API Documentation

API Documentation

November 28, 2014

Contents

1 Package FractalZE

Author: López Ricardo Ezequiel

Organization: UTN FRRE - Matemática Superior

Contact: lopezezequiel.09@gmail.com

Bug: probablemente muchos, fue hecho en 2 dias

To Do: agregar validacion, optimizar, eliminar redundacias, agregar fractales, agregar funciones

1.1 Modules

1.2 Variables

Name	Description
___package___	**Value:** `None`

2 Module FractalZE.arbol

2.1 Variables

Name	Description
___package___	**Value:** `'FractalZE'`

2.2 Class Arbol

FractalZE.fractal.Fractal
└ **FractalZE.arbol.Arbol**

Fractal del arbol. Construccion:

1. Se parte de un segmento.
2. Desde la punta de cada rama se dibujan dos nuevos segmentos de la mitad de la longitud de la rama.
   (a) El primero apuntando 60 grados a la izquierda de la direccion de la rama.
   (b) El segundo apuntando 60 grados a la derecha de la direccion de la rama.
3. Se repite esto sucesivamente con cada rama las veces que se desee.

2.2.1 Methods

___init___(*self*, *l*=`1`, *angle*=`0.523598775598`)

Parameters

`l:` longitud del segmento maximo.
(type=float)

`angle:` angulo ramas en radianes
(type=float)

segmentLength(*self*, *n*=`0`)

Devuelve la longitud de un segmento del termino n.

Parameters

`n:` numero de termino de la sucesion
(type=int)

Return Value

longitud de un segmento.
(type=float)

countSegments(*self*, *n*)

Devuelve la cantidad de segmentos nuevos.

Parameters

`n:` numero de termino de la sucesion

(type=int)

Return Value

cantidad de segmentos.

(type=int)

totalLength(*self*, *n*)

Devuelve la suma de la longitud de todos los segmentos.

Parameters

`n:` numero de termino de la sucesion

(type=int)

Return Value

longitud total.

(type=float)

lindenmayer(*self*, *n*)

Devuelve el fractal del arbol expresado en L-System

Parameters

`n:` numero de termino de la sucesion

(type=int)

Return Value

fractal expresado en L-System.

(type=string

Algoritmo en L-System.

- *variables: 0 1*
- *constantes: [ ]*
- *axioma: 0*
- *reglas:*
  1. *1 -> 11*
  2. *0 -> 1[0]0*

Interpretacion.

1. *Un 0 significa dibujar una hoja (segmento de longitud L/2)*
2. *Un 1 significa dibujar una rama (segmento de longitud L)*
3. *Un [significa guardar posicion y angulo en la pila, y girar X grados en sentido antihorario*
4. *Un] significa extraer posicion y angulo de la pila, y girar X grados en sentido horario*

)

getHeight(*self*, *n*, *angle*=`1.57079632679`)

Devuelve la altura del arbol.

Parameters

`n:` numero de termino de la sucesion

(type=int)

`angle:` no se usa, necesario para la recursion

(type=float)

Return Value

altura del arbol.

(type=float)

getWidth(*self*, *n*=`0`, *angle*=`1.57079632679`)

Devuelve el ancho del arbol.

Parameters

`n:` numero de termino de la sucesion

(type=int)

`angle:` no se usa, necesario para la recursion

(type=float)

Return Value

ancho del arbol.

(type=float)

Inherited from FractalZE.fractal.Fractal(Section 5.2)

graph()

3 Module FractalZE.cantor

3.1 Variables

Name	Description
___package___	**Value:** `'FractalZE'`

3.2 Class Cantor

FractalZE.fractal.Fractal
└
FractalZE.cantor.Cantor

Known Subclasses: FractalZE.quintoCantor.QuintoCantor

Fractal de Cantor. Construccion:

1. Se parte de un segmento.
2. Se lo divide en 3 partes iguales.
3. Se descarta la del medio.
4. Se repite esto sucesivamente con cada segmento las veces que se desee.

3.2.1 Methods

___init___(*self*, *l*=`1.0`, *segmentHeight*=`1`)

Parameters

`l`: longitud del segmento maximo. Default 1

(type=float)

`segmentHeight`: altura que va a tener el segmento al dibujarlo(en pixels)

(type=int)

countSegments(*self*, *n*)

Devuelve la cantidad de segmentos del termino n.

Parameters

`n`: numero de termino de la sucesion

(type=int)

Return Value

cantidad de segmentos.

(type=int)

segmentLength(*self*, *n*)

Devuelve la longitud de un segmento del termino n.

Parameters

`n`: numero de termino de la sucesion

(type=int)

Return Value

longitud de un segmento.

(type=float)

totalLength(*self*, *n*)

Devuelve la suma de la longitud de todos los segmentos de un termino.

Parameters

`n`: numero de termino de la sucesion

(type=int)

Return Value

longitud total.

(type=float)

lindenmayer(*self*, *n*=0)

Devuelve el fractal de cantor expresado en L-System.

Parameters

`n:` termino de la sucesion

(type=int)

Return Value

termino expresado en L-system.

(type=string

Algoritmo en L-System.

- *variables: _ (space)*
- *constantes:*
- *axioma: _*
- *reglas:*
  1. *_ -> _ _*
  2. *(space) -> (space)(space)(space)*

Interpretacion.

1. *Un _(guion bajo) significa dibujar un segmento*
2. *Un (space) significa avanzar la longitud de un segmento sin dibujar*

)

getWidth(*self*, *n*=0)

Devuelve el ancho del fractal.

Return Value

ancho del fractal

(type=float)

getHeight(*self*, *n*=0)

Devuelve la altura del fractal.

Return Value

alto del fractal

(type=float)

Inherited from FractalZE.fractal.Fractal(Section 5.2)

graph()

4 Module FractalZE.dragon

4.1 Variables

Name	Description
___package___	**Value:** `'FractalZE'`

4.2 Class Dragon

FractalZE.fractal.Fractal

FractalZE.dragon.Dragon

4.2.1 Methods

___init___(*self*, *l*=1)

Parameters

`l`: longitud del segmento inicial. Default 1

(type=float)

lindenmayer(*self*, *n*)

Devuelve el fractal del dragón expresado en L-System

Parameters

`n:` numero de termino de la sucesion

(type=int)

Return Value

fractal del dragón expresada en L-System.

(type=string

Algoritmo en L-System.

- *variables: X Y*
- *constantes: F + -*
- *axioma: FX*
- *reglas:*
  1. *X -> X+YF+*
  2. *Y -> -FX-Y*

Interpretacion.

1. *Una F significa dibujar*
2. *Un - significa girar a la izquierda 90 grados*
1. *Un + significa girar a la derecha 90 grados*

)

getWidth(*self*, *n*=0)

getHeight(*self*, *n*=0)

Inherited from FractalZE.fractal.Fractal(Section 5.2)

graph()

5 Module FractalZE.fractal

5.1 Variables

Name	Description
___package___	**Value:** `'FractalZE'`

5.2 Class Fractal

Known Subclasses: FractalZE.cantor.Cantor, FractalZE.sierpinsky.Sierpinsky, FractalZE.koch.Koch, FractalZE.arbol.Arbol, FractalZE.pascal.Pascal, FractalZE.dragon.Dragon

5.2.1 Methods

graph(*self*, *n*=`0`, *mode*=`'1'`, *fillColor*=`'black'`, *lineColor*=`'white'`, *backgroundColor*=`'white'`)

Grafica el fractal.

Parameters

`n`: numero de termino de la sucesion
(type=int)

`mode`: modo imagen(RGB, 1, etc)
(type=string)

`fillColor`: color de relleno
(type=RGB or string)

`lineColor`: color de linea
(type=RGB or string)

`backgroundColor`: color de fondo
(type=RGB or string)

Return Value

imagen del fractal.
(type=Image)

6 Module FractalZE.koch

6.1 Variables

Name	Description
___package___	**Value:** `'FractalZE'`

6.2 Class Koch

FractalZE.fractal.Fractal
└
FractalZE.koch.Koch

Known Subclasses: FractalZE.snowflake.Snowflake

Fractal de la curva de Koch. Cosntruccion:

1. Se parte de un segmento.
2. Se divide el segmento en 3 partes iguales.
3. Se construye un triangulo equilatero sobre el segmento central.
4. Se descarta el segmento central(la base del triangulo).
5. Se repiten los pasos para cada segmento las veces que se quiera.

6.2.1 Methods

___init___(*self*, *l*=1)

Parameters

`l`: longitud del segmento maximo(cuando n=0). Default 1

(type=float)

countSegments(*self*, *n*)

Devuelve la cantidad de segmentos del termino n.

Parameters

`n`: numero de termino de la sucesion

(type=int)

Return Value

cantidad de segmentos.

(type=int)

getHeight(*self*, *n*)

Devuelve la altura del fractal.

Return Value

altura del fractal

(type=float)

getWidth(*self*, *n*=0)

Devuelve el ancho del fractal.

Return Value

ancho del fractal

(type=float)

lindenmayer(*self*, *n*)

Devuelve la curva de koch expresada en L-System

Parameters

`n:` numero de termino de la sucesion

(type=int)

Return Value

curva de koch expresada en L-System.

(type=string

Algoritmo en L-System.

- *variables: _*
- *constantes: +-*
- *axioma: _*
- *reglas:*
  1. *_ -> _+_-_+_*

Interpretacion.

1. *Un _(guion bajo) significa dibujar un segmento*
2. *Un +(mas) significa girar en sentido antihorario 60 grados*
1. *Un -(menos) significa girar en sentido horario 120 grados*

)

segmentLength(*self*, *n*)

Devuelve la longitud de un segmento del termino n.

Parameters

`n`: numero de termino de la sucesion

(type=int)

Return Value

longitud de un segmento.

(type=float)

totalArea(*self*, *n*)

Devuelve el area total debajo de la curva.

Parameters

`n`: numero de termino de la sucesion

(type=int)

Return Value

area total debajo de la curva.

(type=float)

totalLength(*self*, *n*)

Devuelve la suma de la longitud de todos los segmentos.

Parameters

`n`: numero de termino de la sucesion

(type=int)

Return Value

longitud total.

(type=float)

Inherited from FractalZE.fractal.Fractal(Section 5.2)

graph()

7 Module FractalZE.pascal

7.1 Variables

Name	Description
___package___	**Value:** `'FractalZE'`

7.2 Class Pascal

FractalZE.fractal.Fractal
└
FractalZE.pascal.Pascal

Fractal del triángulo de Pascal.

1. Cada fila tiene un boque mas que la anterior.
2. La primer fila tiene un solo bloque que vale 1.
3. Cada fila comienza y termina con un bloque que vale uno.
4. Cada bloque contiene la suma de los valores de los dos bloques superiores.

7.2.1 Methods

___init___(*self*, *blockWidth*=`10.0`, *blockHeight*=`10.0`)

Parameters

`blockWidth:` ancho del bloque.
(type=float)

`blockHeight:` alto del bloque
(type=float)

getWidth(*self*, *n*=`0`)

Devuelve el ancho del fractal.

Return Value

ancho del fractal
(type=float)

getHeight(*self*, *n*=0)

Devuelve la altura del fractal.

Return Value

alto del fractal

(type=float)

Inherited from FractalZE.fractal.Fractal(Section 5.2)

graph()

7.2.2 Instance Variables

Name	Description
fn	yellow

8 Module FractalZE.quintoCantor

8.1 Variables

Name	Description
___package___	**Value:** `'FractalZE'`

8.2 Class QuintoCantor

FractalZE.fractal.Fractal
└ FractalZE.cantor.Cantor
└ **FractalZE.quintoCantor.QuintoCantor**

Fractal de cantor que divide los segmentos en 5 y descarta el del medio.

8.2.1 Methods

segmentLength(*self*, *n*)

Devuelve la longitud de un segmento del termino n.

Parameters

`n`: numero de termino de la sucesion

(type=int)

Return Value

longitud de un segmento.

(type=float)

Overrides: FractalZE.cantor.Cantor.segmentLength

lindenmayer(*self*)

Devuelve el fractal de cantor expresado en L-System.

Parameters

`n`: termino de la sucesion

Return Value

termino expresado en L-system.

(type=string

Algoritmo en L-System.

- *variables: _ (space)*
- *constantes:*
- *axioma: _*
- *reglas:*
  1. *_ -> _ _*
  2. *(space) -> (space)(space)(space)*

Interpretacion.

1. *Un _(guion bajo) significa dibujar un segmento*
2. *Un (space) significa avanzar la longitud de un segmento sin dibujar*

)

Overrides: FractalZE.cantor.Cantor.lindenmayer extit(inherited documentation)

Inherited from FractalZE.cantor.Cantor(Section 3.2)

___init___(), countSegments(), getHeight(), getWidth(), totalLength()

Inherited from FractalZE.fractal.Fractal(Section 5.2)

graph()

9 Module FractalZE.sierpinsky

9.1 Variables

Name	Description
___package___	**Value:** `'FractalZE'`

9.2 Class Sierpinsky

FractalZE.fractal.Fractal
└
FractalZE.sierpinsky.Sierpinsky

Known Subclasses: FractalZE.sierpinskyRectangular.SierpinskyRectangular

Fractal o Triangulo de Sierpinsky. Construccion:

1. Se parte de un triangulo equilatero.
2. Se divide cada lado en dos obteniendose 3 puntos.
3. Al unir los 3 puntos, el triangulo queda dividido en 4 triangulos.
4. Se descarta el del medio, es decir el que esta invertido.
5. Se repite esto sucesivamente con cada triangulo.

9.2.1 Methods

___init___(*self*, *l*=`1`)

Parameters

`l`: longitud del lado maximo(cuando n=0).default = 1.

(type=float)

countTriangles(*self*, *n*)

Devuelve la cantidad de triangulos del termino n.

Parameters

`n`: numero de termino de la sucesion

(type=int)

Return Value

cantidad de triangulos.

(type=int)

triangleArea(*self*, *n*)

Devuelve el area ocupada por un solo triangulo del termino n.

Parameters

`n`: numero de termino de la sucesion

(type=int)

Return Value

area ocupada por un triangulo.

(type=float)

totalArea(*self*, *n*)

Devuelve el area total.

Parameters

`n`: numero de termino de la sucesion

(type=int)

Return Value

area total.

(type=float)

triangulePerimeter(*self*, *n*)

Devuelve el perimetro de un triangulo del termino n.

Parameters

`n`: numero de termino de la sucesion

(type=int)

Return Value

perimetro de un triangulo.

(type=float)

totalPerimeter(*self*, *n*)

Devuelve la suma de los perimetros de todos los triangulos del termino n.

Parameters

`n`: numero de termino de la sucesion

(type=int)

Return Value

perimetro total.

(type=float)

triangleHeight(*self*, *n*)

Devuelve la altura de un triangulo del termino n.

Parameters

`n`: numero de termino de la sucesion

(type=int)

Return Value

altura de un triangulo.

(type=float)

triangleWidth(*self*, *n*)

Devuelve el ancho de un triangulo del termino n.

Parameters

`n`: numero de termino de la sucesion

(type=int)

Return Value

ancho de un triangulo.

(type=float)

getWidth(*self*, *n*=`0`)

Devuelve el ancho del fractal. Siempre es igual a l

Return Value

ancho del fractal

(type=float)

getHeight(*self*, *n*=`0`)

Devuelve la altura del fractal.

Return Value

ancho del fractal

(type=float)

Inherited from FractalZE.fractal.Fractal(Section 5.2)

graph()

10 Module FractalZE.sierpinskyRectangular

10.1 Variables

Name	Description
___package___	**Value:** `'FractalZE'`

10.2 Class SierpinskyRectangular

FractalZE.fractal.Fractal
FractalZE.sierpinsky.Sierpinsky
FractalZE.sierpinskyRectangular.SierpinskyRectangular

Sierpisnky rectangulo isosceles.

10.2.1 Methods

triangulePerimeter(*self*, *n*)

Devuelve el perimetro de un triangulo del termino n.

Parameters

`n`: numero de termino de la sucesion

(type=int)

Return Value

perimetro de un triangulo.

(type=float)

Overrides: FractalZE.sierpinsky.Sierpinsky.triangulePerimeter

triangleHeight(*self*, *n*)

Devuelve la altura de un triangulo del termino n.

Parameters

`n:` numero de termino de la sucesion

(type=int)

Return Value

altura de un triangulo.

(type=float)

Overrides: FractalZE.sierpinsky.Sierpinsky.triangleHeight

Inherited from FractalZE.sierpinsky.Sierpinsky(Section 9.2)

___init___(), countTriangles(), getHeight(), getWidth(), totalArea(), totalPerimeter(), triangleArea(), triangleWidth()

Inherited from FractalZE.fractal.Fractal(Section 5.2)

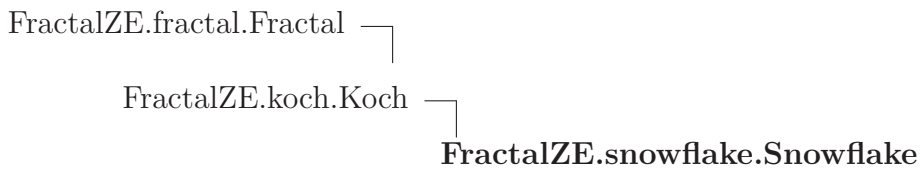
graph()

11 Module FractalZE.snowflake

11.1 Variables

Name	Description
___package___	**Value:** `'FractalZE'`

11.2 Class Snowflake



Fractal de la bola de nieve. Cosntruccion:

1. Se parte de un triangulo equilátero.
2. Se divide cada segmento en 3 partes iguales.
3. Se construye un triangulo equilatero sobre cada mitad de cada segmento.
4. Se descartan los segmentos centrales(la base de los triangulo creados).
5. Se repiten los pasos para cada segmento las veces que se quiera.

11.2.1 Methods

countSegments(*self*, *n*)

Devuelve la cantidad de segmentos del termino n.

Parameters

`n`: numero de termino de la sucesion

(type=int)

Return Value

cantidad de segmentos.

(type=int)

Overrides: FractalZE.koch.Koch.countSegments

totalArea(*self*, *n*)

Devuelve el area total debajo de la curva.

Parameters

`n:` numero de termino de la sucesion

(type=int)

Return Value

area total debajo de la curva.

(type=float)

Overrides: FractalZE.koch.Koch.totalArea

getWidth(*self*, *n*=`0`)

Devuelve el ancho del fractal.

Return Value

ancho del fractal

(type=float)

Overrides: FractalZE.koch.Koch.getWidth

getHeight(*self*, *n*=`0`)

Devuelve la altura del fractal.

Return Value

altura del fractal

(type=float)

Overrides: FractalZE.koch.Koch.getHeight

Inherited from FractalZE.koch.Koch(Section 6.2)

___init___(), lindenmayer(), segmentLength(), totalLength()

Inherited from FractalZE.fractal.Fractal(Section 5.2)

graph()

Index