Tecnológico de Costa Rica

Escuela de Ingeniería en Computación

IC-1801 Taller de Programación

Prof. Mauricio Avilés

Proyecto Programado 2 - Fractales

Motivación

Un fractal es una figura geométrica cuya estructura básica se repite a diferentes escalas. El término fue propuesto por el matemático polaco Benoît Mandelbrot en 1975, y proviene del latín *fractus* que significa fracturado. Los fractales son demasiado irregulares para ser descritos en términos geométricos tradicionales y también son autosimilares, su forma está compuesta por copias más pequeñas de la misma figura. Esta última característica hace posible que los fractales puedan ser generados por medio de procedimientos recursivos. El objetivo de este proyecto programado es pasar del mundo de la matemática abstracta hacia la estética visual a través de la representación en pantalla de diferentes fractales.

Software a desarrollar

Su trabajo consiste en implementar funciones recursivas para generar gráficos por computadora de los siguientes fractales:

1. Triángulo de Sierpinski
2. Alfombra de Sierpinski
3. Hexágono de Sierpinski
4. Curva Koch
5. Círculos de Ravel
6. Curva de Satie
7. Estrella de Debussy

El trabajo no consiste solamente en programar los fractales tal y como se muestran en este documento. Cada fractal debe hacer uso de colores, figuras rellenas y cualquier otra variación que se desee con el objetivo de generar una presentación única y creativa del fractal. La creatividad será tomada muy en cuenta para la revisión del proyecto.

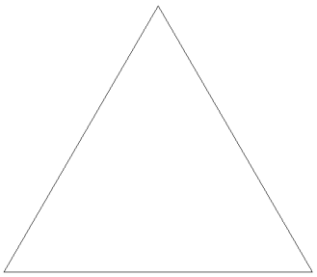
Algunos de estos fractales son muy conocidos y es fácil encontrar código ya implementado en Internet. **NO COPIE** código de ningún lugar. Cualquier copia que se detecte en el código será calificado con nota de cero y el caso será reportado a la dirección de la Escuela de Computación.

Triángulo de Sierpinski

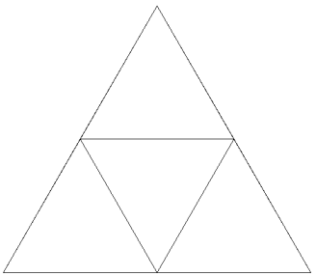
Este fractal consiste en un triángulo equilátero que está divido en cuatro triángulos equiláteros más pequeños. Cada uno de los triángulos ubicados en las esquinas del triángulo principal recibe el mismo tratamiento que el triángulo original, repitiendo el mismo patrón dentro de un área más pequeña. Este fractal puede ser dibujado como una línea continua que no es recta. Observe que cada triángulo tiene como lado la mitad del lado del triángulo anterior.

Debe crear una función recursiva que dibuje el fractal, que utilice como mínimo parámetros para profundidad (cantidad de repeticiones) y tamaño del lado. Puede invocarse como **trianguloSierpinski(profundidad, lado)**. A continuación se muestran algunos ejemplos del resultado que debe mostrar la función utilizando diferentes profundidades.

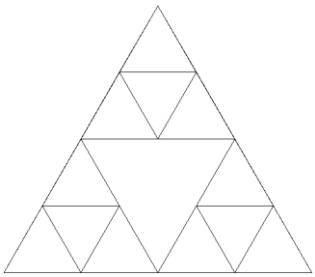
Profundidad = 1



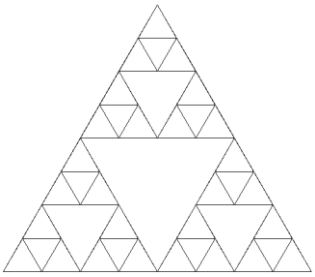
Profundidad = 2



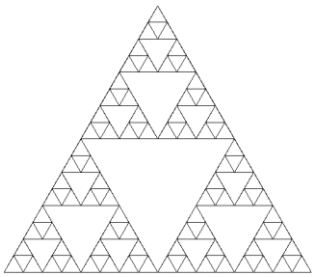
Profunidad=3



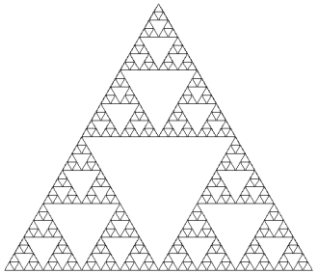
Profundidad=4



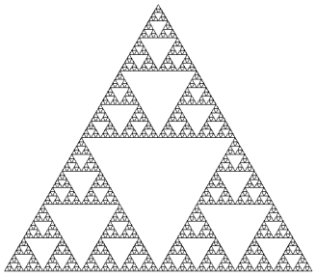
Profundidad=5



Profundidad = 6



Profundidad=7

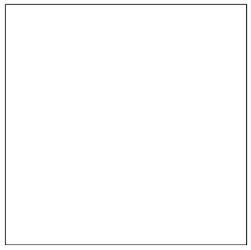


Alfombra de Sierpinski

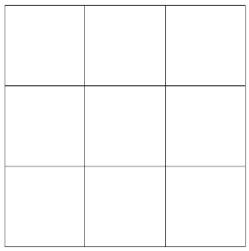
Este fractal consiste en un cuadrado que está dividido en nueve cuadrados iguales. Todos los cuadrados pequeños, excepto el central, reciben el mismo tratamiento que el cuadrado original, repitiendo el mismo patrón. Es un concepto muy similar al triángulo de Sierpinski, sólo que sustituyendo la forma geométrica base. En este caso cada cuadrado tiene como largo un tercio del lado del cuadrado anterior.

Debe crear una función recursiva que dibuje el fractal, que utilice como mínimo parámetros para profundidad y tamaño del lado. Puede invocarse como **alfombraSierpinski(produnfidad, lado)**. A continuación se muestran algunos ejemplos del resultado que debe mostrar la función utilizando diferentes profundidades.

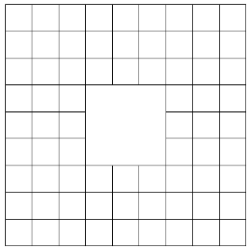
Profundidad=1



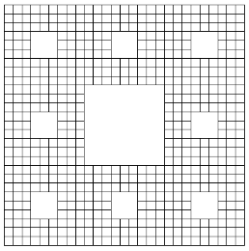
Profundidad=2



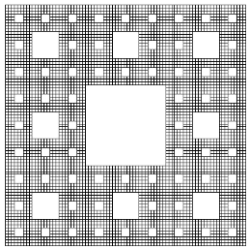
Profundidad=3



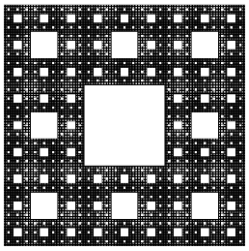
Profundidad=4



Profundidad=5



Profundidad=6

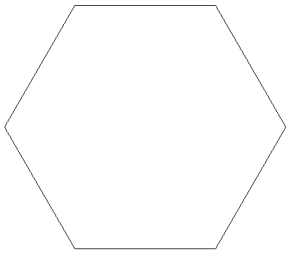


Hexágono de Sierpinski

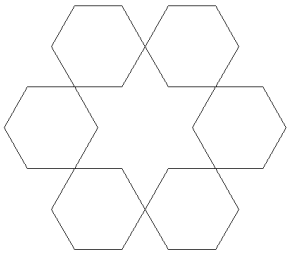
Este fractal sigue la misma idea de los dos fractales anteriores, pero sustituyendo la figura geométrica base, que en este caso es un hexágono. Si la profundidad es 1, se dibuja un hexágono corriente, si no, se dibuja un hexágono (recursivamente) de un tercio del tamaño y se mueve la posición al siguiente vértice del hexágono. Nótese que en este fractal no existe una línea que una los vértices de los hexágonos grandes, si no que queda un espacio vacío.

Debe crear una función recursiva que dibuje el fractal, que utilice como mínimo parámetros para profundidad y tamaño del lado. Puede invocarse como **hexagonoSierpinski(produnfidad, lado)**. A continuación se muestran algunos ejemplos del resultado que debe mostrar la función utilizando diferentes profundidades.

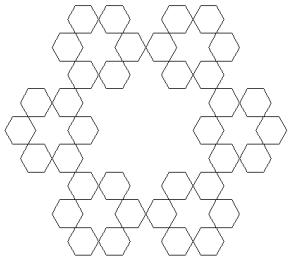
Profundidad=1



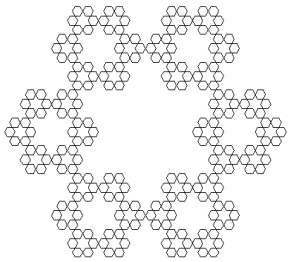
Profundidad=2



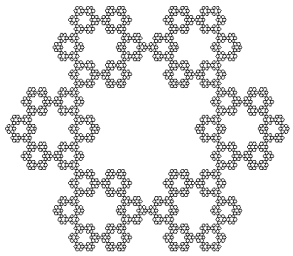
Profundidad=3



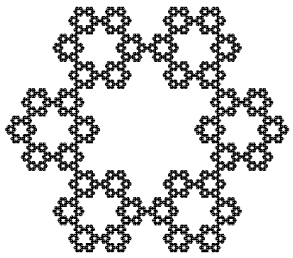
Profundidad=4



Profundidad=5



Profundidad=6



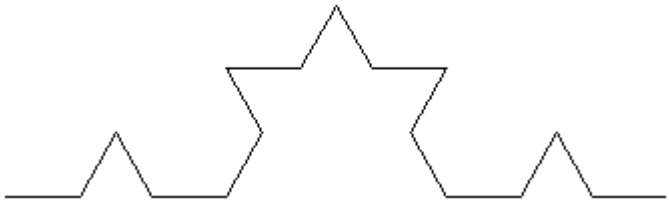
Curva de Koch

Este fractal es un tipo de figura que es conocida como curva. Una curva es una línea continua con ángulos y que no se interseca a sí misma. El principio de este fractal consiste en dibujar una línea, pero esta línea se divide en 4 líneas del tercio del tamaño original. Primero se dibuja una de las líneas de un tercio (recursivamente), luego se hace un giro a la izquierda de 60 grados y se dibuja otra línea (recursivamente) de un tercio, se hace un giro a la derecha de 120 grados, se dibuja otra línea de un tercio (recursivamente), se hace un giro a la izquierda de 30 grados, y se dibuja otra línea (recursivamente) de un tercio.

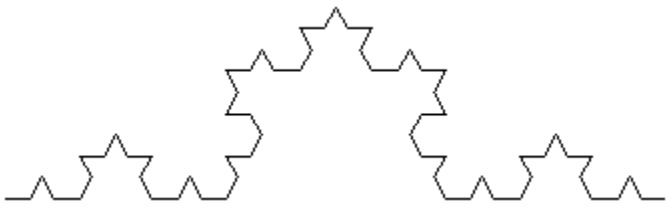
Debe crear una función recursiva que dibuje el fractal, que utilice como mínimo parámetros para profundidad y tamaño del lado. Puede invocarse como **koch(produnfidad, lado)**. A continuación se muestran algunos ejemplos del resultado que debe mostrar la función utilizando diferentes profundidades.

Profundidad=1

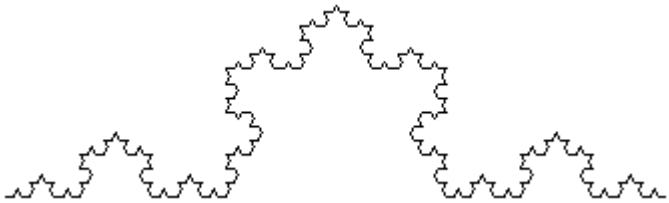
Profundidad=2



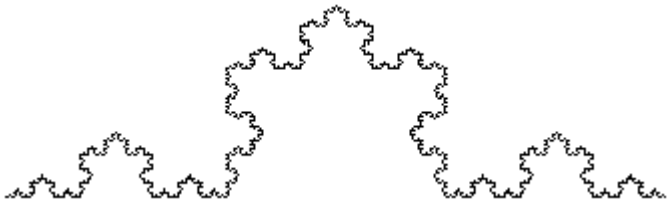
Profundidad=3



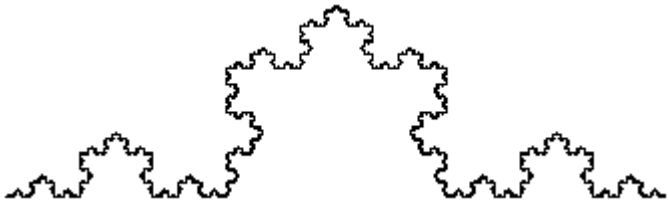
Profundidad=4



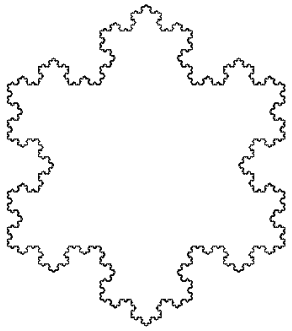
Profundidad=5



Profundidad=6



Note que si se dibujan tres curvas juntas con un giro de 60 al final de cada una se genera una especie de copo de nieve.

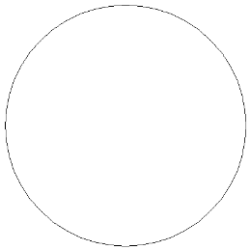


Círculo de Ravel

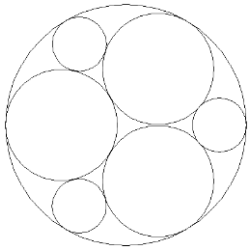
Este fractal consiste de un círculo que contiene seis círculos dentro. Tres de los círculos son más grandes y ocupan el mayor espacio posible dentro del círculo mayor. Los otros tres círculos se ubican en los espacios vacíos entre los círculos grandes, tal y como se ve en la fitura con profundidad 2. Este patrón se repite recursivamente en cada uno de los círculos.

Debe crear una función recursiva que dibuje el fractal, que utilice como mínimo parámetros para profundidad y tamaño del radio. Puede invocarse como **ravel(produnfidad, radio)**. A continuación se muestran algunos ejemplos del resultado que debe mostrar la función utilizando diferentes profundidades. Para este fractal debe investigar el funcionamiento de la función **circle** en Turtle.

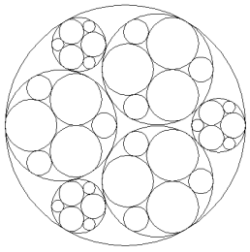
Profundidad=1



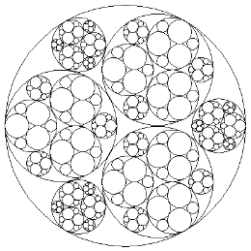
Profundidad=2



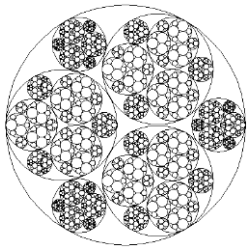
Profundidad=3



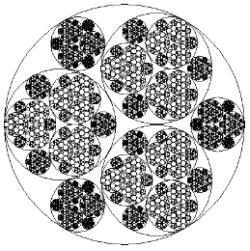
Profundidad=4



Profundidad=5



Profundidad=6



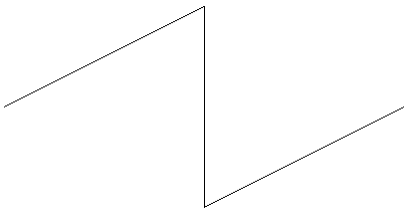
Note como el patrón que forman los círculos grandes se asemeja al patrón producido por el triángulo de Sierpinski.

Curva de Satie

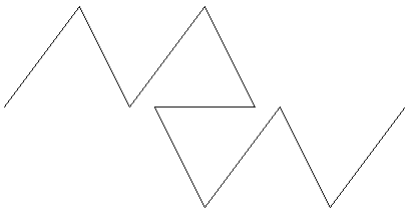
Este es un fractal que sigue una idea similar a la curva de Koch, pero utiliza una figura diferente para dividir cada línea. La base es una línea recta, la cual se divide en 3 líneas, tal y como se muestra en el dibujo que muestra la profundidad=1. La línea vertical es de la mitad del tamaño que separa el inicio y el final. Las otras dos líneas tienen un tamaño que se calcula usando el teorema de Pitágoras. Cada una de estas líneas se sustituye por el mismo patrón en zig-zag, dando origen al fractal.

Debe crear una función recursiva que dibuje el fractal, que utilice como mínimo parámetros para profundidad y tamaño del lado. Puede invocarse como **satie(produnfidad, lado)**. A continuación se muestran algunos ejemplos del resultado que debe mostrar la función utilizando diferentes profundidades.

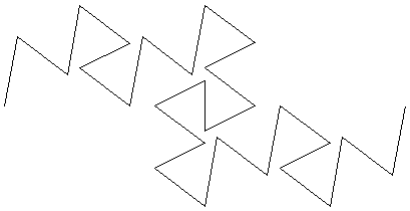
Profundidad=1



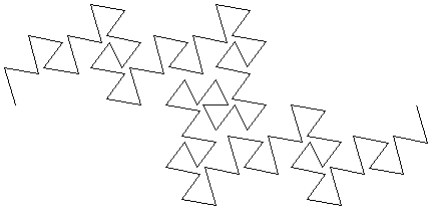
Profundidad=2



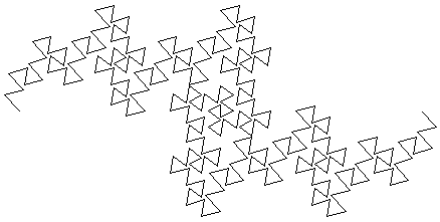
Profundidad=3



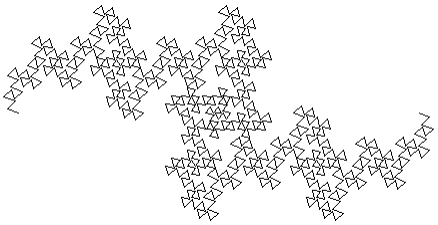
Profundidad=4



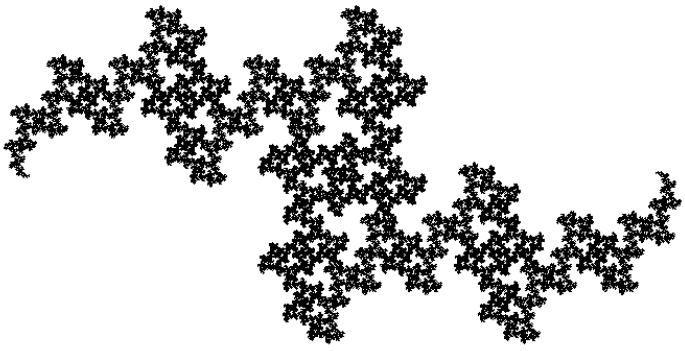
Profundidad=5



Profundidad=6



Con mayor cantidad de iteraciones el fractal toma una forma similar a la siguiente:

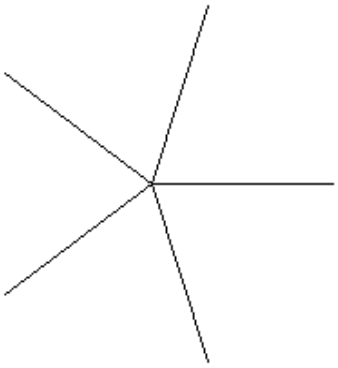


Estrella de Debussy

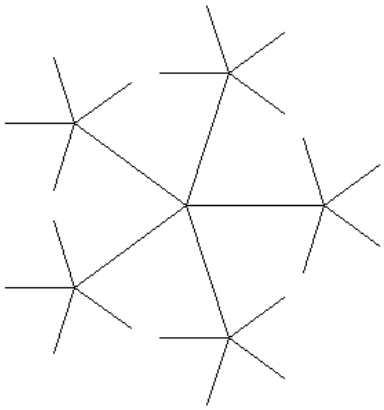
Este fractal es más generalizado que los demás. Consiste en dibujar líneas que formen una estrella o asterisco. La cantidad de picos de la estrella está dado por un parámetro n. Si n es igual a 4, entonces la estrella tiene cuatro picos y cada uno de ellos tendrá estrellas del mismo tipo. Las estrellas "hijas" tendrán un pico menos que la estrella principal, esto es porque cada una nace de un pico de la estrella principal y ese cuenta como uno de sus picos. También requiere otro parámetro que es la relación de tamaño entre una estrella y sus estrellas "hijas". Si la relación es 0.5, entonces las estrellas hijas de una estrella van a ser de la mitad del tamaño que la estrella anterior.

Debe crear una función recursiva que dibuje el fractal, que utilice parámetros para la profundidad, el tamaño del pico, la cantidad de picos y la relación entre el tamaño de la estrella y sus estrellas hijas. Puede invocarse como **debussy(profundidad, lado, n, relacion)**. A continuación se muestran algunos ejemplos con diferentes valores de parámetros.

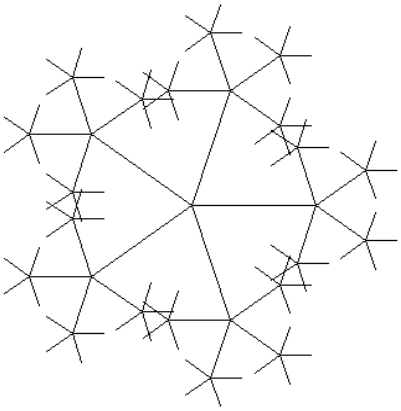
Uno de 5 picos con profundidad 1: debussy(1, 100, 5, 0.5)



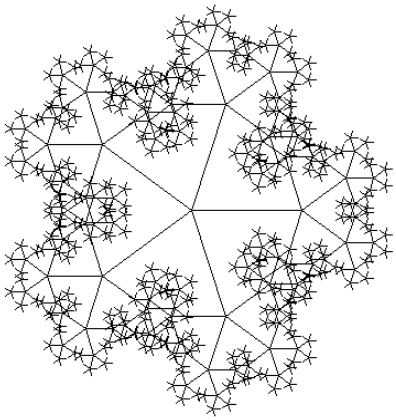
Aumentando profundidad: debussy(2, 100, 5, 0.5)



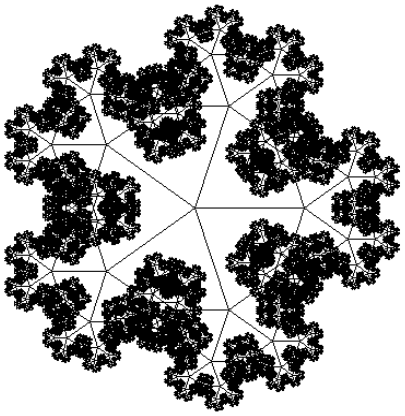
Aumentando profundidad: debussy(3, 100, 5, 0.5)



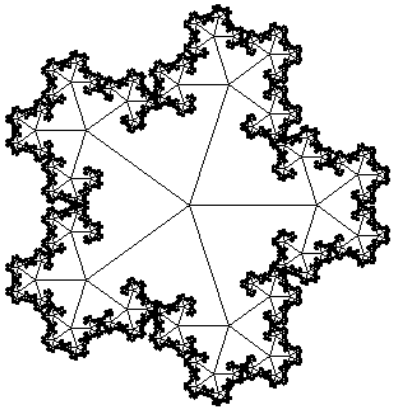
Aumentando profundidad: debussy(5, 100, 5, 0.5)



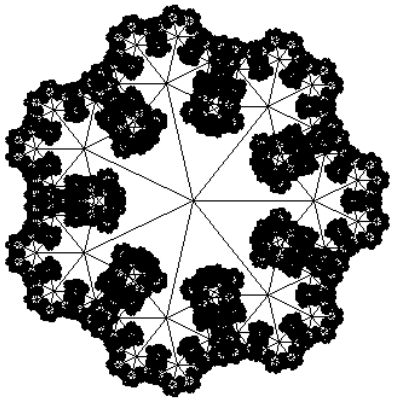
Aumentando profundidad: debussy(8, 100, 5, 0.5)



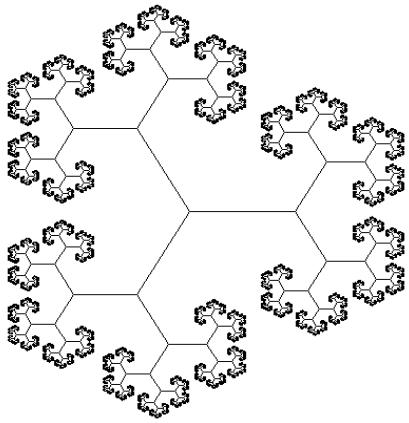
Mismo que el anterior pero cambiando la relación a 0.4. Note como los picos de las estrellas se empequeñecen más rápido: debussy(8, 100, 5, 0.4)



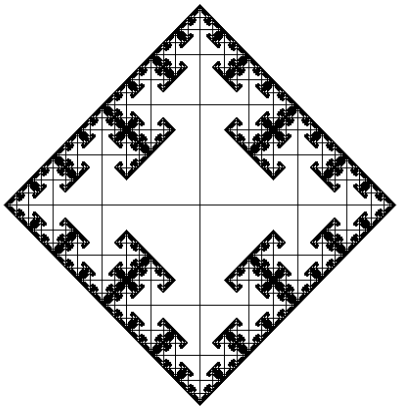
Cambiando la cantidad de picos a 7: debussy(7, 100, 7, 0.4)



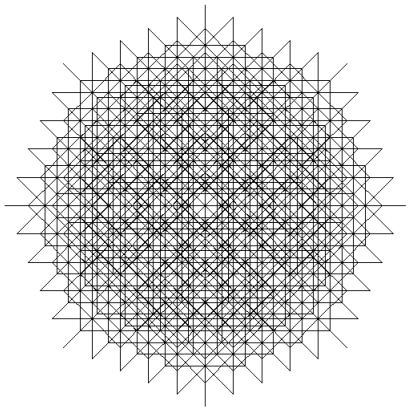
Con un n menor se puede tener mayor profundidad sin afectar el desempeño: debussy(14, 100, 3, 0.6)



Con cuatro ramas: debussy(9, 100, 4, 0.5)



La relación no tiene porqué ser menor que uno, puede ser incluso mayor: debussy(5, 50, 8, 1)



Documentación

Toda función debe llevar como documentación interna comentarios con lo siguiente:

- Descripción de la función
- Entradas
- Salidas
- Restricciones

En cuanto a la documentación externa, debe entregarse un manual de usuario en formato PDF que explique a cualquier persona cómo utilizar las funciones en el *shell* de Python.

Forma de trabajo

Debido a la cantidad de trabajo involucrada, el proyecto se desarrollará en parejas.

Para hacer dibujos en pantalla se utilizará el paquete **turtle** que viene incluido con el lenguaje Python. Este paquete es una implementación de la geometría de la tortuga, concepto matemático desarrollado por Seymour Papert y Wally Feurzig durante los años 60.

Para importar el módulo se utiliza la sentencia: **import turtle**

Algunos métodos que pueden ser de provecho para la tarea son:

turtle.forward(numpixels)	turtle.backward(numpixels)
turtle.right(degrees)	turtle.left(degrees)
turtle.circle(radius)	turtle.reset()
turtle.pencolor(r,g,b)	turtle.penup()
turtle.pendown()	

Puede encontrar más información sobre **turtle** en http://docs.python.org/release/3.1.3/library/turtle.html

Entrega

El tiempo asignado para la tarea programada es de 2 semanas. Debe entregarse un archivo **archivo ZIP** que contenga dos cosas:

1. **Un solo archivo Python** con todas las funciones requeridas.
2. Un archivo PDF con el manual de usuario

Evaluación

La tarea tiene un valor de 20% de la nota final, en el rubro de Proyectos Programados.

Desglose de la evaluación de la tarea programada:

Documentación:	20%
Programación:	80%

Como se mencionó anteriormente, se tomará en cuenta la creatividad para la representación de los fractales. La utilización de diferentes colores, fondos de pantalla o cualquier otro elemento que le dé un matiz artístico a su trabajo, será considerado para la nota de la programada.

Recomendaciones adicionales

Pruebe cada función individualmente. No implemente todo el programa sin verificar el funcionamiento por separado de cada una de sus funciones. Esto dirige a errores que son más difíciles de encontrar.

Recuerde que el trabajo es en equipos, es indispensable la comunicación y la coordinación entre los miembros del subgrupo.

Comparta el conocimiento con los demás compañeros de grupo y de la carrera, la ciencia de la computación es una disciplina que requiere el traspaso libre de conocimientos. Se logran mejores resultados con la colaboración de todos que con el esfuerzo separado de diferentes personas.

No dude en consultar diferentes fuentes para satisfacer las dudas. Aparte de las búsquedas en internet, asegúrese de exponer sus dudas a sus compañeros, profesor y conocidos que estudien la carrera; en la mayoría de las ocasiones es más provechosa conversación de 10 minutos entre personas que están trabajando en lo mismo que pasar horas buscando la respuesta a una duda de forma individual.

No deje la documentación para el final, es buena práctica ir desarrollándola durante todo el transcurso del proyecto. Recuerde que la documentación debe ser concisa y puntual, por lo que en realidad no toma mucho tiempo al realizarla de esta forma.

Plagios no serán tolerados bajo ninguna circunstancia. Cualquier intento de fraude será evaluado con una nota de cero y se enviará una carta al expediente del estudiante. Siempre escriba su propio código.

Referencias

Fractal. (2013, May 8). In *Wikipedia, The Free Encyclopedia*. Retrieved 02:28, May 10, 2013, from http://en.wikipedia.org/w/index.php?title=Fractal&oldid=554087975

Sierpinski triangle. (2013, March 4). In *Wikipedia, The Free Encyclopedia*. Retrieved 02:28, May 10, 2013, from http://en.wikipedia.org/w/index.php?title=Sierpinski_triangle&oldid=542044665

Sierpinski carpet. (2013, February 22). In *Wikipedia, The Free Encyclopedia*. Retrieved 02:28, May 10, 2013, from http://en.wikipedia.org/w/index.php?title=Sierpinski_carpet&oldid=539780666

N-flake. (2015, October 24). In *Wikipedia, The Free Encyclopedia*. Retrieved 22:41, May 1, 2016, from https://en.wikipedia.org/w/index.php?title=N-flake&oldid=687324189

Koch snowflake. (2016, March 31). In *Wikipedia, The Free Encyclopedia*. Retrieved 22:41, May 1, 2016, from https://en.wikipedia.org/w/index.php?title=Koch_snowflake&oldid=712767769