# REPRESENTACIÓN DE FRACTALES CON L - SISTEMAS

WILMAR EDGARDO BEDOYA GONZÁLEZ
Código: 2009240007
C.C. 14325518

Monografía asociada al estudio de un tema específico de interés personal para el maestro en formación y presentada como requisito parcial para obtener el título de Licenciado en Matemáticas de la Universidad Pedagógica Nacional.

Asesor:
JORGE EDGAR PÁEZ ORTEGÓN

UNIVERSIDAD PEDAGÓGICA NACIONAL.
FACULTAD DE CIENCIA Y TECNOLOGÍA.
DEPARTAMENTO DE MATEMÁTICAS.
BOGOTA, D.C. - COLOMBIA.
2016

# RESUMEN ANALÍTICO EN EDUCACIÓN - RAE

| 1.Información General. | |
|---|---|
| **Tipo de documento:** | Trabajo de grado. |
| **Acceso al documento:** | Universidad Pedagógica Nacional. Biblioteca Central. |
| **Título del documento:** | REPRESENTACIÓN DE FRACTALES CON L – SISTEMAS. |
| **Autor:** | Bedoya González, Wilmar Edgardo. |
| **Director:** | Páez Ortegón, Jorge Edgar. |
| **Publicación:** | Bogotá, D.C. Universidad Pedagógica Nacional, 2016. 114 p. |
| **Unidad Patrocinante:** | Universidad Pedagógica Nacional. |
| **Palabras Claves:** | Geometría Fractal, Fractales, Algoritmo, Iteración, Lenguajes de Programación, L-Sistemas, Representación on-line. |

| 2.Descripción. |
|---|
| El siguiente documento presenta un estudio de los diferentes algoritmos usados para construir algunos fractales clásicos, pasando por el lenguaje natural y matemático, luego una estructura codificada como los L-Sistemas, y finalmente un lenguaje estructurado (lenguaje de programación) para la construcción de los códigos que se representan mediante el uso del programa Turtle Graphics Renderer que es un programa gratuito y en línea para la generación de L-Sistemas creado por Kevin Roast en el año 2012. Se presenta una descripción general del programa así como las instrucciones básicas para realizar representaciones gráficas de algunos fractales clásicos y esquemas simples de plantas. |

## 3.Fuentes.

Este documento se fundamenta en 5 libros, 2 artículos de eventos de educación matemática, 6 páginas web, 2 tesis de pregrado. El trabajo se centro en los siguientes documentos:


Campos, D. (2011). Introducción a los Sistemas de Lindenmayer: fractales, autómatas celulares y aplicaciones. Recuperado el 30 de Septiembre de 2014, de Centro de Investigación y de Estudios Avanzados del IPN, Departamento de Computación del CINVESTAV: http://delta.cs.cinvestav.mx/~mcintosh/cellularautomata/Summer_Research_files/Arti_Ver_Inv_2011_DCM.pdf

Estrada, W. (2004). Geometría Fractal: conceptos y procedimientos para la construcción de fractales. Bogotá, D.C., Colombia: Cooperativa Editorial Magisterio.

Luque, B., & Agea, A. (s.f.). Fractales en la red, Los L-Sistemas. Recuperado el 17 de Noviembre de 2015, de Cursos fractales en Departamento Matemática Aplicada y Estadística: http://www.dmae.upm.es/cursofractales/capitulo2/frames.htm

Muñoz, I. (2007). Uso de los Lenguajes de Programación en el Estudio de los Fractales. Trabajo de grado para obtener el título de Licenciada en Matemáticas, Departamento de Matemáticas, Universidad Pedagógica Nacional, Bogotá, D.C., Colombia

Rivero, J. A. (2014). Sistemas de Lindenmayer con L- Parser. (U. P. Madrid, Ed.) Recuperado el 17 de Noviembre de 2015, de Cursos fractales en Departamento Matemática Aplicada y Estadística: http://www.dmae.upm.es/cursofractales/capitulo2/lparse/lparser.htm

Roast, K. (2012). Turtle Graphics Renderer on- line. Recuperado el 17 de Noviembre de 2015, de HTML 5 + Java Script Canvas Demo: http://kevs3d.co.uk/dev/lsystems/#

## 4.Contenidos.

En este trabajo se presentan cuatro capítulos para realizar el estudio de algunas estructuras fractales, el primero se refiere al marco histórico, en el cual se muestran algunas ideas sobre el surgimiento de los Sistemas de Lindenmayer como una rama de la geometría fractal; en el segundo capítulo se trabaja el marco conceptual, en el cual se presentan algunas definiciones de fractal y se trabaja un algoritmo usando el lenguaje natural y matemático; en el tercer capítulo se trabajan los Sistemas de Lindenmayer, se desarrollan los algoritmos de algunos fractales clásicos como la Curva de Koch, el Copo de Nieve de Koch y el Triángulo de Sierpinski, se presentan esquemas simples que simulan los procesos de crecimiento de las plantas y todo esto se representa mediante el uso del programa Turtle Graphics Renderer on-line, creado por Kevin Roast en el año 2012, para resaltar el uso de la tecnología en el estudio de los fractales. Por último, se presentan algunas de las posibles aplicaciones de los L-Sistemas.

## 5.Metodología.

Este trabajo se realizó en cinco fases: búsqueda y consulta, caracterización del objeto de estudio, selección del programa, representación de L-Sistemas, elaboración del documento y ajustes.

En la primera fase, la de búsqueda y consulta, se realizó una revisión bibliográfica de diferentes documentos, artículos y páginas web relacionados con los L-Sistemas y la geometría fractal con el fin de analizar y seleccionar aquellos escritos que contenían información relevante relacionada con el objeto de estudio. En la segunda fase, realizada la revisión bibliográfica, se procedió con la descripción de los sistemas de Lindenmayer, sus características, clasificación y algunas de sus posibles aplicaciones. En la tercera fase, se realizó una revisión de diferentes programas que simulan los procesos de crecimiento de las plantas para seleccionar uno que brindara cierta flexibilidad al introducir las cadenas de caracteres diseñadas para las simulaciones.

Para continuar, en la cuarta fase se presentan las instrucciones básicas para el manejo del programa y se realizan representaciones gráficas de algunos fractales clásicos y esquemas simples de plantas, resaltando el uso de la tecnología en el estudio de la geometría fractal. Por último, se elaboró un documento que contiene el estudio descriptivo sobre los sistemas de Lindenmayer y algunas de las posibles aplicaciones, como una propuesta para los maestros en formación.

## 6.Conclusiones.

Con la realización de este trabajo se pudo observar que existen programas menos conocidos para la generación de fractales y la simulación de procesos de crecimiento; el manejo de estos elementos tecnológicos son una herramienta para el estudio de los algoritmos que los generan ya que antes solo era posible construir tales objetos utilizando el dibujo a mano (lápiz y papel) y ahora se pueden generar diferentes aproximaciones usando las herramientas computacionales en donde cada lenguaje pone en evidencia gran cantidad de objetos, elementos y reglas de construcción de dichos objetos. Con la ayuda del programa Turtle Graphics Renderer on-line es posible generar estudios de diferentes objetos de la geometría fractal y mostrar algunas representaciones gráficas, en particular el programa permitió la visualización de algunos fractales clásicos como la Curva de Koch, el Copo de Nieve de Koch y el Triángulo de Sierpinski, así como la simulación de los procesos de crecimiento de las plantas.

| | |
|---|---|
| **Elaborado por:** | Wilmar Edgardo Bedoya González. |
| **Revisado por:** | Jorge Edgar Páez Ortegón. |

| | | | |
|---|---|---|---|
| **Fecha de elaboración del Resumen:** | 23 | 01 | 2016 |

# TABLA DE CONTENIDO

# ÍNDICE DE IMÁGENES

# ÍNDICE DE TABLAS

# INTRODUCCIÓN.

La geometría Euclideana es la rama de la Matemática encargada de estudiar las propiedades de elementos tales como puntos, rectas, polígonos, planos y algunos objetos tridimensionales. Además, se ocupa del análisis y la construcción bajo algunos parámetros de figuras y formas bastante conocidas, que podemos observar en la cotidianidad y de las cuales se puede estudiar su estructura y forma, entre otras características.

Sin embargo, como ya es sabido, existen muchas formas encontradas en la naturaleza, como las nubes, montañas, algunos comportamientos de los órganos humanos, plantas y sin número de sistemas presentes en fenómenos naturales, para las cuales la geometría tradicional no representa un buen modelo para describir su comportamiento. Es allí cuando la geometría fractal aparece y representa una herramienta que satisface la descripción de aquellas estructuras a las cuales la geometría tradicional no llega.

En este trabajo se presentan cuatro capítulos para realizar el estudio de algunas estructuras fractales, el primero se refiere al marco histórico, en el cual se muestran algunas ideas sobre el surgimiento de los Sistemas de Lindenmayer como una rama de la geometría fractal; en el segundo capítulo se trabaja el marco conceptual, en el cual se presentan algunas definiciones de fractal y se trabaja un algoritmo usando el lenguaje natural y matemático; en el tercer capítulo se trabajan los Sistemas de Lindenmayer, se desarrollan los algoritmos de algunos fractales clásicos como la Curva de Koch, el Copo de Nieve de Koch y el Triángulo de Sierpinski, se muestran esquemas simples que simulan los procesos de crecimiento de las plantas y todo esto se representa mediante el uso del programa

Turtle Graphics Renderer on-line, creado por Kevin Roast en el año 2012, (Roast, 2012) para resaltar el uso de la tecnología en el estudio de los fractales. Por último, se presentan algunas de las posibles aplicaciones de los L-Sistemas.

Dado que uno de los objetivos de este documento es representar simulaciones de los procesos de crecimiento de las plantas, se resalta el uso del programa Turtle Graphics Renderer on-line, (Roast, 2012) como un elemento potente que permite el estudio de los fractales, más allá del simple dibujo, ya que es posible generar conocimiento y manejo desde el estudio del algoritmo, de todos los aspectos presentes en la representación gráfica, puesto que como lo señala Estrada (2004), en su libro: "*Geometría Fractal: conceptos y procedimientos para la construcción de fractales",* "para lograr la efectividad de los algoritmos utilizados y una construcción adecuada del objeto matemático, es necesario conocer las herramientas de programación, ya que cada software tiene definida una *sintaxis* (pero la lógica es la misma para todos) en cuanto conjunto de símbolos y reglas para generar combinaciones válidas entre ellos, a las que se les hace una asignación de significados constituyéndose de esta manera en una *semántica"*.

# JUSTIFICACIÓN.

El propósito de este trabajo es brindar a los estudiantes de la licenciatura en matemáticas, de cualquier semestre, que tengan algunas bases en programación, que sean curiosos y deseen adentrarse en el estudio de esta rama de las matemáticas, algunos algoritmos que muestran los procesos, en especial la iteración que es el corazón mismo de los fractales, para que éstos sirvan como una herramienta de entendimiento y estimulación, además de darles una visión diferente de las matemáticas, no como el conjunto de axiomas y teoremas obtenidos de forma deductiva, sino como un campo en el cual el objeto de estudio puede ser creado y recreado por medio de instrumentos tecnológicos (computadoras, calculadoras y celulares) logrando una interacción de los mismos y de sus diferentes representaciones.

Al realizar el estudio de los L-Sistemas, un aspecto fundamental para su entendimiento es la generación de las imágenes que los representan, éstas imágenes muestran gran cantidad de información (como la forma, los objetos iniciales, la cantidad de objetos después de realizar un proceso, etc.), con respecto al comportamiento de la estructura fractal sobre la cual se está trabajando, es así que la visualización del mismo se convierte en una herramienta indispensable para su estudio.

Para tal fin, se emplea el programa Turtle Graphics Renderer, que es un software gratuito y en línea para la generación de L-Sistemas en un espacio bidimensional creado por Kevin Roast. (Roast, 2012). Esta herramienta es muy flexible ya que permite cambiar numerosos aspectos, tales como el número de iteraciones, el ángulo, las constantes, el axioma definido y hasta cinco reglas de reescritura

diferentes. Además, su creador incluye también en la página web una lista de ejemplos donde el usuario puede interactuar con el algoritmo para observar por ejemplo cómo los fractales pueden relacionarse con la naturaleza.

Así mismo, el desarrollo de este trabajo amplia el campo de estudio de los objetos fractales desde el punto de vista del uso de la tecnología y el modo de trabajar sobre una pantalla, dado que casi siempre el software en matemáticas es usado en la producción de cálculos exclusivamente fuera del alcance de procesos y concepciones de un objeto definido, en este caso los fractales.

# 1. OBJETIVOS DEL TRABAJO DE GRADO.

- ✓ Describir los sistemas de Lindenmayer, sus características, clasificación y algunas de sus posibles aplicaciones.

- ✓ Emplear el programa Turtle Graphics Renderer on-line para ejecutar los algoritmos que permiten la visualización de algunas estructuras fractales mediante la aplicación de la idea básica de los L-sistemas.

- ✓ Representar simulaciones de los procesos de crecimiento de las plantas mediante el uso del programa Turtle Graphics Renderer on-line, en el cual se representan gráficamente fractales y es posible ver las diferentes iteraciones del objeto fractal.

# 2. MARCO HISTÓRICO.

En este capítulo se presentan algunas ideas sobre el surgimiento de los Sistemas de Lindenmayer como una rama de la geometría fractal, atendiendo a la necesidad de explicar fenómenos de la naturaleza a los cuales no podía llegar la geometría Euclidea. Por otro lado, se describen objetos - matemáticos - que rompieron las estructuras tradicionales, siendo necesario ampliar el estudio de las matemáticas.

El surgimiento de algunas ideas geométricas se remonta a varios miles de años atrás. La geometría tuvo sus primeros orígenes muy seguramente al mismo tiempo que la humanidad empezó sus procesos de pensamiento, y los conocimientos iniciales datan probablemente del tiempo en el que el ser humano se estableció en una región y dejó su carácter nómada. (Monroy, 2002)

Según Suárez (2010), los conceptos más antiguos son consecuencia de actividades prácticas como: la clasificación de objetos según su forma, la predicción de fenómenos y la estimación de algunas distancias, entre otras; así los primeros hombres llegaron a formas geométricas a partir de la abstracción de formas que observaban en la naturaleza y las necesidades de su entorno.

Uno de los grandes matemáticos de todos los tiempos: Euclides, fue el artífice de la sistematización y organización de gran cantidad de conocimientos logrados por la humanidad desde tiempos inmemorables, relacionados con los estudios sobre la medición de la tierra. Su trabajo quedó plasmado en trece libros llamados *Los Elementos*, donde desarrolló los fundamentos de la geometría elemental y algunos conceptos de la aritmética, legado que le valió el mérito de llamarse el "Padre de

la Geometría", aunque su trabajo fue más de tipo organizacional que de un verdadero carácter teórico, ya que muchas de las definiciones, axiomas y postulados ya se conocían desde mucho antes.

Los aportes de la Geometría Euclidea llegaron a todas las áreas del conocimiento matemático, generando gran cantidad de teorías en el cálculo, álgebra y análisis, entre otras, y su trascendencia es tal que ha llegado hasta la presente época ya que estos resultados son nuevamente usados y aún representan un campo de estudio y es continuamente analizada por miles de personas dentro y fuera de las áreas de la matemática.

En general, el modelo Euclideo pretende explicar las formas de la naturaleza tomando elementos como rectas, círculos, polígonos, poliedros, entre otros, los cuales efectivamente permiten explicar gran cantidad de fenómenos y estructuras presentes en el entorno, y con los cuales el hombre ha creado el mundo que lo rodea. Pero si se piensa en una cantidad de objetos palpables, cercanos a la cotidianidad, se pueden observar más que figuras planas o tridimensionales, es posible obtener elementos tan complejos y ausentes de una descripción Euclidea, que se deben buscar otros modelos geométricos que acerquen más la razón a los objetos de la naturaleza. (Monroy, 2002)

## 2.1. L-Sistemas y la geometría fractal.

El mundo está constituido por montañas, costas, mares, nubes, plantas, animales, entre otros; sin duda alguna, se vive en un mundo donde se encuentra toda clase de formas y estructuras; formas que se alejan de las representaciones simplificadas que el hombre maneja comúnmente y que por sus limitaciones no es capaz de reproducir, estos objetos se acercan mucho a la irregularidad y el caos, dentro de estructuras que asombran a los matemáticos y estudiosos de los fenómenos de la naturaleza por su gran armonía y simplicidad.

Los orígenes de la geometría fractal se remontan a finales del siglo XIX y comienzos del siglo XX con la aparición de conjuntos de puntos, en el plano o en el espacio, que poseían características bastante especiales como por ejemplo la longitud del conjunto obtenido. (Muñoz, 2007)

Matemáticos como Karkl Weierstrass (1815 – 1897), Georg Philipp Cantor (1845 – 1918), Guiseppe Peano (1858 – 1932), Niels Fabian Helge von Koch (1870 – 1924) y Waclaw Sierpinski (1882 – 1969), entre otros, trabajaron sobre este tipo de "monstruos", que hoy día son conocidos con el nombre de sus creadores, el nacimiento de estos objetos geométricos amplió el campo de estudio de las matemáticas y surgió la necesidad de explorar las estructuras geométricas, aritméticas y analíticas ya existentes.

El estudio de estos "monstruos" como fueron catalogados en un principio, dio origen a un campo del análisis matemático: *la teoría geométrica de la medida*, que tuvo su punto de partida con el concepto de medida y dimensión trabajada por Félix Hausdorff (1868 – 1942), - así como su descripción de la noción popularizada de fractal - quien impulsó y se convirtió en la base para los estudios de Abram Samoilovitch Besicovitch (1891 - 1970), quien le dio estructura a muchas de las propiedades geométricas de los conjuntos que él llamó *irregulares*, esto desde las matemáticas formales. (Suárez, 2010)

Según Muñoz (2007), Gastón Maurice Julia (1893 – 1978) fue uno de los padres de la Teoría de los Sistemas Dinámicos modernos, es recordado por el llamado Conjunto de Julia. En 1918 publicó un hermoso libro, *"Mémoire sur l'itération des fonctions rationnelles"*, Journal de Math. Pure et Appl. 8" (1918), concerniente a la iteración de una función racional $f$. Sus descubrimientos le valieron ganar el *"Grand Prix de l'Académie des Sciences"*. Sin embargo, desde su fama en 1920, su trabajo fue esencialmente olvidado, hasta que Benoit B. Mandelbrot (1924 – 2010) lo hizo resurgir en 1970 con sus experimentos computacionales. Mandelbrot

mostró cómo los fractales pueden aparecer en muchos ámbitos diferentes, tanto en matemáticas como en otros aspectos de la naturaleza. Su obra fue puesta de forma elaborada por primera vez en su libro *"Les objets fractals, form, hasard et dimensión"* – (Los objetos fractales, forma, azar y dimensión) de 1975 y de forma más completa en *"The fractal geometry of nature"* – (La geometría fractal de la naturaleza) de 1982.

El término fractal, fue usado por primera vez por Mandelbrot en 1967 (proviene del latín *fractus* que significa irregular) para llamar a los objetos geométricos con estructura irregular, presentes en muchos de los comportamientos y formas de la naturaleza, por ejemplo: la frontera de separación de dos medios, los distintos procesos de ramificación y la formación de porosidad. Incursionó en el área de las matemáticas gracias a su interés por la representación gráfica de este tipo de conjuntos, y su estudio se centró en el uso de representaciones que obtenía en sus experimentos con el computador, gracias a su trabajo en el centro Thomas Watson de la compañía IBM ya que disponía de una gran cantidad de recursos informáticos, aprovechando para llevar a la práctica tareas sencillas como la iteración y el dibujo de estructuras muy complejas. Esto significó un gran avance ya que en épocas anteriores sólo se habían podido realizar experimentos mentales. Así, el computador se convirtió en lo que el lápiz y el papel fueron para la geometría clásica. Con el paso del tiempo, se aprovechaban recursos como la versatilidad gráfica, la ampliación o reducción (zoom), profundidad, alta gama de colores, la resolución, entre otros; para permitir los procesos iterativos que dan origen a las estructuras fractales. (Mandelbrot, 1982)

Un año después, en 1968, Aristid Lindenmayer (1925 - 1989), biólogo y botánico húngaro, profesor de la Universidad de Utrech, desarrolló un tipo de lenguaje formal para modelar el comportamiento de las células de las plantas. Los denominados L-Sistemas en honor a su creador, fueron diseñados originalmente para proporcionar una descripción formal del desarrollo de organismos

pluricelulares simples y para ilustrar las relaciones de vecindad entre las células vegetales. Más tarde este sistema se ampliaría para describir estructuras de ramificación compleja. (Iglesias & Gálvez, 2011)

Como lo mencionan Luque & Agea (s.f.), Lindenmayer pretendía modelar cómo, partiendo de una célula inicial, se genera un filamento celular cuyo orden en la posición de las células es conocido. En concreto modelaba la bacteria azul-verde "Anabaena catenula". Se enfrentaba a un sistema celular con dos estados citológicos A y B, relativos al tamaño y disponibilidad para la división, que crecía de la siguiente manera:

| Células que realizan la división: | Resultado de la división: |
|---|---|
| A (1 célula) | AB |
| A B (2 células) | AB BA |
| A BBA (4 células) | AB BABAAB |
| A BBA BAA B (8 células) | AB BABAAB BAABAB BA |
| …. | …. |

Tabla 1: Esquema de división celular.

Pero, ¿cuáles eran las reglas de crecimiento del filamento celular?. Si suponemos que una célula en estado A, al dividirse, generará una célula en estado A y una célula en estado B, y lo representamos mediante la terminología A = AB. Así mismo, suponemos que una célula en estado B, al dividirse generará el siguiente resultado B = BA. De esta forma se obtiene un crecimiento como el descrito en la tabla anterior. A la condición inicial (o axioma) y al conjunto de reglas de producción (o sustitución) se le denomina L-Sistema. (Luque & Agea, s.f.)

Los L-Sistemas también son conocidos como "Parallel String-Rewrite Systems". El precursor de estos sistemas fue, en cierto modo, Noam Chomsky, creador de las gramáticas de Chomsky para el estudio teórico de los lenguajes naturales. Las gramáticas de Chomsky son un conjunto de símbolos y reglas de producción que parten de una "frase" inicial; las reglas de producción se aplican secuencialmente

a "la cola" de una en una. De esta forma, Lindenmayer pensó en utilizar este método para desarrollar una axiomática del proceso de desarrollo de organismos pluricelulares puesto que las células se regeneran constantemente y de forma independiente. (Luque & Agea, s.f.)

Actualmente muchos matemáticos continúan con el estudio de este tipo de estructuras configurando la teoría iniciada por grandes personajes que ya hacen parte de la historia, además la exploración mediante nuevas herramientas conceptuales y tecnológicas de los objetos matemáticos ha permitido avanzar en el desarrollo de la teoría geométrica que subyace, sin desconocer el gran interés por su exploración y estudio por parte de los estudiantes y curiosos de las matemáticas, ligado en gran parte al inmenso auge que los computadores han alcanzado en los últimos años.

# 3. MARCO CONCEPTUAL.

Dado que uno de los objetivos de este trabajo es la descripción de los sistemas de Lindenmayer, primero se parte de la definición de fractal, sus elementos y la relación con un lenguaje codificado como los L-Sistemas, este lenguaje suministra herramientas para construir los algoritmos que luego se llevarán a codificar en el programa seleccionado. Por lo tanto, inicialmente se presentan algunas definiciones de fractal de acuerdo con el enfoque que se pretende dar en este trabajo al estudio de tales objetos, relacionando los algoritmos y procesos de construcción; posteriormente se presenta la clasificación de los fractales (lineales o no-lineales) y otros conceptos necesarios para la comprensión del tema.

La geometría fractal es una parte de las matemáticas la cual puede ser encuadrada entre la teoría geométrica de la medida, que se manifiesta con la aparición de conjuntos geométricos de propiedades paradójicas. En estos conjuntos parece existir una discordancia entre su tamaño real y su configuración espacial como conjunto de puntos, este problema fue una de las hipótesis iniciales para establecer una medida adecuada de su tamaño, en una primera instancia, y por otra parte el estudio de las propiedades geométricas. (Muñoz, 2007)

Este tipo de objetos geométricos, los cuales inundaron el mundo científico hace más de una década y que inicialmente fueron considerados "monstruos" por los matemáticos, han venido a establecer un conjunto de reglas diferentes para la descripción de la naturaleza, su lenguaje alcanzó muchos campos del saber científico y de las artes, tanto que aún sigue sorprendiendo la gama de posibilidades y aplicaciones que se les puede atribuir. La geometría fractal, proporciona elementos de trabajo a físicos, químicos, biólogos, economistas, entre

otros, pues ha permitido la reformulación y la resolución de algunos problemas de forma más simple.

A continuación se presentan definiciones de fractal, unas más formales que otras, con las cuales los autores pretenden dar a entender ciertas realidades matemáticas mostrando algunos rasgos comunes entre los objetos, todas estas son aplicables a los fractales y muestran características que los identifican, pero tal vez la mejor forma de definirlos sea señalar los procesos matemáticos que los generan.

### 3.1. ¿Qué es un fractal?

"Un fractal es el producto final que se origina a través de la iteración infinita de un proceso geométrico (algoritmo) bien especificado. Este proceso geométrico es muy elemental y determina perfectamente la estructura final del objeto, debido a la repetición infinita que se ha efectuado". (De Guzmán, 1993)

"Un fractal es una figura geométrica fragmentada, donde cada una de las partes conserva una relación de semejanza con la figura completa". (Estrada, 2004)

Actualmente hay en uso diferentes definiciones de fractal, pero según Rubiano (2011), "cada una de estas definiciones tiene como base dos nociones: *autosimilaridad* y *dimensión*. Ser auto similar significa que, cuando examinamos pequeñas porciones de objeto, la imagen que vemos no es más que una copia de nuestro objeto inicial. Una cualidad adicional la constituye la simplicidad de las funciones que se involucran para generar estructuras extraordinariamente complejas y hermosas".

Se puede pensar en los fractales como una curva en continuo desarrollo. Para ver un fractal, hay que verlo en movimiento, puesto que se desarrolla constantemente,

simulando fenómenos de crecimiento, de aquí que pueden crearse fractales para simular cualquier forma que uno pueda imaginar.

En otros términos, un fractal es una forma que empieza con un objeto (tal como un segmento, un punto, un triángulo, etc.) que es alterado constantemente por medio de la aplicación de una determinada regla (iteración). Ésta regla puede describirse por medio de una fórmula matemática o por medio de palabras, además, los fractales no están necesariamente limitados a una sola regla, sino que pueden estar formados por varias reglas.

Ahora, se tiene que desde el lenguaje del programador, la iteración es cada una de las repeticiones de las acciones contenidas en un bucle (Un bucle en un programa, es un grupo de instrucciones que se repiten hasta que se cumple una determinada condición) del programa.

De esta manera es posible afirmar que los fractales son conjuntos geométricos aparentemente complicados, pero en realidad resultan ser tales que para su descripción, construcción y exploración se requiere muy poca información de tal forma que se pueden construir muchos objetos fractales partiendo de reglas simples aplicadas a un sin número de objetos. A continuación se muestra una clasificación de los fractales presentada por Muñoz (2007), de acuerdo a su linealidad, así:

### 3.2. Fractales lineales.

Son aquellos que se generan a partir de elementos de la geometría tradicional y se construyen con un cambio en la variación de sus escalas. Esto implica que los fractales lineales son exactamente idénticos en todas sus escalas si se tiene en cuenta que esta característica se mantiene si el proceso se aplica infinitamente. Es decir, si vemos una parte específica muy pequeña de una forma fractal la

veremos igual o similar a la forma original del fractal, solamente que más pequeña. El siguiente es un ejemplo de este tipo de fractales:

**3.2.1. Curva de Koch:** Helge von Koch introdujo la curva que lleva su nombre en 1904 y es un ejemplo de una curva que no tiene tangente en ningún punto.

Para construir la curva de Koch se parte de un segmento de cualquier longitud, en particular:

$$K_0 = [0,1]$$

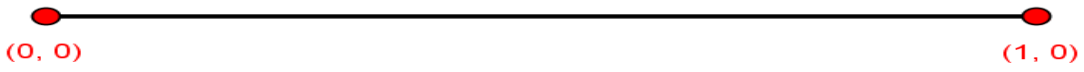



Imagen 1: Curva de Koch - Iteración 0.

que es el objeto inicial. Se divide dicho segmento en tres partes de igual medida y se consideran los segmentos de los extremos, cada uno de ellos de longitud $1/3$, el segmento central se sustituye por dos nuevos segmentos de longitud $1/3$ cada uno de ellos, los cuales formarían con el segmento retirado un triángulo equilátero. Así, se obtienen cuatro segmentos, que forman la curva $K_1$, a continuación los extremos de tales segmentos:

$$K_{11} = (0,0) \text{ y } \left(\frac{1}{3}, 0\right) \qquad K_{12} = \left(\frac{1}{3}, 0\right) \text{ y } \left(\frac{1}{2}, \frac{\sqrt{3}}{6}\right)$$

$$K_{13} = \left(\frac{1}{2}, \frac{\sqrt{3}}{6}\right) \text{ y } \left(\frac{2}{3}, 0\right) \qquad K_{14} = \left(\frac{2}{3}, 0\right) \text{ y } (1,0)$$

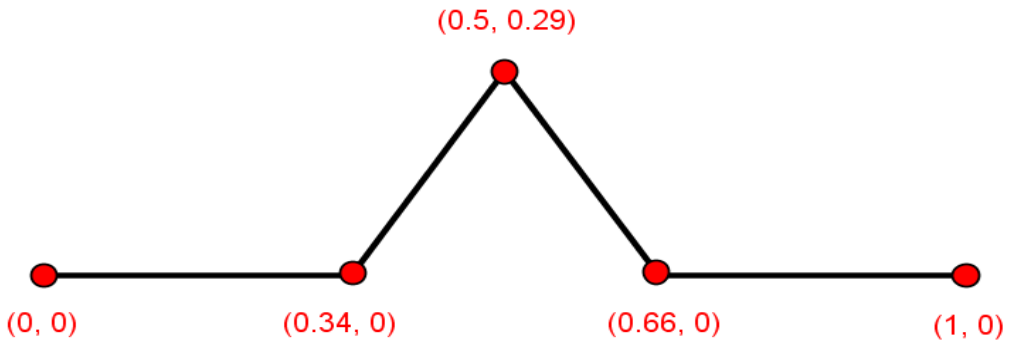



Imagen 2: Curva de Koch - Iteración 1.

Si se repite este proceso sobre cada uno de los segmentos de $K_1$ se obtiene otra curva $K_2$ formada por $4^2$ segmentos de longitud $3^{-2}$ cada uno de ellos, a continuación los extremos de tales segmentos:

$$K_{21} = (0,0) \text{ y } \left(\frac{1}{9}, 0\right)$$

$$K_{22} = \left(\frac{1}{9}, 0\right) \text{ y } \left(\frac{1}{6}, \frac{\sqrt{3}}{18}\right)$$

$$K_{23} = \left(\frac{1}{6}, \frac{\sqrt{3}}{18}\right) \text{ y } \left(\frac{2}{9}, 0\right)$$

$$K_{24} = \left(\frac{2}{9}, 0\right) \text{ y } \left(\frac{1}{3}, 0\right)$$

$$K_{25} = \left(\frac{1}{3}, 0\right) \text{ y } \left(\frac{4}{9}, \frac{\sqrt{3}}{9}\right)$$

$$K_{26} = \left(\frac{4}{9}, \frac{\sqrt{3}}{9}\right) \text{ y } \left(\frac{1}{9}, 0\right)$$

$$K_{27} = \left(\frac{1}{9}, 0\right) \text{ y } \left(\frac{7}{18}, \frac{\sqrt{3}}{18}\right)$$

$$K_{28} = \left(\frac{7}{18}, \frac{\sqrt{3}}{18}\right) \text{ y } \left(\frac{1}{2}, \frac{\sqrt{3}}{6}\right)$$

$$K_{29} = \left(\frac{1}{2}, \frac{\sqrt{3}}{6}\right) \text{ y } \left(\frac{11}{18}, \frac{\sqrt{3}}{18}\right)$$

$$K_{210} = \left(\frac{11}{18}, \frac{\sqrt{3}}{18}\right) \text{ y } \left(\frac{1}{2}, \frac{\sqrt{3}}{18}\right)$$

$$K_{211} = \left(\frac{1}{2}, \frac{\sqrt{3}}{18}\right) \text{ y } \left(\frac{5}{9}, \frac{\sqrt{3}}{9}\right)$$

$$K_{212} = \left(\frac{5}{9}, \frac{\sqrt{3}}{9}\right) \text{ y } \left(\frac{2}{3}, 0\right)$$

$$K_{213} = \left(\frac{2}{3}, 0\right) \text{ y } \left(\frac{7}{9}, 0\right)$$

$$K_{214} = \left(\frac{7}{9}, 0\right) \text{ y } \left(\frac{5}{6}, \frac{\sqrt{3}}{18}\right)$$

$$K_{215} = \left(\frac{5}{6}, \frac{\sqrt{3}}{18}\right) \text{ y } \left(\frac{8}{9}, 0\right)$$

$$K_{216} = \left(\frac{8}{9}, 0\right) \text{ y } (1,0)$$

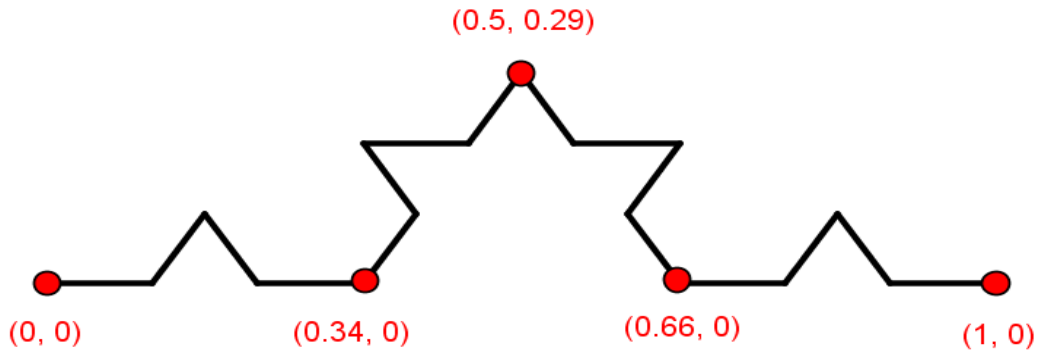



Imagen 3: Curva de Koch - Iteración 2.

En el siguiente paso del proceso se obtienen 64 segmentos de longitud $1/27$ cada uno. Si se continúa de esta forma, en la etapa i-ésima se habrá obtenido una nueva curva con $i = 1,2,3 \ldots$ formada por $4^i$ segmentos de longitud $3^{-i}$ cada uno de ellos. ($i$: corresponde al número de la iteración)

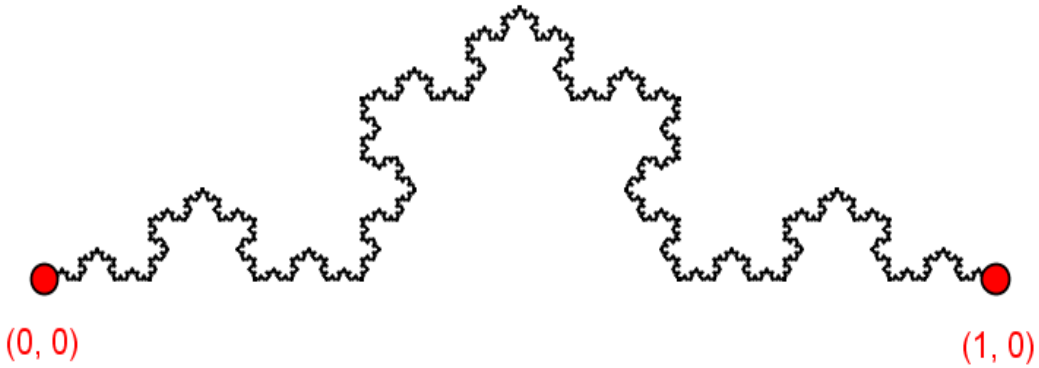



Imagen 4: Curva de Koch - Iteración 5.

La construcción de esta curva en particular será presentada más adelante mediante el uso de los L-Sistemas en el programa seleccionado para este trabajo.

### 3.3. Fractales no lineales.

Los fractales no lineales se generan creando distorsiones no lineales o complejas. Es decir, son fractales que presentan una estructura similar, pero no son exactamente iguales a su original ya que son conjuntos que se construyen a partir de la iteración de una expresión compleja (número complejo). Si vemos de

cerca una parte específica de un fractal no lineal se parecerá al original pero tendrá unas pequeñas variaciones. La mayoría de los objetos naturales y matemáticos se pueden describir con fractales no lineales.

### 3.4. Lenguajes de programación.

Dado que un L-Sistema es un conjunto de reglas y símbolos principalmente utilizados para modelar el proceso de crecimiento de las plantas a través de un lenguaje que intenta estar relativamente próximo al lenguaje humano o natural, se hace necesario detallar que la solución de problemas mediante el uso del computador nos ha llevado a desarrollar programas o aplicaciones que permitan describir de manera lógica y ordenada las instrucciones para brindarle salida a la situación.

La serie de pasos organizados que describe el proceso que se debe seguir, para dar solución a un problema específico se denomina algoritmo. Un algoritmo es un conjunto finito de instrucciones o pasos que sirven para ejecutar una tarea o resolver un problema obteniendo una solución, teniendo en cuenta que debe de ser definido, finito y preciso. Por preciso se entiende que cada paso a seguir tiene un orden; finito implica que tiene un determinado número de pasos, o sea, que tiene un fin; y definido, que si se sigue el mismo proceso más de una vez se llega al mismo resultado. La estructura básica de un algoritmo es:

- ✓ Inicio.
- ✓ Datos de entrada (operaciones básicas).
- ✓ Procesamiento de los datos.
- ✓ Datos de salida.
- ✓ Fin.

La construcción de estos programas debe ser realizada mediante un lenguaje utilizado para controlar el comportamiento de una máquina, particularmente el

computador. El lenguaje de programación permite a un programador especificar de manera precisa sobre qué datos una computadora debe operar, cómo deben ser almacenados y transmitidos los mismos y, qué acciones debe tomar bajo una variada gama de circunstancias.

Los procesadores usados en las computadoras son capaces de entender y actuar según lo indican los programas escritos en un lenguaje fijo llamado lenguaje de máquina. Todo programa escrito en otro lenguaje puede ser ejecutado de dos maneras:

- ✓ Mediante un programa que va adaptando las instrucciones conforme son encontradas. A este proceso se le denomina interpretar y a los programas que lo hacen se les conoce como intérpretes.

- ✓ Traduciendo este programa al programa equivalente escrito en lenguaje máquina. A este proceso se le denomina compilar y al traductor se le conoce como compilador.

# 4. LOS SISTEMAS DE LINDENMAYER.

## 4.1. ¿Quién los inventó?

El concepto de Sistemas de Lindenmayer o L-Sistemas fue concebido por el biólogo y botánico teórico húngaro Aristid Lindenmayer de la Universidad de Utrecht, en 1968. El comportamiento de algunos objetos naturales y la subdivisión en el organismo de los humanos en muchos casos obedecen a modelos fractales. Una de las forma para mostrar dichos comportamientos fue desarrollada por Lindenmayer, quien en el año de 1968 publicó *“Mathematical Models for Cellular Interaction in Development”*. En éste articulo mostró un modelo de interacción celular donde empleaba un nuevo lenguaje y nuevas reglas de construcción, en donde partiendo de una célula inicial, se genera un filamento celular cuyo orden en la posición de las células es conocido. Sin embargo, fueron dos de sus estudiantes, Ben Hesper y Pauline Hogeweg los primeros en darse cuenta del potencial de los L-Sistemas para representar plantas (en un principio). En 1970, Hesper y Hogeweg en dos semanas crearon el primer programa que, a partir de una secuencia de 5000 caracteres generados por un L-Sistema, imprimió algo que se parecía mucho a una hoja vegetal. Esta primera idea digital no le gustó mucho a Lindenmayer, ya que él consideraba que no debía perder tiempo y "desperdiciar" recursos en esas cosas. (Campos, 2011)

Posteriormente, cuando en 1986 Lindenmayer conoció a Przemyslaw Prusinkiewicz se convenció plenamente de las posibilidades de los L-Sistemas y empezó también a utilizarlos para generar imágenes cada vez más realistas de plantas. Juntos, en 1990, escribieron el libro *"La belleza algorítmica de las plantas" (The Algorithmic Beauty of Plants)*. (Muñoz, 2007)

## 4.2. ¿Qué es un L-Sistema?

Un L-Sistema es un lenguaje, un conjunto de reglas y símbolos[1] principalmente utilizados para modelar el proceso de crecimiento de las plantas, aunque también puede modelar la morfología de una gran variedad de organismos. El concepto central de los L-Sistemas es el de reescritura o la recursividad de los procesos, una técnica para definir objetos complejos reemplazando sucesivamente "partes" de un objeto inicial simple denominado axioma, mediante un conjunto de reglas de reescritura o de producción. Un algoritmo en L-Sistemas consiste en una gramática que posee símbolos y reglas de sustitución que a partir de reglas simples construye una estructura compleja, la cual genera una cadena de caracteres que pueden ser interpretados en términos gráficos. (Tagtachian & Argumedo, 1995)

Los L-Sistemas se basan en la reescritura de un código que genera una estructura sencilla, sustituyendo una parte de la misma por otra más compleja elaborada a partir de unas determinadas reglas, de tal manera que a cada paso (iteración, que es la repetición de una operación un sin número de veces), la estructura aumenta su complejidad. Las reglas de producción de las nuevas partes pueden ser las mismas que se usaron para crear las partes anteriores, y de esta forma se pueden crear formas en las cuales cada parte de la misma se parece al total, lo que se conoce como *autosimilaridad*.

## 4.3. Elementos de un L-Sistema.

Para la generación de los elementos geométricos de una estructura se pueden usar diferentes códigos o lenguajes de programación que incluyan funciones de creación de dichos elementos. Estas funciones normalmente requerirán coordenadas y otros datos por lo que su representación gráfica aumentará la complejidad del código, dificultando la visualización de las reglas de generación.

[1] Como el español, tiene símbolos (las letras del abecedario, signos de puntuación, etc.) y reglas para utilizarlos (la ortografía, la forma de acentuar las palabras, etc.)

Por otra parte es ineludible el uso de bucles para la reescritura de las partes del mismo que tengan que ser modificadas en cada iteración, y la complejidad asociada a la implementación de estos bucles puede acabar siendo bastante alta según el lenguaje de programación que se use. Por lo anterior, para el desarrollo de este trabajo se propone el manejo de un lenguaje que simplifique el uso de bucles y de funciones para la generación de elementos geométricos. (Rivero, 2014)

Este modelo depende del comportamiento de su alfabeto y de las reglas que se especifiquen, y está constituido como un conjunto por los siguientes elementos:

$$G = \{V, S, \omega, P\}$$

$V$: El alfabeto, es un conjunto de símbolos que pueden ser reemplazados (variables o símbolos no terminales) y se utilizan para componer cadenas.

$S$: Es un conjunto de símbolos que se mantienen fijos (constantes o símbolos terminales).

$\omega$: El axioma, es la cadena que describe al sistema en su estado inicial, formado por un (os) símbolo (s) de $V$.

$P$: Reglas de producción, son las transformaciones que serán aplicadas al axioma y, sucesivamente, a las cadenas generadas. Definen la forma en la que las variables pueden ser reemplazadas por combinaciones de constantes y otras variables. Las reglas de producción generan cadenas formadas únicamente por los símbolos del alfabeto y por lo tanto todas las cadenas pertenecerán al lenguaje definido por el L-Sistema.

Una consideración importante es que las reglas de producción se aplican simultáneamente a todos los símbolos de la cadena de entrada, sea ésta el

axioma o las cadenas resultantes de cada derivación. Esta propiedad refleja el origen biológico de los L-Sistemas, ya que los organismos vivos crecen simultáneamente en “todas” sus partes y no secuencialmente. Cabe señalar que en cada iteración, la estructura del sistema aumenta su complejidad por lo que hay que ser cuidadosos al derivar demasiadas veces el sistema. Además se pueden incorporar a la definición del sistema un conjunto de parámetros para su interpretación ya que debemos tener en cuenta que no le hemos dado ningún significado a los símbolos del alfabeto. Estos sistemas hipotéticamente podrían describir muy diversos procesos reales según sus significados (gráficos o sonoros); podrían representar la reproducción de células o el crecimiento de ramas en un árbol. Para generar imágenes se requiere que los símbolos en el modelo hagan referencia a elementos de un dibujo, interpretando cada constante en el L-Sistema como una operación de dibujo y para generar sonidos, que se asocien a una nota musical, por mencionar algunos. (Campos, 2011)

### 4.4. Sistema DOL.

Si definimos un símbolo del alfabeto y su respectiva regla de producción, iteramos y obtenemos la nueva cadena del lenguaje (esta es la idea básica de cómo funciona un L-Sistema). Luego, si definimos otra regla que incluye al símbolo original de axioma, por lo que ahora hay dos reglas que se pueden aplicar al mismo elemento, se genera así un “problema” porque este tipo de sistemas tienden a ser muy complejos por la extensa gama de posibilidades que pueden ocurrir. Así que, ¿cómo decidir que regla aplicar?, definimos entonces un **sistema DOL** (**D**eterministic **O**-context **L**-systems; sistemas determinísticos libres de contexto), que son los más sencillos que existen y operan bajo las siguientes condiciones:

- ✓ El lado izquierdo de la regla de producción debe ser un solo símbolo del alfabeto.
- ✓ Un mismo símbolo no puede producir distintas cadenas del lenguaje.

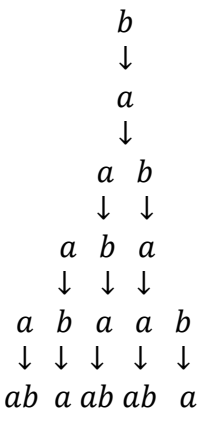
Con estas restricciones se asegura que para cualquier axioma o cadena introducida en el sistema DOL, ésta se podrá derivar en otra única cadena.

Para explicarlos de forma sencilla, Rivero (2014) presenta un ejemplo que propusieron Prusinkiewicz y Lindenmayer en 1991. Supongamos que tenemos dos caracteres $a$ y $b$ (letras que componen nuestro alfabeto), y a cada letra le asignamos la siguiente regla de reescritura:

$$a \rightarrow \text{ab} \quad \text{y} \quad b \rightarrow \text{a}$$

de tal forma que en cada paso de iteración cada letra $a$ del código se sustituirá por la cadena $ab$ y cada letra $b$ se sustituirá por la letra $a$. Para comenzar el proceso necesitaremos una cadena de caracteres inicial (axioma), formada por letras de nuestro alfabeto. Supongamos que empezamos por la cadena formada únicamente por la letra $b$. Comenzamos el proceso de reescritura y obtenemos el siguiente resultado al cabo de cinco pasos:

| | |
|---|---|
| Axioma: | $b$ |
| Paso 1: | $a$ |
| Paso 2: | $ab$ |
| Paso 3: | $aba$ |
| Paso 4: | $abaab$ |
| Paso 5: | $abaababa$ |



De esta forma sencilla se ha obtenido un L-Sistema a partir de elementos y reglas de sustitución muy elementales; sin embargo, hay que tener en cuenta que no se le ha dado ningún significado a los caracteres $a$ y $b$. Este sistema hipotéticamente podría describir diferentes procesos reales según su significado;

podría representar por ejemplo la reproducción de ciertas células o el crecimiento de las ramas en un árbol.

### 4.5. Interpretación gráfica de las cadenas de caracteres.

La idea básica de la interpretación de las cadenas de caracteres es la "representación de la tortuga", ésta consiste en suponer que estamos parados arriba de una tortuga la cual puede avanzar o retroceder, rotar y escribir mientras avanza o avanzar sin escribir. (estas son las ideas introducidas por el lenguaje LOGO) Luego, se puede decir que el "estado de la tortuga" viene dado por las coordenadas cartesianas donde se halla la tortuga y el ángulo de referencia se interpreta como la dirección en que se encuentra avanzando la tortuga. Al introducir estos datos se puede definir la trayectoria que recorrerá la tortuga. (Tagtachian & Argumedo, 1995)

Para darle un significado geométrico a las cadenas de caracteres, se empleará un tipo de gráficos sencillos cuyo uso está generalizado en la creación de los Sistemas de Lindenmayer. Estos gráficos estarán constituidos por elementos simples descritos únicamente por su posición y su orientación. Así, solamente se trabajará con un espacio bidimensional. En la siguiente tabla, se pueden encontrar los símbolos básicos (caracteres) y sus significados geométricos dentro de los L-Sistemas.

| SÍMBOLO | SIGNIFICADO |
|---|---|
| L | Longitud del segmento. |
| F | Avanzar (dibujar) un segmento de longitud L. |
| G | Moverse hacia adelante, sin dibujar, un segmento de longitud L. |
| X e Y | Son variables que no poseen significado geométrico. Es decir, son ignoradas a la hora de interpretar las cadenas de símbolos gráficamente. Se usan para conseguir el orden correcto de símbolos gráficos deseados. |
| + | Girar al contrario de las manecillas del reloj un ángulo $\theta$. |
| - | Girar en el sentido de las manecillas del reloj un ángulo $\theta$. |

Tabla 2: El Lenguaje de los L-Sistemas.

Con estos elementos definidos, podemos empezar a usar el programa Turtle Graphics Renderer (Roast, 2012), que es un software gratuito y en línea para la generación de L-Sistemas en un espacio bidimensional creado por Kevin Roast, al cual se puede acceder desde el link: http://kevs3d.co.uk/dev/lsystems/# o a través del buscador predeterminado por el usuario. Esta herramienta es muy flexible ya que permite cambiar numerosos aspectos, tales como las iteraciones, el ángulo, constantes, axioma y hasta cinco reglas de reescritura diferentes. En la página también se incluye una lista de ejemplos donde el usuario puede interactuar con el algoritmo para observar por ejemplo cómo los fractales pueden relacionarse con la naturaleza. A continuación se presentan las imágenes del entorno gráfico del programa junto con su descripción, así:

DESCRIPCIÓN — ENTORNO GRÁFICO DEL PROGRAMA

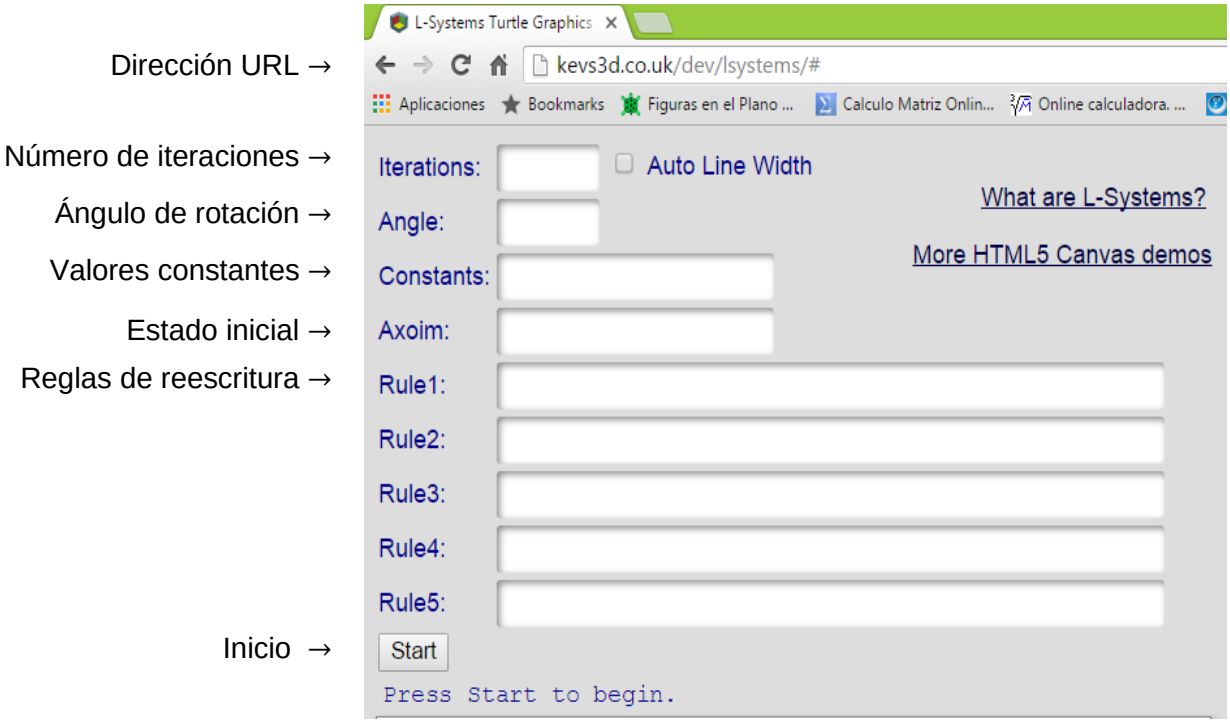



Imagen 5: Entorno gráfico Turtle Graphics Renderer on-line.

Como se puede observar, en el programa Turtle Graphics Renderer on-line no se requiere asignar una longitud predeterminada al segmento, sino que es el

programa el que la asigna ya que variando este parámetro lo único que se conseguiría es variar la escala del objeto representado, pero no la forma del objeto en sí.

La siguiente imagen muestra la pantalla de la vista gráfica que ofrece el programa, al igual que el botón de inicio para representar el L-Sistema que se haya ingresado, aunque basta con darle "Enter" para que el programa se ejecute.

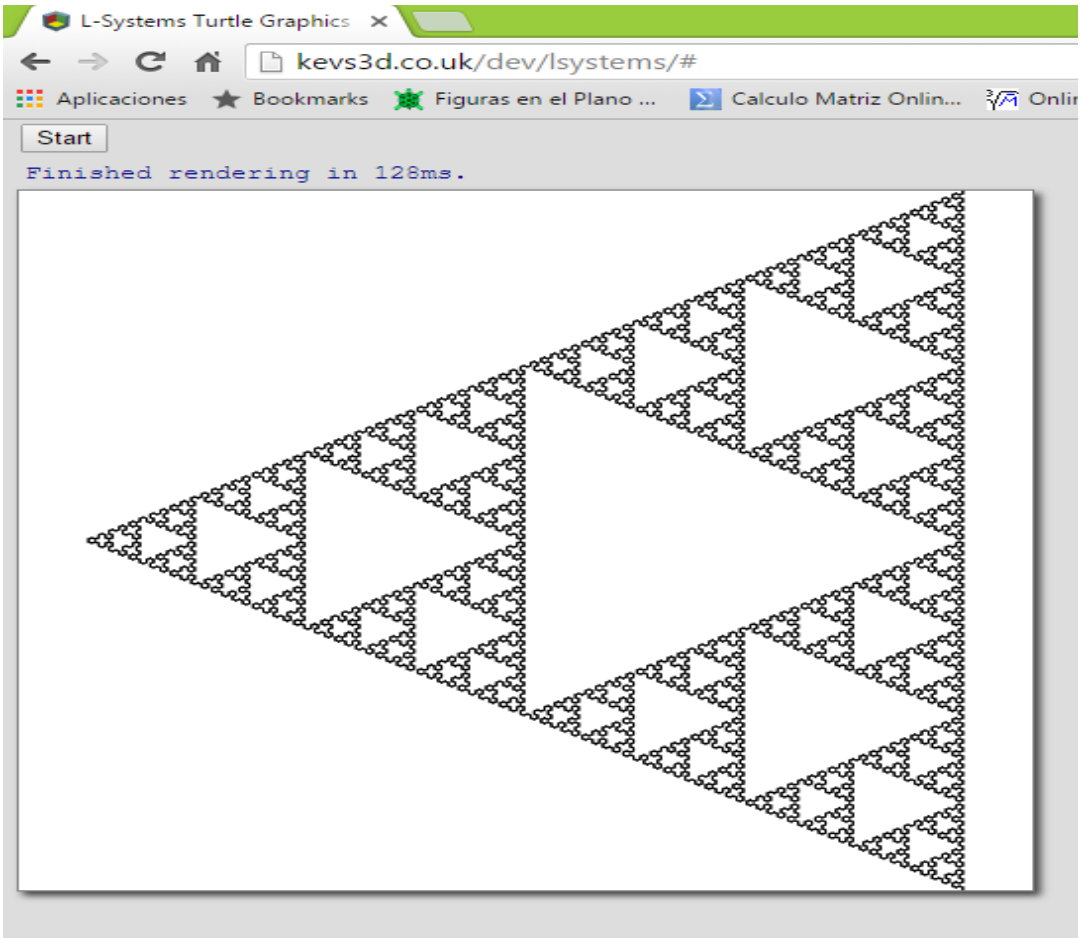



Imagen 6: Vista gráfica Turtle Graphics Renderer on-line.

A continuación se representan algunas construcciones de fractales con los L-Sistemas empleando el programa Turtle Graphics Renderer on-line.

## 4.6. L–Sistemas en Turtle Graphics Renderer on-line.

### 4.6.1. La Curva de Koch.

El código que se debe ingresar en este programa para generar un L-Sistema, como por ejemplo la curva de Koch, tendrá los siguientes caracteres:

| DESCRIPCIÓN | CÓDIGO |
|---|---|
| Número de iteraciones (van aumentando) | 0 |
| Ángulo predeterminado | $60°$ |
| Axioma (estado inicial del L-Sistema) | $F$ |
| Reglas de reescritura | $F = F + F - -F + F$ |

Tabla 3: Código Curva de Koch en L-Sistemas. Iteración 0.

Al ingresar este código en el programa, con un valor de cero iteraciones, el resultado será la representación gráfica del axioma o estado inicial indicado "F", es decir, en este caso se dibujará un segmento de longitud L. Sin embargo, según se mencionó anteriormente en el programa no se requiere asignar una longitud predeterminada al segmento, sino que es el programa el que la asigna.

Imagen 7: Construcción Von Koch - Iteración 0.

Ahora veamos que sucede al realizar la primera iteración del código.

| DESCRIPCIÓN | CÓDIGO |
|---|---|
| Número de iteraciones (van aumentando) | 1 |
| Ángulo predeterminado | $60°$ |
| Axioma (estado inicial del L-Sistema) | $F$ |
| Reglas de reescritura | $F = F + F - -F + F$ |

Tabla 4: Código Curva de Koch en L-Sistemas. Iteración 1.

A continuación se presenta la imagen de la pantalla luego de introducir el código en el programa:

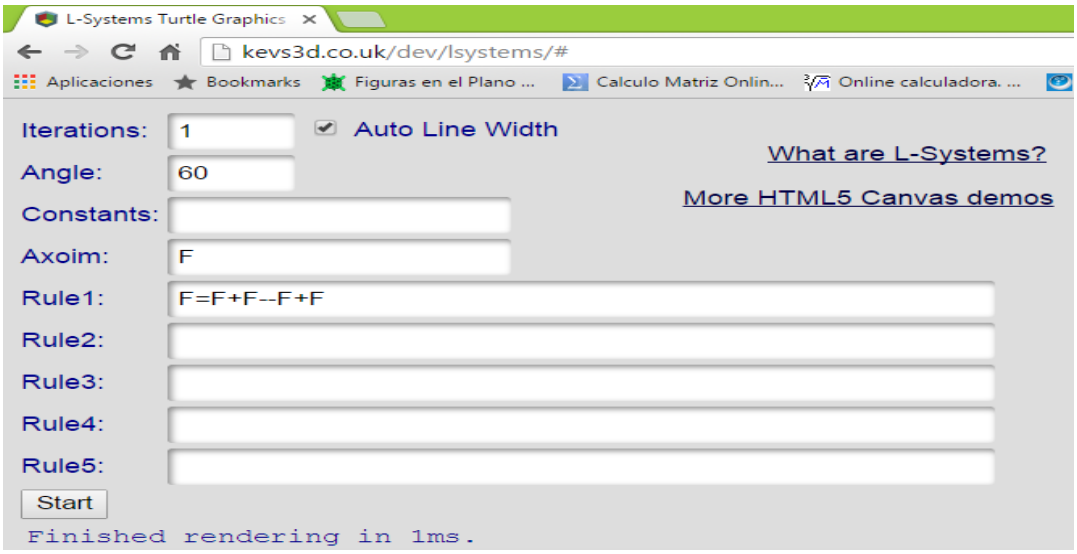


Imagen 8: Código Curva de Koch en Turtle Graphics Renderer. Iteración 1.

Al hacer click en inicio, "Start en el programa", se imprimirá la primer cadena de caracteres establecidos, de tal forma que en cada uno de los caracteres de la regla de reescritura dada, en este ejemplo: $F+F--F+F$, cada letra $F$ del código se sustituirá por el axioma establecido, en este caso $F$, donde cada $F$ significa avanzar (dibujar) un segmento de longitud L, la cual no se requiere asignar sino que es el programa el que la asigna. Así mismo, cada $+$ significa girar un ángulo de $60°$ (establecido por el usuario) al contrario de las manecillas del reloj y cada $-$ significa girar un ángulo de $60°$ (establecido por el usuario) en el sentido de las manecillas del reloj.

$$\text{Axioma:} \quad F$$

$$\text{Regla de reescritura:} \quad F = F + F - -F + F$$

$$\text{Iteración 0:} \quad F$$

$$\text{Iteración 1:} \quad \overbrace{F + F - -F + F}$$

En palabras más sencillas, la primera iteración sería: Avanza (dibuja) una longitud L, gira un ángulo de $60°$ al contrario de las manecillas del reloj, avanza nuevamente (dibuja) una longitud L, gira el doble del ángulo de $60°$ en el sentido

de las manecillas del reloj (puesto que el signo – aparece dos veces), luego avanza (dibuja) una longitud L, gira un ángulo de $60°$ al contrario de las manecillas del reloj y avanza (dibuja) una longitud L.

La siguiente imagen muestra la forma de construcción del L-Sistema en la primera iteración en el programa Turtle Graphics Renderer on-line, así:

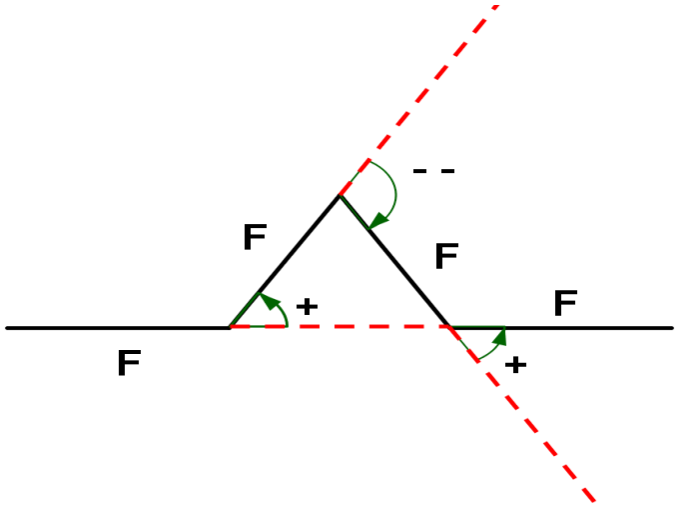



Imagen 9: Construcción Von Koch - Iteración 1.

En la imagen anterior, se puede observar que el segmento inicial de longitud L, se ha dividido en tres partes de igual medida, cada uno de ellos de longitud $^1/_3$ L. Se consideran los segmentos de los extremos y el segmento central se sustituye por dos nuevos segmentos de longitud $^1/_3$ L cada uno de ellos, los cuales formaron con el segmento retirado un triángulo equilátero. Así, se obtienen cuatro segmentos que forman la figura mostrada.

Ahora veamos que sucede al realizar la segunda iteración del código.

| DESCRIPCIÓN | CÓDIGO |
|---|---|
| Número de iteraciones | 2 |
| Ángulo predeterminado | $60°$ |
| Axioma (estado inicial del L-Sistema) | $F$ |
| Reglas de reescritura | $F = F + F - -F + F$ |

Tabla 5: Código Curva de Koch en L-Sistemas. Iteración 2.

A continuación se presenta la imagen de la pantalla luego de introducir el código en el programa:

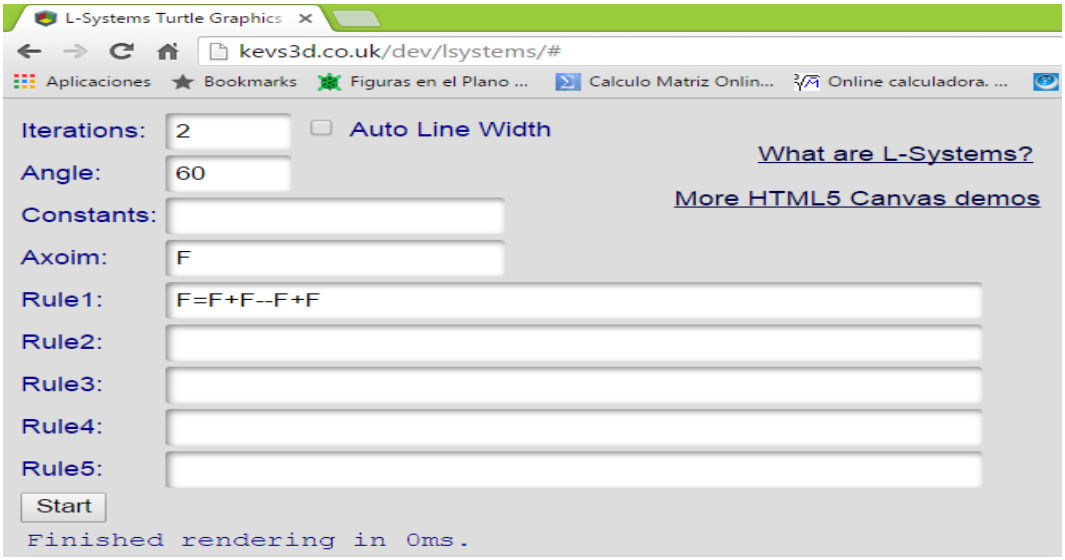



Imagen 10: Código Curva de Koch en Turtle Graphics Renderer. Iteración 2.

Al hacer click en "Start", se inicia un proceso de reescritura de la primer cadena de caracteres impresa en el paso anterior, de tal forma que en cada uno de los caracteres establecidos en la primera iteración: $F + F - -F + F$, cada letra $F$ del código se sustituirá por la cadena de la regla dada: $F + F - -F + F$. Los signos $+$ y $-$ conservan su condición y se interpretan de la manera preestablecida.

Axioma: $F$

Regla: $F = F + F - -F + F$

Iteración 0: $F$

Iteración 1: $\overbrace{F \quad + \quad F \quad - - \quad F \quad + \quad F}$

Iteración 2: $\overbrace{F + F - -F + F}\ +\ \overbrace{F + F - -F + F}\ -\ -\ \overbrace{F + F - -F + F}\ +\ \overbrace{F + F - -F + F}$

Gráficamente se puede observar que al repetir el proceso sobre cada uno de los cuatro segmentos de la primera iteración, se obtiene otra curva formada por $4^2$ segmentos de longitud $3^{-2}$. Sin embargo, según se mencionó antes en el programa no se requiere asignar una longitud predeterminada al segmento, sino que es el programa el que la asigna. La siguiente imagen muestra la construcción del L-Sistema en la segunda iteración en el programa Turtle Graphics Renderer on-line, así:

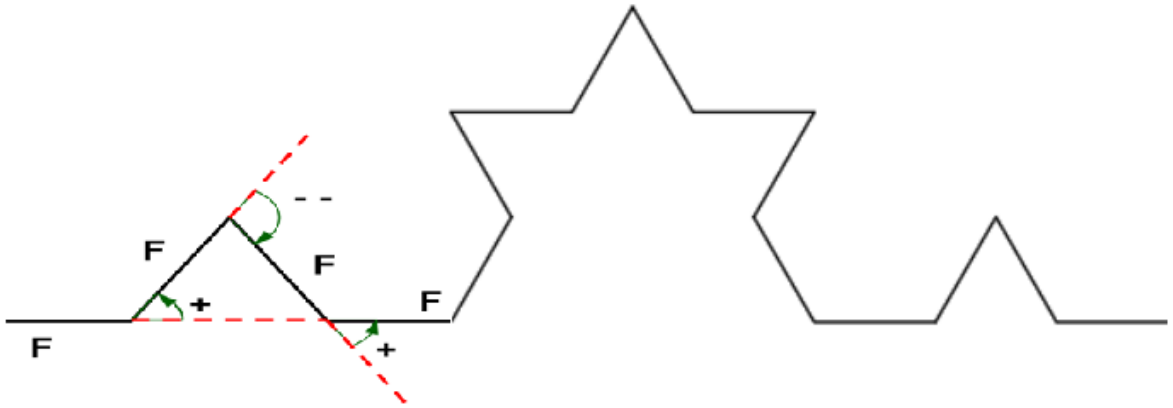



Imagen 11: Construcción Von Koch - Iteración 2.

Ahora veamos que sucede al realizar la tercera iteración del código.

| DESCRIPCIÓN | CÓDIGO |
|---|---|
| Número de iteraciones | 3 |
| Ángulo predeterminado | $60°$ |
| Axioma (estado inicial del L-Sistema) | $F$ |
| Reglas de reescritura | $F = F + F - -F + F$ |

Tabla 6: Código Curva de Koch en L-Sistemas. Iteración 3.

A continuación se presenta la imagen de la pantalla luego de introducir el código en el programa:

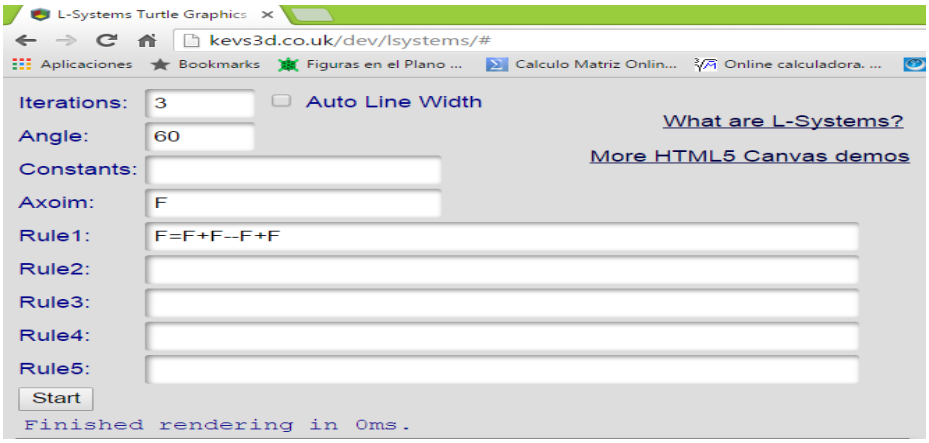



Imagen 12: Código Curva de Koch en Turtle Graphics Renderer. Iteración 3.

Al hacer click en "Start", nuevamente se inicia un proceso de reescritura de la cadena de caracteres impresa en el paso anterior, de tal forma que en cada uno de los caracteres establecidos en la segunda iteración: $F+F--F+F+F+F--F+F--F+F--F+F+F+F--F+F$, cada letra $F$ del código se sustituirá por la cadena de la regla dada: $F+F--F+F$. Los signos $+$ y $-$ conservan su condición y se interpretan de la misma manera.

$$
\begin{array}{cccc}
\text{Iteración 0} & \text{Iteración 1} & \text{Iteración 2} & \text{Iteración 3} \\
\downarrow & \downarrow & \downarrow & \downarrow \\
F & \{ \; F+F--F+F \; \{ & \begin{array}{l} F+F--F+F \\ + \\ F+F--F+F \\ + \\ F+F--F+F \\ + \\ F+F--F+F \end{array} & \begin{array}{l} \{F+F--F+F\ + \\ F+F--F+F--\\ F+F--F+F\ + \\ F+F--F+F\} \\ -- \\ \{F+F--F+F\ + \\ F+F--F+F--\\ F+F--F+F\ + \\ F+F--F+F\} \\ + \\ \{F+F--F+F\ + \\ F+F--F+F--\\ F+F--F+F\ + \\ F+F--F+F\} \end{array}
\end{array}
\; \} \; \}
$$

Gráficamente se puede observar que al repetir el proceso sobre cada uno de los segmentos de la segunda iteración, se obtiene otra curva formada por $4^3$ segmentos de longitud $3^{-3}$. Sin embargo, según se mencionó antes en el programa no se requiere asignar una longitud predeterminada al segmento, sino que es el programa el que la asigna. La siguiente imagen muestra la construcción del L-Sistema en la tercera iteración en el programa Turtle Graphics Renderer on-line, así:

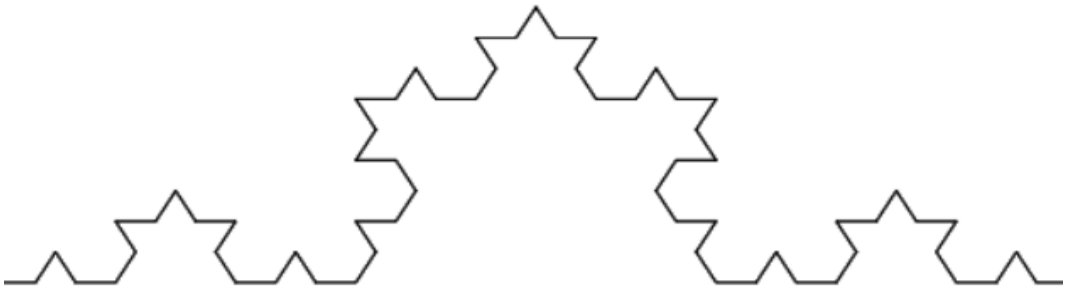


Imagen 13: Construcción Von Koch - Iteración 3.

Si continuamos iterando, el código para la cuarta iteración es:

| DESCRIPCIÓN | CÓDIGO |
|---|---|
| Número de iteraciones | 4 |
| Ángulo predeterminado | $60°$ |
| Axioma (estado inicial del L-Sistema) | $F$ |
| Reglas de reescritura | $F = F + F - -F + F$ |

Tabla 7: Código Curva de Koch en L-Sistemas. Iteración 4.

Gráficamente se puede observar que al repetir el proceso sobre cada uno de los segmentos de la tercera iteración, se obtiene otra curva formada por $4^4$ segmentos de longitud $3^{-4}$. La siguiente imagen muestra la construcción del L-Sistema en la cuarta iteración en el programa Turtle Graphics Renderer on-line, así:

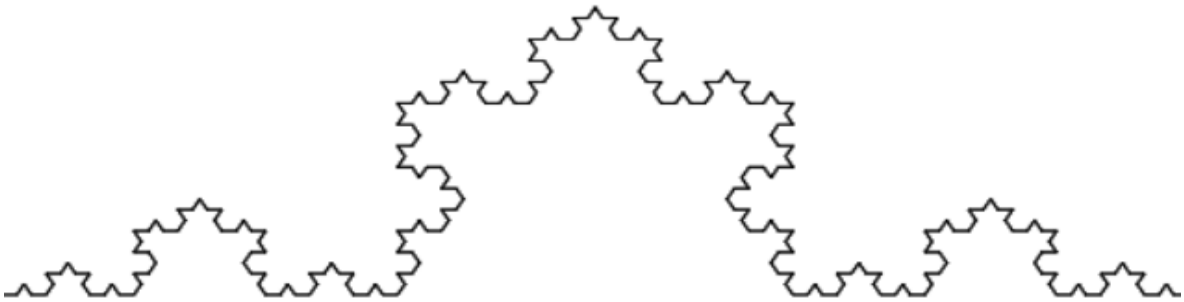


Imagen 14: Construcción Von Koch - Iteración 4.

Si continuamos iterando, el código para la quinta iteración es:

| DESCRIPCIÓN | CÓDIGO |
|---|---|
| Número de iteraciones | 5 |
| Ángulo predeterminado | $60°$ |
| Axioma (estado inicial del L-Sistema) | $F$ |
| Reglas de reescritura | $F = F + F - -F + F$ |

Tabla 8: Código Curva de Koch en L-Sistemas. Iteración 5.

Gráficamente se puede observar que al repetir el proceso sobre cada uno de los segmentos de la cuarta iteración, se obtiene otra curva formada por $4^5$ segmentos de longitud $3^{-5}$. La siguiente imagen muestra la construcción del L-Sistema en la quinta iteración en el programa Turtle Graphics Renderer on-line, así:

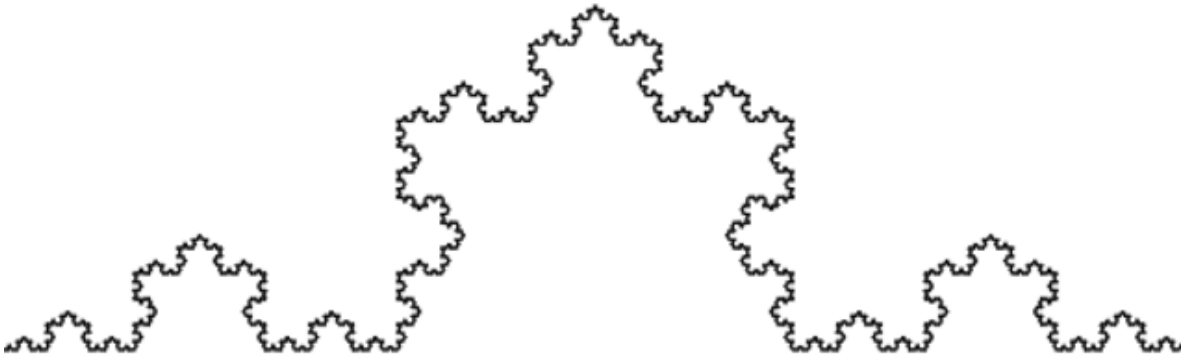


Imagen 15: Construcción Von Koch - Iteración 5.

Si se continúa de esta forma, en la etapa i-ésima se habrá obtenido una curva formada por $4^i$ segmentos de longitud $3^{-i}$ cada uno de ellos, donde $i$: corresponde al número de la iteración realizada, obteniéndose de este modo en cada iteración una figura autosimilar más compleja.

### 4.6.2. Copo de nieve de Koch.

El L-Sistema que se debe ingresar en este programa para generar el copo de nieve de Koch, tendrá los siguientes caracteres:

| DESCRIPCIÓN | CÓDIGO |
|---|---|
| Número de iteraciones (van aumentando) | 0 |
| Ángulo predeterminado | $60°$ |
| Axioma (estado inicial del L-Sistema) | $F++F++F$ |
| Reglas de reescritura | $F = F-F++F-F$ |

Tabla 9: Código Copo de Nieve en L-Sistemas. Iteración 0.

Luego de introducir correctamente el código deseado, el usuario empieza a elegir el número de iteraciones que desea realizar. En las siguientes imágenes se muestran las construcciones del L-Sistema en la iteración 0 y 1 con el programa Turtle Graphics Renderer on-line, así:

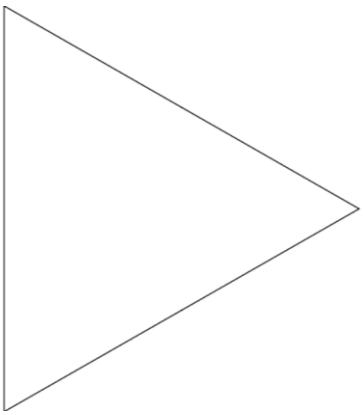


Imagen 16: Construcción Copo de Nieve de Koch - Iteración 0.

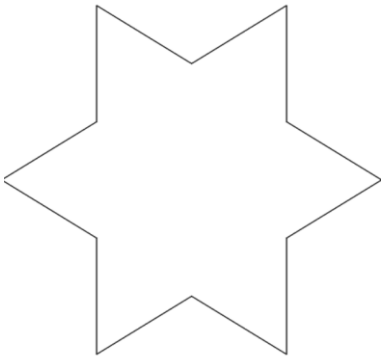


Imagen 17: Construcción Copo de Nieve de Koch - Iteración 1.

En la siguiente tabla se muestra la secuencia de la reescritura del axioma si se realizaran tres iteraciones.

| PASO | Secuencia del L–Sistema |
|---|---|
| Axioma | $F++F++F$ |
| Regla 1 | $F=F-F++F-F$ |
| Iteración 1 | $F++F++F-F++F++F++F++F-F++F++F$ |
| Iteración 2 | $F++F++F++F++F++F++F++F-F++F++F++F++F$<br>$++F++F++F++F++F++F++F++F++F$<br>$++F++F++F++F++F-F++F++F++F++F$<br>$++F++F++F++F$ |
| Iteración 3 | $F++F++F++F++F++F++F++F++F++F++F++F$<br>$++F++F++F++F++F++F++F++F++F++F$<br>$++F++F++F++F++F++F-F++F++F++F++F$<br>$++F++F++F++F++F++F++F++F++F++F$<br>$++F++F++F++F++F++F++F++F++F++F$<br>$++F++F++F++F++F++F++F++F++F++F$<br>$++F++F++F++F++F++F++F++F++F++F$<br>$++F++F++F++F++F-F++F++F++F++F$<br>$++F++F++F++F++F++F++F++F++F++F$<br>$++F++F++F++F++F++F++F++F++F++F$<br>$++F++F++F++F$ |
| Iteración 4 | … |

Tabla 10: Copo de Nieve. Secuencia del L–Sistema. Paso 1 – 3.

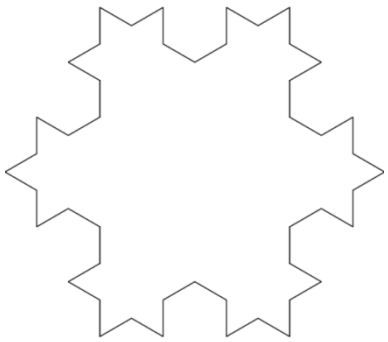


Imagen 18: Construcción Copo de Nieve de Koch - Iteración 2.

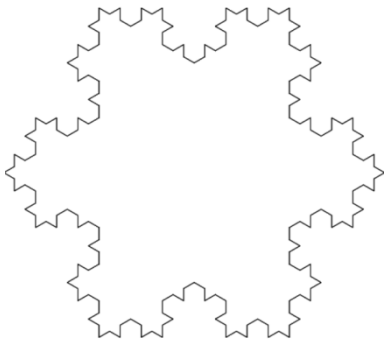


Imagen 19: Construcción Copo de Nieve de Koch - Iteración 3.

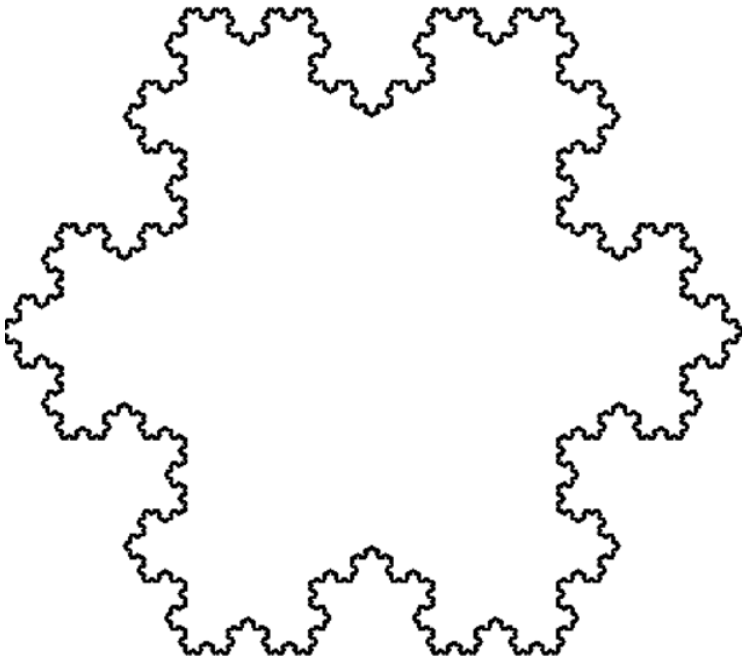


Imagen 20: Construcción Copo de Nieve de Koch - Iteración 9.

### 4.6.3. Triángulo de Sierpinski.

El L-Sistema que se debe ingresar en este programa para generar el Triángulo de Sierpinski, tendrá los siguientes caracteres:

| DESCRIPCIÓN | CÓDIGO |
|---|---|
| Número de iteraciones (van aumentando) | 0 |
| Ángulo predeterminado | $120°$ |
| Axioma (estado inicial del L-Sistema) | $F - G - G$ |
| Regla 1 | $F = F - G + F + G - F$ |
| Regla 2 | G=GG |

Tabla 11: Código Triángulo de Sierpinski en L-Sistemas. Iteración 0.

Luego de introducir correctamente el código deseado, el usuario empieza a elegir el número de iteraciones que desea realizar. En las siguientes imágenes se muestran las construcciones del L-Sistema desde la iteración 0 hasta la iteración 5 con el programa Turtle Graphics Renderer on-line, así:

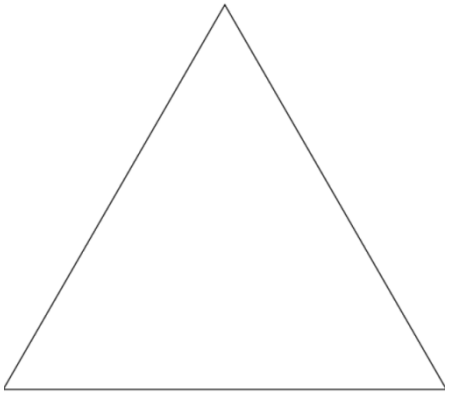


Imagen 21: Construcción Triángulo de Sierpinski - Iteración 0.

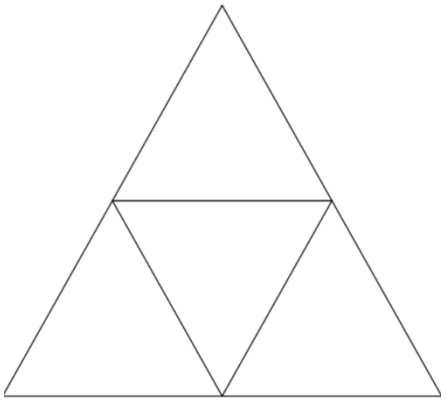


Imagen 22: Construcción Triángulo de Sierpinski - Iteración 1.

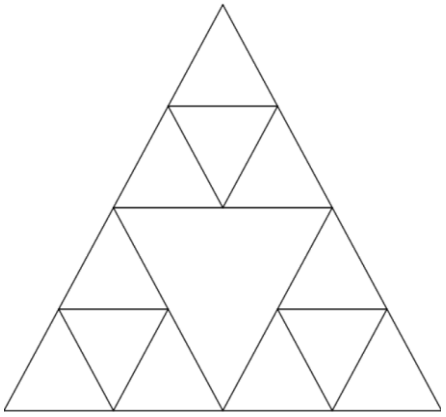


Imagen 23: Construcción Triángulo de Sierpinski - Iteración 2.

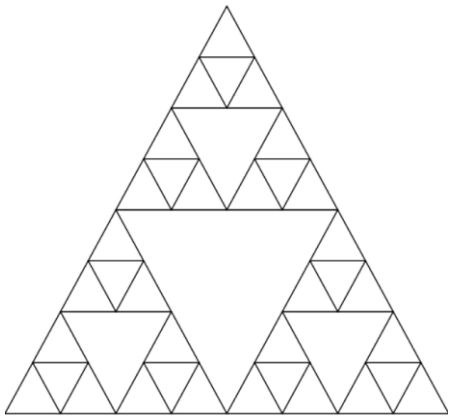


Imagen 24: Construcción Triángulo de Sierpinski - Iteración 3.

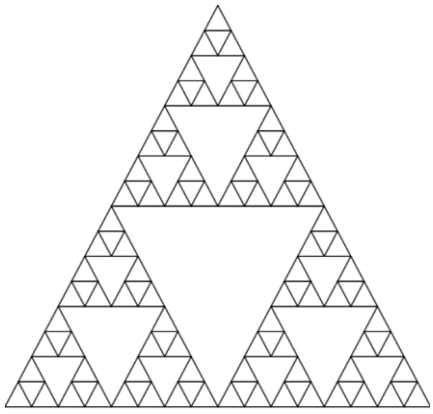


Imagen 25: Construcción Triángulo de Sierpinski - Iteración 4.

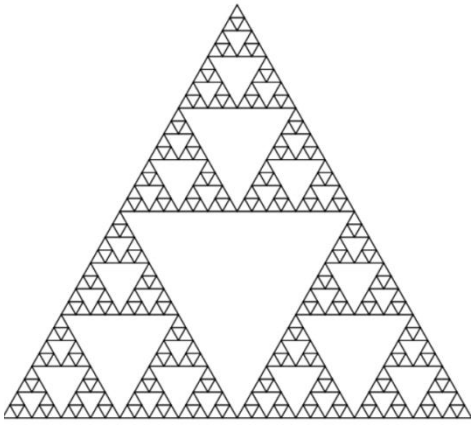


Imagen 26: Construcción Triángulo de Sierpinski - Iteración 5.

## 4.7. L-Sistemas ramificados.

Los ejemplos presentados hasta el momento, con tan sólo unas pocas reglas nos dan una idea de las inmensas posibilidades de los L-Sistemas. Sin embargo, utilizando solamente estas reglas siempre obtendríamos estructuras que siguen una línea continua, sin posibilidad de ramificarse. Con el objetivo de crear ramificaciones se introducirán en el alfabeto dos nuevos símbolos " [ " y " ] " para delimitar una rama, y se dispone una memoria (pila) para almacenarla y usarla posteriormente. El ejemplo que se presenta a continuación genera una estructura bidimensional en forma de planta muy sencilla:

| DESCRIPCIÓN | CÓDIGO |
|---|---|
| Iteraciones: | 1 |
| Ángulo: | $36°$ |
| Axioma: | $F$ |
| Regla 1 | $F = F[+F]F$ |

Tabla 12: Código Planta 1 en L-Sistemas. Iteración 1.

Cada vez que el programa se encuentre con los símbolos " [ " y " ] " (paréntesis cuadrados), introducirá el contenido de la memoria almacenada (pila) en el código, y volverá a situar el punto geométrico a partir del cual continúa el trazado del dibujo en el lugar en que se encontraba antes de interpretar los paréntesis cuadrados. En la iteración 0 la pila está vacía, ya que todavía no se ha introducido nada en ella; por lo tanto, el programa no introducirá ningún contenido cuando lea por primera vez los paréntesis cuadrados, simplemente interpretará el código que ya existe dentro. No obstante, el punto de dibujo sí retornará a su posición anterior para dibujar una nueva rama. Al llegar a la última línea se introducirá en la pila toda la estructura generada en la primera iteración, y se volverá a incluir entera en el código cada vez que aparezcan los paréntesis cuadrados.

En palabras más sencillas, la primera iteración sería: Avanza (dibuja) un segmento de longitud L, almacena la posición en la memoria, gira un ángulo de $36°$ al contrario de las manecillas del reloj, avanza nuevamente (dibuja) un segmento de longitud L, regresa a la posición almacenada en la memoria y luego avanza (dibuja) un segmento de longitud L. Este proceso se vuelve a repetir según el número de iteraciones que se le indique al programa. El siguiente esquema expresa esta idea un poco mejor:

Axioma: $F$

Regla: $F = F[+F]F$

Iteración 0: $F$

Iteración 1:

| $F$ | $[$ | $+$ | $F$ | $]$ | $F$ |
|---|---|---|---|---|---|
| ↓ | ↓ | ↓ | ↓ | ↓ | ↓ |
| Avanza (dibuja) un segmento de longitud L | Almacena la posición en la memoria | Gira un ángulo de $-36°$ | Avanza (dibuja) un segmento de longitud L | Regresa a la posición almacenada en la memoria | Avanza (dibuja) un segmento de longitud L |

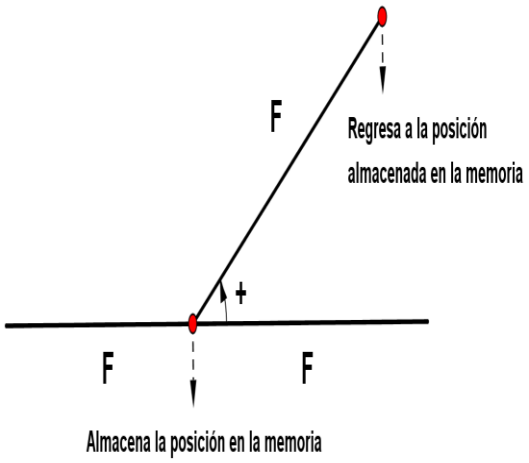



Imagen 27: Construcción Planta 1 - Iteración 1.

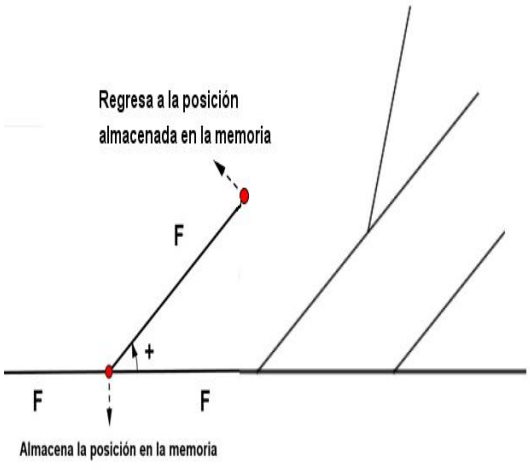



Imagen 28: Construcción Planta 1 - Iteración 2.

En las siguientes imágenes se muestran las construcciones del L-Sistema desde la iteración 1 hasta la iteración 6 con el programa Turtle Graphics Renderer on-line, así:

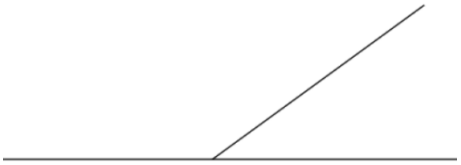


Imagen 27: Construcción Planta 1 - Iteración 1.

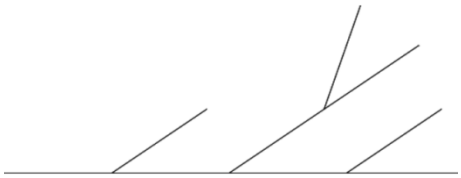


Imagen 28: Construcción Planta 1 - Iteración 2.

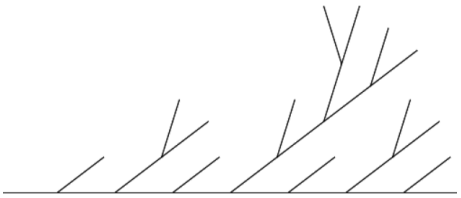


Imagen 29: Construcción Planta 1 - Iteración 3.

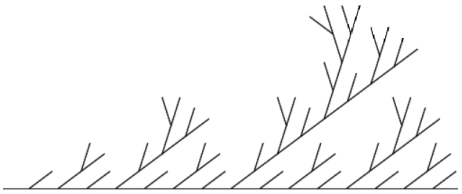


Imagen 30: Construcción Planta 1 - Iteración 4.

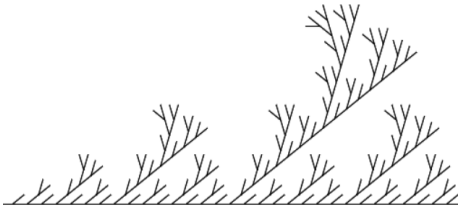


Imagen 31: Construcción Planta 1 - Iteración 5.

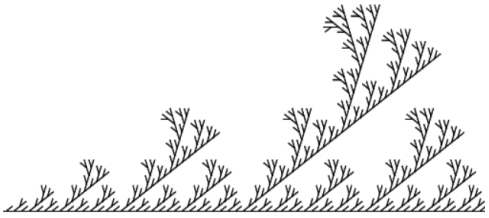


Imagen 32: Construcción Planta 1 - Iteración 6.

A continuación se presenta otro ejemplo de un L-Sistema ramificado haciendo uso de los paréntesis cuadrados para simular una planta un poco más elaborada.

| DESCRIPCIÓN | CÓDIGO |
|---|---|
| Iteraciones: | 5 |
| Ángulo: | $30°$ |
| Axioma: | $F$ |
| Regla 1: | $F = F[-F]F[+F][F]$ |

Tabla 13: Código Planta 2 en L-Sistemas. Iteración 5.

Lo que se observará al ejecutar el L-Sistema en el programa será la combinación del caso anterior pero en ambos sentidos (en el sentido de las manecillas del reloj y al contrario de las manecillas del reloj) puesto que en la primera iteración se dibuja un segmento, se almacena la posición en la memoria, se gira un ángulo de $30°$ *en el sentido de las manecillas del reloj*, vuelve a dibujar un segmento, se regresa a la posición almacenada en la memoria y luego se avanza dibujando un segmento para iniciar nuevamente el proceso en el sentido contrario, es decir, se dibuja un segmento, se almacena la posición en la memoria, se gira un ángulo de $30°$ *al contrario de las manecillas del reloj*, se dibuja un segmento, se regresa a la posición almacenada en la memoria y por último se dibuja un segmento. Este proceso se vuelve a repetir según el número de iteraciones que se le indique al programa.

En las siguientes imágenes se muestran las construcciones del L-Sistema desde la iteración 1 hasta la iteración 5 con el programa Turtle Graphics Renderer on-line, así:

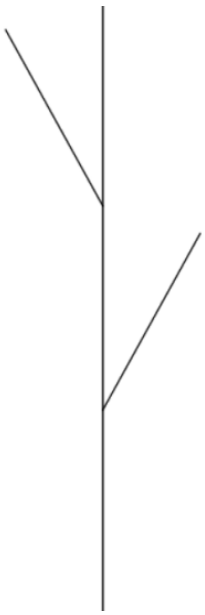


Imagen 33: Construcción Planta 2 - Iteración 1.

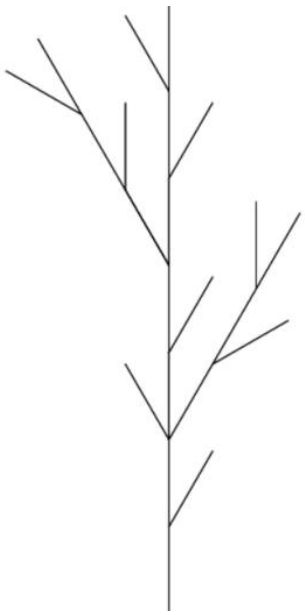


Imagen 34: Construcción Planta 2 - Iteración 2.

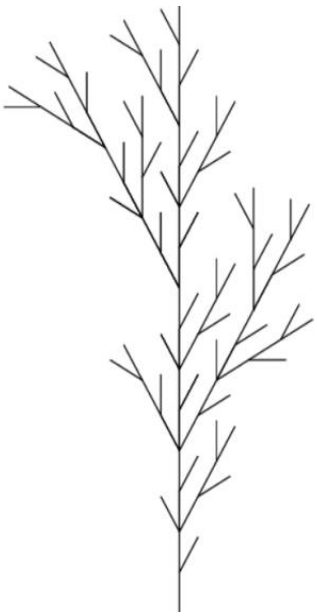


Imagen 35: Construcción Planta 2 - Iteración 3.

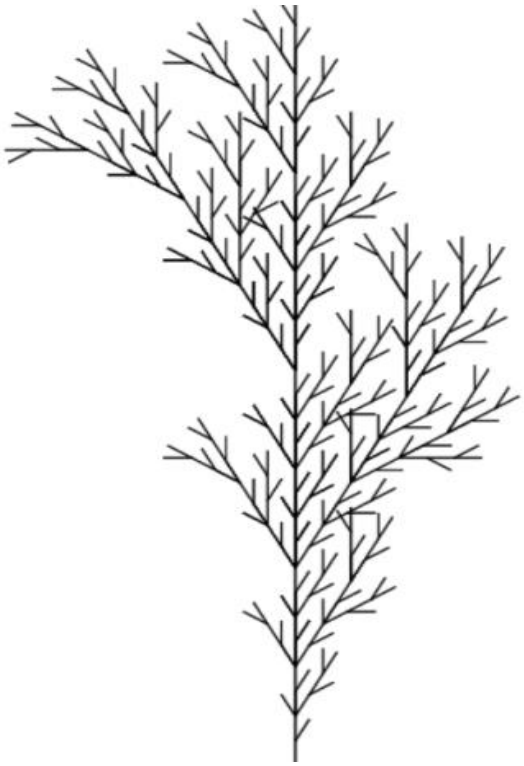


Imagen 36: Construcción Planta 2 - Iteración 4.

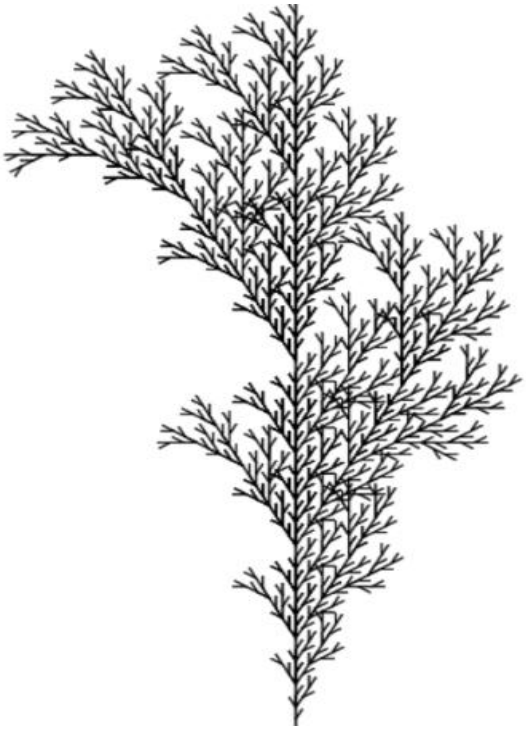


Imagen 37: Construcción Planta 2 - Iteración 5.

A continuación se presentan dos ejemplos más de los L-Sistemas ramificados haciendo uso de los paréntesis cuadrados para simular plantas y arbustos.

L-Sistema: Planta 3
Iteraciones: 6
Ángulo: $18°$
Axioma: $X$
Regla 1: $X = F[+X]F[-X] + X$
Regla 2: $F = FF$

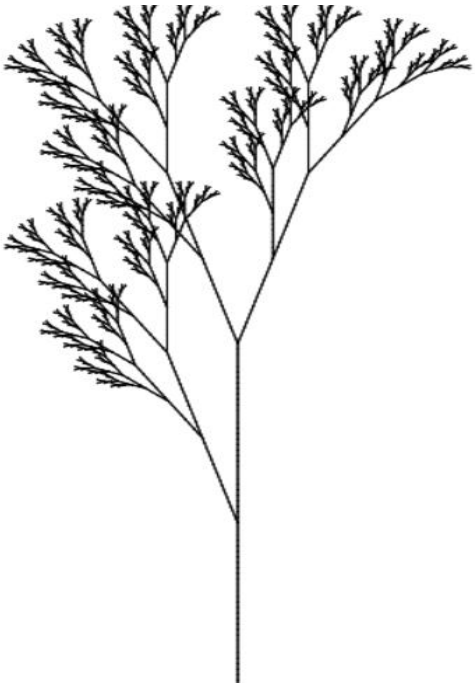


Imagen 38: Construcción Planta 3 - Iteración 6.

L-Sistema: Arbusto 1
Iteraciones: 5
Ángulo: $22{,}5°$
Axioma: $F$
Regla 1: $F = FF - [-F + F + F] + [+F - F - F]$

Imagen 39: Construcción Arbusto 1 - Iteración 5.

### 4.8. L-Sistemas estocásticos.

Hasta ahora los sistemas creados son demasiado "perfectos", y ello es debido a que se usa un patrón que se repite de forma idéntica a lo largo de todo el proceso sin posibilidad de variación. A los L-Sistemas descritos hasta el momento se les llama L-Sistemas paramétricos, y en ellos siempre se reproducirá la misma secuencia en el desarrollo. En las plantas reales, sin embargo, podemos observar una cierta aleatoriedad en sus ramificaciones, a pesar de que se mantiene una cierta organización general que da un aspecto típico a cada especie. Para introducir un cierto grado de aleatoriedad en nuestro sistema emplearemos el símbolo " $\sim$ " (símbolo de negación) de la siguiente forma:

| | |
|---|---|
| $\sim$ | Realiza un giro de manera aleatoria. |
| $\sim(x)$ | Gira un máximo de x grados de manera aleatoria. El valor de x se determina al momento de establecer las reglas de reescritura. |

Ahora veamos el resultado de aplicar esta nueva condición en el ejemplo de la Planta 2 de un L-Sistema ramificado con $30°$ de aleatoriedad máxima en la primera rama generada.

| | |
|---|---|
| Iteraciones: | 5 |
| Ángulo: | $30°$ |
| Axioma: | $F$ |
| Regla 1: | $F = F[\sim(30) - F]F[+F][F]$ |

Tabla 14: Código Aleatorio Planta 2 en L-Sistemas. Iteración 5.

En las siguientes imágenes se muestran las construcciones desde la iteración 1 hasta la iteración 5 con el programa Turtle Graphics Renderer on-line, así:

Imagen 40: Construcción Planta 2 Aleatoria - Iteración 1.

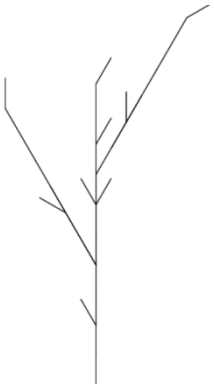


Imagen 41: Construcción Planta 2 Aleatoria - Iteración 2.

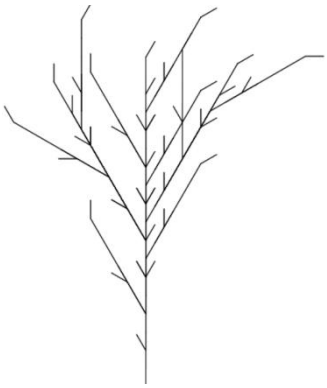


Imagen 42: Construcción Planta 2 Aleatoria - Iteración 3.

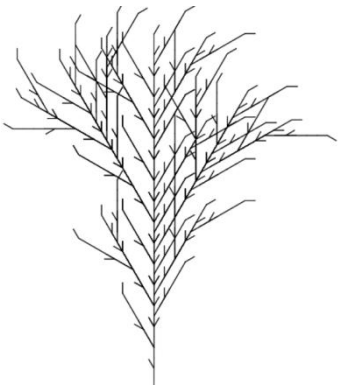


Imagen 43: Construcción Planta 2 Aleatoria - Iteración 4.

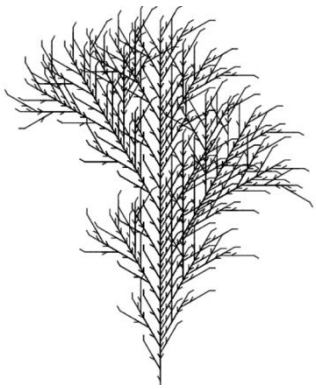


Imagen 44: Construcción Planta 2 Aleatoria - Iteración 5.

Como se puede observar, el resultado obtenido poco tiene que ver con el ejemplo de la Planta 2 de un L-Sistema ramificado, presentado anteriormente. Nos encontramos ante un modelo en tres dimensiones, lo cual era de esperar ya que los giros que proporciona el comando " ~ " pueden ser en cualquier dirección del espacio. Por otra parte la planta parece mucho más real, y es más difícil identificar el patrón utilizado, a pesar de que se ha generado con una forma básica muy similar a la anterior. Debido a la aleatoriedad introducida, si volviéramos a generar la imagen a partir del mismo código podríamos obtener en cada caso un resultado distinto.

### 4.9. Mecanismos de control para el entorno.

Los sistemas vistos hasta el momento son sistemas aislados, sin un entorno que pueda modificar sus procesos de crecimiento. Sin embargo, en los organismos reales, se dan distintas formas de interacción con el medio ambiente, como pueden ser el contacto con otros objetos, la gravedad o la competencia por recursos naturales como la luz, que orientan en un cierto sentido el crecimiento de la estructura. Para simular la presencia de gravedad, cuyo efecto sobre una estructura ramificada sería el de favorecer el crecimiento hacia arriba y tirar de las ramas hacia el suelo, en nuestro sistema emplearemos el símbolo " $t$ " (símbolo de nuestro alfabeto) de la siguiente forma:

| | |
|---|---|
| $t$ | Realiza una corrección por gravedad de valor 0,2 |
| $t\ (\ x\ )$ | Realiza una corrección por gravedad de valor x. El valor de x se determina al momento de establecer las reglas de reescritura. |

Es importante aclarar que el efecto de la gravedad se aplica sea cual sea el ángulo a partir del cual se empieza a generar el L-Sistema, es decir, aunque la planta nazca inclinada, crecerá hacia arriba y las ramas caerán hacia abajo según la dirección de la gravedad.

Supongamos que aplicamos corrección por gravedad a cada tramo de rama del ejemplo de la Planta 2 de un L-Sistema ramificado, presentado anteriormente.

| | |
|---|---|
| Iteraciones: | 5 |
| Ángulo: | $30°$ |
| Axioma: | $F$ |
| Regla 1: | $F = t(0.1)F[\sim(30) - t(0.1)F]t(0.1)F[+t(0.1)F][F]$ |

Tabla 15: Código corrección por gravedad Planta 2. Iteración 5.

En las siguientes imágenes se muestran las construcciones desde la iteración 1 hasta la iteración 5 con el programa Turtle Graphics Renderer on-line, así:

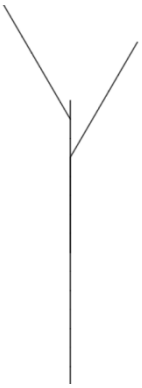


Imagen 45: Planta 2 Corrección por gravedad - Iteración 1.

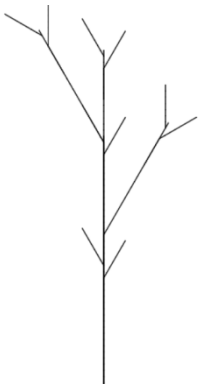


Imagen 46: Planta 2 Corrección por gravedad - Iteración 2.

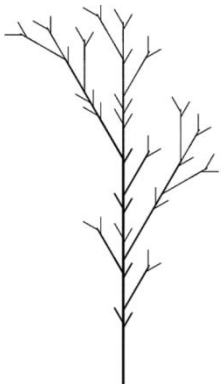


Imagen 47: Planta 2 Corrección por gravedad - Iteración 3.

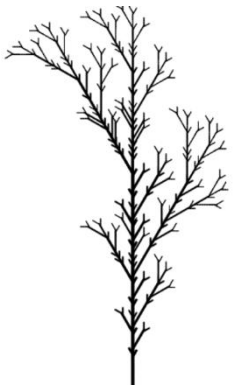


Imagen 48: Planta 2 Corrección por gravedad - Iteración 4.

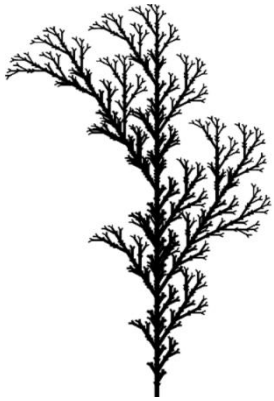


Imagen 49: Planta 2 Corrección por gravedad - Iteración 5.

Como podemos observar, las ramas se deforman simulando la presencia de gravedad y el efecto es más notable cuanto mayor es la longitud de las mismas. La deformación no se realizará aplicando un giro constante en cada tramo de la rama, sino que este giro dependerá de la orientación de la misma siendo máximo cuando la rama se encuentre en posición horizontal y nulo en el caso de que la rama se encuentre en posición vertical. También se puede apreciar una vez más las distintas orientaciones que tienen las ramas debido a la aleatoriedad introducida, de la cual se habló anteriormente.

### 4.10. Ejemplos de L-Sistemas con color.

En las siguientes imágenes se muestran algunas de las construcciones realizadas en las que se simulan procesos de crecimiento, incluyendo el comando para agregar un poco de color a las representaciones de los tallos y las hojas mediante el uso del símbolo "$C$" (símbolo de nuestro alfabeto) acompañado del color elegido de la siguiente forma:

| | |
|---|---|
| $C0$ | Aplica color "café" a la sección definida por el usuario. |
| $C1$ | Aplica color "verde oscuro" a la sección definida por el usuario. |
| $C2$ | Aplica color "verde claro" a la sección definida por el usuario. |
| $C3$ | Aplica color "verde más claro" a la sección definida por el usuario. |

Con la presentación de estos ejemplos se espera que el usuario ponga en juego sus capacidades para reproducir, modificar y crear diferentes estructuras en las que se pueden identificar patrones básicos de desarrollo o simular procesos de crecimiento, aprovechando las bondades del programa Turtle Graphics Renderer on-line ya que permite cambiar numerosos aspectos, tales como el número de iteraciones, el ángulo, las constantes, el axioma definido y hasta cinco reglas de reescritura diferentes. Además, se pueden explorar los ejemplos de la página web donde es posible interactuar con los algoritmos para observar cómo los fractales pueden relacionarse con la naturaleza.

L-Sistema: Planta 2

Iteraciones: 5

Ángulo: $30°$

Axioma: $F$

Regla 1: $F = C0F[-C1F]C0F[+C1F][C1F]$

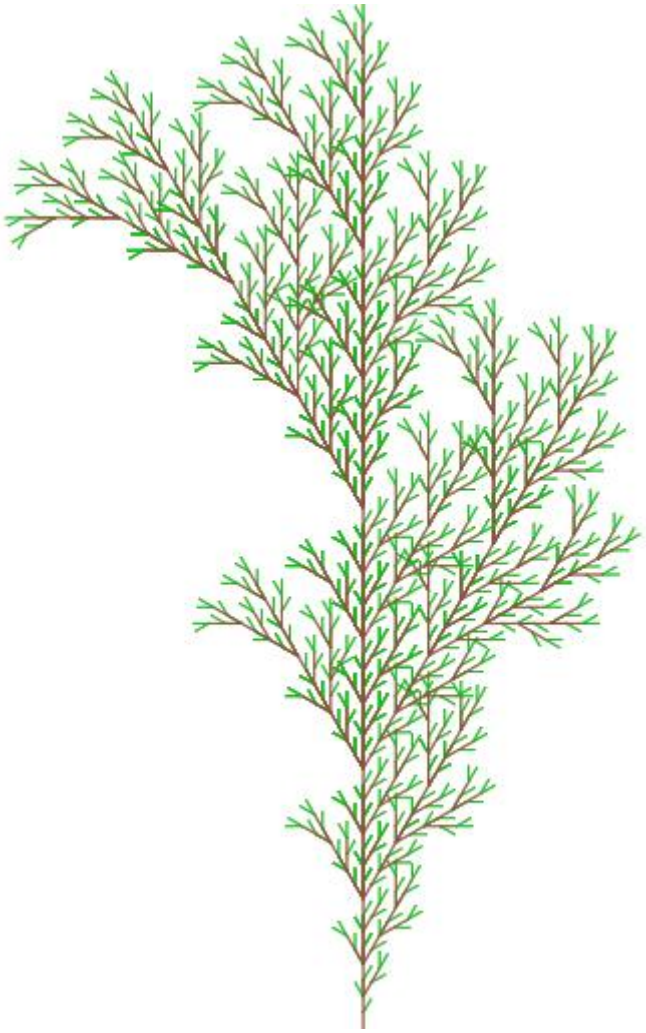


Imagen 50: Construcción Planta 2 Color - Iteración 5.

L-Sistema: Planta 3

Iteraciones: 7

Ángulo: $18°$

Axioma: $X$

Regla 1: $X = C0F[+C1X]C0F[-C1X] + C2X$

Regla 2: $F = FF$

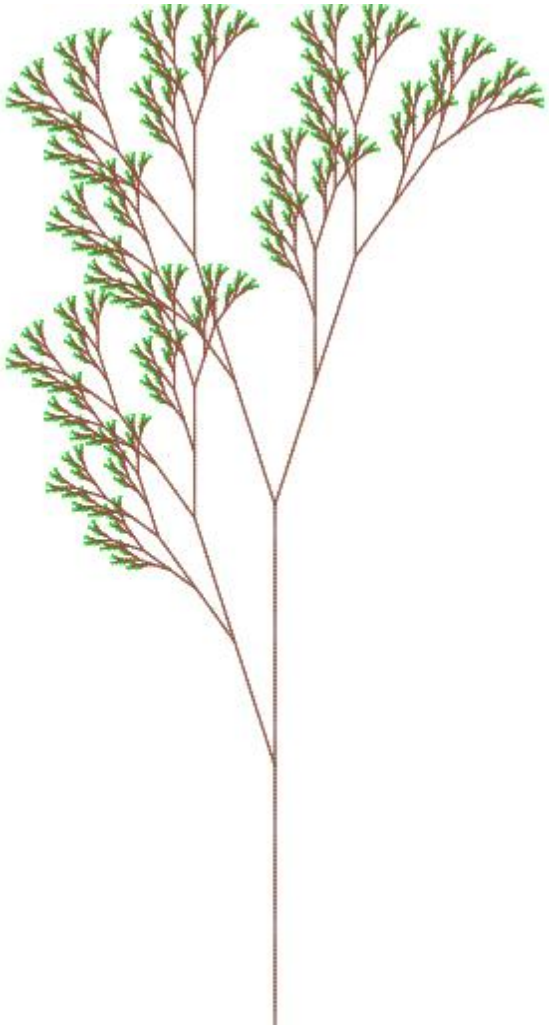


Imagen 51: Construcción Planta 3 Color - Iteración 7.

L-Sistema: Planta 4

Iteraciones: 5

Ángulo: $25°$

Axioma: $FX$

Regla 1: $F = C0FF - [C1 - F + F] + [C2 + F - F]$

Regla 2: $X = C0FF + [C1 + F] + [C3 - F]$

Imagen 52: Construcción Planta 4 Color - Iteración 5.

| | |
|---|---|
| L-Sistema: | Planta 5 |
| Iteraciones: | 6 |
| Ángulo: | $25°$ |
| Constante: | $X$ |
| Axioma: | $X$ |
| Regla 1: | $X = C0F - [C2[X] + C3X] + C1F[C3 + FX] - X$ |
| Regla 2: | $F = FF$ |

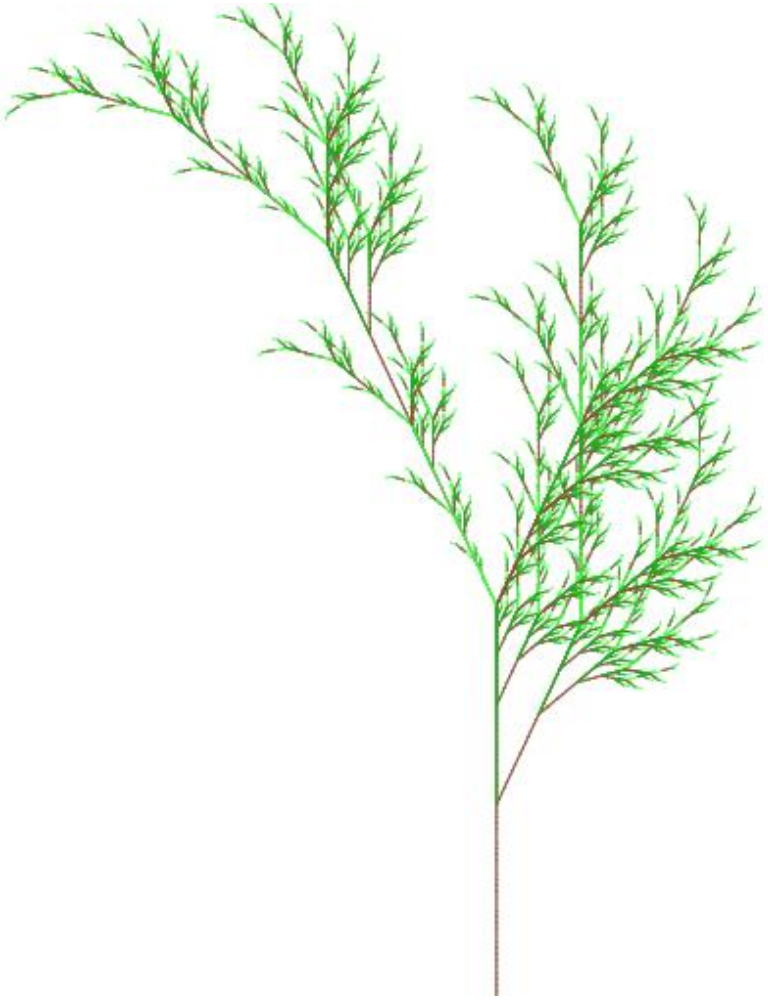


Imagen 53: Construcción Planta 5 Color - Iteración 6.

L-Sistema: Arbusto 1

Iteraciones: 5

Ángulo: $22{,}5°$

Axioma: $F$

Regla 1: $F = C0FF - [C1 - F + F + F] + [C2 + F - F - F]$

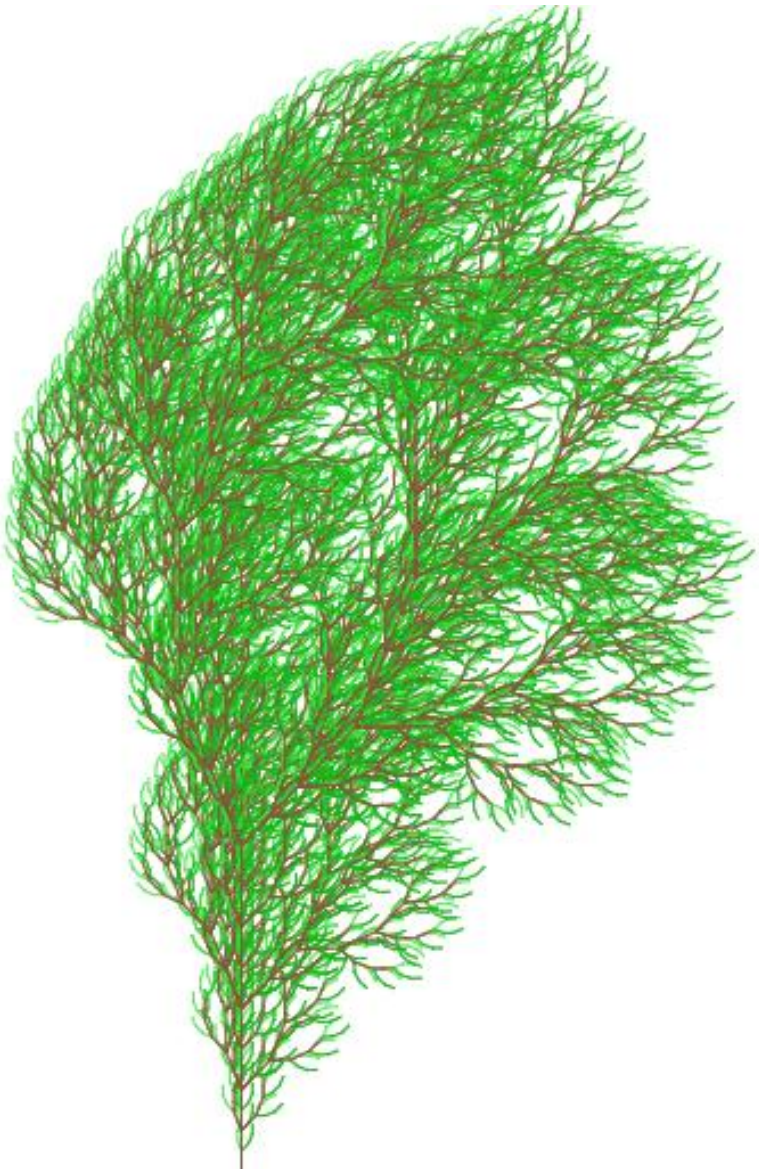


Imagen 54: Construcción Arbusto 1 Color - Iteración 5.

L-Sistema: Carpeta de Sierpinski

Iteraciones: 4

Ángulo: $90°$

Axioma: $F$

Regla 1: $F = C0F + F - F - F - G + C1F + F + F - F$

Regla 2: $G = GGG$

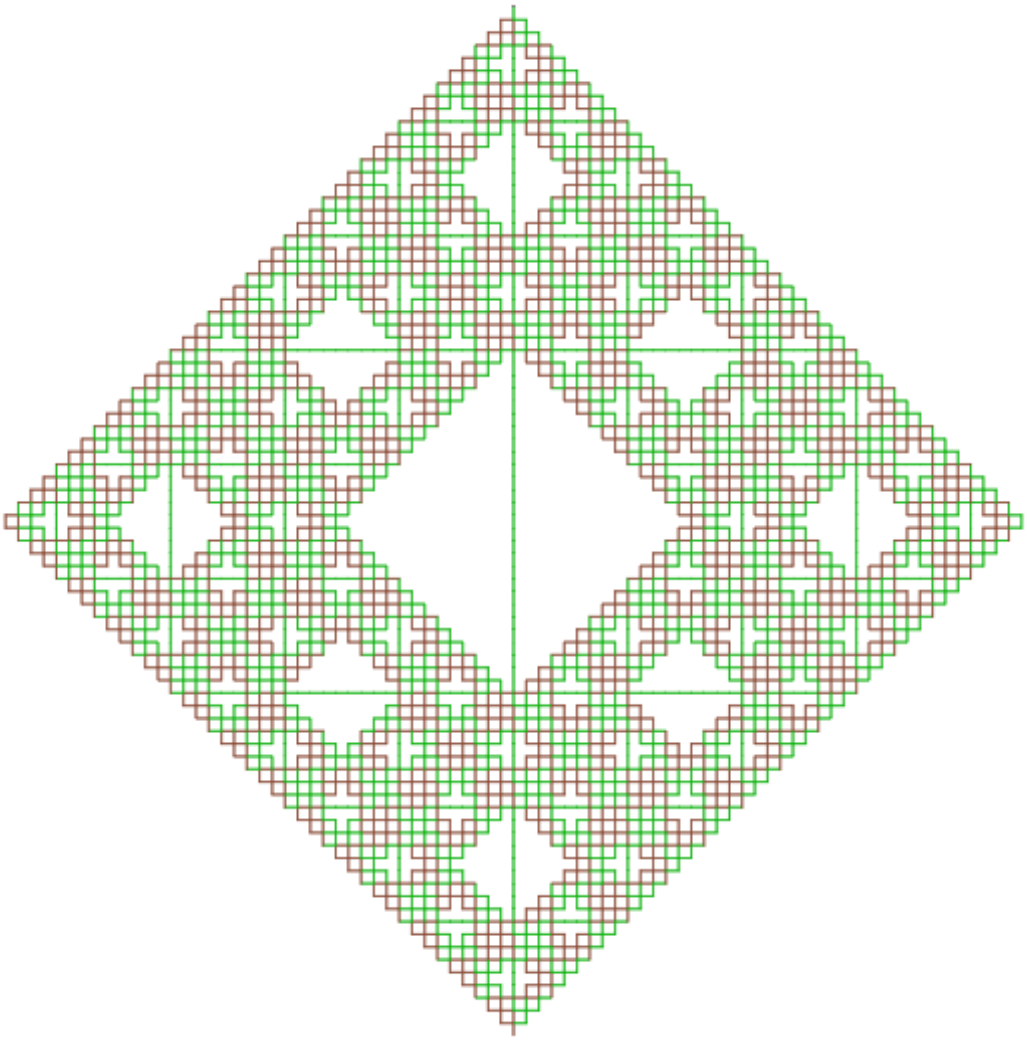


Imagen 55: Construcción Carpeta de Sierpinski Color - Iteración 4.

### 4.11. Algoritmos de las construcciones presentadas.

Luego de representar algunas simulaciones de los procesos de crecimiento de las plantas y algunos fractales lineales mediante el uso del programa Turtle Graphics Renderer on-line, a continuación se presentan los algoritmos extraídos del programa (código fuente), aclarando que dichos algoritmos no pueden ser exportados como un código html para ser ejecutados desde otra plataforma distinta a la que se brinda en línea.

| | |
|---|---|
| L-Sistema: | Curva de Koch |
| Iteraciones: | Desde 0 hasta 5 |
| Ángulo: | $60°$ |
| Axioma: | $F$ |
| Regla 1: | $F = F + F - -F + F$ |

```
<!doctype html>
<html>
<head>
<title>L-Systems Turtle Graphics Renderer - HTML5 Canvas - by Kevin Roast</title>
<script src="scripts/lsystems.js?v=1"></script>
<script async defer src="http://s7.addthis.com/js/250/addthis_widget.js#pubid=xa-4e4813104686cad8"></script>
<script>
function openHelp()
{document.getElementById("help").style.display = "block";}
function closeHelp()
{document.getElementById("help").style.display = "none";}
</script>
<style type="text/css">
Body
{background-color: #ddd;
font-family: GeezaPro,Arial,Helvetica;
color: darkblue;
font-size: 0.9em;}
a, a:visited, a:active, a:hover
{color: #000044;}
```

```
.example
{border: 1px solid lightgrey;
cursor: pointer;
-moz-box-shadow: 3px 3px 4px #444;
-webkit-box-shadow: 3px 3px 4px #444;
box-shadow: 3px 3px 4px #444;}
.examples-title
{height: 24px;
text-align: center;}
.examples
{float: right;
overflow-y: auto;
overflow-x: hidden;
padding-right: 18px;
padding-top: 42px;
xborder: 1px solid darkgrey;}
.ex-left
{float: left;
padding-right: 6px;}
.ex-right
{width: 250px;
margin-bottom: 2px;}
Canvas
{background-color: white;
border: 1px solid grey;
-moz-box-shadow: 3px 3px 6px #444;
-webkit-box-shadow: 3px 3px 6px #444;
box-shadow: 3px 3px 6px #444;}
input[type="text"]
{border: none;
-moz-box-shadow: inset 2px 2px 5px #888;
-webkit-box-shadow: inset 2px 2px 5px #888;
box-shadow: inset 2px 2px 5px #888;
-moz-border-radius: 3px;
-webkit-border-radius: 3px;
border-radius: 3px;
margin: 1px;
padding: 5px;}
</style>
```

```
</head> <body>
<div style="position:absolute;right:1em;min-width:12em">
<!-- AddThis Button BEGIN -->
<div class="addthis_toolbox addthis_default_style addthis_32x32_style">
<a class="addthis_button_preferred_1"></a>
<a class="addthis_button_preferred_2"></a>
<a class="addthis_button_compact"></a>
<a class="addthis_counter addthis_bubble_style"></a> <div>
<!-- AddThis Button END --> <div>
<div id="lsystems" style="width:512px; float:left">
<div style="position:absolute;left:380px;top:32px">
<a href="#" onclick="openHelp()">What are L-Systems?</a> <div>
<div style="position:absolute;left:338px;top:64px">
<a href="../index.html">More HTML5 Canvas demos</a> </div> <div>
<table border="0" cellspacing="1" cellpadding="1"> <tr>
<td><span>Iterations:</span></td> <td>
<input id="iterations" type="text" value="1" style="width:4em" />
<div style="display:inline"> <input id="linewidths" type="checkbox" />
<label for="linewidths">Auto Line Width</label> </div> </td> </tr> <tr>
<td><span>Angle:</span></td>
<td><input id="angle" type="text" value="60" style="width:4em" /></td>
</tr> <tr> <td><span>Constants:</span></td>
<td><input id="constants" type="text" value="" style="width:12em" /></td>
</tr> <tr> <td><span>Axoim:</span></td>
<td><input id="axiom" type="text" value="F" style="width:12em"/></td>
</tr> <tr> <td><span>Rule1:</span></td>
<td><input id="rule1" type="text" value="F=F+F--F+F" style="width:30em" /></td>
</tr> <tr> <td><span>Rule2:</span></td>
<td><input id="rule2" type="text" value="" style="width:30em" /></td>
</tr> <tr> <td colspan="2"><input id="start" type="button" value="Start"
onclick="startHandler()"/> </td>
</tr> </table> </div>
<div id="status" style="padding:4px;font-family:courier new">Press Start to begin.</div>
<div> <canvas id="canvas" width="512" height="512"></canvas>
</div> </div> </div>
</body>
</html>
```

| | |
|---|---|
| L-Sistema: | Copo de nieve de Koch |
| Iteraciones: | Desde 0 hasta 9 |
| Ángulo: | $60°$ |
| Axioma: | $F++F++F$ |
| Regla 1: | $F = F - F++F - F$ |

```
<!doctype html>
<html>
<head>
<title>L-Systems Turtle Graphics Renderer - HTML5 Canvas - by Kevin Roast</title>
<script src="scripts/lsystems.js?v=1"></script>
<script async defer src="http://s7.addthis.com/js/250/addthis_widget.js#pubid=xa-4e4813104686cad8"></script>
<script>
function openHelp()
{document.getElementById("help").style.display = "block";}
function closeHelp()
{document.getElementById("help").style.display = "none";}
</script>
<style type="text/css">
Body
{background-color: #ddd;
font-family: GeezaPro,Arial,Helvetica;
color: darkblue;
font-size: 0.9em;}
a, a:visited, a:active, a:hover
{color: #000044;}
.example
{border: 1px solid lightgrey;
cursor: pointer;
-moz-box-shadow: 3px 3px 4px #444;
-webkit-box-shadow: 3px 3px 4px #444;
box-shadow: 3px 3px 4px #444;}
.examples-title
{height: 24px;
text-align: center;}
.examples
{float: right;
```

```
overflow-y: auto;
overflow-x: hidden;
padding-right: 18px;
padding-top: 42px;
xborder: 1px solid darkgrey;}
.ex-left
{float: left;
padding-right: 6px;}
.ex-right
{width: 250px;
margin-bottom: 2px;}
Canvas
{background-color: white;
border: 1px solid grey;
-moz-box-shadow: 3px 3px 6px #444;
-webkit-box-shadow: 3px 3px 6px #444;
box-shadow: 3px 3px 6px #444;}
input[type="text"]
{border: none;
-moz-box-shadow: inset 2px 2px 5px #888;
-webkit-box-shadow: inset 2px 2px 5px #888;
box-shadow: inset 2px 2px 5px #888;
-moz-border-radius: 3px;
-webkit-border-radius: 3px;
border-radius: 3px;
margin: 1px;
padding: 5px;}
</style>
</head> <body>
<div style="position:absolute;right:1em;min-width:12em">
<!-- AddThis Button BEGIN -->
<div class="addthis_toolbox addthis_default_style addthis_32x32_style">
<a class="addthis_button_preferred_1"></a>
<a class="addthis_button_preferred_2"></a>
<a class="addthis_button_compact"></a>
<a class="addthis_counter addthis_bubble_style"></a> <div>
<!-- AddThis Button END --> <div>
<div id="lsystems" style="width:512px; float:left">
<div style="position:absolute;left:380px;top:32px">
```

<a href="#" onclick="openHelp()">What are L-Systems?</a> 

<a href="../index.html">More HTML5 Canvas demos</a>  
<table border="0" cellspacing="1" cellpadding="1"> <tr>
<td>Iterations:</td> <td>
<input id="iterations" type="text" value="1" style="width:4em" />
 <input id="linewidths" type="checkbox" />
<label for="linewidths">Auto Line Width</label>  </td> </tr> <tr>
<td>Angle:</td>
<td><input id="angle" type="text" value="60" style="width:4em" /></td>
</tr> <tr> <td>Constants:</td>
<td><input id="constants" type="text" value="" style="width:12em" /></td>
</tr> <tr> <td>Axoim:</td>
<td><input id="axiom" type="text" value="F++F++F" style="width:12em"/></td>
</tr> <tr> <td>Rule1:</td>
<td><input id="rule1" type="text" value="F=F-F++F-F" style="width:30em" /></td>
</tr> <tr> <td>Rule2:</td>
<td><input id="rule2" type="text" value="" style="width:30em" /></td>
</tr> <tr> <td colspan="2"><input id="start" type="button" value="Start" onclick="startHandler()"/> </td>
</tr> </table> 
Press Start to begin.
 <canvas id="canvas" width="512" height="512"></canvas>

</body>
</html>

| | |
|---|---|
| L-Sistema: | Triángulo de Sierpinski |
| Iteraciones: | Desde 0 hasta 5 |
| Ángulo: | $120°$ |
| Axioma: | $F - G - G$ |
| Regla 1: | $F = F - G + F + G - F$ |
| Regla 2: | $G = GG$ |

<!doctype html>
<html>
<head>
<title>L-Systems Turtle Graphics Renderer - HTML5 Canvas - by Kevin Roast</title>
<script src="scripts/lsystems.js?v=1"></script>
<script async defer src="http://s7.addthis.com/js/250/addthis_widget.js#pubid=xa-

```
4e4813104686cad8"></script>
<script>
function openHelp()
{document.getElementById("help").style.display = "block";}
function closeHelp()
{document.getElementById("help").style.display = "none";}
</script>
<style type="text/css">
Body
{background-color: #ddd;
font-family: GeezaPro,Arial,Helvetica;
color: darkblue;
font-size: 0.9em;}
a, a:visited, a:active, a:hover
{color: #000044;}
.example
{border: 1px solid lightgrey;
cursor: pointer;
-moz-box-shadow: 3px 3px 4px #444;
-webkit-box-shadow: 3px 3px 4px #444;
box-shadow: 3px 3px 4px #444;}
.examples-title
{height: 24px;
text-align: center;}
.examples
{float: right;
overflow-y: auto;
overflow-x: hidden;
padding-right: 18px;
padding-top: 42px;
xborder: 1px solid darkgrey;}
.ex-left
{float: left;
padding-right: 6px;}
.ex-right
{width: 250px;
margin-bottom: 2px;}
Canvas
{background-color: white;
```

```
border: 1px solid grey;
-moz-box-shadow: 3px 3px 6px #444;
-webkit-box-shadow: 3px 3px 6px #444;
box-shadow: 3px 3px 6px #444;}
input[type="text"]
{border: none;
-moz-box-shadow: inset 2px 2px 5px #888;
-webkit-box-shadow: inset 2px 2px 5px #888;
box-shadow: inset 2px 2px 5px #888;
-moz-border-radius: 3px;
-webkit-border-radius: 3px;
border-radius: 3px;
margin: 1px;
padding: 5px;}
</style>
</head> <body>
<div style="position:absolute;right:1em;min-width:12em">
<!-- AddThis Button BEGIN -->
<div class="addthis_toolbox addthis_default_style addthis_32x32_style">
<a class="addthis_button_preferred_1"></a>
<a class="addthis_button_preferred_2"></a>
<a class="addthis_button_compact"></a>
<a class="addthis_counter addthis_bubble_style"></a> <div>
<!-- AddThis Button END --> <div>
<div id="lsystems" style="width:512px; float:left">
<div style="position:absolute;left:380px;top:32px">
<a href="#" onclick="openHelp()">What are L-Systems?</a> <div>
<div style="position:absolute;left:338px;top:64px">
<a href="../index.html">More HTML5 Canvas demos</a> </div> <div>
<table border="0" cellspacing="1" cellpadding="1"> <tr>
<td><span>Iterations:</span></td> <td>
<input id="iterations" type="text" value="1" style="width:4em" />
<div style="display:inline"> <input id="linewidths" type="checkbox" />
<label for="linewidths">Auto Line Width</label> </div> </td> </tr> <tr>
<td><span>Angle:</span></td>
<td><input id="angle" type="text" value="120" style="width:4em" /></td>
</tr> <tr> <td><span>Constants:</span></td>
<td><input id="constants" type="text" value="" style="width:12em" /></td>
</tr> <tr> <td><span>Axoim:</span></td>
```

<td><input id="axiom" type="text" value="F-G-G" style="width:12em"/></td>
</tr> <tr> <td>Rule1:</td>
<td><input id="rule1" type="text" value="F=F-G+F+G-F" style="width:30em" /></td>
</tr> <tr> <td>Rule2:</td>
<td><input id="rule2" type="text" value="G=GG" style="width:30em" /></td>
</tr> <tr> <td colspan="2"><input id="start" type="button" value="Start" onclick="startHandler()"/> </td>
</tr> </table> 
Press Start to begin.
 <canvas id="canvas" width="512" height="512"></canvas>

</body>
</html>

| | |
|---|---|
| L-Sistema: | Planta 1 |
| Iteraciones: | Desde 1 hasta 6 |
| Ángulo: | $36°$ |
| Axioma: | $F$ |
| Regla 1: | $F = F[+F]F$ |

<!doctype html>
<html>
<head>
<title>L-Systems Turtle Graphics Renderer - HTML5 Canvas - by Kevin Roast</title>
<script src="scripts/lsystems.js?v=1"></script>
<script async defer src="http://s7.addthis.com/js/250/addthis_widget.js#pubid=xa-4e4813104686cad8"></script>
<script>
function openHelp()
{document.getElementById("help").style.display = "block";}
function closeHelp()
{document.getElementById("help").style.display = "none";}
</script>
<style type="text/css">
Body
{background-color: #ddd;
font-family: GeezaPro,Arial,Helvetica;
color: darkblue;
font-size: 0.9em;}
a, a:visited, a:active, a:hover
{color: #000044;}

```
.example
{border: 1px solid lightgrey;
cursor: pointer;
-moz-box-shadow: 3px 3px 4px #444;
-webkit-box-shadow: 3px 3px 4px #444;
box-shadow: 3px 3px 4px #444;}
.examples-title
{height: 24px;
text-align: center;}
.examples
{float: right;
overflow-y: auto;
overflow-x: hidden;
padding-right: 18px;
padding-top: 42px;
xborder: 1px solid darkgrey;}
.ex-left
{float: left;
padding-right: 6px;}
.ex-right
{width: 250px;
margin-bottom: 2px;}
Canvas
{background-color: white;
border: 1px solid grey;
-moz-box-shadow: 3px 3px 6px #444;
-webkit-box-shadow: 3px 3px 6px #444;
box-shadow: 3px 3px 6px #444;}
input[type="text"]
{border: none;
-moz-box-shadow: inset 2px 2px 5px #888;
-webkit-box-shadow: inset 2px 2px 5px #888;
box-shadow: inset 2px 2px 5px #888;
-moz-border-radius: 3px;
-webkit-border-radius: 3px;
border-radius: 3px;
margin: 1px;
padding: 5px;}
</style>
```

```
</head> <body>
<div style="position:absolute;right:1em;min-width:12em">
<!-- AddThis Button BEGIN -->
<div class="addthis_toolbox addthis_default_style addthis_32x32_style">
<a class="addthis_button_preferred_1"></a>
<a class="addthis_button_preferred_2"></a>
<a class="addthis_button_compact"></a>
<a class="addthis_counter addthis_bubble_style"></a> <div>
<!-- AddThis Button END --> <div>
<div id="lsystems" style="width:512px; float:left">
<div style="position:absolute;left:380px;top:32px">
<a href="#" onclick="openHelp()">What are L-Systems?</a> <div>
<div style="position:absolute;left:338px;top:64px">
<a href="../index.html">More HTML5 Canvas demos</a> </div> <div>
<table border="0" cellspacing="1" cellpadding="1"> <tr>
<td><span>Iterations:</span></td> <td>
<input id="iterations" type="text" value="1" style="width:4em" />
<div style="display:inline"> <input id="linewidths" type="checkbox" />
<label for="linewidths">Auto Line Width</label> </div> </td> </tr> <tr>
<td><span>Angle:</span></td>
<td><input id="angle" type="text" value="36" style="width:4em" /></td>
</tr> <tr> <td><span>Constants:</span></td>
<td><input id="constants" type="text" value="" style="width:12em" /></td>
</tr> <tr> <td><span>Axoim:</span></td>
<td><input id="axiom" type="text" value="F" style="width:12em"/></td>
</tr> <tr> <td><span>Rule1:</span></td>
<td><input id="rule1" type="text" value="F=F[+F]F" style="width:30em" /></td>
</tr> <tr> <td><span>Rule2:</span></td>
<td><input id="rule2" type="text" value="" style="width:30em" /></td>
</tr> <tr> <td colspan="2"><input id="start" type="button" value="Start"
onclick="startHandler()"/> </td>
</tr> </table> </div>
<div id="status" style="padding:4px;font-family:courier new">Press Start to begin.</div>
<div> <canvas id="canvas" width="512" height="512"></canvas>
</div> </div> </div>
</body>
</html>
```

| | |
|---|---|
| L-Sistema: | Planta 2 |
| Iteraciones: | Desde 1 hasta 5 |
| Ángulo: | $30°$ |
| Axioma: | $F$ |
| Regla 1: | $F = F[-F]F[+F][F]$ |

```
<!doctype html>
<html>
<head>
<title>L-Systems Turtle Graphics Renderer - HTML5 Canvas - by Kevin Roast</title>
<script src="scripts/lsystems.js?v=1"></script>
<script async defer src="http://s7.addthis.com/js/250/addthis_widget.js#pubid=xa-4e4813104686cad8"></script>
<script>
function openHelp()
{document.getElementById("help").style.display = "block";}
function closeHelp()
{document.getElementById("help").style.display = "none";}
</script>
<style type="text/css">
Body
{background-color: #ddd;
font-family: GeezaPro,Arial,Helvetica;
color: darkblue;
font-size: 0.9em;}
a, a:visited, a:active, a:hover
{color: #000044;}
.example
{border: 1px solid lightgrey;
cursor: pointer;
-moz-box-shadow: 3px 3px 4px #444;
-webkit-box-shadow: 3px 3px 4px #444;
box-shadow: 3px 3px 4px #444;}
.examples-title
{height: 24px;
text-align: center;}
.examples
{float: right;
overflow-y: auto;
```

```
overflow-x: hidden;
padding-right: 18px;
padding-top: 42px;
xborder: 1px solid darkgrey;}
.ex-left
{float: left;
padding-right: 6px;}
.ex-right
{width: 250px;
margin-bottom: 2px;}
Canvas
{background-color: white;
border: 1px solid grey;
-moz-box-shadow: 3px 3px 6px #444;
-webkit-box-shadow: 3px 3px 6px #444;
box-shadow: 3px 3px 6px #444;}
input[type="text"]
{border: none;
-moz-box-shadow: inset 2px 2px 5px #888;
-webkit-box-shadow: inset 2px 2px 5px #888;
box-shadow: inset 2px 2px 5px #888;
-moz-border-radius: 3px;
-webkit-border-radius: 3px;
border-radius: 3px;
margin: 1px;
padding: 5px;}
</style>
</head> <body>
<div style="position:absolute;right:1em;min-width:12em">
<!-- AddThis Button BEGIN -->
<div class="addthis_toolbox addthis_default_style addthis_32x32_style">
<a class="addthis_button_preferred_1"></a>
<a class="addthis_button_preferred_2"></a>
<a class="addthis_button_compact"></a>
<a class="addthis_counter addthis_bubble_style"></a> <div>
<!-- AddThis Button END --> <div>
<div id="lsystems" style="width:512px; float:left">
<div style="position:absolute;left:380px;top:32px">
<a href="#" onclick="openHelp()">What are L-Systems?</a> <div>
```

<a href="../index.html">More HTML5 Canvas demos</a>  
<table border="0" cellspacing="1" cellpadding="1"> <tr>
<td>Iterations:</td> <td>
<input id="iterations" type="text" value="1" style="width:4em" />
 <input id="linewidths" type="checkbox" />
<label for="linewidths">Auto Line Width</label>  </td> </tr> <tr>
<td>Angle:</td>
<td><input id="angle" type="text" value="30" style="width:4em" /></td>
</tr> <tr> <td>Constants:</td>
<td><input id="constants" type="text" value="" style="width:12em" /></td>
</tr> <tr> <td>Axoim:</td>
<td><input id="axiom" type="text" value="F" style="width:12em"/></td>
</tr> <tr> <td>Rule1:</td>
<td><input id="rule1" type="text" value="F=F[-F]F[+F][F]" style="width:30em" /></td>
</tr> <tr> <td>Rule2:</td>
<td><input id="rule2" type="text" value="" style="width:30em" /></td>
</tr> <tr> <td colspan="2"><input id="start" type="button" value="Start" onclick="startHandler()"/> </td>
</tr> </table> 
Press Start to begin.
 <canvas id="canvas" width="512" height="512"></canvas>

</body>
</html>

| | |
|---|---|
| L-Sistema: | Planta 3 |
| Iteraciones: | 6 |
| Ángulo: | $18°$ |
| Axioma: | $X$ |
| Regla 1: | $X = F[+X]F[-X] + X$ |
| Regla 2: | $F = FF$ |

<!doctype html>
<html>
<head>
<title>L-Systems Turtle Graphics Renderer - HTML5 Canvas - by Kevin Roast</title>
<script src="scripts/lsystems.js?v=1"></script>
<script async defer src="http://s7.addthis.com/js/250/addthis_widget.js#pubid=xa-4e4813104686cad8"></script>

```
<script>
function openHelp()
{document.getElementById("help").style.display = "block";}
function closeHelp()
{document.getElementById("help").style.display = "none";}
</script>
<style type="text/css">
Body
{background-color: #ddd;
font-family: GeezaPro,Arial,Helvetica;
color: darkblue;
font-size: 0.9em;}
a, a:visited, a:active, a:hover
{color: #000044;}
.example
{border: 1px solid lightgrey;
cursor: pointer;
-moz-box-shadow: 3px 3px 4px #444;
-webkit-box-shadow: 3px 3px 4px #444;
box-shadow: 3px 3px 4px #444;}
.examples-title
{height: 24px;
text-align: center;}
.examples
{float: right;
overflow-y: auto;
overflow-x: hidden;
padding-right: 18px;
padding-top: 42px;
xborder: 1px solid darkgrey;}
.ex-left
{float: left;
padding-right: 6px;}
.ex-right
{width: 250px;
margin-bottom: 2px;}
Canvas
{background-color: white;
border: 1px solid grey;
```

```
-moz-box-shadow: 3px 3px 6px #444;
-webkit-box-shadow: 3px 3px 6px #444;
box-shadow: 3px 3px 6px #444;}
input[type="text"]
{border: none;
-moz-box-shadow: inset 2px 2px 5px #888;
-webkit-box-shadow: inset 2px 2px 5px #888;
box-shadow: inset 2px 2px 5px #888;
-moz-border-radius: 3px;
-webkit-border-radius: 3px;
border-radius: 3px;
margin: 1px;
padding: 5px;}
</style>
</head> <body>
<div style="position:absolute;right:1em;min-width:12em">
<!-- AddThis Button BEGIN -->
<div class="addthis_toolbox addthis_default_style addthis_32x32_style">
<a class="addthis_button_preferred_1"></a>
<a class="addthis_button_preferred_2"></a>
<a class="addthis_button_compact"></a>
<a class="addthis_counter addthis_bubble_style"></a> <div>
<!-- AddThis Button END --> <div>
<div id="lsystems" style="width:512px; float:left">
<div style="position:absolute;left:380px;top:32px">
<a href="#" onclick="openHelp()">What are L-Systems?</a> <div>
<div style="position:absolute;left:338px;top:64px">
<a href="../index.html">More HTML5 Canvas demos</a> </div> <div>
<table border="0" cellspacing="1" cellpadding="1"> <tr>
<td><span>Iterations:</span></td> <td>
<input id="iterations" type="text" value="6" style="width:4em" />
<div style="display:inline"> <input id="linewidths" type="checkbox" />
<label for="linewidths">Auto Line Width</label> </div> </td> </tr> <tr>
<td><span>Angle:</span></td>
<td><input id="angle" type="text" value="18" style="width:4em" /></td>
</tr> <tr> <td><span>Constants:</span></td>
<td><input id="constants" type="text" value="" style="width:12em" /></td>
</tr> <tr> <td><span>Axoim:</span></td>
<td><input id="axiom" type="text" value="X" style="width:12em"/></td>
```

</tr> <tr> <td>Rule1:</td>
<td><input id="rule1" type="text" value= "X = F[+X]F[−X] + X" style="width:30em" /></td>
</tr> <tr> <td>Rule2:</td>
<td><input id="rule2" type="text" value="F=FF" style="width:30em" /></td>
</tr> <tr> <td colspan="2"><input id="start" type="button" value="Start" onclick="startHandler()"/> </td>
</tr> </table> 
Press Start to begin.
 <canvas id="canvas" width="512" height="512"></canvas>

</body>
</html>

| | |
|---|---|
| L-Sistema: | Arbusto 1 |
| Iteraciones: | 5 |
| Ángulo: | $22{,}5°$ |
| Axioma: | $F$ |
| Regla 1: | $F = FF - [-F + F + F] + [+F - F - F]$ |

<!doctype html>
<html>
<head>
<title>L-Systems Turtle Graphics Renderer - HTML5 Canvas - by Kevin Roast</title>
<script src="scripts/lsystems.js?v=1"></script>
<script async defer src="http://s7.addthis.com/js/250/addthis_widget.js#pubid=xa-4e4813104686cad8"></script>
<script>
function openHelp()
{document.getElementById("help").style.display = "block";}
function closeHelp()
{document.getElementById("help").style.display = "none";}
</script>
<style type="text/css">
Body
{background-color: #ddd;
font-family: GeezaPro,Arial,Helvetica;
color: darkblue;
font-size: 0.9em;}
a, a:visited, a:active, a:hover
{color: #000044;}

```
.example
{border: 1px solid lightgrey;
cursor: pointer;
-moz-box-shadow: 3px 3px 4px #444;
-webkit-box-shadow: 3px 3px 4px #444;
box-shadow: 3px 3px 4px #444;}
.examples-title
{height: 24px;
text-align: center;}
.examples
{float: right;
overflow-y: auto;
overflow-x: hidden;
padding-right: 18px;
padding-top: 42px;
xborder: 1px solid darkgrey;}
.ex-left
{float: left;
padding-right: 6px;}
.ex-right
{width: 250px;
margin-bottom: 2px;}
Canvas
{background-color: white;
border: 1px solid grey;
-moz-box-shadow: 3px 3px 6px #444;
-webkit-box-shadow: 3px 3px 6px #444;
box-shadow: 3px 3px 6px #444;}
input[type="text"]
{border: none;
-moz-box-shadow: inset 2px 2px 5px #888;
-webkit-box-shadow: inset 2px 2px 5px #888;
box-shadow: inset 2px 2px 5px #888;
-moz-border-radius: 3px;
-webkit-border-radius: 3px;
border-radius: 3px;
margin: 1px;
padding: 5px;}
</style>
```

```
</head> <body>
<div style="position:absolute;right:1em;min-width:12em">
<!-- AddThis Button BEGIN -->
<div class="addthis_toolbox addthis_default_style addthis_32x32_style">
<a class="addthis_button_preferred_1"></a>
<a class="addthis_button_preferred_2"></a>
<a class="addthis_button_compact"></a>
<a class="addthis_counter addthis_bubble_style"></a> <div>
<!-- AddThis Button END --> <div>
<div id="lsystems" style="width:512px; float:left">
<div style="position:absolute;left:380px;top:32px">
<a href="#" onclick="openHelp()">What are L-Systems?</a> <div>
<div style="position:absolute;left:338px;top:64px">
<a href="../index.html">More HTML5 Canvas demos</a> </div> <div>
<table border="0" cellspacing="1" cellpadding="1"> <tr>
<td><span>Iterations:</span></td> <td>
<input id="iterations" type="text" value="5" style="width:4em" />
<div style="display:inline"> <input id="linewidths" type="checkbox" />
<label for="linewidths">Auto Line Width</label> </div> </td> </tr> <tr>
<td><span>Angle:</span></td>
<td><input id="angle" type="text" value="22,5" style="width:4em" /></td>
</tr> <tr> <td><span>Constants:</span></td>
<td><input id="constants" type="text" value="" style="width:12em" /></td>
</tr> <tr> <td><span>Axoim:</span></td>
<td><input id="axiom" type="text" value="F" style="width:12em"/></td>
</tr> <tr> <td><span>Rule1:</span></td>
<td><input id="rule1" type="text" value="F = FF − [−F + F + F] + [+F − F − F]"
style="width:30em" /></td>
</tr> <tr> <td><span>Rule2:</span></td>
<td><input id="rule2" type="text" value="" style="width:30em" /></td>
</tr> <tr> <td colspan="2"><input id="start" type="button" value="Start"
onclick="startHandler()"/> </td>
</tr> </table> </div>
<div id="status" style="padding:4px;font-family:courier new">Press Start to begin.</div>
<div> <canvas id="canvas" width="512" height="512"></canvas>
</div> </div> </div>
</body>
</html>
```

| | |
|---|---|
| L-Sistema: | Planta 2 - Aleatoria |
| Iteraciones: | 5 |
| Ángulo: | $30°$ |
| Axioma: | $F$ |
| Regla 1: | $F = F[\sim(30) - F]F[+F][F]$ |

```
<!doctype html>
<html>
<head>
<title>L-Systems Turtle Graphics Renderer - HTML5 Canvas - by Kevin Roast</title>
<script src="scripts/lsystems.js?v=1"></script>
<script async defer src="http://s7.addthis.com/js/250/addthis_widget.js#pubid=xa-4e4813104686cad8"></script>
<script>
function openHelp()
{document.getElementById("help").style.display = "block";}
function closeHelp()
{document.getElementById("help").style.display = "none";}
</script>
<style type="text/css">
Body
{background-color: #ddd;
font-family: GeezaPro,Arial,Helvetica;
color: darkblue;
font-size: 0.9em;}
a, a:visited, a:active, a:hover
{color: #000044;}
.example
{border: 1px solid lightgrey;
cursor: pointer;
-moz-box-shadow: 3px 3px 4px #444;
-webkit-box-shadow: 3px 3px 4px #444;
box-shadow: 3px 3px 4px #444;}
.examples-title
{height: 24px;
text-align: center;}
.examples
{float: right;
overflow-y: auto;
```

```
overflow-x: hidden;
padding-right: 18px;
padding-top: 42px;
xborder: 1px solid darkgrey;}
.ex-left
{float: left;
padding-right: 6px;}
.ex-right
{width: 250px;
margin-bottom: 2px;}
Canvas
{background-color: white;
border: 1px solid grey;
-moz-box-shadow: 3px 3px 6px #444;
-webkit-box-shadow: 3px 3px 6px #444;
box-shadow: 3px 3px 6px #444;}
input[type="text"]
{border: none;
-moz-box-shadow: inset 2px 2px 5px #888;
-webkit-box-shadow: inset 2px 2px 5px #888;
box-shadow: inset 2px 2px 5px #888;
-moz-border-radius: 3px;
-webkit-border-radius: 3px;
border-radius: 3px;
margin: 1px;
padding: 5px;}
</style>
</head> <body>
<div style="position:absolute;right:1em;min-width:12em">
<!-- AddThis Button BEGIN -->
<div class="addthis_toolbox addthis_default_style addthis_32x32_style">
<a class="addthis_button_preferred_1"></a>
<a class="addthis_button_preferred_2"></a>
<a class="addthis_button_compact"></a>
<a class="addthis_counter addthis_bubble_style"></a> <div>
<!-- AddThis Button END --> <div>
<div id="lsystems" style="width:512px; float:left">
<div style="position:absolute;left:380px;top:32px">
<a href="#" onclick="openHelp()">What are L-Systems?</a> <div>
```

<a href="../index.html">More HTML5 Canvas demos</a>  
<table border="0" cellspacing="1" cellpadding="1"> <tr>
<td>Iterations:</td> <td>
<input id="iterations" type="text" value="5" style="width:4em" />
 <input id="linewidths" type="checkbox" />
<label for="linewidths">Auto Line Width</label>  </td> </tr> <tr>
<td>Angle:</td>
<td><input id="angle" type="text" value="30" style="width:4em" /></td>
</tr> <tr> <td>Constants:</td>
<td><input id="constants" type="text" value="" style="width:12em" /></td>
</tr> <tr> <td>Axoim:</td>
<td><input id="axiom" type="text" value="F" style="width:12em"/></td>
</tr> <tr> <td>Rule1:</td>
<td><input id="rule1" type="text" value="F = F[~(30) − F]F[+F][F]" style="width:30em" /></td>
</tr> <tr> <td>Rule2:</td>
<td><input id="rule2" type="text" value="" style="width:30em" /></td>
</tr> <tr> <td colspan="2"><input id="start" type="button" value="Start" onclick="startHandler()"/> </td>
</tr> </table> 
Press Start to begin.
 <canvas id="canvas" width="512" height="512"></canvas>

</body>
</html>

| | |
|---|---|
| L-Sistema: | Planta 2 - Corrección por gravedad |
| Iteraciones: | 5 |
| Ángulo: | $30°$ |
| Axioma: | $F$ |
| Regla 1: | $F = t(0.1)F[\sim(30) - t(0.1)F]t(0.1)F[+t(0.1)F][F]$ |

<!doctype html>
<html>
<head>
<title>L-Systems Turtle Graphics Renderer - HTML5 Canvas - by Kevin Roast</title>
<script src="scripts/lsystems.js?v=1"></script>
<script async defer src="http://s7.addthis.com/js/250/addthis_widget.js#pubid=xa-4e4813104686cad8"></script>
<script>

```
function openHelp()
{document.getElementById("help").style.display = "block";}
function closeHelp()
{document.getElementById("help").style.display = "none";}
</script>
<style type="text/css">
Body
{background-color: #ddd;
font-family: GeezaPro,Arial,Helvetica;
color: darkblue;
font-size: 0.9em;}
a, a:visited, a:active, a:hover
{color: #000044;}
.example
{border: 1px solid lightgrey;
cursor: pointer;
-moz-box-shadow: 3px 3px 4px #444;
-webkit-box-shadow: 3px 3px 4px #444;
box-shadow: 3px 3px 4px #444;}
.examples-title
{height: 24px;
text-align: center;}
.examples
{float: right;
overflow-y: auto;
overflow-x: hidden;
padding-right: 18px;
padding-top: 42px;
xborder: 1px solid darkgrey;}
.ex-left
{float: left;
padding-right: 6px;}
.ex-right
{width: 250px;
margin-bottom: 2px;}
Canvas
{background-color: white;
border: 1px solid grey;
-moz-box-shadow: 3px 3px 6px #444;
```

```
-webkit-box-shadow: 3px 3px 6px #444;
box-shadow: 3px 3px 6px #444;}
input[type="text"]
{border: none;
-moz-box-shadow: inset 2px 2px 5px #888;
-webkit-box-shadow: inset 2px 2px 5px #888;
box-shadow: inset 2px 2px 5px #888;
-moz-border-radius: 3px;
-webkit-border-radius: 3px;
border-radius: 3px;
margin: 1px;
padding: 5px;}
</style>
</head> <body>
<div style="position:absolute;right:1em;min-width:12em">
<!-- AddThis Button BEGIN -->
<div class="addthis_toolbox addthis_default_style addthis_32x32_style">
<a class="addthis_button_preferred_1"></a>
<a class="addthis_button_preferred_2"></a>
<a class="addthis_button_compact"></a>
<a class="addthis_counter addthis_bubble_style"></a> <div>
<!-- AddThis Button END --> <div>
<div id="lsystems" style="width:512px; float:left">
<div style="position:absolute;left:380px;top:32px">
<a href="#" onclick="openHelp()">What are L-Systems?</a> <div>
<div style="position:absolute;left:338px;top:64px">
<a href="../index.html">More HTML5 Canvas demos</a> </div> <div>
<table border="0" cellspacing="1" cellpadding="1"> <tr>
<td><span>Iterations:</span></td> <td>
<input id="iterations" type="text" value="5" style="width:4em" />
<div style="display:inline"> <input id="linewidths" type="checkbox" />
<label for="linewidths">Auto Line Width</label> </div> </td> </tr> <tr>
<td><span>Angle:</span></td>
<td><input id="angle" type="text" value="30" style="width:4em" /></td>
</tr> <tr> <td><span>Constants:</span></td>
<td><input id="constants" type="text" value="" style="width:12em" /></td>
</tr> <tr> <td><span>Axoim:</span></td>
<td><input id="axiom" type="text" value="F" style="width:12em"/></td>
</tr> <tr> <td><span>Rule1:</span></td>
```

<td><input id="rule1" type="text" value="$F = t(0.1)F[\sim(30) - t(0.1)F]t(0.1)F[+t(0.1)F][F]$" style="width:30em" /></td>
</tr> <tr> <td>Rule2:</td>
<td><input id="rule2" type="text" value="" style="width:30em" /></td>
</tr> <tr> <td colspan="2"><input id="start" type="button" value="Start" onclick="startHandler()"/> </td>
</tr> </table> 
Press Start to begin.
 <canvas id="canvas" width="512" height="512"></canvas>

</body>
</html>

| | |
|---|---|
| L-Sistema: | Planta 2 - Color |
| Iteraciones: | 5 |
| Ángulo: | $30°$ |
| Axioma: | $F$ |
| Regla 1: | $F = C0F[-C1F]C0F[+C1F][C1F]$ |

<!doctype html>
<html>
<head>
<title>L-Systems Turtle Graphics Renderer - HTML5 Canvas - by Kevin Roast</title>
<script src="scripts/lsystems.js?v=1"></script>
<script async defer src="http://s7.addthis.com/js/250/addthis_widget.js#pubid=xa-4e4813104686cad8"></script>
<script>
function openHelp()
{document.getElementById("help").style.display = "block";}
function closeHelp()
{document.getElementById("help").style.display = "none";}
</script>
<style type="text/css">
Body
{background-color: #ddd;
font-family: GeezaPro,Arial,Helvetica;
color: darkblue;
font-size: 0.9em;}
a, a:visited, a:active, a:hover
{color: #000044;}
.example

```
{border: 1px solid lightgrey;
cursor: pointer;
-moz-box-shadow: 3px 3px 4px #444;
-webkit-box-shadow: 3px 3px 4px #444;
box-shadow: 3px 3px 4px #444;}
.examples-title
{height: 24px;
text-align: center;}
.examples
{float: right;
overflow-y: auto;
overflow-x: hidden;
padding-right: 18px;
padding-top: 42px;
xborder: 1px solid darkgrey;}
.ex-left
{float: left;
padding-right: 6px;}
.ex-right
{width: 250px;
margin-bottom: 2px;}
Canvas
{background-color: white;
border: 1px solid grey;
-moz-box-shadow: 3px 3px 6px #444;
-webkit-box-shadow: 3px 3px 6px #444;
box-shadow: 3px 3px 6px #444;}
input[type="text"]
{border: none;
-moz-box-shadow: inset 2px 2px 5px #888;
-webkit-box-shadow: inset 2px 2px 5px #888;
box-shadow: inset 2px 2px 5px #888;
-moz-border-radius: 3px;
-webkit-border-radius: 3px;
border-radius: 3px;
margin: 1px;
padding: 5px;}
</style>
</head> <body>
```

```
<div style="position:absolute;right:1em;min-width:12em">
<!-- AddThis Button BEGIN -->
<div class="addthis_toolbox addthis_default_style addthis_32x32_style">
<a class="addthis_button_preferred_1"></a>
<a class="addthis_button_preferred_2"></a>
<a class="addthis_button_compact"></a>
<a class="addthis_counter addthis_bubble_style"></a> <div>
<!-- AddThis Button END --> <div>
<div id="lsystems" style="width:512px; float:left">
<div style="position:absolute;left:380px;top:32px">
<a href="#" onclick="openHelp()">What are L-Systems?</a> <div>
<div style="position:absolute;left:338px;top:64px">
<a href="../index.html">More HTML5 Canvas demos</a> </div> <div>
<table border="0" cellspacing="1" cellpadding="1"> <tr>
<td><span>Iterations:</span></td> <td>
<input id="iterations" type="text" value="5" style="width:4em" />
<div style="display:inline"> <input id="linewidths" type="checkbox" />
<label for="linewidths">Auto Line Width</label> </div> </td> </tr> <tr>
<td><span>Angle:</span></td>
<td><input id="angle" type="text" value="30" style="width:4em" /></td>
</tr> <tr> <td><span>Constants:</span></td>
<td><input id="constants" type="text" value="" style="width:12em" /></td>
</tr> <tr> <td><span>Axoim:</span></td>
<td><input id="axiom" type="text" value="F" style="width:12em"/></td>
</tr> <tr> <td><span>Rule1:</span></td>
<td><input id="rule1" type="text" value="F = C0F[−C1F]C0F[+C1F][C1F]"
style="width:30em" /></td>
</tr> <tr> <td><span>Rule2:</span></td>
<td><input id="rule2" type="text" value="" style="width:30em" /></td>
</tr> <tr> <td colspan="2"><input id="start" type="button" value="Start"
onclick="startHandler()"/> </td>
</tr> </table> </div>
<div id="status" style="padding:4px;font-family:courier new">Press Start to begin.</div>
<div> <canvas id="canvas" width="512" height="512"></canvas>
</div> </div> </div>
</body>
</html>
```

| | |
|---|---|
| L-Sistema: | Planta 3 - Color |
| Iteraciones: | 7 |
| Ángulo: | $18°$ |
| Axioma: | $X$ |
| Regla 1: | $X = C0F[+C1X]C0F[-C1X] + C2X$ |
| Regla 2: | $F = FF$ |

```
<!doctype html>
<html>
<head>
<title>L-Systems Turtle Graphics Renderer - HTML5 Canvas - by Kevin Roast</title>
<script src="scripts/lsystems.js?v=1"></script>
<script async defer src="http://s7.addthis.com/js/250/addthis_widget.js#pubid=xa-4e4813104686cad8"></script>
<script>
function openHelp()
{document.getElementById("help").style.display = "block";}
function closeHelp()
{document.getElementById("help").style.display = "none";}
</script>
<style type="text/css">
Body
{background-color: #ddd;
font-family: GeezaPro,Arial,Helvetica;
color: darkblue;
font-size: 0.9em;}
a, a:visited, a:active, a:hover
{color: #000044;}
.example
{border: 1px solid lightgrey;
cursor: pointer;
-moz-box-shadow: 3px 3px 4px #444;
-webkit-box-shadow: 3px 3px 4px #444;
box-shadow: 3px 3px 4px #444;}
.examples-title
{height: 24px;
text-align: center;}
.examples
{float: right;
```

```
overflow-y: auto;
overflow-x: hidden;
padding-right: 18px;
padding-top: 42px;
xborder: 1px solid darkgrey;}
.ex-left
{float: left;
padding-right: 6px;}
.ex-right
{width: 250px;
margin-bottom: 2px;}
Canvas
{background-color: white;
border: 1px solid grey;
-moz-box-shadow: 3px 3px 6px #444;
-webkit-box-shadow: 3px 3px 6px #444;
box-shadow: 3px 3px 6px #444;}
input[type="text"]
{border: none;
-moz-box-shadow: inset 2px 2px 5px #888;
-webkit-box-shadow: inset 2px 2px 5px #888;
box-shadow: inset 2px 2px 5px #888;
-moz-border-radius: 3px;
-webkit-border-radius: 3px;
border-radius: 3px;
margin: 1px;
padding: 5px;}
</style>
</head> <body>
<div style="position:absolute;right:1em;min-width:12em">
<!-- AddThis Button BEGIN -->
<div class="addthis_toolbox addthis_default_style addthis_32x32_style">
<a class="addthis_button_preferred_1"></a>
<a class="addthis_button_preferred_2"></a>
<a class="addthis_button_compact"></a>
<a class="addthis_counter addthis_bubble_style"></a> <div>
<!-- AddThis Button END --> <div>
<div id="lsystems" style="width:512px; float:left">
<div style="position:absolute;left:380px;top:32px">
```

```
<a href="#" onclick="openHelp()">What are L-Systems?</a> <div>
<div style="position:absolute;left:338px;top:64px">
<a href="../index.html">More HTML5 Canvas demos</a> </div> <div>
<table border="0" cellspacing="1" cellpadding="1"> <tr>
<td><span>Iterations:</span></td> <td>
<input id="iterations" type="text" value="7" style="width:4em" />
<div style="display:inline"> <input id="linewidths" type="checkbox" />
<label for="linewidths">Auto Line Width</label> </div> </td> </tr> <tr>
<td><span>Angle:</span></td>
<td><input id="angle" type="text" value="18" style="width:4em" /></td>
</tr> <tr> <td><span>Constants:</span></td>
<td><input id="constants" type="text" value="" style="width:12em" /></td>
</tr> <tr> <td><span>Axoim:</span></td>
<td><input id="axiom" type="text" value="X" style="width:12em"/></td>
</tr> <tr> <td><span>Rule1:</span></td>
<td><input id="rule1" type="text" value="X = C0F[+C1X]C0F[−C1X] + C2X"
style="width:30em" /></td>
</tr> <tr> <td><span>Rule2:</span></td>
<td><input id="rule2" type="text" value="F=FF" style="width:30em" /></td>
</tr> <tr> <td colspan="2"><input id="start" type="button" value="Start"
onclick="startHandler()"/> </td>
</tr> </table> </div>
<div id="status" style="padding:4px;font-family:courier new">Press Start to begin.</div>
<div> <canvas id="canvas" width="512" height="512"></canvas>
</div> </div> </div>
</body>
</html>
```

| | |
|---|---|
| L-Sistema: | Planta 4 - Color |
| Iteraciones: | 5 |
| Ángulo: | $25°$ |
| Axioma: | $FX$ |
| Regla 1: | $F = C0FF - [C1 - F + F] + [C2 + F - F]$ |
| Regla 2: | $X = C0FF + [C1 + F] + [C3 - F]$ |

```
<!doctype html>
<html>
<head>
<title>L-Systems Turtle Graphics Renderer - HTML5 Canvas - by Kevin Roast</title>
<script src="scripts/lsystems.js?v=1"></script>
<script async defer src="http://s7.addthis.com/js/250/addthis_widget.js#pubid=xa-4e4813104686cad8"></script>
<script>
function openHelp()
{document.getElementById("help").style.display = "block";}
function closeHelp()
{document.getElementById("help").style.display = "none";}
</script>
<style type="text/css">
Body
{background-color: #ddd;
font-family: GeezaPro,Arial,Helvetica;
color: darkblue;
font-size: 0.9em;}
a, a:visited, a:active, a:hover
{color: #000044;}
.example
{border: 1px solid lightgrey;
cursor: pointer;
-moz-box-shadow: 3px 3px 4px #444;
-webkit-box-shadow: 3px 3px 4px #444;
box-shadow: 3px 3px 4px #444;}
.examples-title
{height: 24px;
text-align: center;}
.examples
{float: right;
overflow-y: auto;
overflow-x: hidden;
padding-right: 18px;
padding-top: 42px;
xborder: 1px solid darkgrey;}
.ex-left
{float: left;
```

```
padding-right: 6px;}
.ex-right
{width: 250px;
margin-bottom: 2px;}
Canvas
{background-color: white;
border: 1px solid grey;
-moz-box-shadow: 3px 3px 6px #444;
-webkit-box-shadow: 3px 3px 6px #444;
box-shadow: 3px 3px 6px #444;}
input[type="text"]
{border: none;
-moz-box-shadow: inset 2px 2px 5px #888;
-webkit-box-shadow: inset 2px 2px 5px #888;
box-shadow: inset 2px 2px 5px #888;
-moz-border-radius: 3px;
-webkit-border-radius: 3px;
border-radius: 3px;
margin: 1px;
padding: 5px;}
</style>
</head> <body>
<div style="position:absolute;right:1em;min-width:12em">
<!-- AddThis Button BEGIN -->
<div class="addthis_toolbox addthis_default_style addthis_32x32_style">
<a class="addthis_button_preferred_1"></a>
<a class="addthis_button_preferred_2"></a>
<a class="addthis_button_compact"></a>
<a class="addthis_counter addthis_bubble_style"></a> <div>
<!-- AddThis Button END --> <div>
<div id="lsystems" style="width:512px; float:left">
<div style="position:absolute;left:380px;top:32px">
<a href="#" onclick="openHelp()">What are L-Systems?</a> <div>
<div style="position:absolute;left:338px;top:64px">
<a href="../index.html">More HTML5 Canvas demos</a> </div> <div>
<table border="0" cellspacing="1" cellpadding="1"> <tr>
<td><span>Iterations:</span></td> <td>
<input id="iterations" type="text" value="5" style="width:4em" />
<div style="display:inline"> <input id="linewidths" type="checkbox" />
```

<label for="linewidths">Auto Line Width</label>  </td> </tr> <tr>
<td>Angle:</td>
<td><input id="angle" type="text" value="25" style="width:4em" /></td>
</tr> <tr> <td>Constants:</td>
<td><input id="constants" type="text" value="" style="width:12em" /></td>
</tr> <tr> <td>Axoim:</td>
<td><input id="axiom" type="text" value="FX" style="width:12em"/></td>
</tr> <tr> <td>Rule1:</td>
<td><input id="rule1" type="text" value="F = C0FF − [C1 − F + F] + [C2 + F − F]" style="width:30em" /></td>
</tr> <tr> <td>Rule2:</td>
<td><input id="rule2" type="text" value="X = C0FF + [C1 + F] + [C3 − F]" style="width:30em" /></td>
</tr> <tr> <td colspan="2"><input id="start" type="button" value="Start" onclick="startHandler()"/> </td>
</tr> </table> 
Press Start to begin.
 <canvas id="canvas" width="512" height="512"></canvas>

</body>
</html>

| | |
|---|---|
| L-Sistema: | Planta 5 - Color |
| Iteraciones: | 6 |
| Ángulo: | $25°$ |
| Constante: | $X$ |
| Axioma: | $X$ |
| Regla 1: | $X = C0F - [C2[X] + C3X] + C1F[C3 + FX] - X$ |
| Regla 2: | $F = FF$ |

<!doctype html>
<html>
<head>
<title>L-Systems Turtle Graphics Renderer - HTML5 Canvas - by Kevin Roast</title>
<script src="scripts/lsystems.js?v=1"></script>
<script async defer src="http://s7.addthis.com/js/250/addthis_widget.js#pubid=xa-4e4813104686cad8"></script>
<script>
function openHelp()
{document.getElementById("help").style.display = "block";}

```
function closeHelp()
{document.getElementById("help").style.display = "none";}
</script>
<style type="text/css">
Body
{background-color: #ddd;
font-family: GeezaPro,Arial,Helvetica;
color: darkblue;
font-size: 0.9em;}
a, a:visited, a:active, a:hover
{color: #000044;}
.example
{border: 1px solid lightgrey;
cursor: pointer;
-moz-box-shadow: 3px 3px 4px #444;
-webkit-box-shadow: 3px 3px 4px #444;
box-shadow: 3px 3px 4px #444;}
.examples-title
{height: 24px;
text-align: center;}
.examples
{float: right;
overflow-y: auto;
overflow-x: hidden;
padding-right: 18px;
padding-top: 42px;
xborder: 1px solid darkgrey;}
.ex-left
{float: left;
padding-right: 6px;}
.ex-right
{width: 250px;
margin-bottom: 2px;}
Canvas
{background-color: white;
border: 1px solid grey;
-moz-box-shadow: 3px 3px 6px #444;
-webkit-box-shadow: 3px 3px 6px #444;
box-shadow: 3px 3px 6px #444;}
```

```
input[type="text"]
{border: none;
-moz-box-shadow: inset 2px 2px 5px #888;
-webkit-box-shadow: inset 2px 2px 5px #888;
box-shadow: inset 2px 2px 5px #888;
-moz-border-radius: 3px;
-webkit-border-radius: 3px;
border-radius: 3px;
margin: 1px;
padding: 5px;}
</style>
</head> <body>
<div style="position:absolute;right:1em;min-width:12em">
<!-- AddThis Button BEGIN -->
<div class="addthis_toolbox addthis_default_style addthis_32x32_style">
<a class="addthis_button_preferred_1"></a>
<a class="addthis_button_preferred_2"></a>
<a class="addthis_button_compact"></a>
<a class="addthis_counter addthis_bubble_style"></a> <div>
<!-- AddThis Button END --> <div>
<div id="lsystems" style="width:512px; float:left">
<div style="position:absolute;left:380px;top:32px">
<a href="#" onclick="openHelp()">What are L-Systems?</a> <div>
<div style="position:absolute;left:338px;top:64px">
<a href="../index.html">More HTML5 Canvas demos</a> </div> <div>
<table border="0" cellspacing="1" cellpadding="1"> <tr>
<td><span>Iterations:</span></td> <td>
<input id="iterations" type="text" value="6" style="width:4em" />
<div style="display:inline"> <input id="linewidths" type="checkbox" />
<label for="linewidths">Auto Line Width</label> </div> </td> </tr> <tr>
<td><span>Angle:</span></td>
<td><input id="angle" type="text" value="25" style="width:4em" /></td>
</tr> <tr> <td><span>Constants:</span></td>
<td><input id="constants" type="text" value="X" style="width:12em" /></td>
</tr> <tr> <td><span>Axoim:</span></td>
<td><input id="axiom" type="text" value="X" style="width:12em"/></td>
</tr> <tr> <td><span>Rule1:</span></td>
<td><input id="rule1" type="text" value="X = C0F − [C2[X] + C3X] + C1F[C3 + FX] − X"
style="width:30em" /></td>
```

</tr> <tr> <td>Rule2:</td>
<td><input id="rule2" type="text" value="F = FF" style="width:30em" /></td>
</tr> <tr> <td colspan="2"><input id="start" type="button" value="Start" onclick="startHandler()"/> </td>
</tr> </table> 
Press Start to begin.
 <canvas id="canvas" width="512" height="512"></canvas>

</body>
</html>

| | |
|---|---|
| L-Sistema: | Arbusto 1 - Color |
| Iteraciones: | 5 |
| Ángulo: | $22{,}5°$ |
| Axioma: | $F$ |
| Regla 1: | $F = C0FF - [C1 - F + F + F] + [C2 + F - F - F]$ |

<!doctype html>
<html>
<head>
<title>L-Systems Turtle Graphics Renderer - HTML5 Canvas - by Kevin Roast</title>
<script src="scripts/lsystems.js?v=1"></script>
<script async defer src="http://s7.addthis.com/js/250/addthis_widget.js#pubid=xa-4e4813104686cad8"></script>
<script>
function openHelp()
{document.getElementById("help").style.display = "block";}
function closeHelp()
{document.getElementById("help").style.display = "none";}
</script>
<style type="text/css">
Body
{background-color: #ddd;
font-family: GeezaPro,Arial,Helvetica;
color: darkblue;
font-size: 0.9em;}
a, a:visited, a:active, a:hover
{color: #000044;}
.example
{border: 1px solid lightgrey;
cursor: pointer;

```
-moz-box-shadow: 3px 3px 4px #444;
-webkit-box-shadow: 3px 3px 4px #444;
box-shadow: 3px 3px 4px #444;}
.examples-title
{height: 24px;
text-align: center;}
.examples
{float: right;
overflow-y: auto;
overflow-x: hidden;
padding-right: 18px;
padding-top: 42px;
xborder: 1px solid darkgrey;}
.ex-left
{float: left;
padding-right: 6px;}
.ex-right
{width: 250px;
margin-bottom: 2px;}
Canvas
{background-color: white;
border: 1px solid grey;
-moz-box-shadow: 3px 3px 6px #444;
-webkit-box-shadow: 3px 3px 6px #444;
box-shadow: 3px 3px 6px #444;}
input[type="text"]
{border: none;
-moz-box-shadow: inset 2px 2px 5px #888;
-webkit-box-shadow: inset 2px 2px 5px #888;
box-shadow: inset 2px 2px 5px #888;
-moz-border-radius: 3px;
-webkit-border-radius: 3px;
border-radius: 3px;
margin: 1px;
padding: 5px;}
</style>
</head> <body>
<div style="position:absolute;right:1em;min-width:12em">
<!-- AddThis Button BEGIN -->
```

```
<div class="addthis_toolbox addthis_default_style addthis_32x32_style">
<a class="addthis_button_preferred_1"></a>
<a class="addthis_button_preferred_2"></a>
<a class="addthis_button_compact"></a>
<a class="addthis_counter addthis_bubble_style"></a> <div>
<!-- AddThis Button END --> <div>
<div id="lsystems" style="width:512px; float:left">
<div style="position:absolute;left:380px;top:32px">
<a href="#" onclick="openHelp()">What are L-Systems?</a> <div>
<div style="position:absolute;left:338px;top:64px">
<a href="../index.html">More HTML5 Canvas demos</a> </div> <div>
<table border="0" cellspacing="1" cellpadding="1"> <tr>
<td><span>Iterations:</span></td> <td>
<input id="iterations" type="text" value="5" style="width:4em" />
<div style="display:inline"> <input id="linewidths" type="checkbox" />
<label for="linewidths">Auto Line Width</label> </div> </td> </tr> <tr>
<td><span>Angle:</span></td>
<td><input id="angle" type="text" value="22,5" style="width:4em" /></td>
</tr> <tr> <td><span>Constants:</span></td>
<td><input id="constants" type="text" value="" style="width:12em" /></td>
</tr> <tr> <td><span>Axoim:</span></td>
<td><input id="axiom" type="text" value="F" style="width:12em"/></td>
</tr> <tr> <td><span>Rule1:</span></td>
<td><input id="rule1" type="text" value="F = C0FF − [C1 − F + F + F] + [C2 + F − F − F]"
style="width:30em" /></td>
</tr> <tr> <td><span>Rule2:</span></td>
<td><input id="rule2" type="text" value="" style="width:30em" /></td>
</tr> <tr> <td colspan="2"><input id="start" type="button" value="Start"
onclick="startHandler()"/> </td>
</tr> </table> </div>
<div id="status" style="padding:4px;font-family:courier new">Press Start to begin.</div>
<div> <canvas id="canvas" width="512" height="512"></canvas>
</div> </div> </div>
</body>
</html>
```

| | |
|---|---|
| L-Sistema: | Carpeta de Sierpinski |
| Iteraciones: | 4 |
| Ángulo: | $90°$ |
| Axioma: | $F$ |
| Regla 1: | $F = C0F + F - F - F - G + C1F + F + F - F$ |
| Regla 2: | $G = GGG$ |

```
<!doctype html>
<html>
<head>
<title>L-Systems Turtle Graphics Renderer - HTML5 Canvas - by Kevin Roast</title>
<script src="scripts/lsystems.js?v=1"></script>
<script async defer src="http://s7.addthis.com/js/250/addthis_widget.js#pubid=xa-4e4813104686cad8"></script>
<script>
function openHelp()
{document.getElementById("help").style.display = "block";}
function closeHelp()
{document.getElementById("help").style.display = "none";}
</script>
<style type="text/css">
Body
{background-color: #ddd;
font-family: GeezaPro,Arial,Helvetica;
color: darkblue;
font-size: 0.9em;}
a, a:visited, a:active, a:hover
{color: #000044;}
.example
{border: 1px solid lightgrey;
cursor: pointer;
-moz-box-shadow: 3px 3px 4px #444;
-webkit-box-shadow: 3px 3px 4px #444;
box-shadow: 3px 3px 4px #444;}
.examples-title
{height: 24px;
text-align: center;}
```

```
.examples
{float: right;
overflow-y: auto;
overflow-x: hidden;
padding-right: 18px;
padding-top: 42px;
xborder: 1px solid darkgrey;}
.ex-left
{float: left;
padding-right: 6px;}
.ex-right
{width: 250px;
margin-bottom: 2px;}
Canvas
{background-color: white;
border: 1px solid grey;
-moz-box-shadow: 3px 3px 6px #444;
-webkit-box-shadow: 3px 3px 6px #444;
box-shadow: 3px 3px 6px #444;}
input[type="text"]
{border: none;
-moz-box-shadow: inset 2px 2px 5px #888;
-webkit-box-shadow: inset 2px 2px 5px #888;
box-shadow: inset 2px 2px 5px #888;
-moz-border-radius: 3px;
-webkit-border-radius: 3px;
border-radius: 3px;
margin: 1px;
padding: 5px;}
</style>
</head> <body>
<div style="position:absolute;right:1em;min-width:12em">
<!-- AddThis Button BEGIN -->
<div class="addthis_toolbox addthis_default_style addthis_32x32_style">
<a class="addthis_button_preferred_1"></a>
<a class="addthis_button_preferred_2"></a>
<a class="addthis_button_compact"></a>
<a class="addthis_counter addthis_bubble_style"></a> <div>
<!-- AddThis Button END --> <div>
```

```
<div id="lsystems" style="width:512px; float:left">
<div style="position:absolute;left:380px;top:32px">
<a href="#" onclick="openHelp()">What are L-Systems?</a> <div>
<div style="position:absolute;left:338px;top:64px">
<a href="../index.html">More HTML5 Canvas demos</a> </div> <div>
<table border="0" cellspacing="1" cellpadding="1"> <tr>
<td><span>Iterations:</span></td> <td>
<input id="iterations" type="text" value="4" style="width:4em" />
<div style="display:inline"> <input id="linewidths" type="checkbox" />
<label for="linewidths">Auto Line Width</label> </div> </td> </tr> <tr>
<td><span>Angle:</span></td>
<td><input id="angle" type="text" value="90" style="width:4em" /></td>
</tr> <tr> <td><span>Constants:</span></td>
<td><input id="constants" type="text" value="" style="width:12em" /></td>
</tr> <tr> <td><span>Axoim:</span></td>
<td><input id="axiom" type="text" value="F" style="width:12em"/></td>
</tr> <tr> <td><span>Rule1:</span></td>
<td><input id="rule1" type="text" value="F = C0F + F − F − F − G + C1F + F + F − F"
style="width:30em" /></td>
</tr> <tr> <td><span>Rule2:</span></td>
<td><input id="rule2" type="text" value="G = GGG" style="width:30em" /></td>
</tr> <tr> <td colspan="2"><input id="start" type="button" value="Start"
onclick="startHandler()"/> </td>
</tr> </table> </div>
<div id="status" style="padding:4px;font-family:courier new">Press Start to begin.</div>
<div> <canvas id="canvas" width="512" height="512"></canvas>
</div> </div> </div>
</body>
</html>
```

# 5. APLICACIONES.

En este capítulo se describen algunos ejemplos que según menciona Campos (2011), son estructuras con características de L-Sistemas presentes en campos como la botánica, la biología, el arte y la arquitectura entre otros.

Este capítulo es esencialmente informativo. Con éste se pretende que el lector se informe acerca de la presencia de las ideas de los L-Sistemas en algunos campos, y que esto sirva para que le dé una mayor importancia al estudio de esta rama de la geometría fractal.

## 5.1. Botánica y Biología.

Como lo afirma Campos, (2011, pág. 11) podemos pasear por un bosque y darnos cuenta de la gran variedad de formas, tamaños, colores, texturas y distribución espacial de las plantas que lo forman. Algunos botánicos se han dedicado a estudiar y describir estas características, añadiendo otras como el crecimiento, orientación y funcionalidad de las hojas, ramas y troncos de las plantas.

La mayoría de ellos con fines científicos pero es innegable que en nuestros días el mundo virtual ocupa un lugar muy importante, así, en el interés por modelar y generar paisajes naturales lo más realistas posibles, los L-Sistemas tienen un gran campo de acción con sus mayores logros en la industria cinematográfica y los video juegos.

Otras áreas "nuevas" pueden ser el estudio de la productividad de las cosechas debido a la diferente distribución de plantas originadas por diferentes técnicas de

siembra, riegos y cuidados, o la propagación de enfermedades y plagas de insectos y los diferentes métodos para combatirlas.

### 5.2. Arte y Arquitectura.

Los L-Sistemas se han usado para describir procesos complejos de desarrollo, incluyendo composiciones musicales, estructuras arquitectónicas, diseño de patrones geométricos, etcétera. Estas gramáticas han sido útiles para indagar la forma en que puede inducirse las operaciones que realizan artistas, artesanos y diseñadores para generar piezas de alta complejidad que nunca son dos veces la misma. Y es que en cada obra, se pueden reconocer ciertos patrones y similitudes en los rasgos que con paciencia y dedicación podemos representar con uno de estos sistemas. Algunos ejemplos pueden ser tapetes, tejidos y pinturas de culturas indígenas que tienen arreglos muy singulares. (Campos, 2011, pág. 12)

### 5.3. Otras áreas.

Simulación y optimización de procesos, diseño y planificación de ciudades, interacción de organismos y ecosistemas, propagación de enfermedades, de insectos, sistemas de riego, de cultivo, reconstrucción de plantas extintas, vida artificial, redes neuronales, estudio de sistemas complejos, representación de estrellas, burbujas, lluvia, etcétera. (Campos, 2011, pág. 12)

# 6. CONCLUSIONES.

Con la realización de este trabajo se pudo observar que existen programas menos conocidos para la generación de fractales y la simulación de procesos de crecimiento o estructuras en las que se pueda identificar un patrón básico de desarrollo; el manejo de estos elementos tecnológicos son una herramienta para el estudio de los algoritmos que los generan ya que antes solo era posible construir tales objetos utilizando el dibujo a mano (lápiz y papel) y ahora se pueden generar diferentes aproximaciones usando las herramientas computacionales en donde cada lenguaje pone en evidencia gran cantidad de objetos, elementos y reglas de construcción de dichos objetos.

La aplicación de este tipo de actividades en el aula, bajo la mediación de elementos tecnológicos ofrece posibilidades interesantes en los procesos de enseñanza - aprendizaje, ya que permite crear experiencias en las cuales se puede calcular, graficar, modelar, explorar, visualizar, clasificar, comparar y simular objetos que resultan difíciles de manipular con lápiz y papel. Además, el computador se presenta como una herramienta de auto aprendizaje y generación de conocimiento personal a nivel del software utilizado y la manera de trabajar sobre él.

Por otra parte, en cuanto al ejercicio docente, resulta interesante terminar este documento reflexionando sobre los enfoques de trabajo del profesor, ya que es evidente la intención de partir desde una expresión "terminada" en busca de deducir la regla o la gramática que la produjo cambiando así el paradigma general del conocimiento deductivo por el inductivo, lo cual resulta bastante útil en la solución de muchas situaciones del diario vivir. Mi pregunta es: ¿No será un error

el uso inadecuado (o no uso) de las nuevas tecnologías en combinación con una enseñanza de las matemáticas demasiado tradicional donde la repetición de fórmulas es lo más importante, un impedimento para el aprendizaje?

Por último, se espera que este documento sirva como punto de partida para futuros trabajos, en particular de los educadores matemáticos, ya que no podemos seguir marginados de la realidad, se hace necesario estudiar las posibilidades que brindan las nuevas tecnologías y desplegar toda nuestra creatividad, imaginación y fundamentación matemática para encontrar mejores formas de llevar el conocimiento al aula y potenciar el desarrollo integral de nuestros estudiantes.

# 7. BIBLIOGRAFÍA.


Campos, D. (2011). *Introducción a los Sistemas de Lindenmayer: fractales, autómatas celulares y aplicaciones*. Recuperado el 30 de Septiembre de 2014, de Centro de Investigación y de Estudios Avanzados del IPN, Departamento de Computación del CINVESTAV: http://delta.cs.cinvestav.mx/~mcintosh/cellularautomata/Summer_Research_files/Arti_Ver_Inv_2011_DCM.pdf

De Guzmán, M. (1993). *Estructuras Fractales y sus Aplicaciones.* Barcelona: Labor.

Estrada, W. (2004). *Geometría Fractal: conceptos y procedimientos para la construcción de fractales.* Bogotá, D.C., Colombia: Cooperativa Editorial Magisterio.

Frame, M., Mandelbrot, B., & Neger, N. (2010). *Fractal Geometry*. Recuperado el 30 de Septiembre de 2014, de Yale University: http://classes.yale.edu/fractals/

Iglesias, A., & Gálvez, A. (Agosto de 2011). *Open Course Ware*. (Universidad de Cantabria) Recuperado el 30 de Septiembre de 2014, de Aula Virtual: http://ocw.unican.es/ensenanzas-tecnicas/visualizacion-e-interaccion-grafica/otros-recursos-2/Lsistemas.pdf

Lindenmayer, A., & Prusinkiewicz, P. (1990). *The Algorithmic Beauty of Plants.* New York: Springer - Verlag.

Luque, B., & Agea, A. (s.f.). *Fractales en la red, Los L-Sistemas.* Recuperado el 17 de Noviembre de 2015, de Cursos fractales en Departamento Matemática Aplicada y Estadística: http://www.dmae.upm.es/cursofractales/capitulo2/frames.htm

Mandelbrot, B. (1982). *The Fractal Geometry of Nature.* New York: W. H. Freeman and Co.

Monroy, C. (2002). *Curvas Fractales.* México, D.F.: Alfaomega.

Muñoz, I. (2007). *Uso de los Lenguajes de Programación en el Estudio de los Fractales.* Trabajo de grado para obtener el título de Licenciada en Matemáticas, Departamento de Matemáticas, Universidad Pedagógica Nacional, Bogotá, D.C., Colombia.

Rivero, J. A. (Agosto de 2014). *Sistemas de Lindenmayer con L- Parser.* (U. P. Madrid, Ed.) Recuperado el 17 de Noviembre de 2015, de Cursos fractales en Departamento Matemática Aplicada y Estadística: http://www.dmae.upm.es/cursofractales/capitulo2/lparse/lparser.htm

Roast, K. (Agosto de 2012). *Turtle Graphics Renderer on- line*. Obtenido de HTML 5 + Java Script Canvas Demo: http://kevs3d.co.uk/dev/lsystems/#

Rubiano, G. (2011). *Iteración y Fractales (con Mathematica).* Bogotá, D.C., Colombia: Universidad Nacional de Colombia.

Suárez, P. (2010). El aprendizaje de la geometría fractal de la naturaleza en la Licenciatura de la matemáticas de la UPTC (nociones básicas). *TO-04371 - Tesis BC-UPN, xxi*(186 h). Tunja, Boyacá, Colombia.

Tagtachian, S., & Argumedo, C. (1995). *Generación de estructuras arbóreas tridimensionales basados en los Sistemas de Lindenmayer.* (C. CAO, Ed.) Recuperado el 30 de Septiembre de 2014, de Departamento de Computación, Facultad de Ciencias Exactas y Naturales, Universidad de Buenos Aires: http://cumincades.scix.net/data/works/att/ae1f.content.pdf