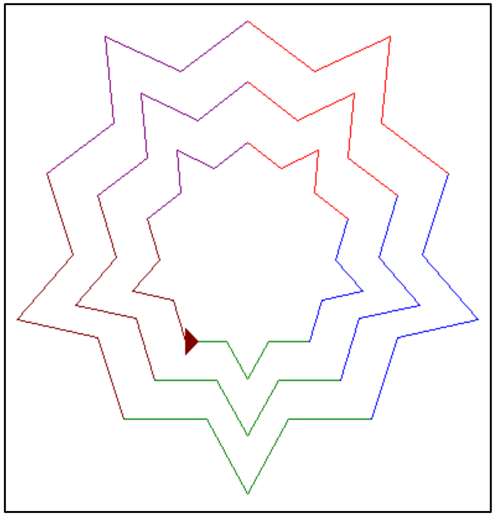
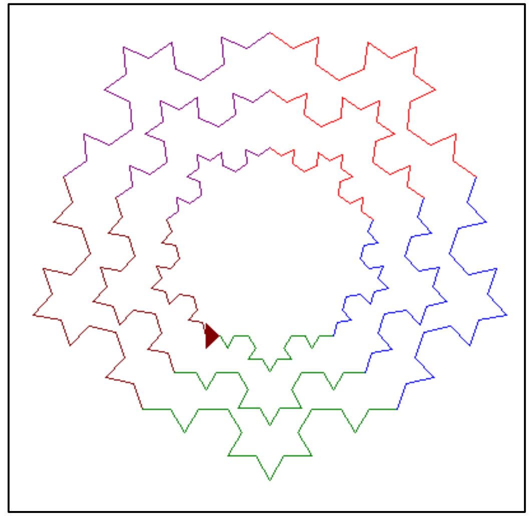
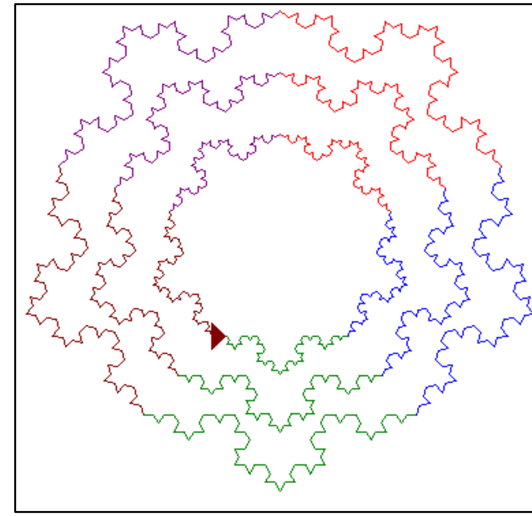
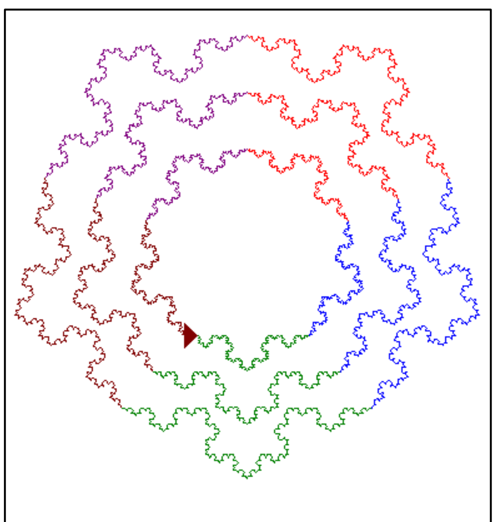
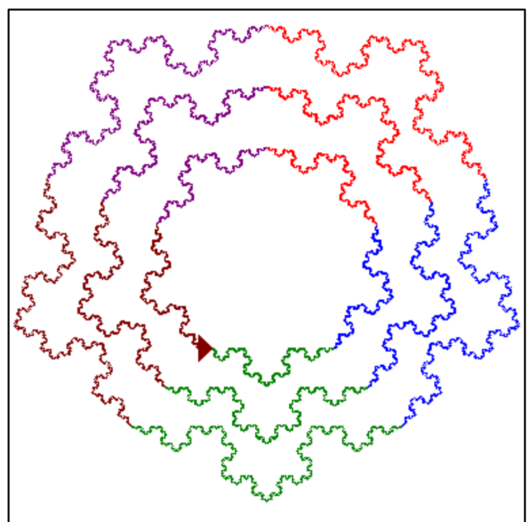
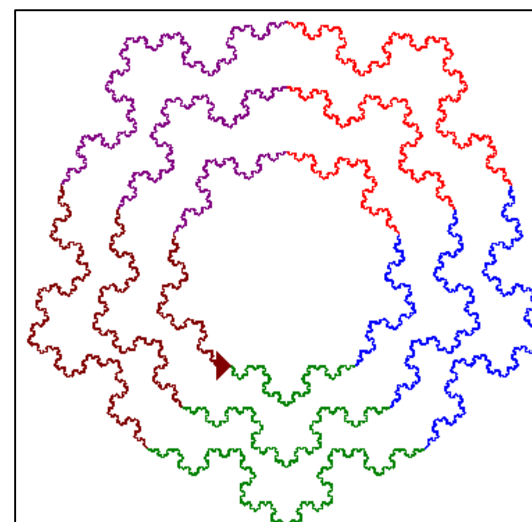
Python Animation (overlap) No.5



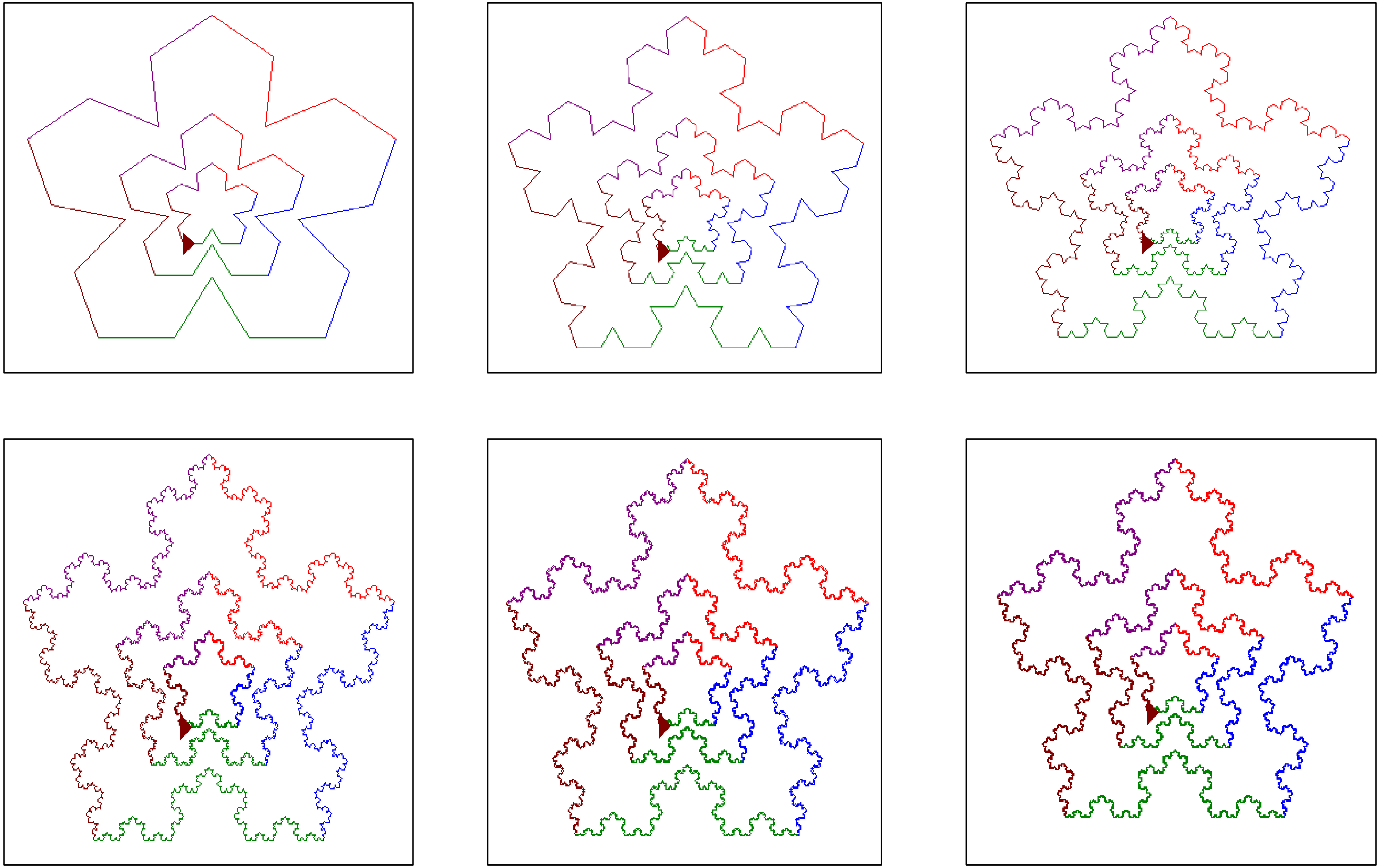
Python Animation (overlap) No.6



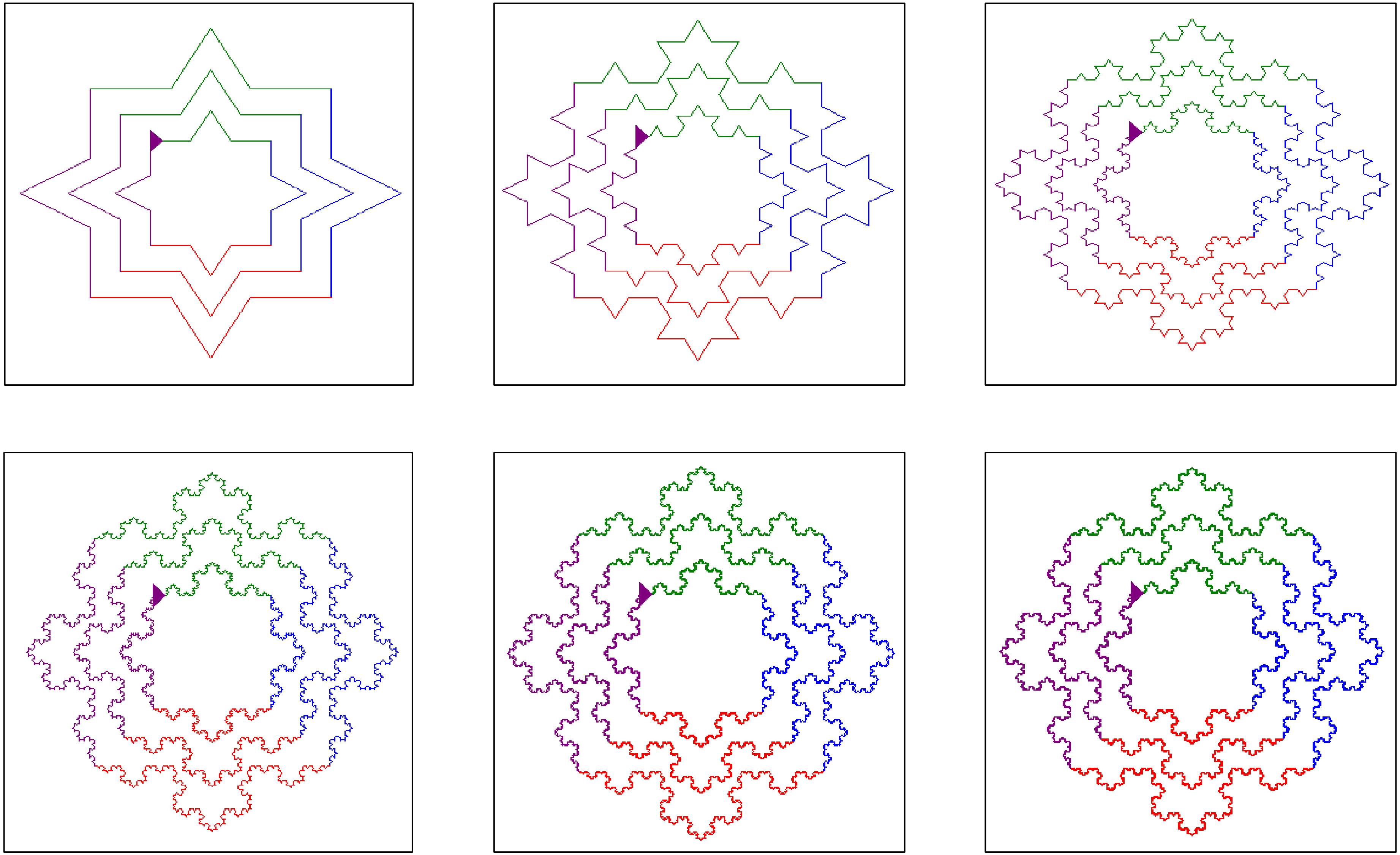
Python Animation (overlap) No.7

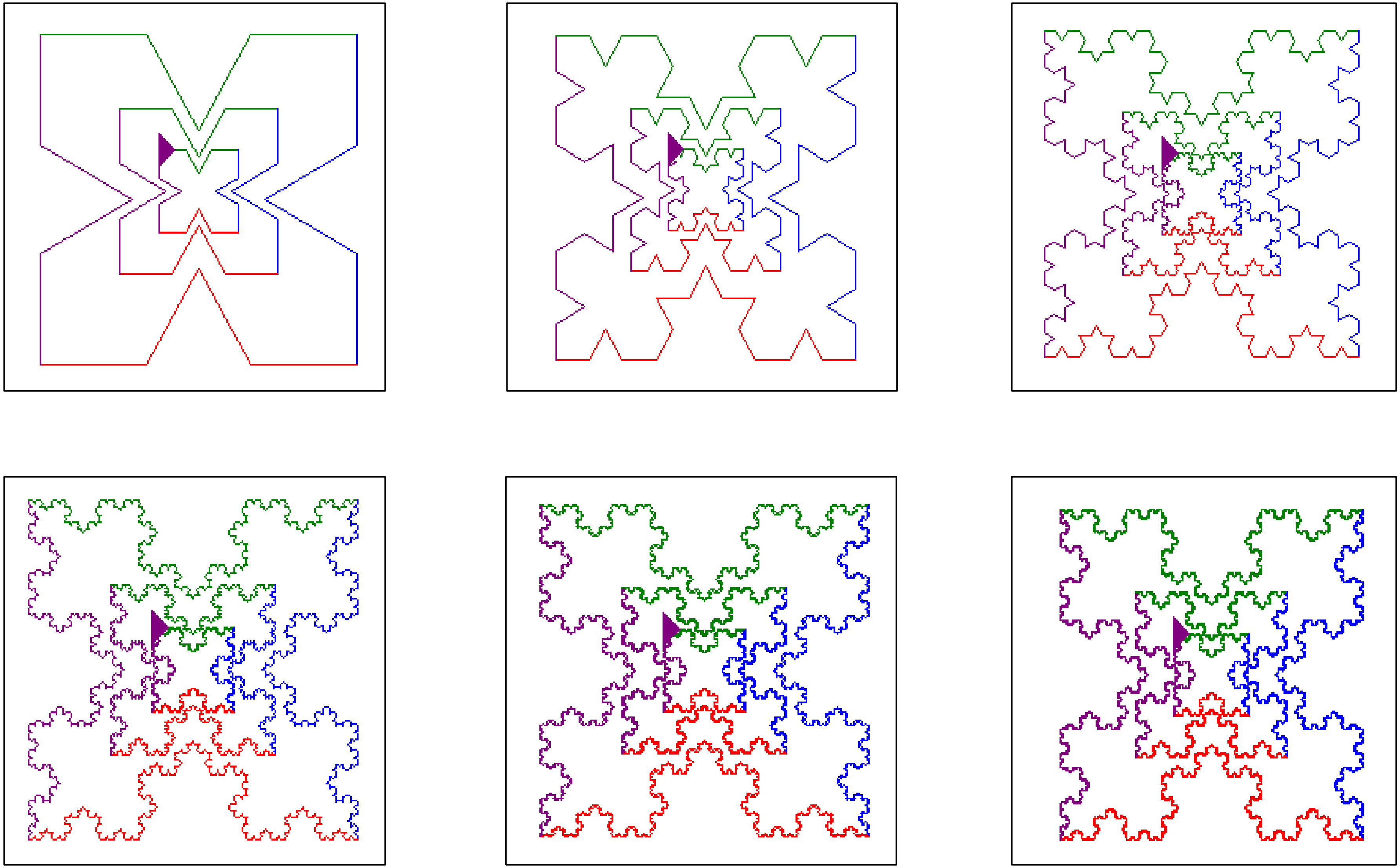
Python Animation (overlap) No.8



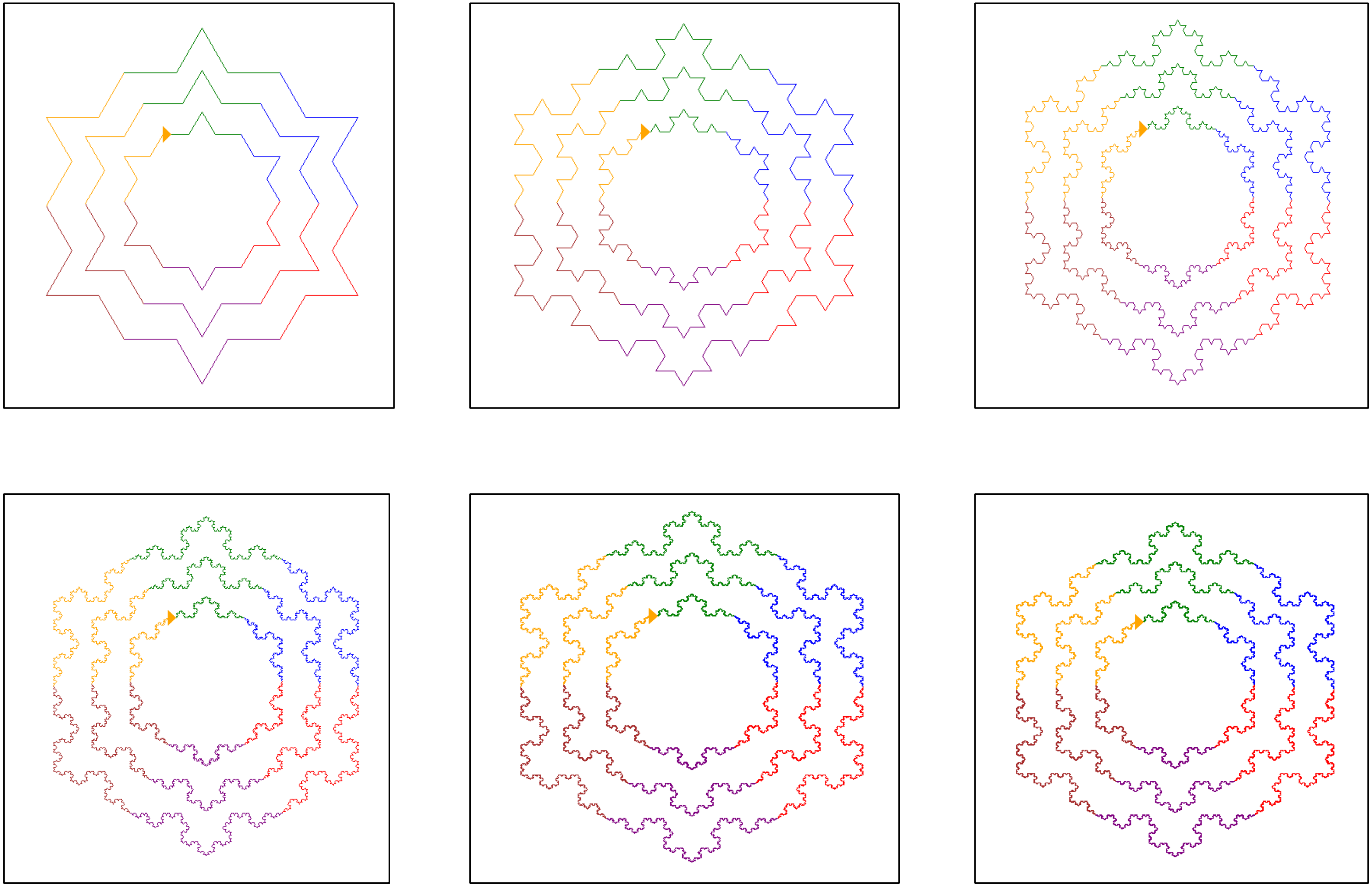
Python Animation (overlap) No.9



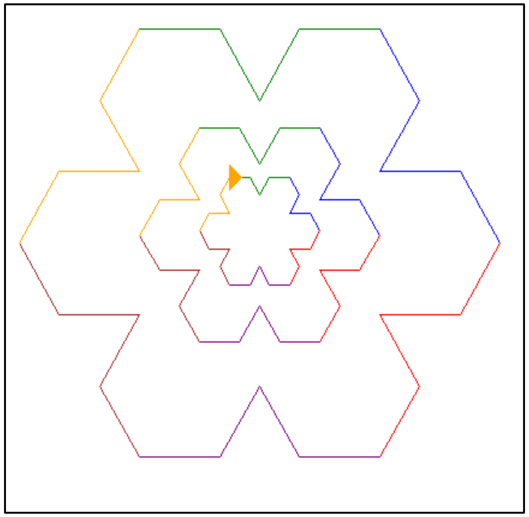
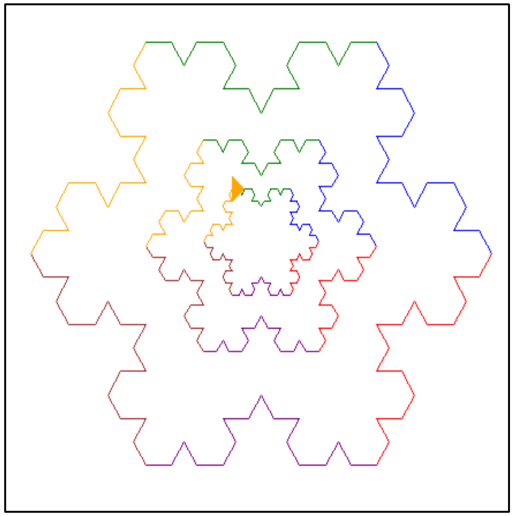
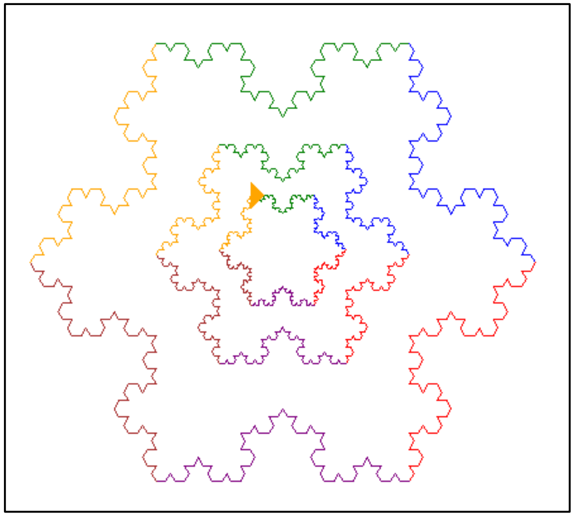
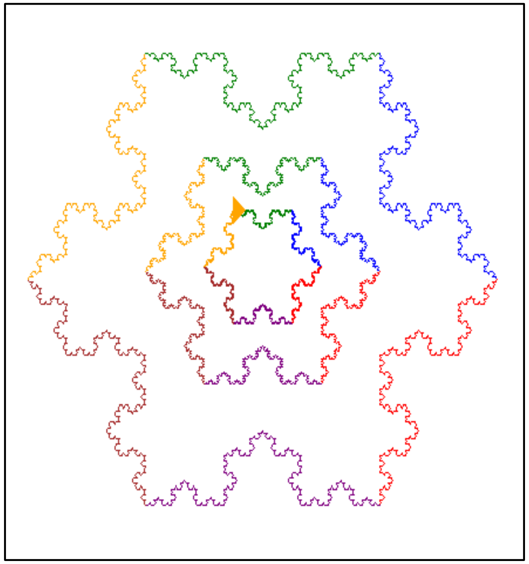
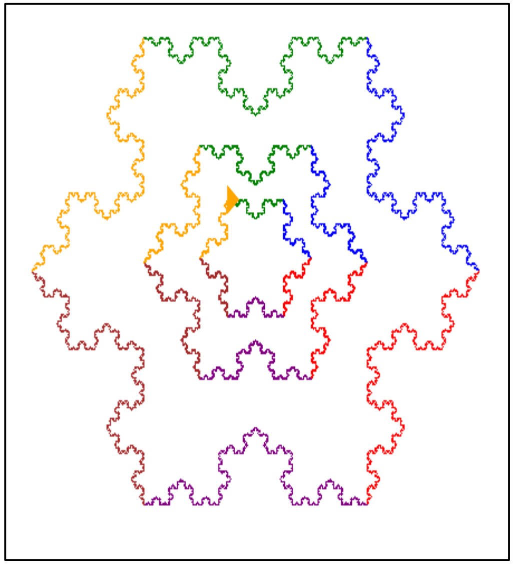
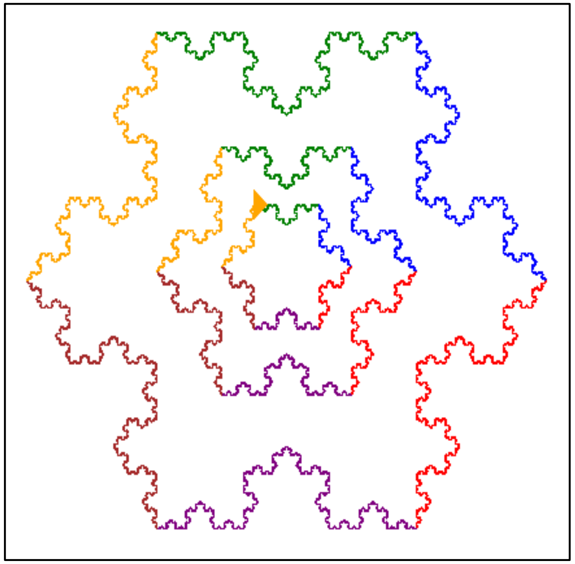
Python Animation (overlap) No.10



Python Animation (overlap) No.11



Python Animation (overlap) No.12

Outline

- Introduction
- Well-known fractal
    - -Sirpienski gasket
    - -Koch Curve
    - -Barnsley Fern
- Fractal Properties
- Program & Algorithm
- New Fractal
- Conclusion

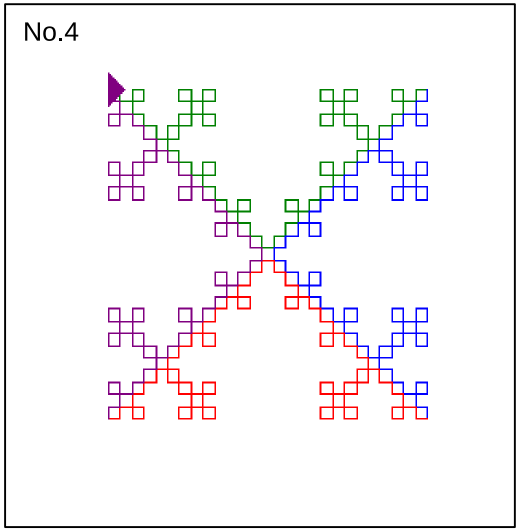
Conclusion

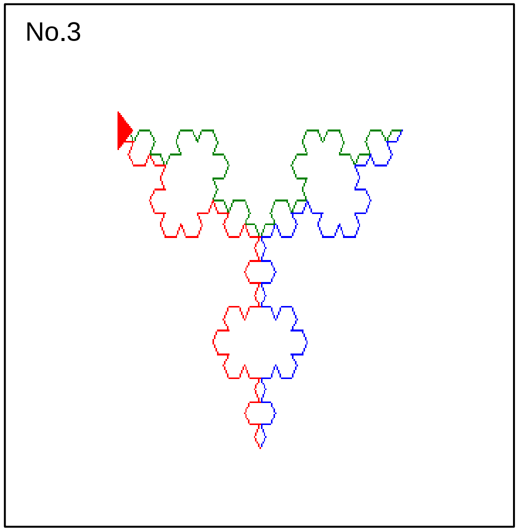
- Implementing the resistor value based on fractal concept during this development regarding this theme.
- The gist concept of fractal providing a progress in the electronic circuit output value when tempering with resistor value aside from the finite areas compiled within small chips.
- We tried a few more types of shapes of fractals troubleshooting the small area circumstances resulting in the addition of resistor value due to large perimeter gain.

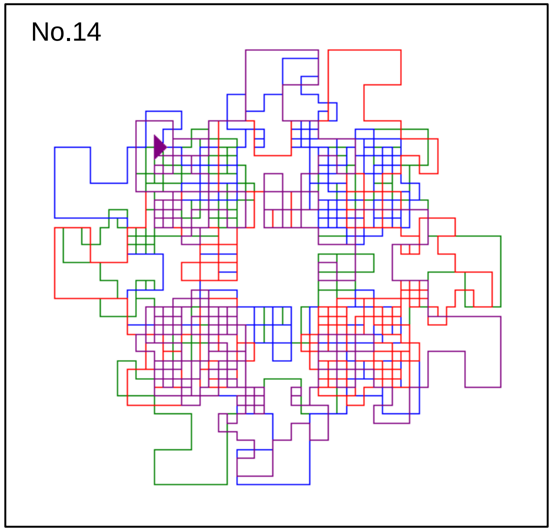
Future Works

Perimeter ÷ Total Area

Ranking of a long line with a limited area (at Iteration=6)





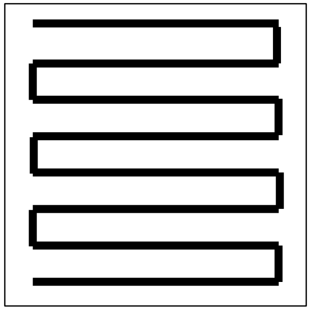


When actually designing a resistor…

- Lines should not touch each other
- Limitation of processing accuracy

Are there any fractal-specific effects?

➔ Comparison with non-fractal shapes



Thank You for Listening

質疑応答

Q. たとえば以下のような図形は
小さな四角に囲まれた部分の面積の合計ではなく
全体の専有面積（赤枠部）を考えるべきではありませんか？

A. 今回は小さな四角に囲まれた部分の面積の合計しか計算していないので
今後、全体の専有面積や、使われず無駄になっている空間の有効利用も考えたいです。