Pintando el caos con Python

Isabel Ruiz Buriticá

PyCon 2018 – Medellin, Colombia

¿Caos?

“ Las nubes no son esferas, las montañas no son conos, las costas no son círculos, las cortezas de los árboles no son lisas, ni los relámpagos viajan en una línea recta.
La naturaleza no solamente exhibe un grado mayor sino también un nivel diferente de complejidad.

Benoit Mandelbrot ”

¿Qué es un Fractal?

Forma geométrica irregular o fragmentada que se puede dividir en partes, cada una de las cuales es, aproximadamente una copia de tamaño reducido del conjunto entero



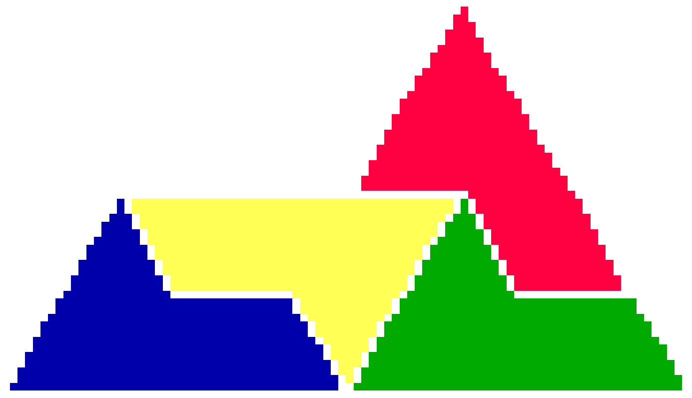
Características: Encontrando orden en el Caos

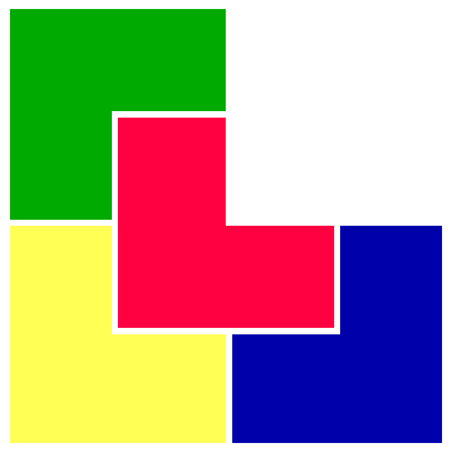


Auto-Similitud

- Su estructura está definida por escalas o subestructuras más pequeñas que parecen copias del todo

- Con el cambio de escala, posición y orientación las copias del conjunto son semejantes, pero no idénticas

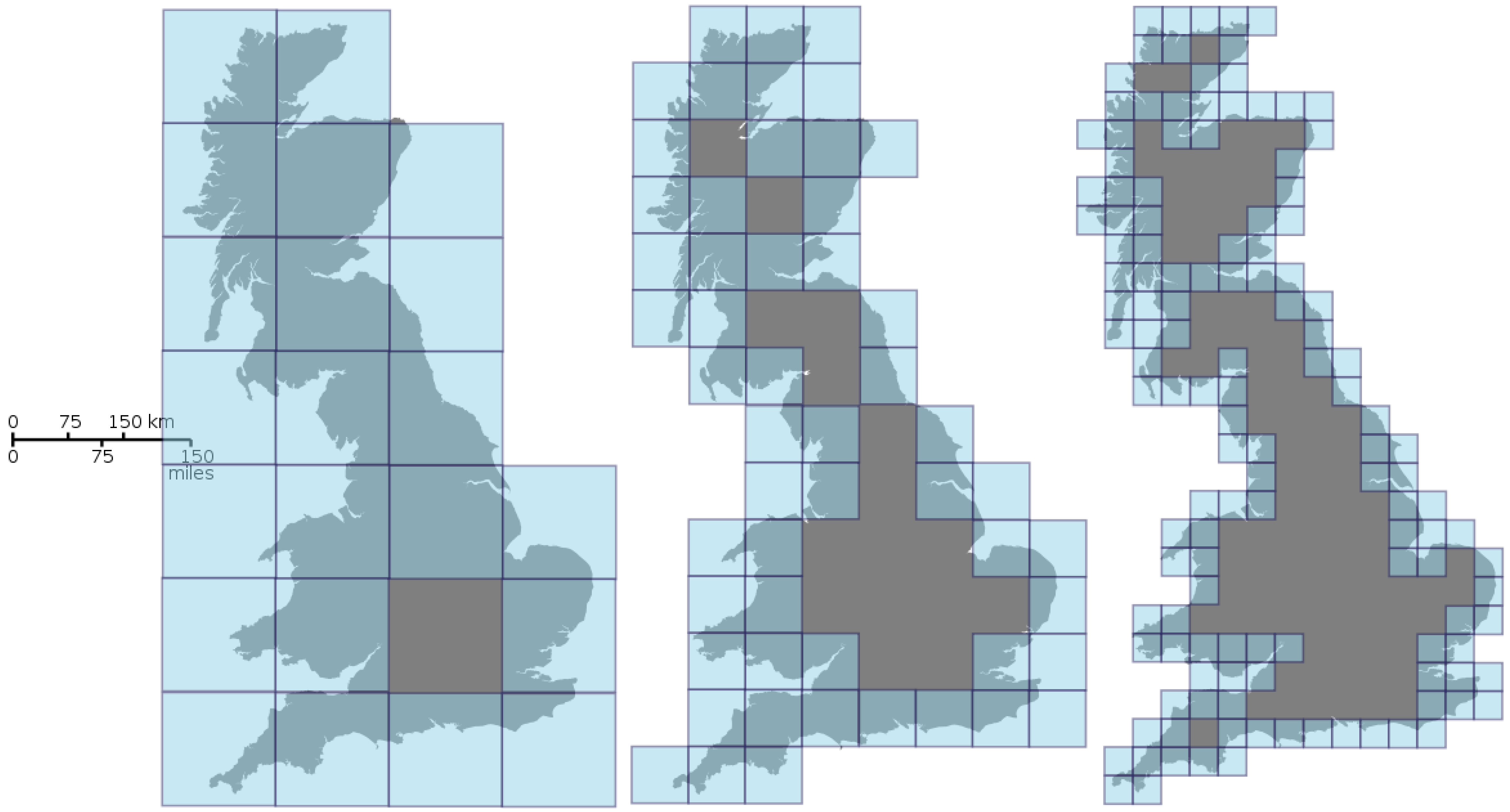




¡Muchas figuras que son auto-similares no son fractales!

Geometría Fractal

- Una forma es un fractal si no puede ser definida por la geometría euclidiana.
- Dimensión Fractal, parámetro que mide el grado de complejidad y rugosidad de un fractal.

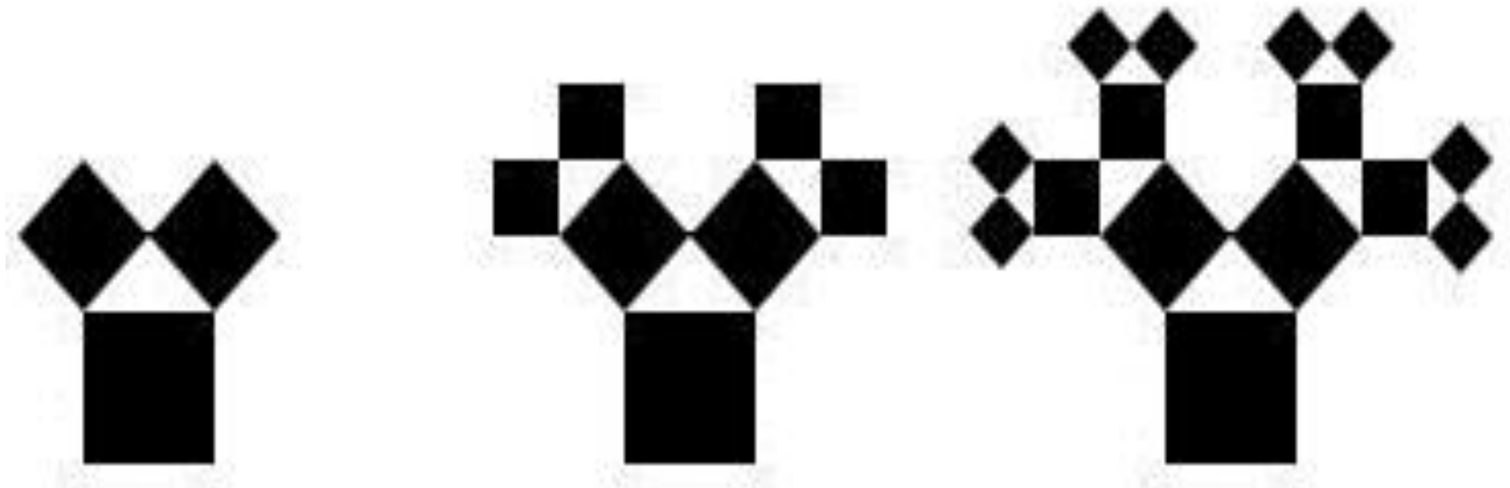


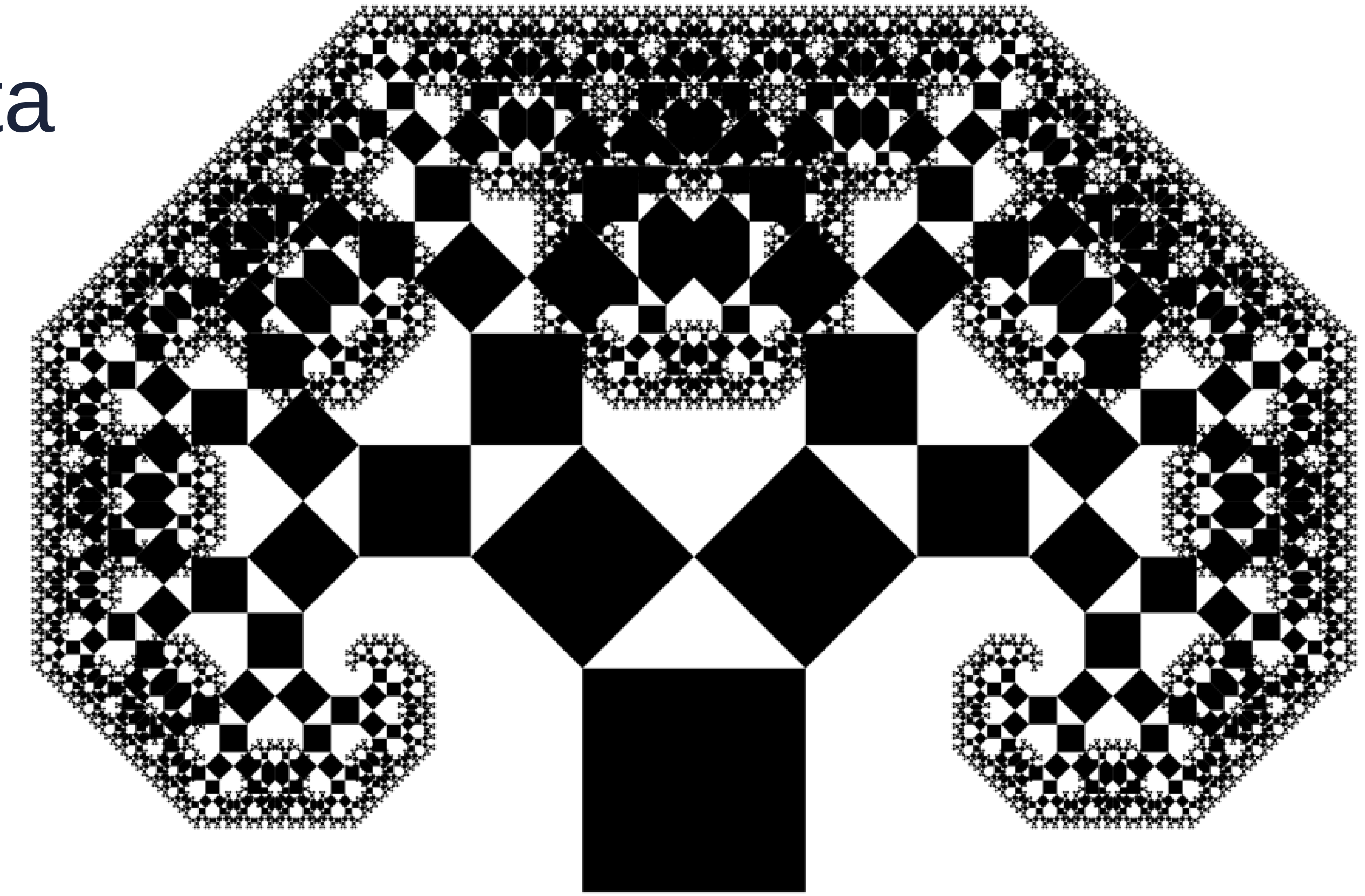
¿Cuánto mide la costa de Gran Bretaña?

Recursividad

- Proceso mediante el cual una función se llama a sí misma de forma repetida, hasta que se satisface alguna determinada condición.

- Miles y miles de iteraciones

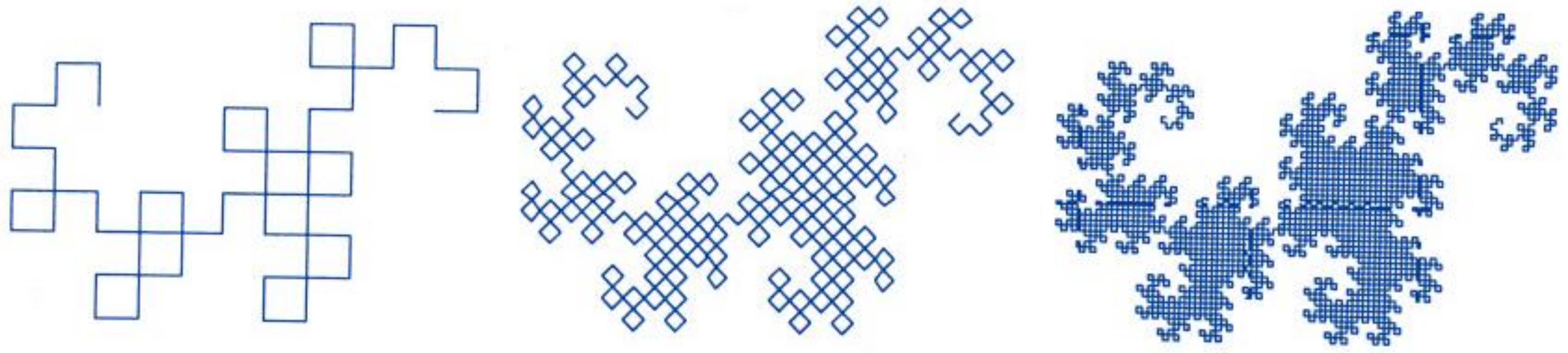




Monstrous Matemáticos

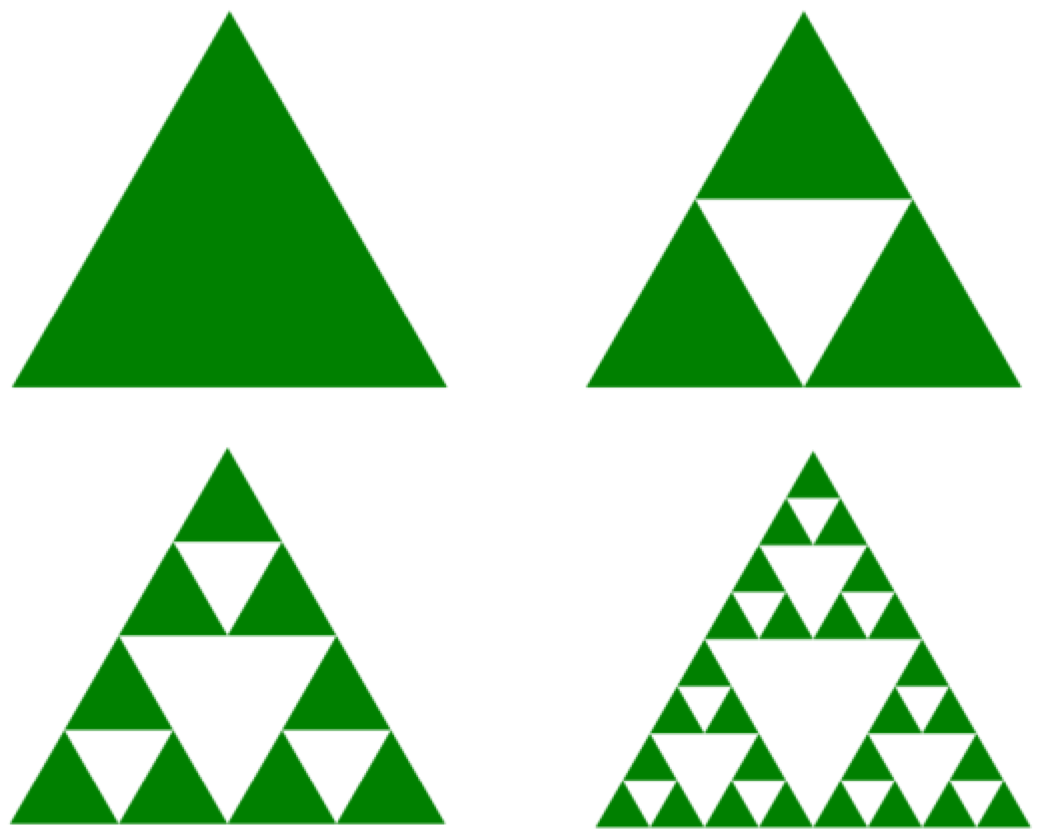
Conjunto de Cantor

Curva de Dragón

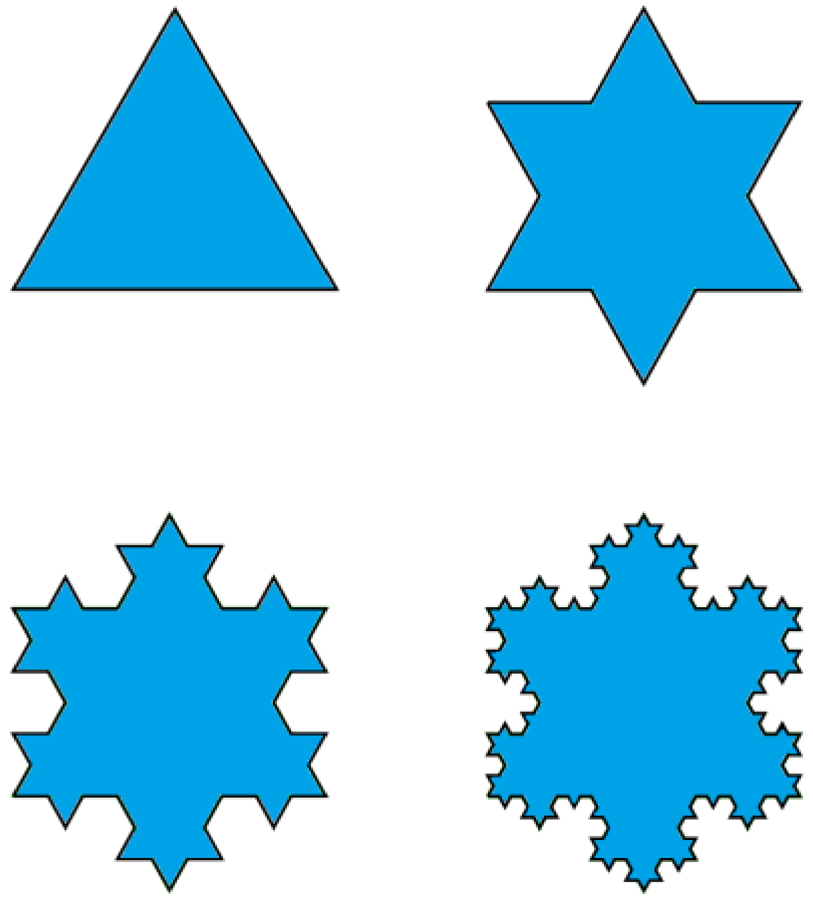


Alfombra de Sierpinski

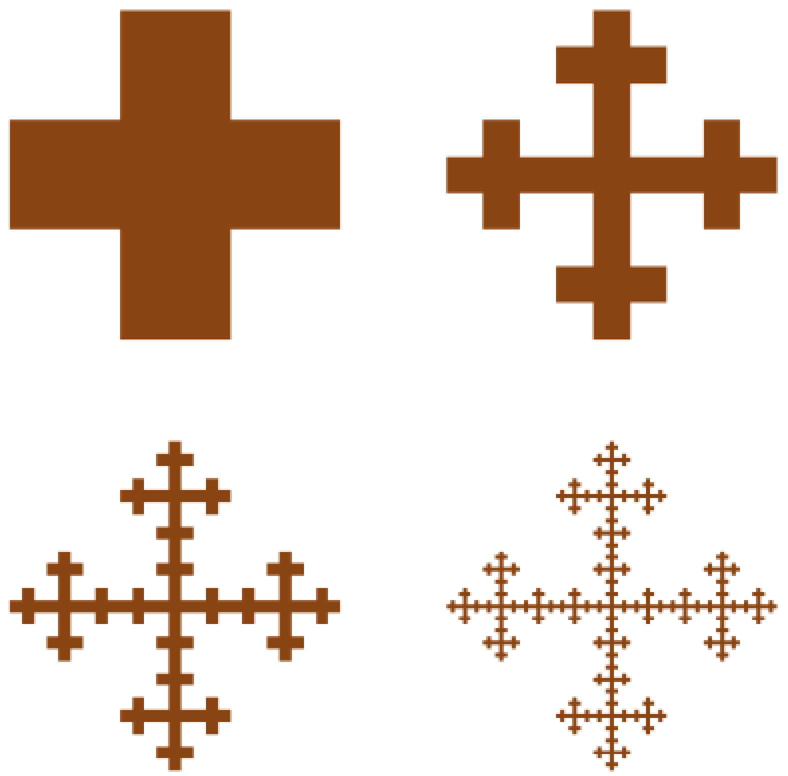
Triangulo de Sierpinski



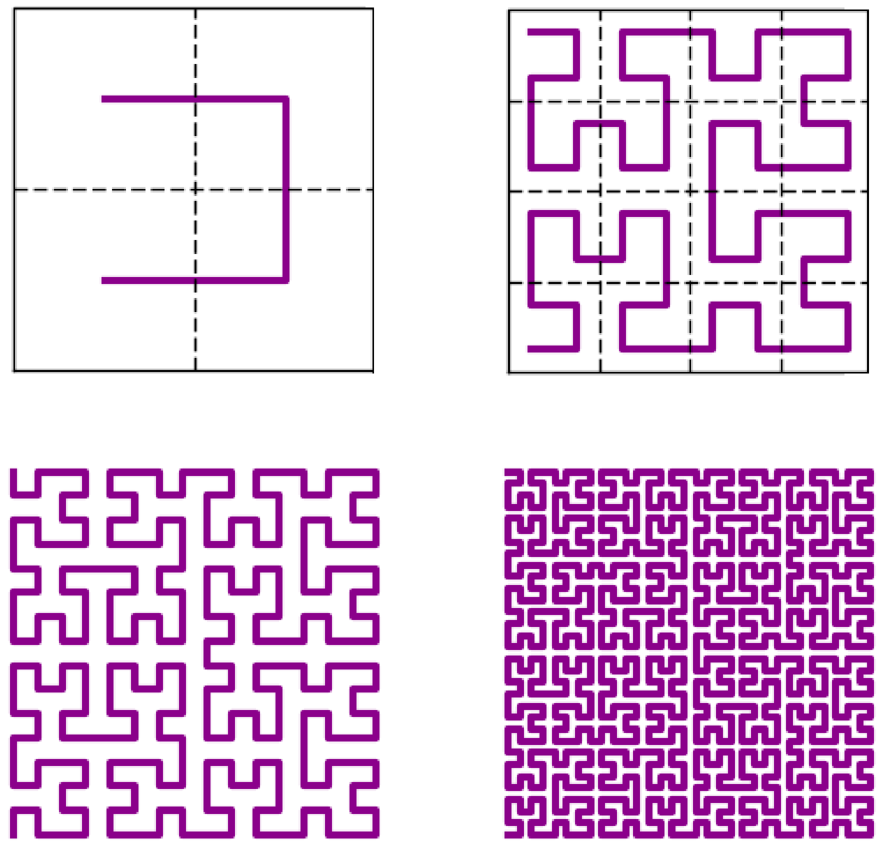
Curva de Koch



Fractal de Vicsek

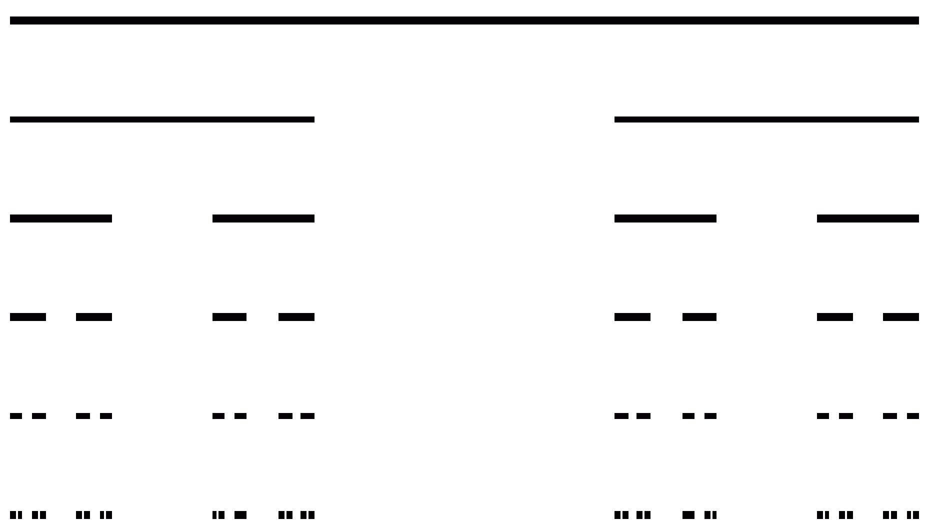


Curva de Hilbert



Dibujando el Conjunto de Cantor

1. Comienza con una línea
2. Divídela en tres partes y borra el segmento medio
3. Repite el paso dos con las líneas restantes, una y otra y otra vez



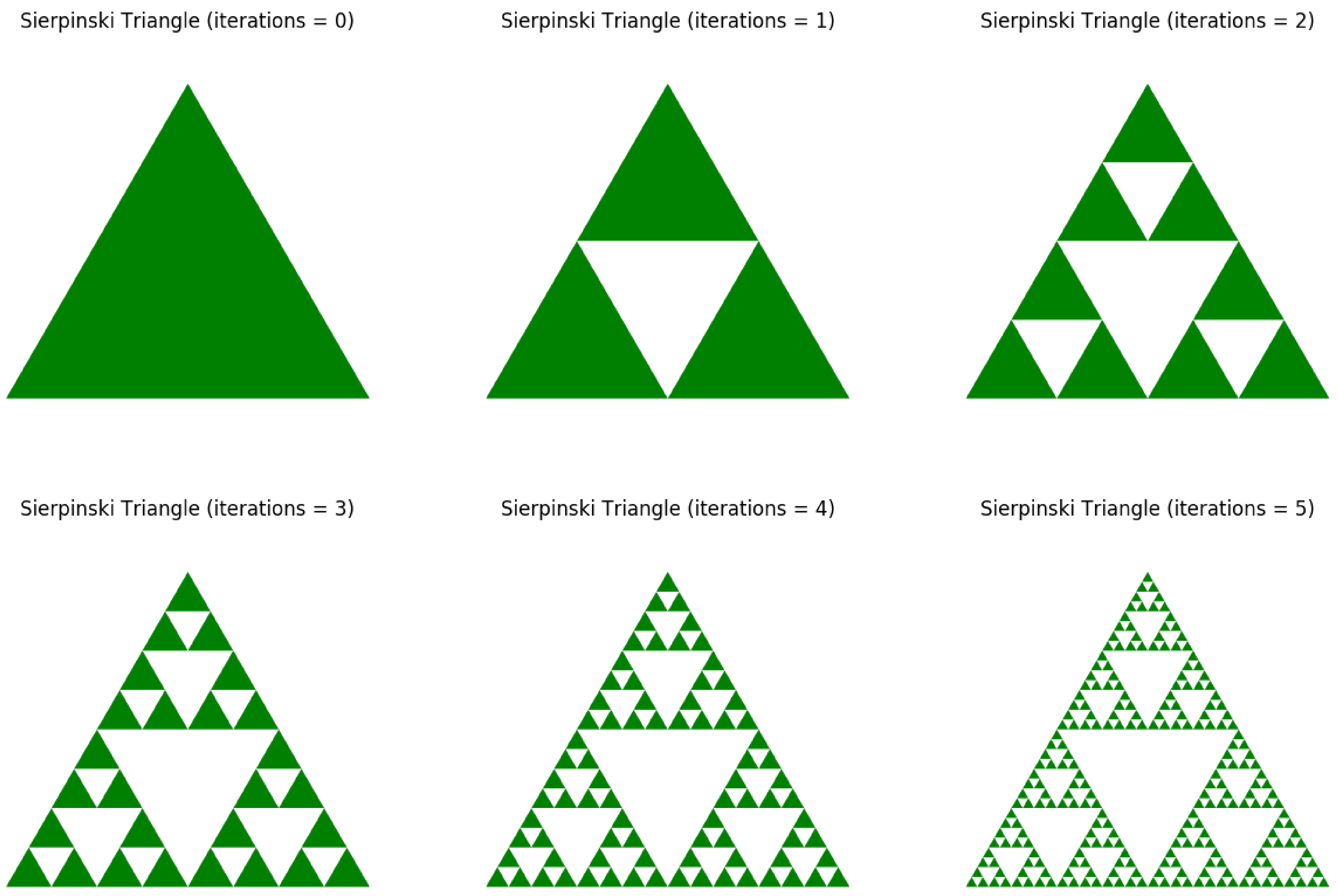
```
#!/usr/bin/env python3
from tkinter import Tk, Canvas

w, h = 1000, 450
win = Canvas(Tk(),width=w,height=h)

def cantor_set(x,y,l):
        # doing while the line is > 1 pixel
  1     if l > 1:
            # draw horizontal line
      2     win.create_line(x,y,x+l,y)
            # move 50 pixels down for next generation
            y = y + 50
            # left hand offspring
      3     cantor_set(x,y,l/3)
            # right hand offspring
      4     cantor_set(x+2/3*l,y,l/3)

win.pack()
cantor_set(10,10,w-20)
win.mainloop()
```

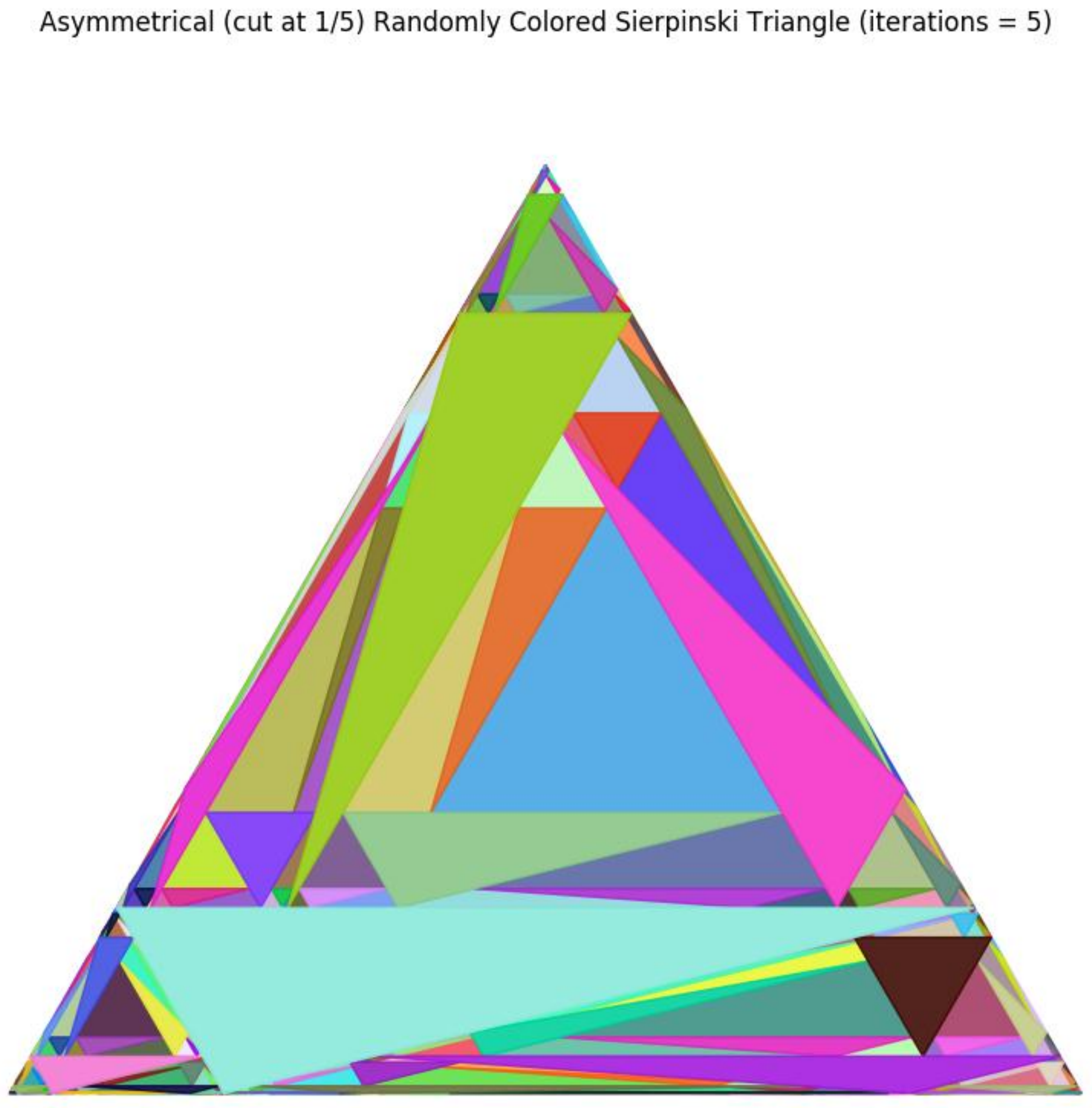
Dibujando el Triangulo de Sierpinski

1. Comience con un triángulo equilátero
2. Subdivídelo en cuatro triángulos equiláteros congruentes más pequeños y elimina el triángulo central.
3. Repite el paso 2 con cada uno de los triángulos más pequeños restantes, una y otra y otra vez

```python
from tkinter import *
import math

def sierpinski(canvas, x, y, size, level):
    x = float(x)
    y = float(y)
    # base case: a Sierpinsky triangle at level 0
    if (level == 0):
        canvas.create_polygon(x, y, x+size, y,
                              x+size/2, y-size*math.sqrt(3)/2,
                              fill="green")
    # recursive case: shrink the previous level of Sierpinski
    # triangle, and repeat it three times
    else:
        sierpinski(canvas, x, y, size/2, level-1)
        sierpinski(canvas, x+size/2, y, size/2, level-1)
        sierpinski(canvas, x+size/4, y-size*math.sqrt(3)/4,
                   size/2, level-1)

root = Tk()
myCanvas = Canvas(root, width=600, height=600)
myCanvas.pack()
sierpinski(myCanvas, 50, 500, 500, 3)
root.mainloop()
```

Dibujando el Triangulo de Sierpinski



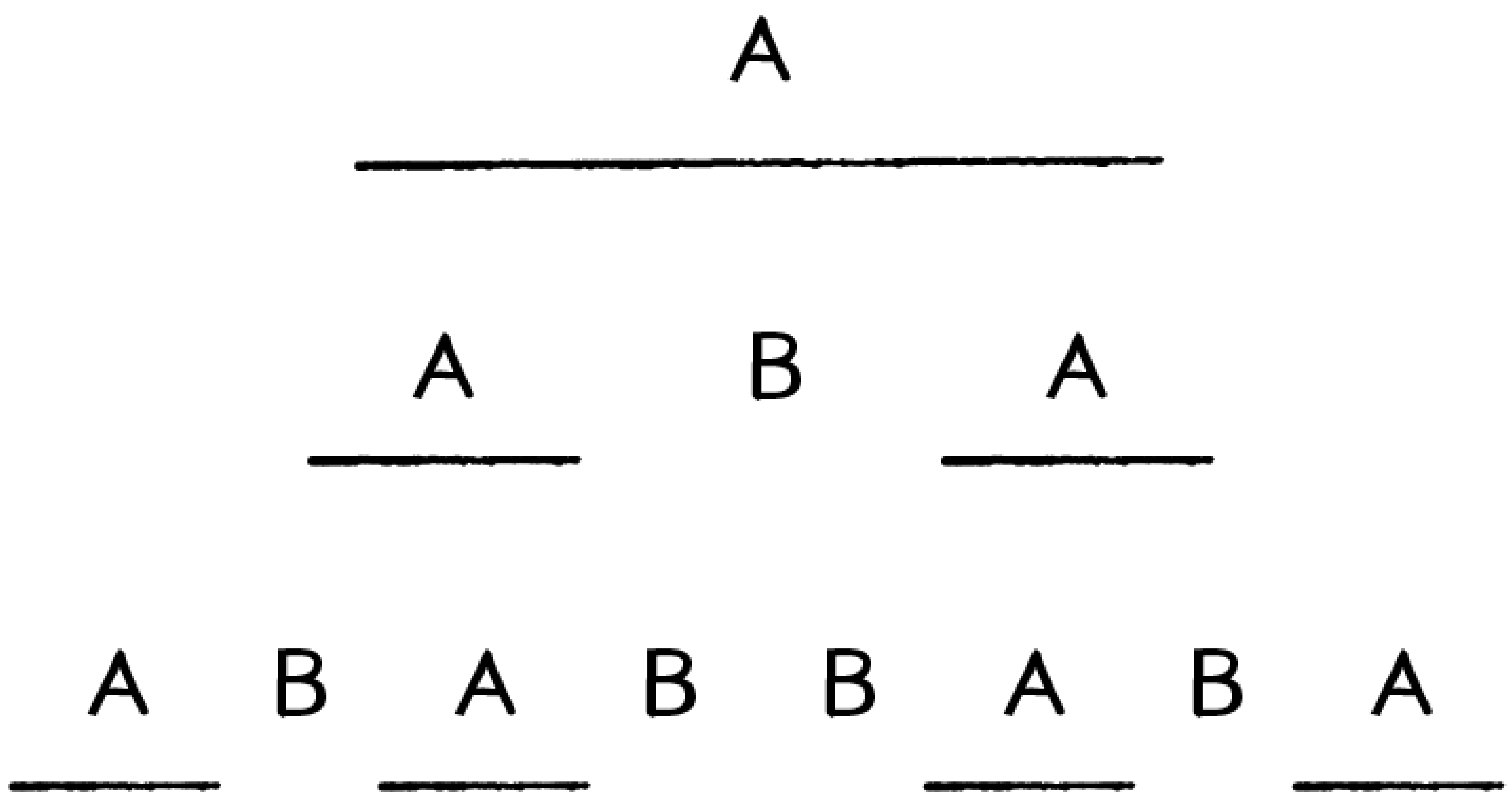


L-System

- 1968, el botánico húngaro Aristid Lindenmayer
  Sistema basado en la gramática para modelar los patrones de crecimiento de las plantas
- Esta formado por **Alfabeto, Axioma y reglas de generación.**
- Alfabeto más usado: “FG+-[]”:

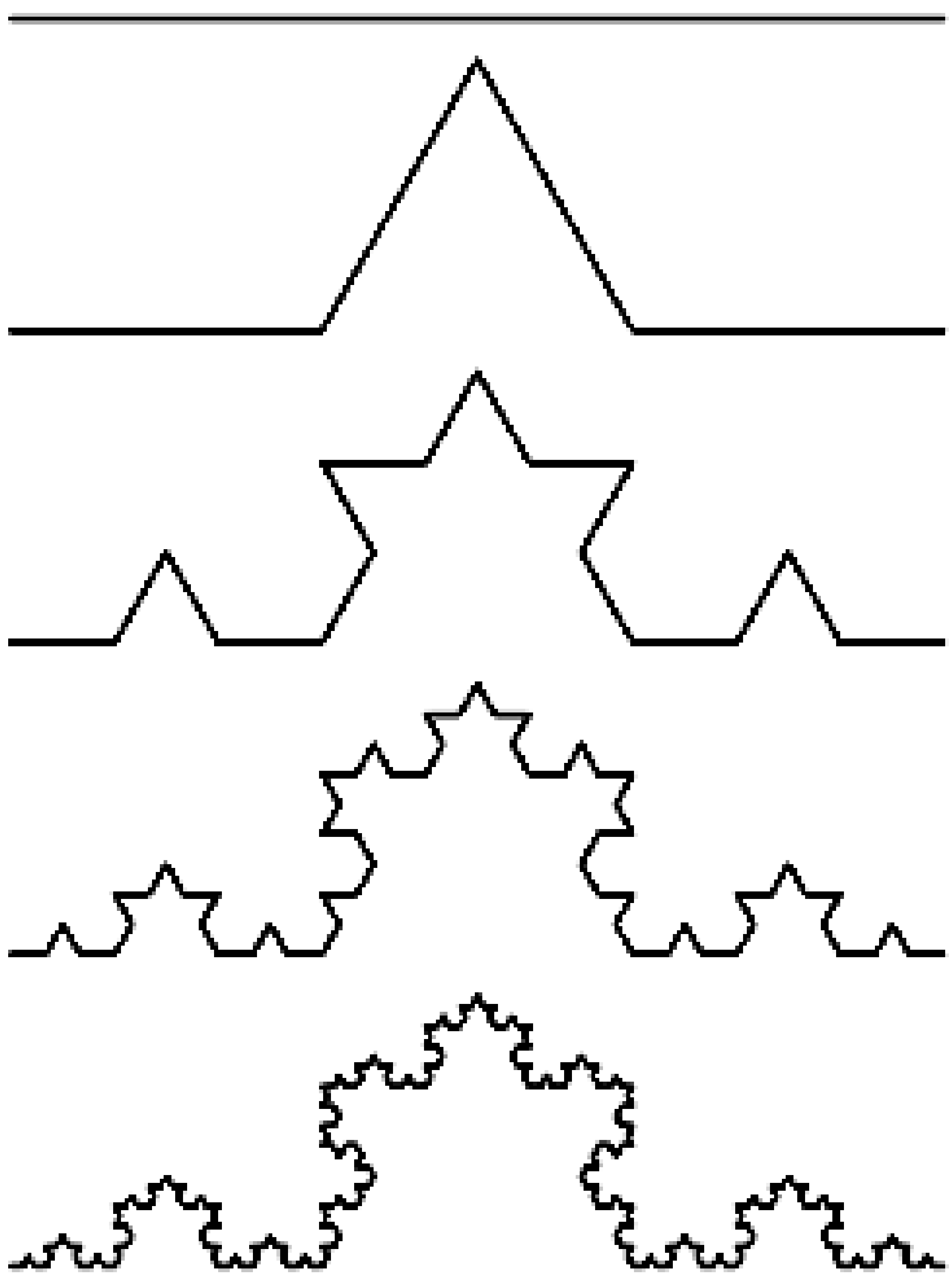
  F: Dibujar una línea y avanzar
  G: Desplazarse hacia adelante (sin dibujar)
  +: Doble a la derecha (R)
  -: Girar a la izquierda (L)
  [: Guardar ubicación actual
  ]: Restaurar ubicación anterior

Alfabeto: AB
axioma: A
Reglas: (A → AB) (B → A)



Dibujando la curva del Koch

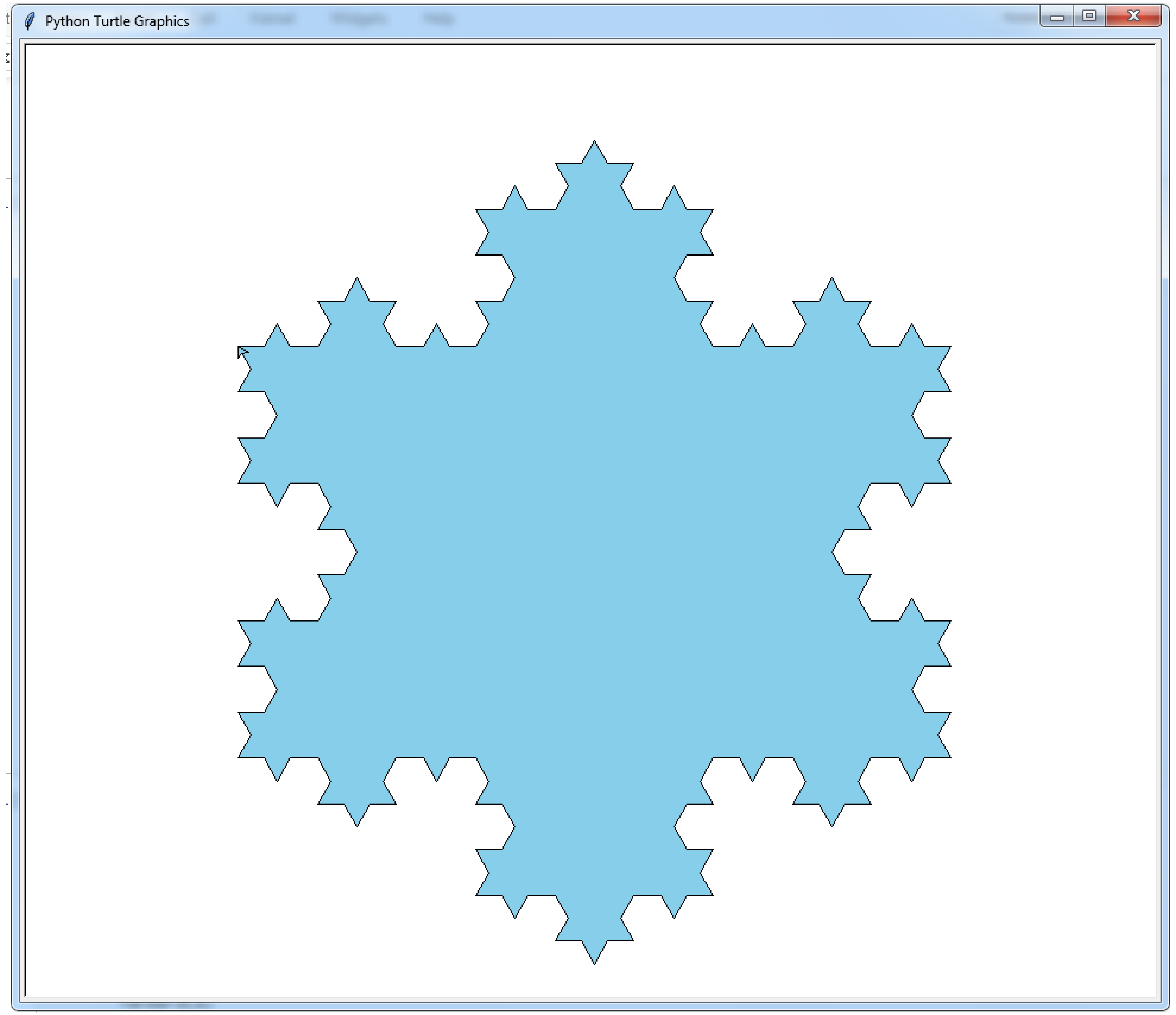
1. Dibuja una línea
2. Divídela en tres partes iguales
3. Borra la línea central y reemplaza por dos partes de igual longitud haciendo un ángulo de 60 grados
4. Repite para los cuatro segmentos restantes
5. Igual para cada lado del triangulo



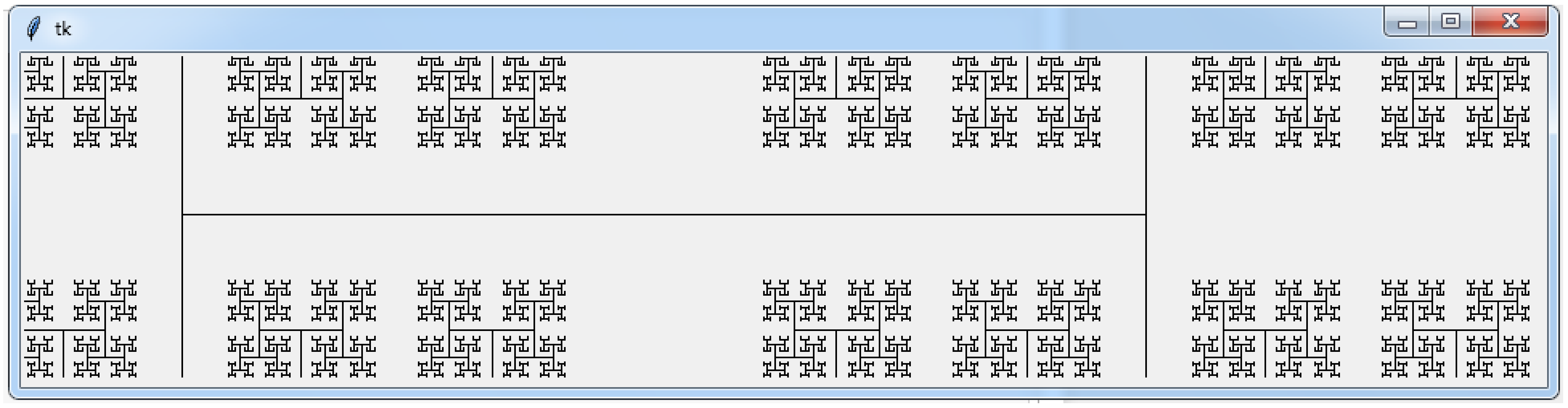
Alfabeto : F+-
Axioma : F
Reglas: $F \rightarrow F-F+F-F$

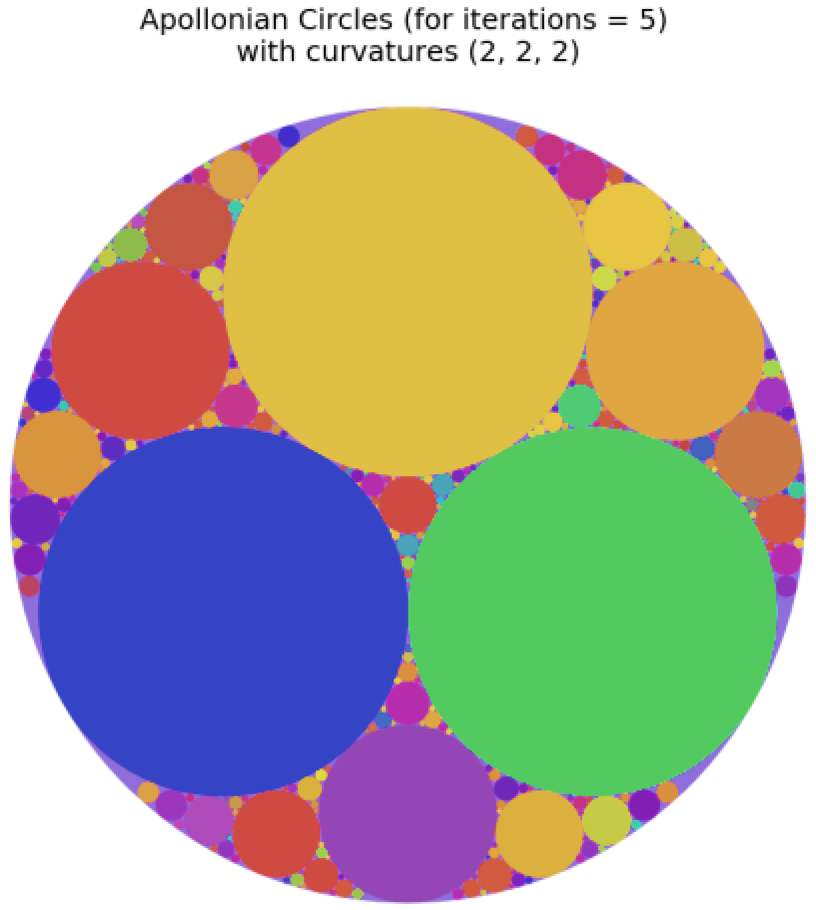
Dibujando la curva del Koch

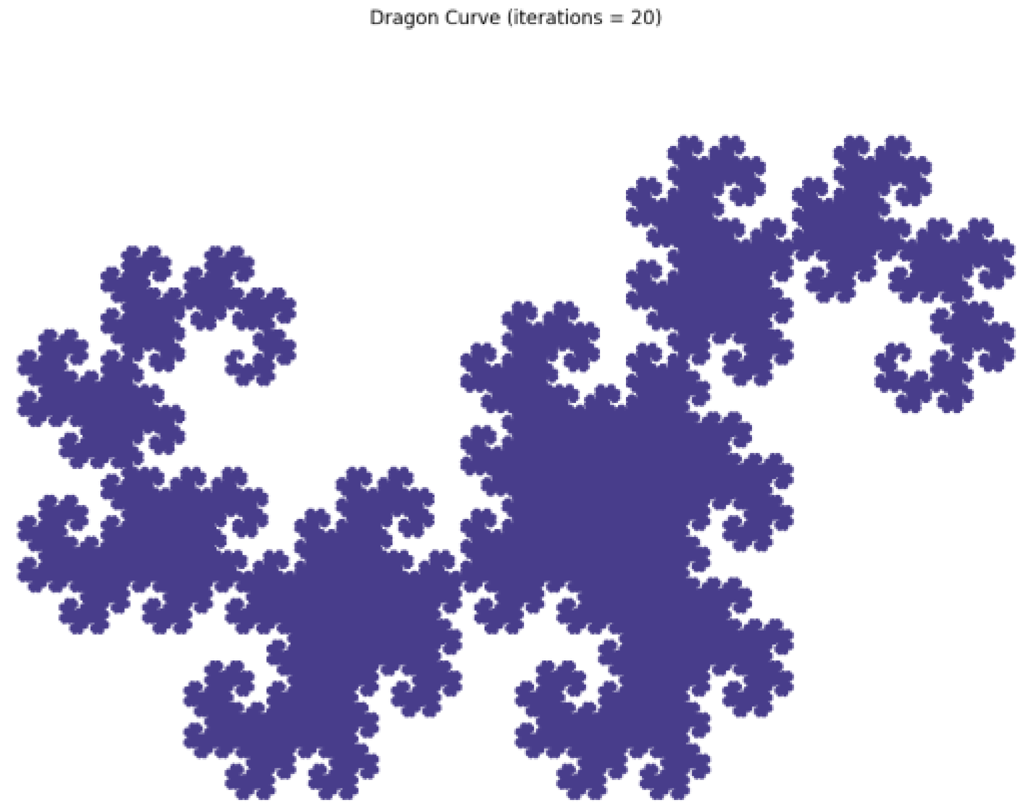
```
#!/usr/bin/env python3
import turtle

koch = 'FRFRF' # axiom for Koch snowflake
iterations = 3 #the number of generations
startLength = 200 #Length of the generation 0 line

#pick the pen up and move cursor to a good starting point
turtle.penup()
turtle.setpos(-startLength*3/2,startLength*3/2/2)
turtle.speed(0)

#make the L-System we want to process
for i in range(iterations):
    koch = koch.replace("F","FLFRFLF")

turtle.pendown()
#draw line in black, fill sky blue
turtle.color('black','skyblue')
turtle.begin_fill()

for move in koch:
    if move == "F":
        turtle.forward(startLength / (3 ** (iterations - 1)))
    elif move == "L":
        turtle.left(60)
    elif move == "R":
        turtle.right(120)

turtle.end_fill() #fill any enclosed areas
turtle.mainloop()
```

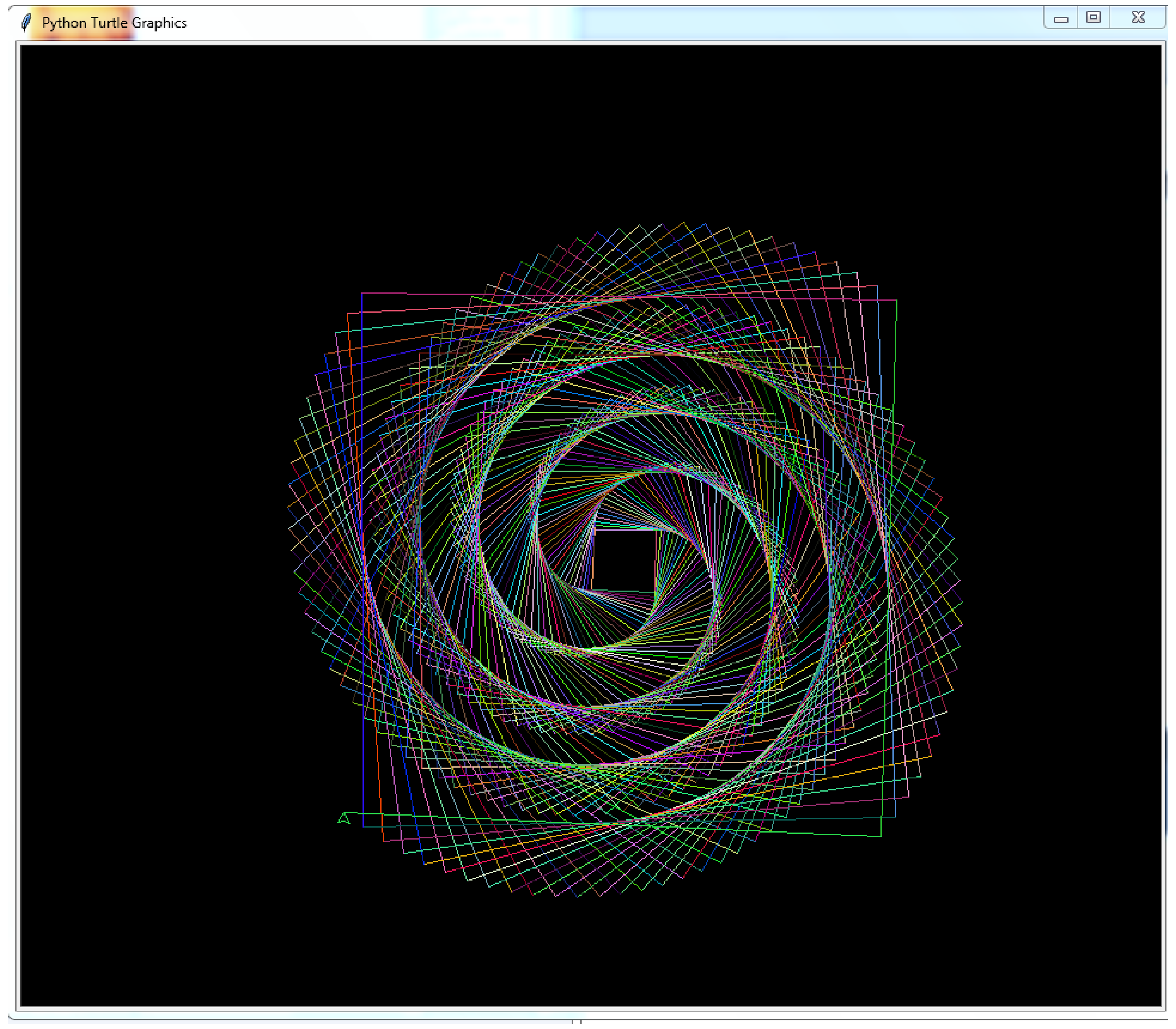


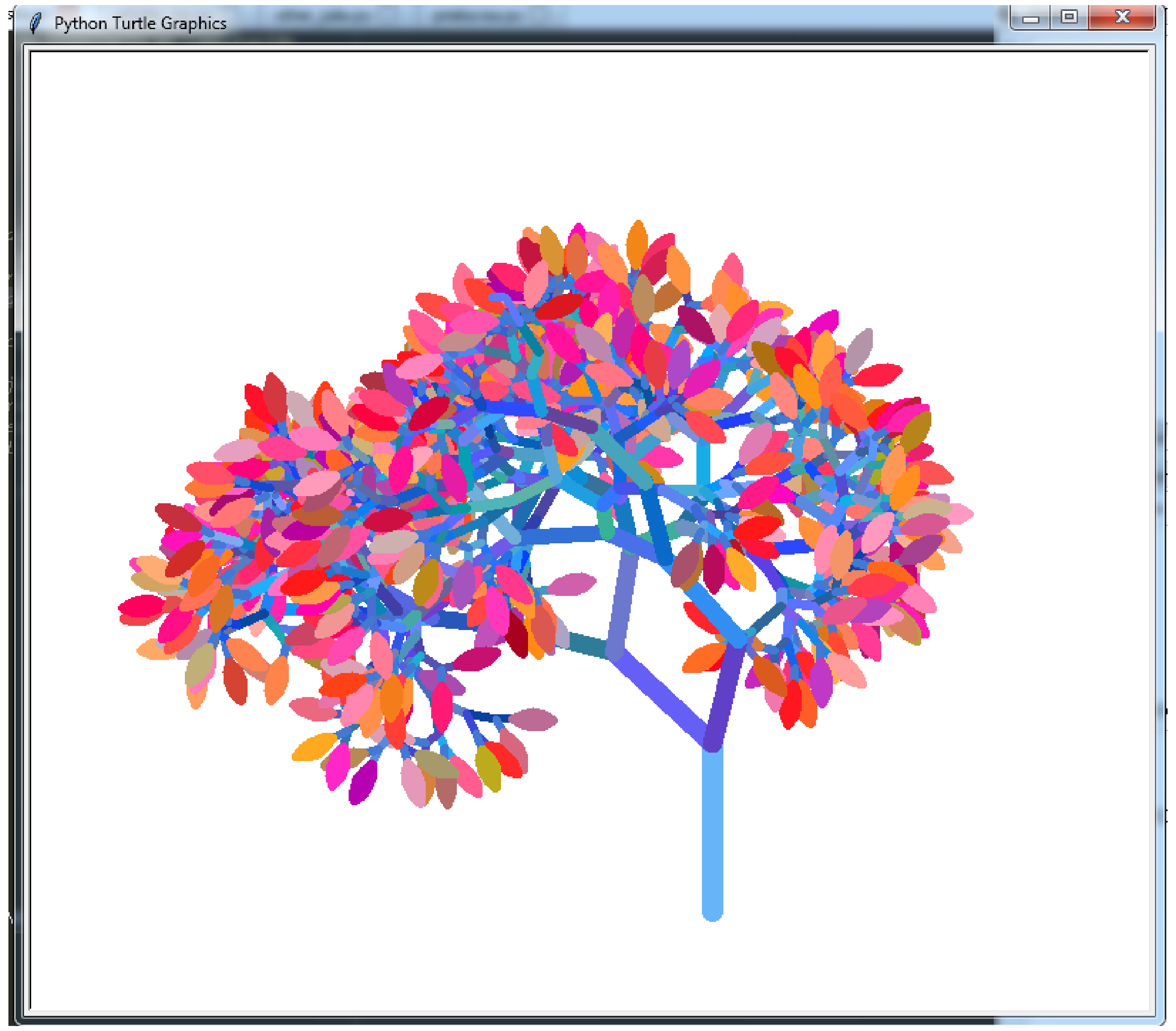
Más recursividad

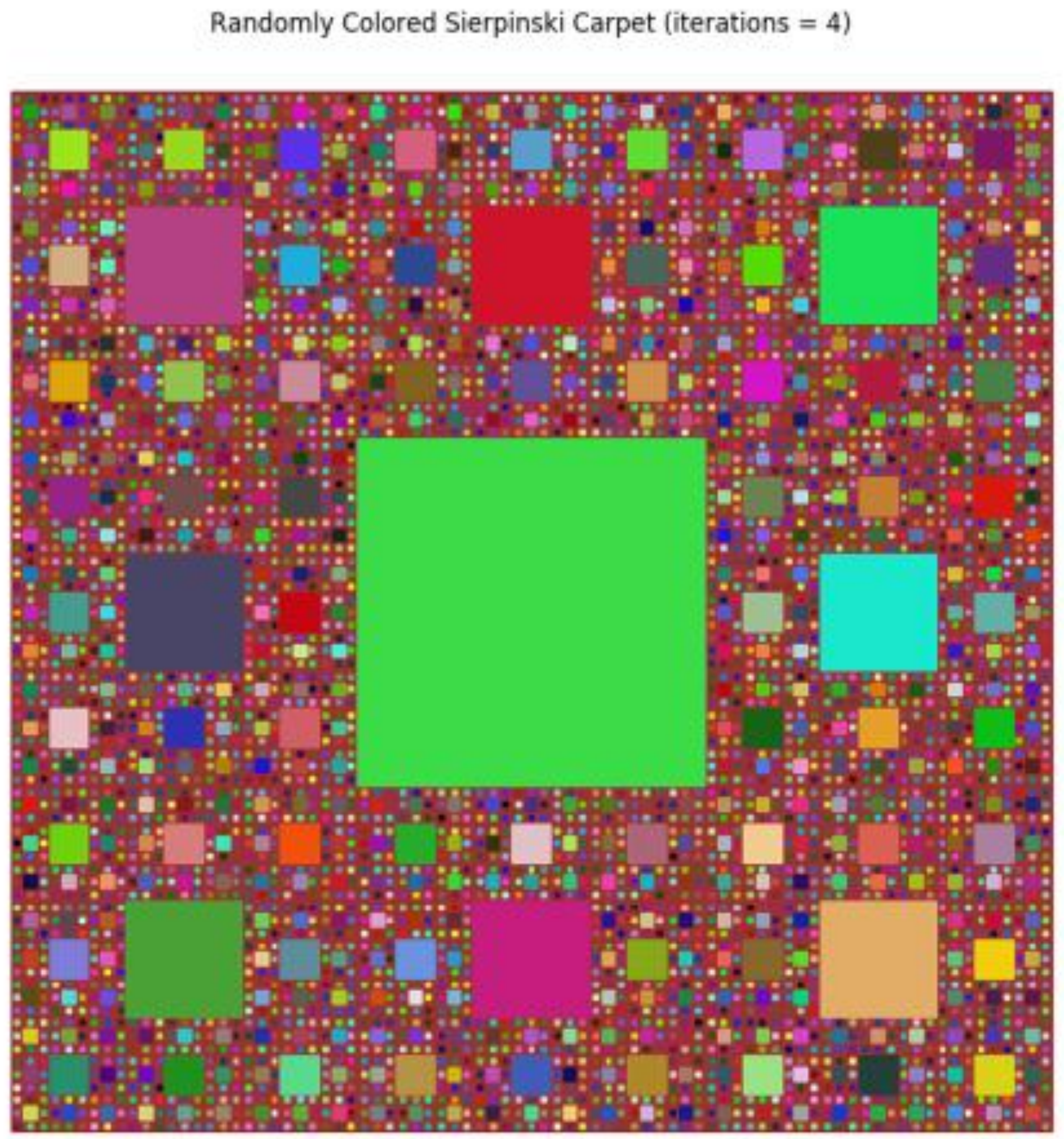






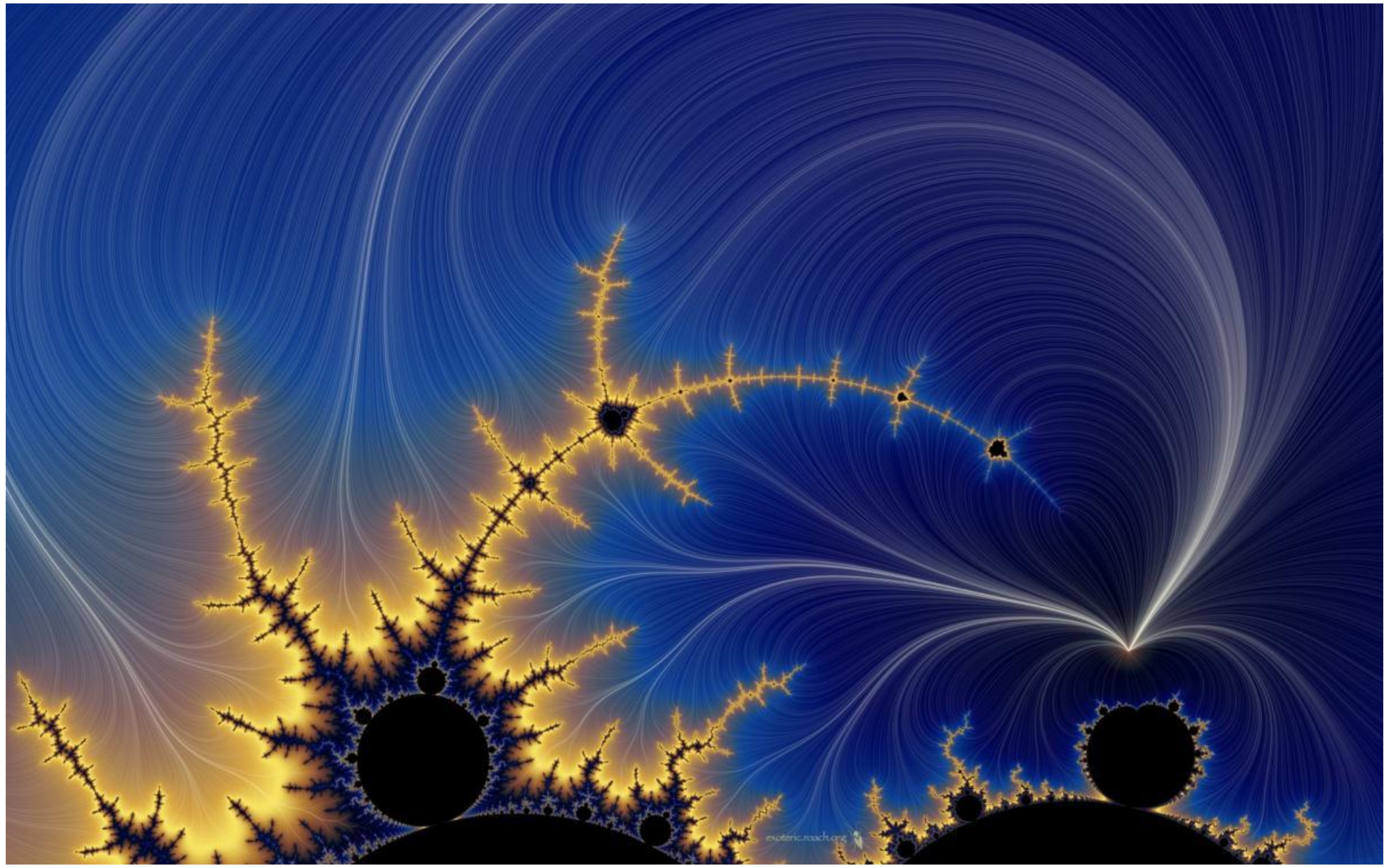






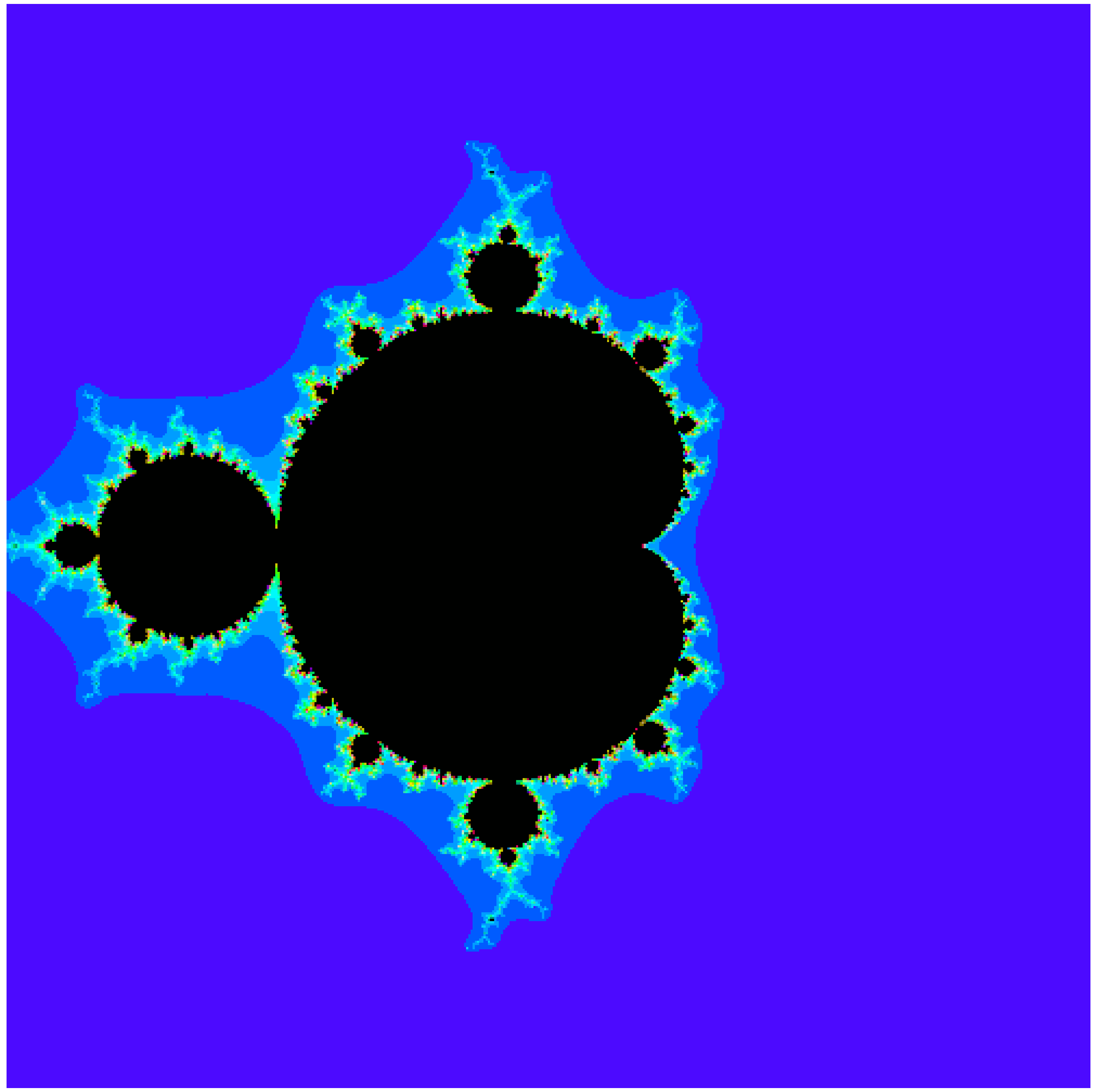
Fractales estocásticos

- Caos, comportamiento impredecible de distintos sistemas que varían de acuerdo a las condiciones iniciales
- Julia propuso iterar una ecuación dentro de un bloque de repetición, y generar un conjunto de números
- Mandelbrot transformó los números de Julia en coordenadas en un plano complejo



Conjunto de Mandelbrot

- Es el más conocido de los conjuntos fractales, y el más estudiado
- A menudo se representa el conjunto mediante el algoritmo de tiempo de escape
- Cualquier punto que va a cero se pinta de negro y cualquiera que vaya al infinito se pinta de una serie de colores



Conjunto de Julia

$$Z_{n+1} = Z_n^2 + c$$

```python
import matplotlib.pyplot as plt
import numpy as np
import numba

# Building the Julia set
def py_julia_fractal(z_re, z_im, j, c):
    for m in range(len(z_re)):
        for n in range(len(z_im)):
            # assigning the initial value to z
            z = z_re[m] + 1j * z_im[n]
            for t in range(256):
                # defining the mathematical function
                z = z ** 2 + c
                # the number comes out of the set
                if np.abs(z) > 2.0:
                    j[m, n] = t
                    break
```

1

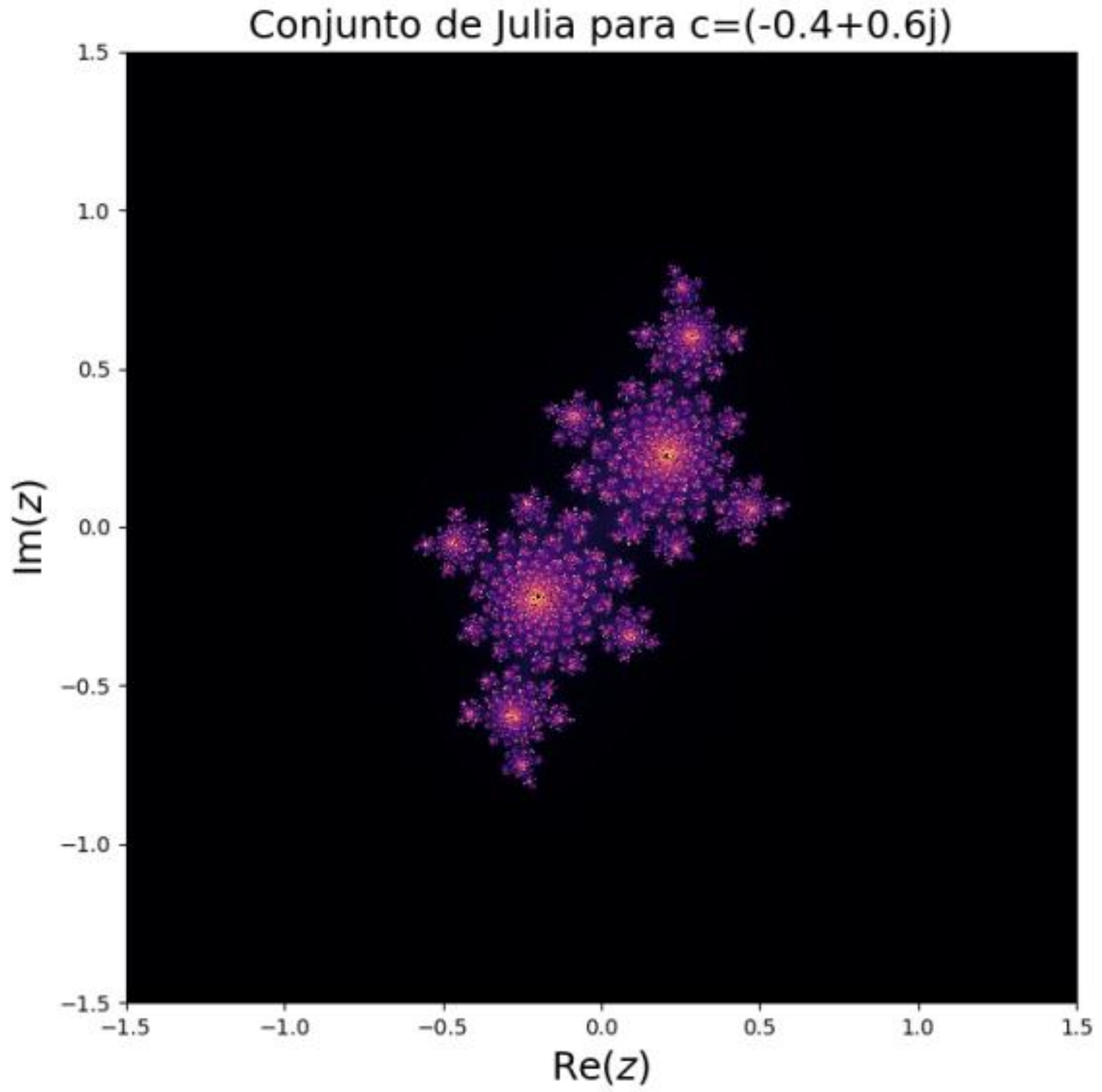
```python
# compiling the function
jit_julia_fractal = numba.jit(nopython=True)(py_julia_fractal)

# number of partitions
N = 1024
# Dimensioned numbers of the set
j = np.zeros((N, N), np.int64)
# constant of the set
c = np.complex(0.05, 0.68)
z_real = np.linspace(-2.5, 2.5, N)
z_imag = np.linspace(-2.5, 2.5, N)
```

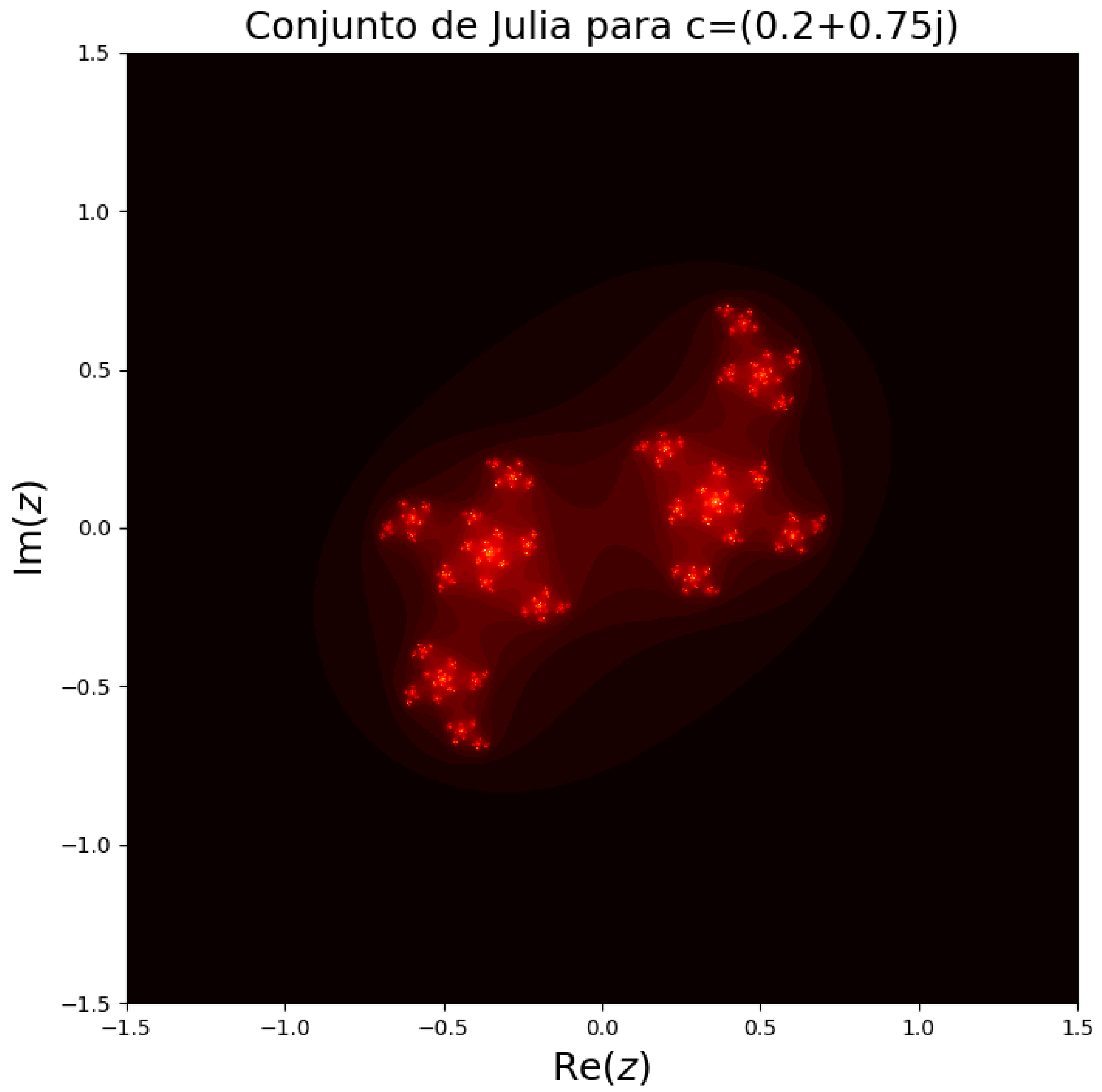
2

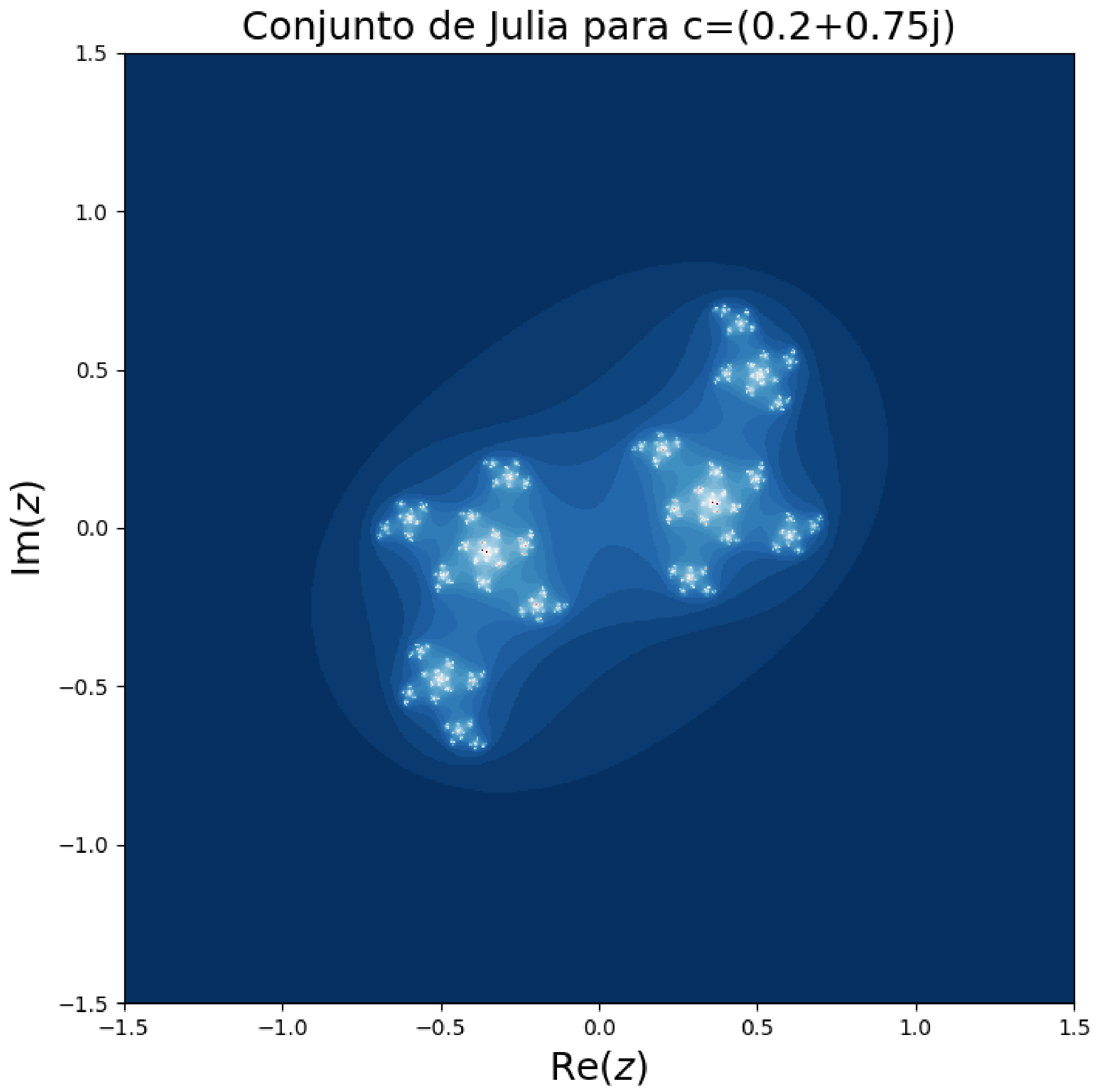
```python
# call the function
jit_julia_fractal(z_real, z_imag, j, c)

# plot the fig
fig, ax = plt.subplots(figsize=(8, 8))
ax.imshow(j, cmap=plt.cm.hot, extent=[-1.5, 1.5, -1.5, 1.5])
ax.set_title('Conjunto de Julia para c='+str(c),fontsize=18)
ax.set_xlabel("$\mathrm{Re}(z)$", fontsize=18)
ax.set_ylabel("$\mathrm{Im}(z)$", fontsize=18)
plt.show()
```

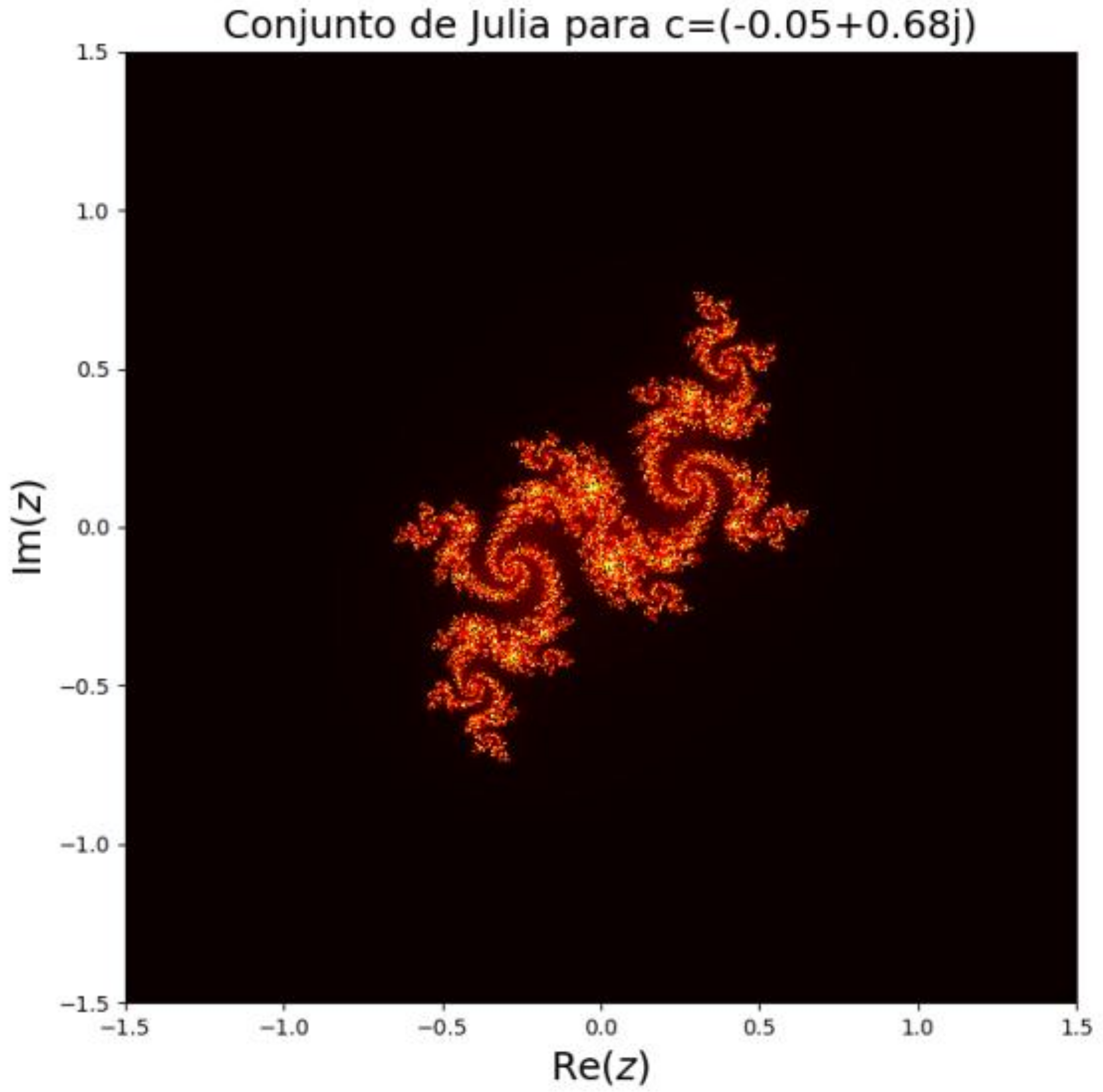
3

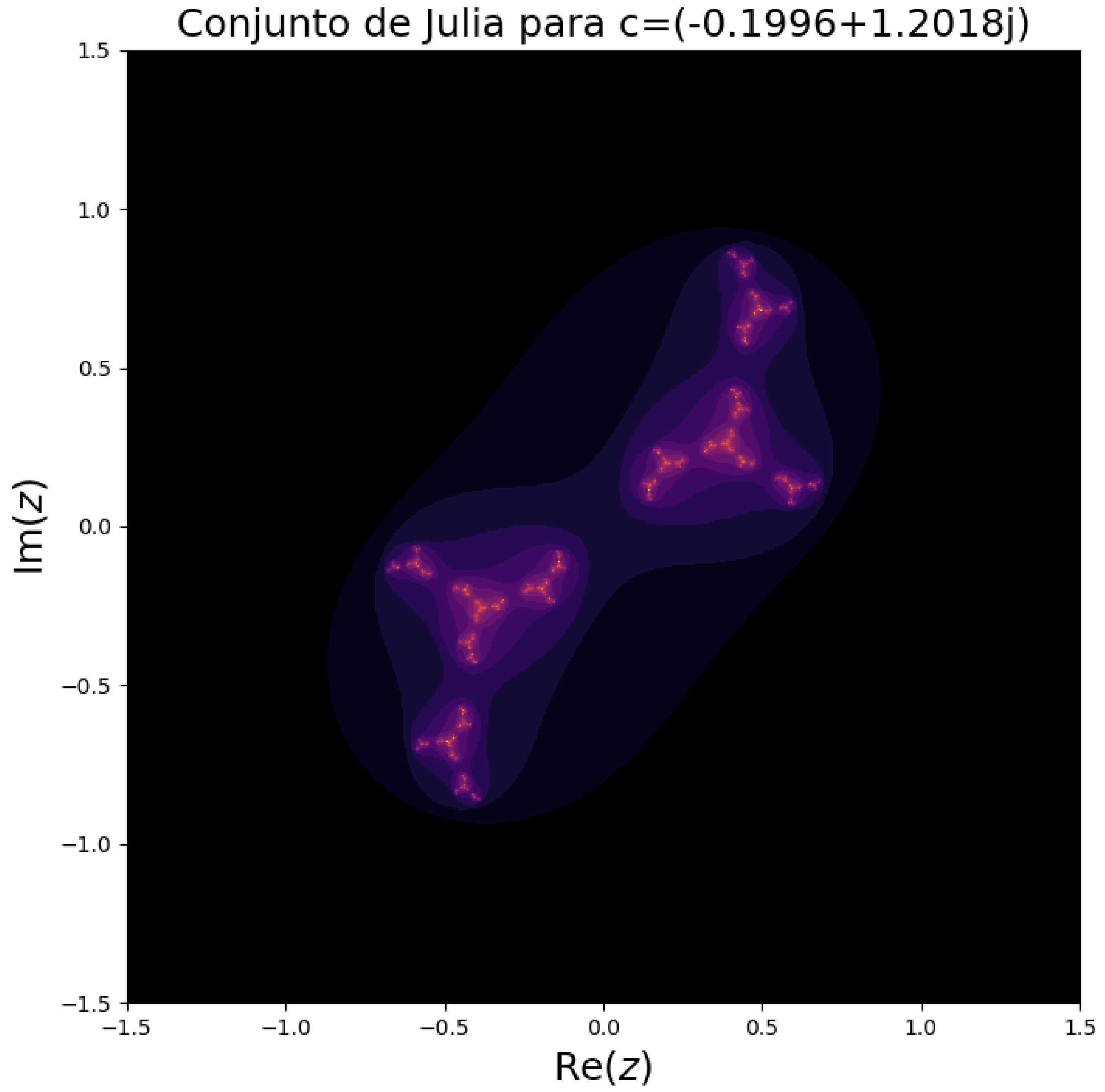
Conjunto de Julia para múltiples C

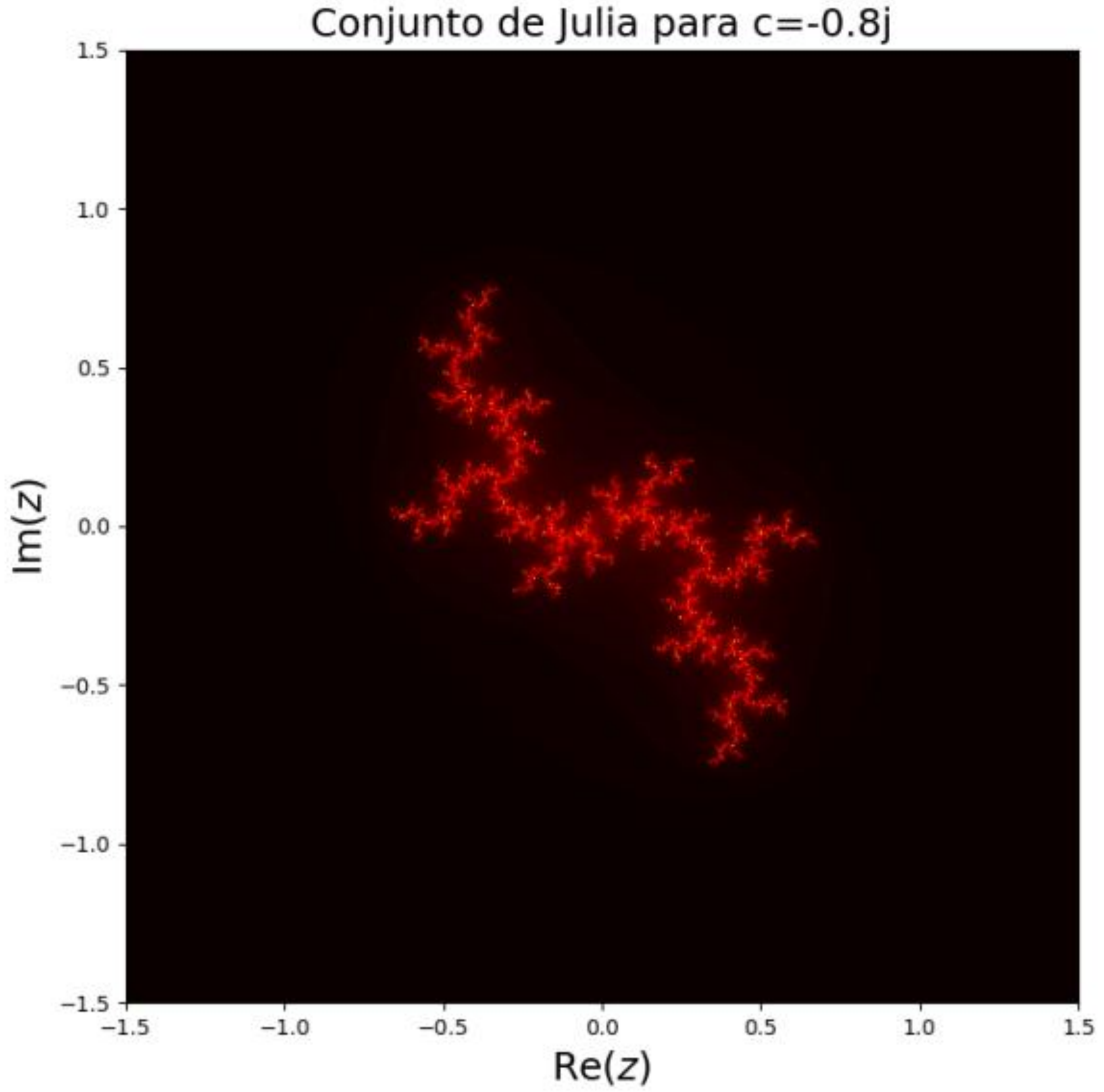












Aplicaciones

1980 Estudios de animación Pixar, **Loren Carpenter**: primera animación fractal por computadoras.

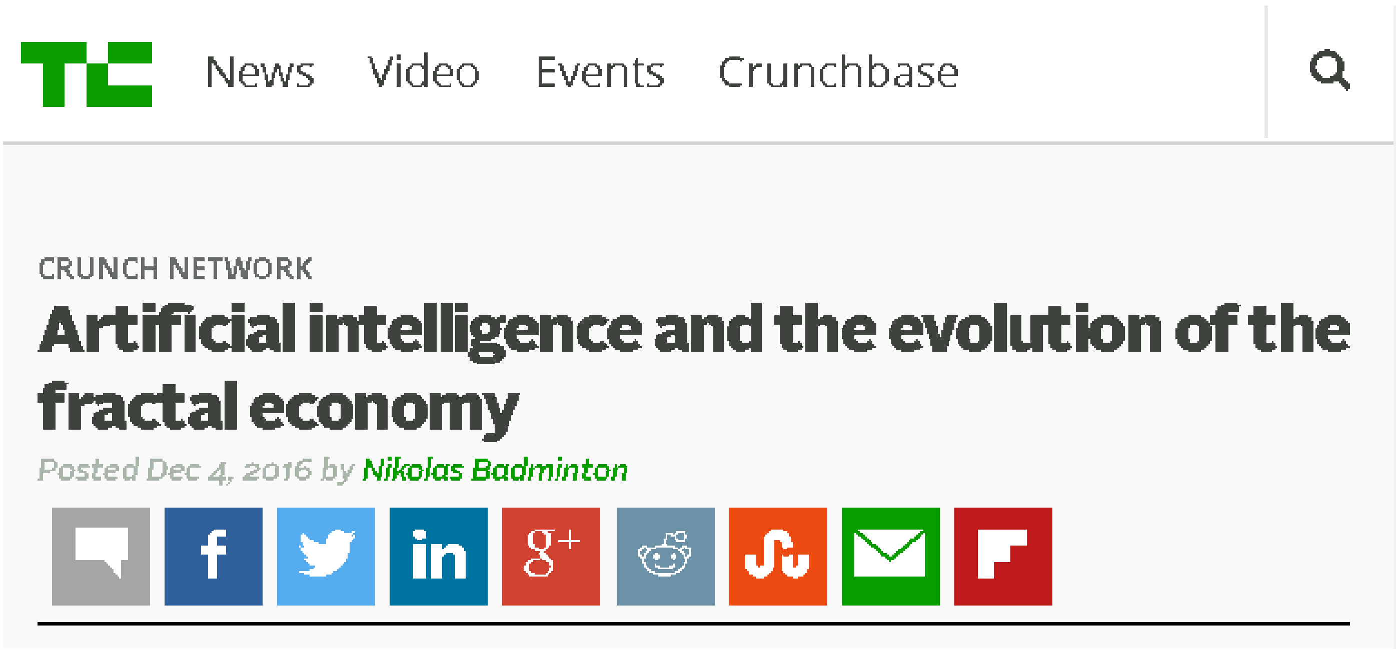
1985 IBM, **Benoit Mandelbrot**, corrección de ruido en transmisiones por cableado telefónico

1990 Iterated systems, **Michael Barnsley**, Algoritmo fractal de comprensión de imágenes

1990 Fractal Antenna Sys, **Nathan Cohen**, antena fractal con mayor rango de frecuencia

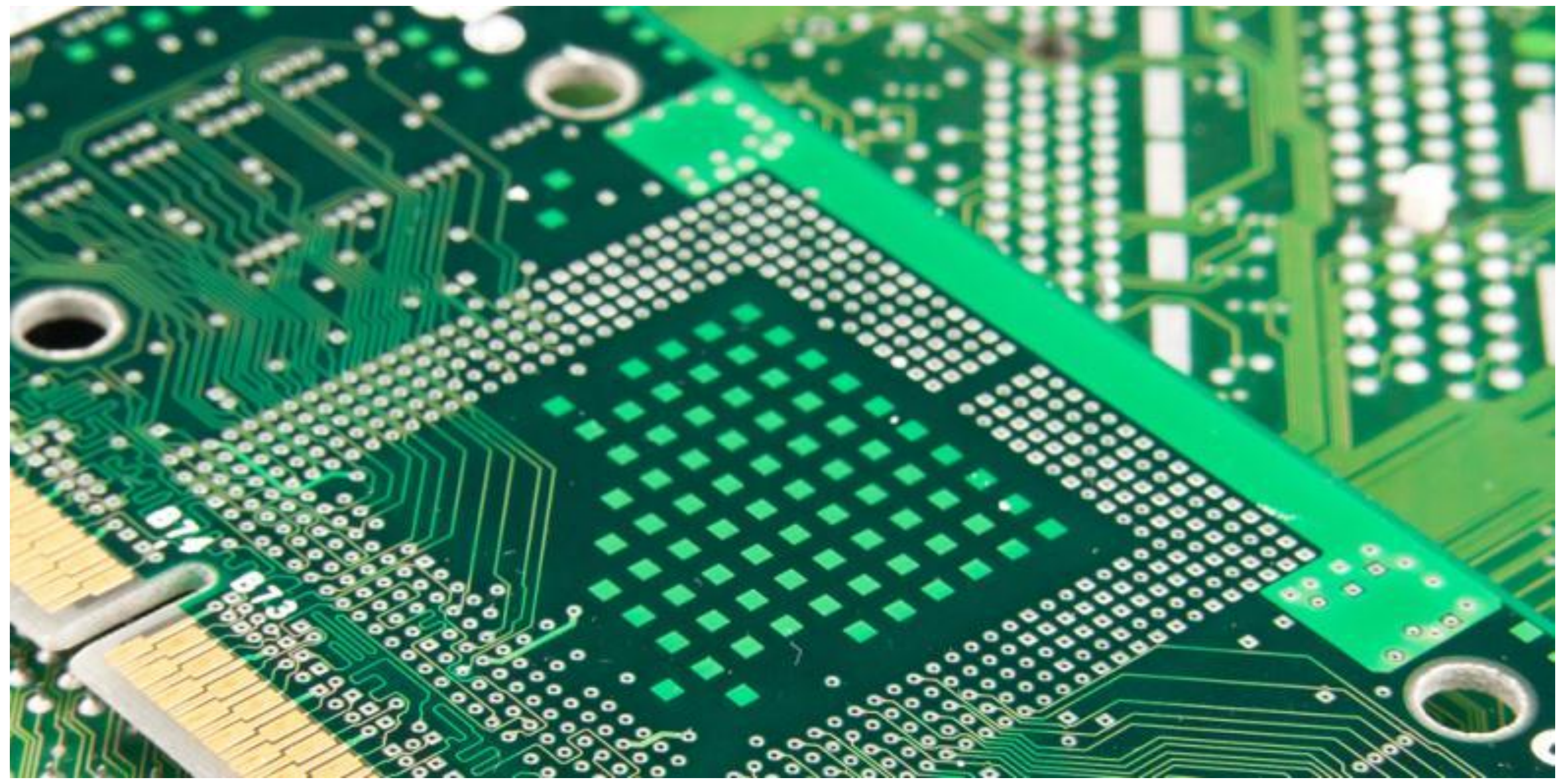
2002 Universidad de Harvard, **Ary Goldberger**, Electrodinámica fractal del latido del corazón

Hoy Inteligencia Artificial y fractales



Salvador Dalí, el rostro de la guerra

La gran Ola, katsushika Hokusai



Repositorio github

El código utilizado en ésta presentación se encuentra en el siguiente repositorio:

https://github.com/iris9112/Pycon2018_Fractals

Gracias!

¿Preguntas?

iris9112@gmail.com