```
import numpy as np

class Lagton(object):
    """docstring for Lagton."""
    MAP = {\
            'N': (-1, 0),\
            'E':(0, 1),\
            'S': (1, 0),\
            'W':(0, -1)\
            }
    DIRECTIONS=['N', 'E', 'S', 'W']

    def __init__(self, N=256):
        self.grid=np.zeros((N,N), np.uint)
        self.aliveValue=1
        self.deadValue=0
        self.ant_cell=(int(N/2), int(N/2))
        self.ant_direction=3

    def getStates(self):
        return self.grid

    def getGrid(self):
        '''
        Same as getStates()
        '''
        return self.getStates()

    def evolve(self):
        def rotate(mode):
            if mode=='C':
                self.ant_direction+=1
                if self.ant_direction >= 4:
                    self.ant_direction=0
            elif mode=='A':
                self.ant_direction-=1
                if self.ant_direction <= -4:
                    self.ant_direction=0

        def move():
            move_r, move_c =
Lagton.MAP[Lagton.DIRECTIONS[self.ant_direction]]
            self.ant_cell = (self.ant_cell[0]+move_r,
self.ant_cell[1]+move_c)

        def toggle(value):
            updated=np.array(self.grid)
            updated[self.ant_cell[0]][self.ant_cell[1]]= value
            return updated
        #change direction
        if self.grid[self.ant_cell[0]][self.ant_cell[1]] ==
self.deadValue:
```

```
            rotate('C')
            value=self.aliveValue
        elif self.grid[self.ant_cell[0]][self.ant_cell[1]] ==
self.aliveValue:
            rotate('A')
            value=self.deadValue
        #toggle cell
        self.grid=toggle(value)

        #move
        move()

    def show(self):
        row, column = self.grid.shape
        for c in range(0, column):
            line=""
            for r in range(0, row):
                line+="{}".format(self.grid[r][c])
            print(line)
        print("ant at {} direction {}".format(self.ant_cell,
Lagton.DIRECTIONS[self.ant_direction]))

if __name__=="__main__":
    life=Lagton(11)
    life.show()
    life.evolve()
    life.show()
    life.evolve()
    life.show()
    life.evolve()
    life.show()
    life.evolve()
    life.show()
    life.evolve()
    life.show()


import numpy as np


COLOR_DICT = {'BLACK':[0,0,0],'AQUA':[0,102,102], 'BLUE':[0,0,255],
'PURPLE':[153,0,76], 'RED':[255,0,0], 'ORANGE':[255,128,0],
'YELLOW':[255,255,0], 'PINK':[255,102,178], 'WHITE':[255,255,255]}


class Lagton(object):
    """docstring for Lagton."""
    MAP = {\
            'N': (-1, 0),\
            'E':(0, 1),\
            'S': (1, 0),\
            'W':(0, -1)\
            }
    DIRECTIONS=['N', 'E', 'S', 'W']
```

```
    def __init__(self, N=256):
        self.grid=np.zeros((N,N,3), np.uint)
        self.ant_cell=(int(N/2), int(N/2))
        self.ant_direction=3

    def getStates(self):
        return self.grid

    def getGrid(self):
        '''
        Same as getStates()
        '''
        return self.getStates()

    def evolve(self):
        def rotate(mode):
            if mode=='C':
                self.ant_direction+=1
                if self.ant_direction >= 4:
                    self.ant_direction=0
            elif mode=='A':
                self.ant_direction-=1
                if self.ant_direction <= -4:
                    self.ant_direction=0

        def move():
            move_r, move_c =
Lagton.MAP[Lagton.DIRECTIONS[self.ant_direction]]
            self.ant_cell = (self.ant_cell[0]+move_r,
self.ant_cell[1]+move_c)

        def toggle(value):
            updated=np.array(self.grid)
            updated[self.ant_cell[0]][self.ant_cell[1]]=value
            return updated
        #change direction
        if (self.grid[self.ant_cell[0]][self.ant_cell[1]] ==
COLOR_DICT['BLACK']).all():
            rotate('A')
            value=COLOR_DICT['AQUA']
            self.grid=toggle(value)
        elif (self.grid[self.ant_cell[0]][self.ant_cell[1]] ==
COLOR_DICT['AQUA']).all():
            rotate('C')
            value=COLOR_DICT['BLUE']
            self.grid=toggle(value)
        elif (self.grid[self.ant_cell[0]][self.ant_cell[1]] ==
COLOR_DICT['BLUE']).all():
            rotate('C')
            value=COLOR_DICT['PURPLE']
            self.grid=toggle(value)
        elif (self.grid[self.ant_cell[0]][self.ant_cell[1]] ==
COLOR_DICT['PURPLE']).all():
```

```
            rotate('C')
            value=COLOR_DICT['RED']
            self.grid=toggle(value)
        elif (self.grid[self.ant_cell[0]][self.ant_cell[1]] ==
COLOR_DICT['RED']).all():
            rotate('C')
            value=COLOR_DICT['ORANGE']
            self.grid=toggle(value)
        elif (self.grid[self.ant_cell[0]][self.ant_cell[1]] ==
COLOR_DICT['ORANGE']).all():
            rotate('C')
            value=COLOR_DICT['YELLOW']
            self.grid=toggle(value)
        elif (self.grid[self.ant_cell[0]][self.ant_cell[1]] ==
COLOR_DICT['YELLOW']).all():
            rotate('A')
            value=COLOR_DICT['PINK']
            self.grid=toggle(value)
        elif (self.grid[self.ant_cell[0]][self.ant_cell[1]] ==
COLOR_DICT['PINK']).all():
            rotate('A')
            value=COLOR_DICT['WHITE']
            self.grid=toggle(value)
        elif (self.grid[self.ant_cell[0]][self.ant_cell[1]] ==
COLOR_DICT['WHITE']).all():
            rotate('C')
            value=COLOR_DICT['BLACK']
            self.grid=toggle(value)
        #toggle cell


        #move
        move()

    def show(self):
        row, column, color = self.grid.shape
        for c in range(0, column):
            line=""
            for r in range(0, row):
                line+="{}".format(self.grid[r][c])
            print(line)
        print("ant at {} direction {}".format(self.ant_cell,
Lagton.DIRECTIONS[self.ant_direction]))

if __name__=="__main__":
    life=Lagton(11)
    life.show()
    life.evolve()
    life.show()
    life.evolve()
    life.show()
    life.evolve()
    life.show()
    life.evolve()
```

```
    life.show()
    life.evolve()
    life.show()
# -*- coding: utf-8 -*-
"""
Game of life script with animated evolution

Created on Tue Jan 15 12:37:52 2019

@author: shakes
"""

#import lagton

import lagton_colors as lagton

N = 64

#create the game of life object
life = lagton.Lagton(N)
life.evolve()
cells = life.getStates() #initial state

#-------------------------------
#plot cells
import matplotlib.pyplot as plt
import matplotlib.animation as animation

fig = plt.figure()

plt.gray()

img = plt.imshow(cells, animated=True)

def animate(i):
    """perform animation step"""
    global life

    life.evolve()
    cellsUpdated = life.getStates()

    img.set_array(cellsUpdated)

    return img,

interval = 1 #ms

#animate 24 frames with interval between them calling animate
function at each frame
ani = animation.FuncAnimation(fig, animate, frames=24,
interval=interval, blit=True)

plt.show()
```