El Juego de La Vida

Reglas:

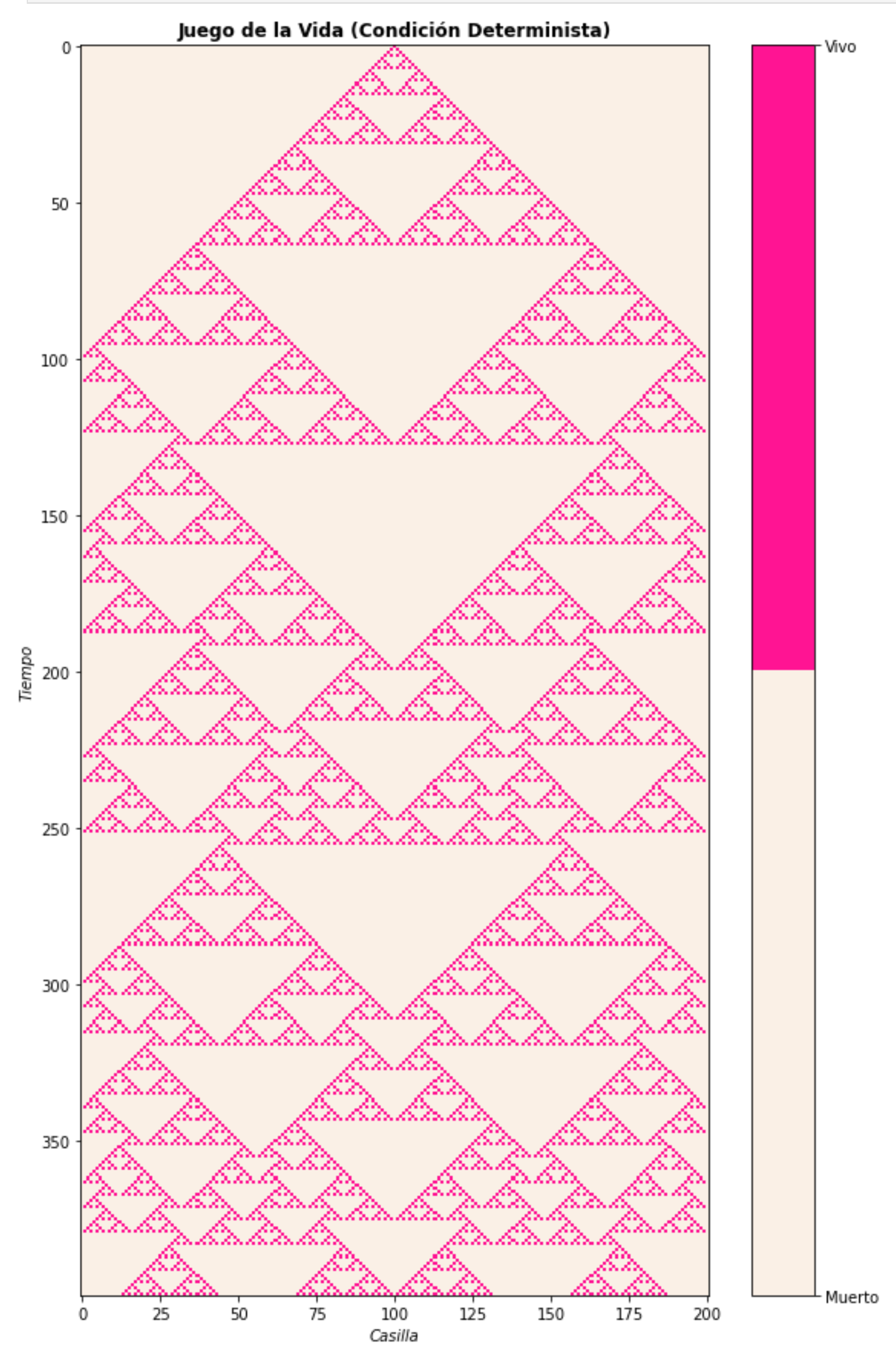
1. Si la casilla tiene sólo un vecino vivo, entonces la casilla del siguiente renglon (en esa misma columna) vive.
2. Si la casilla no tiene vecinos vivos, es decir, tiene 2 vecinos muertos, entonces la casilla del siguiente renglon (en esa misma columna) muere.
3. Si la casilla tiene 2 vecinos vivos, entonces la casilla del siguiente renglon (en esa misma columna) muere.

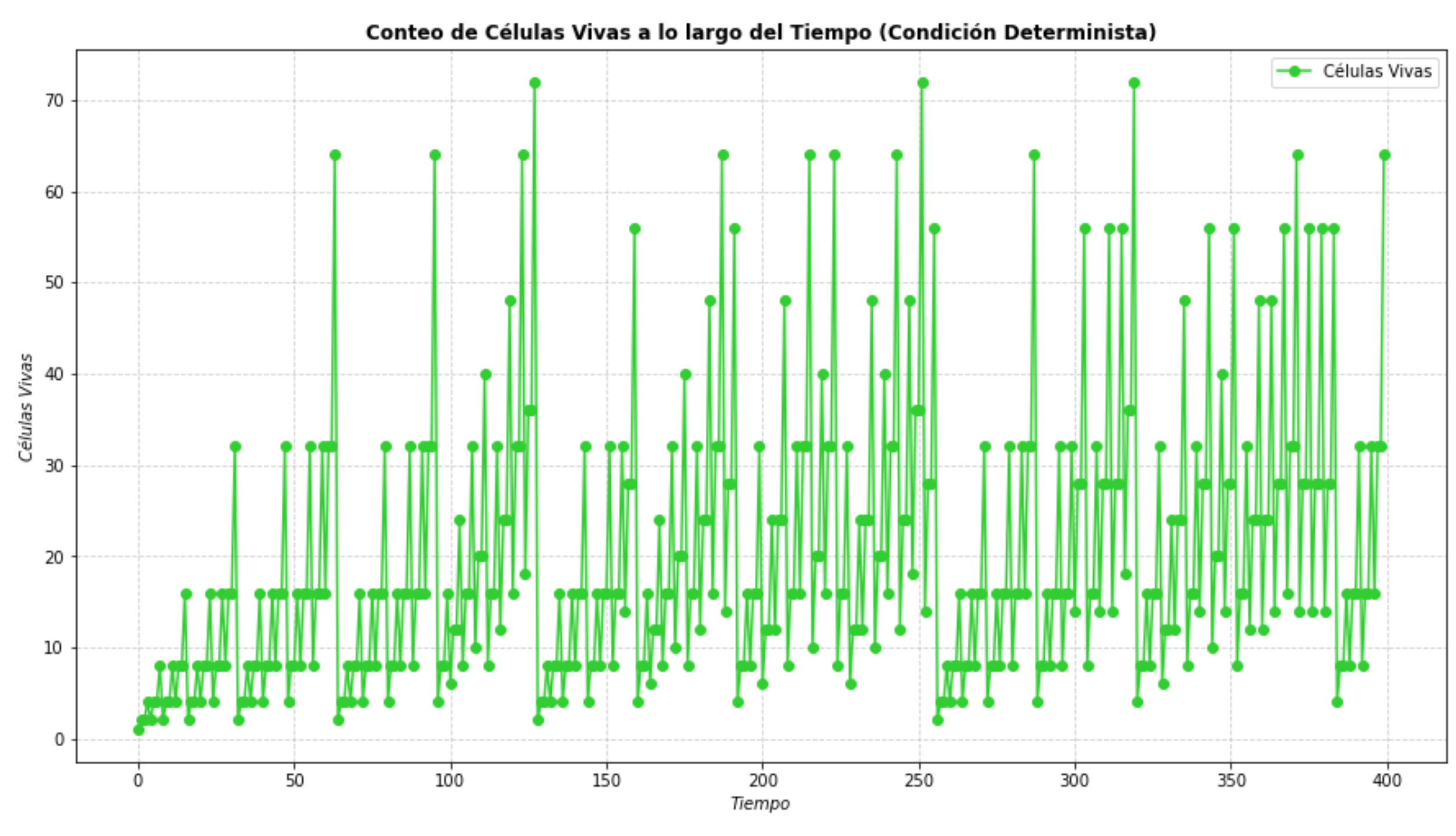
Instrucciones:

- En clase vimos el juego de la vida de Conway (la versión 1D). Presenta tu solución computacional (debes mostrar tu código) en un tablero de 200 cuadros y con 400 pasos de tiempo. Considera dos opciones:
  - a) Condiciones iniciales deterministas.
  - b) Condiciones iniciales aleatorias.

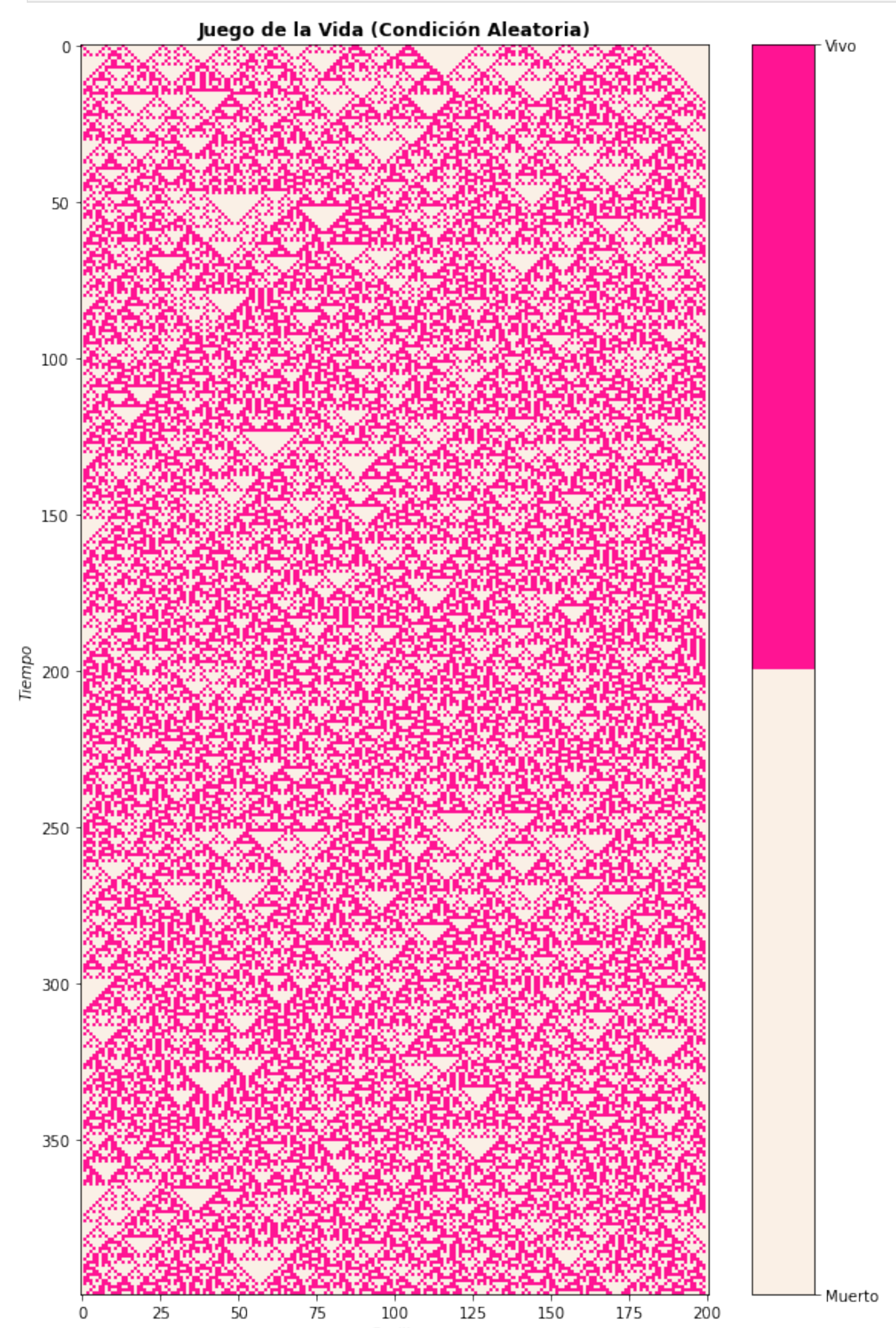
```
In [ ]: # Importamos las librerias necesarias
import numpy as np
import matplotlib.pyplot as plt
from matplotlib.colors import ListedColormap

# Función que recibe un tamaño de tablero y vector de celdas con condiciones iniciales y devuelve un gráfico simulando el juego de la vida
def gameOfLife(n_columnas, n_renglones, celulas, condicion):
    conteo = [np.sum(celulas[0])] # Lista para almacenar el conteo de células vivas en cada renglón
    for i in range(1, n_renglones):
        for j in range(1, n_columnas - 1):
            vecinos = [celulas[i-1, j-1], celulas[i-1, j+1]]
            if any(v == 1 for v in vecinos) and any(v == 0 for v in vecinos):
                celulas[i,j] = 1
        conteo.append(np.sum(celulas[i])) # Contamos las células vivas y las agregamos a la lista de conteos

    # Generamos la gráfica de nuestro juego
    plt.figure(figsize=(10, 15))
    plt.imshow(celulas, interpolation='nearest', cmap=ListedColormap(['linen', 'deeppink']))  # Colormap personalizado para representar "vivo" y "muerto"
    plt.xlabel('Casilla', fontstyle='italic')
    plt.ylabel('Tiempo', fontstyle='italic')
    plt.title('Juego de la Vida (Condición %s)' %condicion, fontweight='bold')
    plt.colorbar(ticks=[0, 1]).set_ticklabels(['Muerto', 'Vivo'])

    # Generamos la gráfica del conteo de células vivas en cada renglón
    plt.figure(figsize=(15, 8))
    plt.plot(range(0, n_renglones), conteo, marker='o', linestyle='-', color='limegreen', label='Células Vivas')
    plt.xlabel('Tiempo', fontstyle='italic')
    plt.ylabel('Células Vivas', fontstyle='italic')
    plt.title('Conteo de Células Vivas a lo largo del Tiempo (Condición %s)' %condicion, fontweight='bold')
    plt.grid(True, linestyle='--', alpha=0.6)
    plt.legend(loc='best')

    plt.show()
```

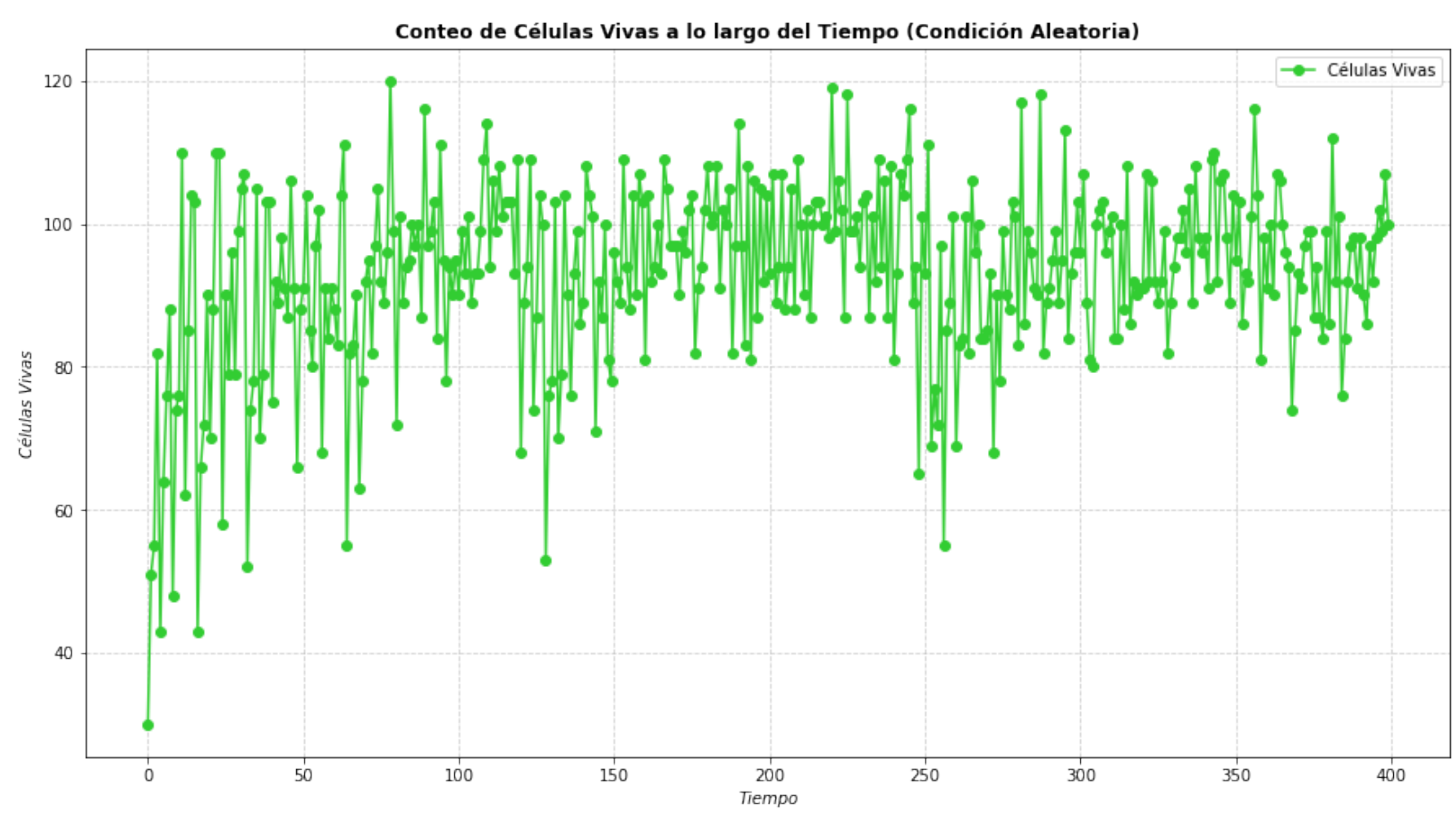
```
In [ ]: # Definimos el tamaño de nuestro tablero
n_columnas, n_renglones = 201, 400

# Matriz con 0's que representan todas las casillas muertas
celulas = np.zeros((n_renglones, n_columnas))

# Definimos la primer casilla viva (condición determinista)
celulas[0, int((n_columnas-1)/2)] = 1

# Reproducimos, de acuerdo a las reglas, para cada renglón
gameOfLife(n_columnas, n_renglones, celulas, "Determinista")
```





```
In [ ]: # Definimos el tamaño de nuestro tablero
n_columnas, n_renglones = 201, 400

# Matriz con 0's que representan todas las casillas muertas
celulas = np.zeros((n_renglones, n_columnas))

# Definimos las primeras 30 casillas vivas (condición aleatoria)
celulas[0,np.random.choice(n_columnas, 30, replace=False)] = 1

# Reproducimos, de acuerdo a las reglas, para cada renglón
gameOfLife(n_columnas, n_renglones, celulas, "Aleatoria")
```





- En el apartado 'introducción a sistemas dinámicos' hay una presentación que se titula 'material auxiliar'. Resuelve las preguntas para los dos casos: el triángulo de Sierpinski y el fractal de Koch

Triángulo de Sierpinski

1. ¿Cuántos triángulos completos hay en la iteración k?

$$N_k = 3^k$$

con k = {0,1,2,3,...,n}, donde k=0 representa nuestro triángulo equilátero inicial.

Notemos que cuando k tiende a ∞, tenemos que:

$$lim_{k\to\infty} N_k = \infty$$

2. ¿Cuál es el área de cada triángulo completo en la iteración k?
Sea a_0 el primer triángulo equilatero en k=0, el área de cada triángulo lleno está dado por:

$$A_k = (\frac{1}{4})^k \cdot a_0$$

puesto que, si nos fijamos en el caso k=1, tenemos 3 tríangulos completos y 1 vacío; por lo que c/u representa 1/4 del área total.

Notemos que cuando k tiende a ∞, tenemos que:

$$lim_{k\to\infty} A_k = 0$$

3. ¿Cuál es el área total del objeto en la iteración k?
Para encontrar la expresión, basta con hacer el producto de cada triángulo en k iteraciones por la cantidad de triángulos totales

$$A_{Total}(k) = N \cdot A_k = (\frac{3}{4})^k \cdot a_0$$

Notemos que cuando k tiende a ∞, tenemos que:

$$lim_{k\to\infty} A_{Total}(k) = 0$$

Copo de Koch

1. ¿Cuántos lados tiene el objeto en la iteración k?

$$N_k = 3 \cdot 4^k$$

con k = {0,1,2,3,...,n}, donde k=0 representa nuestro triángulo equilátero inicial.

Notemos que cuando k tiende a ∞, tenemos que:

$$lim_{k\to\infty} N_k = \infty$$

2. ¿Cuál es el perímetro del objeto en la iteración k?
Sea "ℓ" la longitud original de cada lado del triángulo equilatero en k=0, entonces la longitud de cada lado en la iteración k está dado por:

$$L_k = \frac{\ell}{3^k}$$

Por lo que, el perímetro del copo de Koch en la iteración k es:

$$P_k = N_k \cdot L_k = 3 \cdot \ell \cdot (\frac{4}{3})^k = P_0 \cdot (\frac{4}{3})^k$$

Notemos que cuando k tiende a ∞, tenemos que:

$$lim_{k\to\infty} P_k = \infty$$

3. ¿Cuál es el área del objeto en la iteración k?
Sea a_0 el primer triángulo equilatero en k=0, notemos que podemos dividirlo en 9 triángulos equilateros idénticos por la relación de semejanza entre sus áreas. Por lo que, el área de cada triangulo en la iteración k está dado por:

$$a_k = \frac{1}{9^k} \cdot a_0$$

Por lo que, para calcular el área añadida por todos los triángulos en una cierta iteración k, basta con calcular el producto del área de cada triángulo por el número total de triángulos, o su equivalente, la cantidad total de lados del triángulo:

$$A_k = N_k \cdot a_n = 3 \cdot 4^{k-1} \cdot \frac{1}{9^k} \cdot a_0 = \frac{3}{4} \cdot (\frac{4}{9})^k \cdot a_0$$

Finalmente, para calcular el área total del Copo de Koch podemos aproximarlo mediante la suma de todos los triángulos en cada iteración, tal que:

$$A_{Total}(k) = a_0 + \sum_{i=1}^{k} A_i \ = a_0 + \sum_{i=1}^{k} \frac{3}{4} \cdot a_0 \cdot (\frac{4}{9})^i$$

Donde

$$\sum_{i=1}^{k} \frac{3}{4} \cdot a_0 \cdot (\frac{4}{9})^i$$

es la serie geométrica, tal que:

$$\sum_{i=0}^{n} ar^i = a \cdot \frac{1 - r^{n+1}}{1 - r}$$

Por lo que, resolvemos:

$$A_{Total}(k) = a_0 + \frac{3}{4} \cdot a_0 \cdot (\frac{1 - \frac{4}{9}^{k+1}}{1 - \frac{4}{9}} - 1) = a_0 \cdot (1 - \frac{3}{4} + \frac{3}{4} \cdot (\frac{1 - \frac{4}{9}^{k+1}}{\frac{5}{9}}))$$

Notemos que nuestra expresión llega hasta k, por lo que nos fijamos en el caso cuando k tiende a infinito, tal que:

$$lim_{k\to\infty} A_{Total}(k) = a_0 \cdot (1 - \frac{3}{4} + \frac{3}{4} \cdot (\frac{1-0}{\frac{5}{9}})) = a_0 \cdot (\frac{1}{4} + \frac{3}{4} \cdot \frac{9}{5})$$

$$lim_{k\to\infty} A_{Total}(k) = \frac{8}{5} \cdot a_0$$