fractal Documentation

Release 0.0.1

Les Collaborateurs Illégitimes

Feb 27, 2019

CONTENTS:

CHAPTER ONE

FUNCTION DOCUMENTATION

class main.**Figures**(*im*, *mode=None*)

A lot of function to create some well-know shapes

static _int(*value*)

Make a tuple of float coordinate into tuple of int coordinate

Parameters **value** (*tuple*) – Tuple to convert

Returns new tuple with int values

Return type tuple(int, int)

blanc_manger(*origin*, *finish*, *iterations*, *color=None*, *width=0*)

Trace blanc manger curve

Parameters

- **origin** (*tuple*) – coordinate of the starting point
- **finish** (*tuple*) – coordinate of the ending point
- **iterations** (*int*) – iterations for the drawings
- **color** (*tuple*) – color to use for the lines
- **width** (*int*) – the line width, in pixels

static complex_to_point(*point*)

Transform tuple to complex

Parameters **point** (*complex*) – Point to convert

Returns tuple representation of point

Return type tuple

homothety(*point*, *center=0j*, *size=0*)

Homothety of point in complex plane

Parameters

- **point** (*tuple or complex*) – point (or list of point) to make homothety
- **center** (*tuple or complex*) – center of homothety
- **size** (*float*) – size of homothety

Returns Homothety of point (or list of homothety of points)

Return type tuple or list of tuples

static point_to_complex(*point*)

Transform tuple to complex

Parameters **point** (*tuple*) – Point to convert

Returns Complex representation of point

Return type complex

rotation(*point*, *center=0j*, *angle=0*)

Rotate point in complex plane

Parameters

- **point** (*tuple or complex*) – point (or list of point) to rotate
- **center** (*tuple or complex*) – center of rotation
- **angle** (*float*) – angle of rotation

Returns Rotated point (or list of rotated points)

Return type tuple or list of tuples

translation(*point*, *vect*)

Translate point in complex plane

Parameters

- **point** (*tuple or complex*) – point (or list of point) to translate
- **vect** (*tuple or complex*) – vector of translation

Returns Translated point (or list of translated points)

Return type tuple or list of tuples

von_koch_curve(*origin*, *finish*, *iterations=1*, *color=None*, *width=0*)

Draw the von koch curve on image.

Parameters

- **origin** (*tuple*) – coordinate of the starting point
- **finish** (*tuple*) – coordinate of the ending point
- **iterations** (*int*) – iterations for the drawings
- **color** (*tuple*) – color to use for the lines
- **width** (*int*) – the line width, in pixels

von_koch_curve_flake(*origin*, *radius*, *iterations*, *angle=0*, *color=None*, *width=0*)

Draw the von koch flake on image.

Parameters

- **origin** (*tuple*) – coordinate of the center of circumscribed circle of main triangle
- **radius** (*float*) – radius of circumscribed circle of main triangle
- **iterations** (*int*) – iterations for the drawings
- **angle** (*float*) – rotation of main triangle
- **color** (*tuple*) – color to use for the lines
- **width** (*int*) – the line width, in pixels

class main.**Lsystem**(**args*, ***kwargs*)

Draw a L system

_backward(*distance*)

Backward pen of distance

Parameters **distance** (*float*) – Distance to backward

_forward(*distance*)

Forward pen of distance

Parameters **distance** (*float*) – Distance to forward

_left(*angle*)

Turn pen to left of angle

Parameters **angle** (*float*) – Angle to rotate

_restore()

Restore last pen state

_right(*angle*)

Turn pen to right of angle

Parameters **angle** (*float*) – Angle to rotate

_save()

Save state of pen

backward(*distance*)

Return a lambda function which make pen backward of distance

Parameters **distance** (*float*) – Distance to build function

Returns lambda function to make pen backward

Return type lambda

dragon(*size*, *recursions*, *color=None*, *width=0*)

Trace Dragon curve

Parameters

- **size** (*float*) – Lenght of a segment
- **recursions** (*int*) – number of recursions
- **color** (*tuple*) – color of drawing
- **width** (*int*) – width of drawing

draw_l(*start*, *replacement*, *constants*, *nb_recursive*, *color=(255, 255, 255)*, *width=0*)

Draw a L system

Parameters

- **start** (*str*) – Axiome
- **replacement** (*dict*) – Dictionary which contain replacement values (F->F+F-F-F+F)
- **constants** (*dict*) – Dictionary which contain all elements with there function
- **nb_recursive** (*int*) – Number of recursion
- **color** (*tuple*) – Color to use for the drawing
- **width** (*int*) – The line width, in pixels

forward(*distance*)

Return a lambda function which make pen forward of distance

Parameters **distance** (*float*) – Distance to build function

Returns lambda function to make pen forward

Return type lambda

fractal_binary_tree(*size*, *recursions*, *color=None*, *width=0*)

Draw fractal binary tree

Parameters

- **size** (*float*) – Lenght of a segment
- **recursions** (*int*) – number of recursions
- **color** (*tuple*) – color of drawing
- **width** (*int*) – width of drawing

fractal_plant(*size*, *recursions*, *color=None*, *width=0*)

Draw the fractal plant

Parameters

- **size** (*float*) – Lenght of a segment
- **recursions** (*int*) – number of recursions
- **color** (*tuple*) – color of drawing
- **width** (*int*) – width of drawing

koch_curve_right_angle(*size*, *recursions*, *color=None*, *width=0*)

Draw koch curve with right angle

Parameters

- **size** (*float*) – Lenght of a segment
- **recursions** (*int*) – number of recursions
- **color** (*tuple*) – color of drawing
- **width** (*int*) – width of drawing

left(*angle*)

Return a lambda function which make pen turning of angle radians to left

Parameters **angle** (*float*) – Angle to build function

Returns lambda function to make pen turning left

Return type lambda

nothing()

restore()

Return a lambda function which restore state of pen

Returns lambda function to restore pen state

Return type lambda

right(*angle*)

Return a lambda function which make pen turning of angle radians to right

Parameters **`angle`** (*`float`*) – Angle to build function

Returns lambda function to make pen turning right

Return type lambda

`save`()

Return a lambda function which save state of pen

Returns lambda function to save pen state

Return type lambda

`set_pos`(*x*, *y*)

Set position of pen

Parameters

- **`x`** (*`float`*) – x coordinate
- **`y`** (*`float`*) – y coordinate

`sierpinski_triangle`(*size*, *recursions*, *color=None*, *width=0*)

Draw the sierpinski triangle

Parameters

- **`size`** (*`float`*) – Lenght of a segment
- **`recursions`** (*`int`*) – number of recursions
- **`color`** (*`tuple`*) – color of drawing
- **`width`** (*`int`*) – width of drawing

`class` `main.`**`State`**

State of Lsystem

CHAPTER

TWO

INDICES AND TABLES

- genindex
- modindex
- search

PYTHON MODULE INDEX

INDEX