“””

Jhosue Abel Nacimba Suntasig

Programación Gr1

Fecha entrega: lunes 4 de septiembre del 2023

“””

# Importa bibliotecas a ocuparse

import numpy as np

import matplotlib.pyplot as plt

import matplotlib.animation as animation

# Genera una cuadrícula con celdas vivas/muertas, aleatoriamente

def random\_grid(rows, cols):

return np.random.choice([0, 1], size=(rows, cols), p=[0.7, 0.3])

# Crea una cuadrícula inicial basada en una matriz de ceros y unos

def glider\_gun\_grid(rows, cols):

grid = np.zeros((rows, cols), dtype=int)

glider\_gun = np.array([

[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],

[0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0],

[0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0],

[0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0],

[0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0],

[0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],

[0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0],

[0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 0],

])

grid[1:9, 1:17] = glider\_gun

return grid

# Actualiza los frames acorde a las reglas del código

def update(frame):

global grid, running

if running:

new\_grid = grid.copy()

rows, cols = grid.shape

for i in range(rows):

for j in range(cols):

neighbors\_sum = np.sum(grid[max(0, i-1):min(rows, i+2), max(0, j-1):min(cols, j+2)]) - grid[i, j]

if grid[i, j] == 1:

if neighbors\_sum < 2 or neighbors\_sum > 3:

new\_grid[i, j] = 0

else:

if neighbors\_sum == 3:

new\_grid[i, j] = 1

grid = new\_grid

mat.set\_array(grid)

return mat,

# Se ocupa para interactuar con el código, pausando o corriendo el mismo con un click

def onClick(event):

global running

running = not running

# Muestra una configuración inicial, ocupando la cuadricula creada con glider\_gun\_grid

rows, cols = 50, 50

grid = random\_grid(rows, cols)

# Configurar la figura

fig, ax = plt.subplots()

mat = ax.matshow(grid, cmap='gray')

# Forma parte del evento de onClick, para interactuar con la animación

running = True

fig.canvas.mpl\_connect('button\_press\_event', onClick)

# Definir la animación

ani = animation.FuncAnimation(fig, update, interval=200, blit=True)

# Muestra la animación

plt.show()