UNIVERSIDAD DE EL SALVADOR
FACULTAD DE CIENCIAS NATURALES Y MATEMÁTICA
ESCUELA DE MATEMÁTICA

Universidad de El Salvador
Hacia la libertad por la cultura

"PROPUESTA METODOLÓGICA PARA EL ESTUDIO DE LAS NOCIONES FUNDAMENTALES DE SISTEMAS DINÁMICOS, GEOMETRÍA FRACTAL Y TEORÍA DEL CAOS USANDO EL SOFTWARE MATHEMATICA".

TRABAJO DE GRADUACIÓN PRESENTADO POR:
OSCAR ARMANDO HERNÁNDEZ MORALES.

PARA OPTAR AL TÍTULO DE:
LICENCIADO EN MATEMÁTICA.

ASESORES:
MSc. MARTÍN GUERRA CÁCERES.
Lic. RIQUELMI SALVADOR CARDONA FUENTES.

CIUDAD UNIVERSITARIA, AGOSTO DEL 2009.

UNIVERSIDAD DE EL SALVADOR

MSc. RUFINO ANTONIO QUEZADA SÁNCHEZ.
RECTOR

Lic. DOUGLAS VLADIMIR ALFARO CHÁVEZ.
SECRETARIO GENERAL

Dr. RAFAEL ANTONIO GÓMEZ ESCOTO.
DECANO

LICDA. MARIA TRINIDAD TRIGUEROS DE CASTRO.
SECRETARIA

ING. CARLOS MAURICIO CANJURA.
DIRECTOR DE LA ESCUELA DE MATEMÁTICA

CIUDAD UNIVERSITARIA, AGOSTO DEL 2009.

Índice

Introducción

Las matemáticas, como muchas otras áreas del pensamiento, han sufrido en el tercio central del siglo XX el impacto de la corriente filosófica estructuralista. Esta tendía a desplazar el centro de atención hacia los problemas de fundamentación por una parte, y por otra subrayaba la importancia de las estructuras abstractas como la de conjunto, grupo u otras, que se presentan en diversas áreas de las matemáticas. En general la corriente estructuralista impregna a las matemáticas de los métodos del álgebra y es compañera inevitable de una tendencia hacia la abstracción.

El estructuralismo ha estado lejos de ser un factor determinante en el desarrollo de la producción matemática en el último siglo, ya que el volumen ingente de investigación volcada hacia las aplicaciones ha pesado de forma decisiva en el resultado global. Sin embargo, es en el ámbito de la enseñanza de las matemáticas donde la influencia del estructuralismo ha sido más profunda, penetrando en los programas a todos los niveles educativos y provocando que al estudiar matemáticas, los estudiantes se queden con la impresión de que no hay nada nuevo en matemáticas desde Euclides o Pitágoras, es decir, desde hace más de 2000 años. Con un poco de suerte, algunos se cree que las matemáticas dejaron de desarrollarse después de la creación del cálculo diferencial e integral (hace unos 300 años), en cambio no tenemos la misma impresión sobre otras ciencias como física, química o biología.

Parte del problema es que los temas que se han investigado recientemente en matemáticas requieren de conocimientos más avanzados que los propuestos en los Programas de Estudio y quedan fuera temas tan interesantes como geometrías no euclidianas. Los fractales, justamente, nos dan la oportunidad de abordar algunos resultados obtenidos en estos últimos treinta años que son sencillos de entender en una primera aproximación, nos muestran una teoría en pleno desarrollo en la que hay muchas propiedades que se sospechan pero aun no se han podido demostrar, y que tienen aplicaciones a veces insospechables.

La geometría fractal, cuyos primeros desarrollos datan de finales del siglo XIX, ha recibido durante los últimos treinta años, desde la publicación de los trabajos de Mandelbrot, una atención y un auge crecientes. Lejos de ser simplemente una herramienta de generación de impresionantes paisajes virtuales, la geometría fractal viene avalada por la teoría geométrica de la medida y por innumerables aplicaciones en ciencias tan dispares como la Física, la Química, la Economía o, incluso, la Informática.

A pesar de lo anteriormente expuesto son muchas las circunstancias positivas que confluyen en esta área que pueden ser aprovechadas para que la revolución, que ya ha tenido lugar en la investigación matemática, tenga una prolongación natural en el aula. Dentro de las cuales podemos mencionar la teoría matemática denominada sistemas de funciones iteradas, desarrollada en 1981 por Hutchinson, se convertiría a finales de la década de los 80 con los trabajos de Barnsley en una de las técnicas más innovadoras y prometedoras en el campo de la compresión de imágenes; misma que llevara fácilmente a las primeras nociones de fractales y que ayudaría a analizar desde un enfoque diferente una ecuación muy simple pero de una gran utilidad en muchas áreas (física, biología, economía, etc.) como lo es la ecuación logística.

Al mundo de los fractales muchos llegan atraídos por el estallido de color de alguna representación del conjunto de Mandelbrot o de un conjunto de Julia. Sin embargo, una vez que se profundiza en la magia de los fractales, uno no sabe que admirar más, si las cascadas multicolor o la belleza de las matemáticas que las engendran. Aunque este trabajo, tan solo nos permite asomarnos por primera vez a este mundo, encontramos una introducción desde cero a los principios elementales de la revolución fractal. Hay también quienes disfrutan desde hace ya tiempo generando intrincados bosques fractales o fotografías a profundidades abismales tras iterar fórmulas sencillas mediante programas confeccionados, incluso, por ellos mismos. Algunos serían los geómetras del siglo XXI armados esta vez de ordenadores cada vez más rápidos que, aun así, cuando se trata de fractales, siempre parecen quedarse pequeños. Conceptos como los de dimensión de Hausdorff o conjunto autosemejante son de vital importancia para abordar con ciertas garantías de éxito la exploración de nuevos continentes fractales, razón por la cual serán presentados en este trabajo de investigación con un enfoque intuitivo, dado que está es una mera revisión de los conceptos elementales de una manera totalmente constructiva.

1

Introducción a los sistemas dinámicos discretos

1.1. Iteración de funciones

Los sistemas dinámicos que se estudiaran están íntimamente relacionados con la idea de iteración de una función, es decir, con el comportamiento del conjunto $\{x_0,\ f(x_0),\ f^2(x_0),\ f^3(x_0),\ \ldots\ f^n(x_0),\ \ldots\}$, donde x_0 es un punto del dominio de f y $f^n = f \circ f^{n-1}$. Para ello es necesario conocer algunas de las funciones predeterminadas de *Mathematica* que permiten iterar una función.

Considérese por ejemplo que se desea estudiar iteraciones asociadas con la función raíz cuadrada, que en *Mathematica* se indica por *Sqrt*[1]. Claro está que se deben considerar únicamente números no negativos. Por ejemplo, si se toma el valor inicial $x = 16$, se puede calcular diez iteraciones utilizando la función *NestList*:

```
NestList[Sqrt, 16, 10]
```

$$\left\{16,\ 4,\ 2,\ \sqrt{2},\ 2^{1/4},\ 2^{1/8},\ 2^{1/16},\ 2^{1/32},\ 2^{1/64},\ 2^{1/128},\ 2^{1/256}\right\}$$

Tal como está el resultado anterior no indica demasiado, ya que *Mathematica* trabaja en forma simbólica salvo que se le indique lo contrario. Así, si se desea obtener como resultado una lista de números, tal como en una calculadora, es imprescindible utilizar la función *N* ó escribir 16.0 en vez de 16 para indicar que se está trabajando con valores aproximados.

```
NestList[Sqrt, N[16], 10]
```

```
{16., 4., 2., 1.41421, 1.18921, 1.09051, 1.04427, 1.0219, 1.01089, 1.00543, 1.00271}
```

Se dice que esta lista de números son los primeros once elementos de la **Órbita** del **Valor Inicial** 16.

Actividad 1:

Construya los primeros quince elementos de la órbita para cada uno de los siguientes valores iniciales:

a. 10, 100, 1000, y 100 000.

b. 0.1, 0.01, 0.0001 y 0.00001.

c. ¿Observa alguna regularidad en los resultados obtenidos?

d. Corrobore sus observaciones aumentando el número de iteraciones.

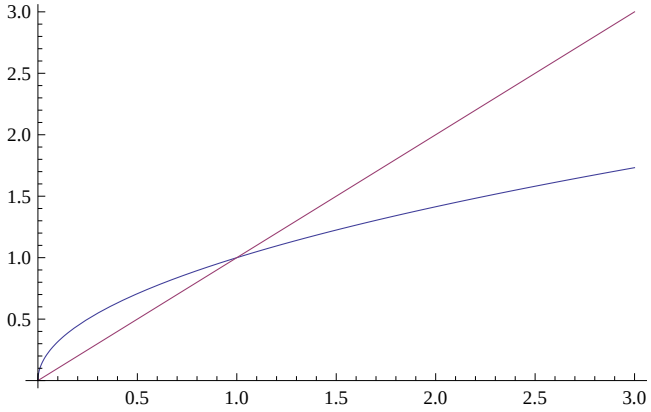
e. ¿Ocurre un nuevo fenómeno si los valores iniciales están cercanos a uno?

Basado en los experimentos anteriores y dado que la órbita para el valor inicial igual a uno consta únicamente de dicho número (como puede comprobarse), podría cometerse el error de afirmar que para todos los valores iniciales la órbita respectiva, para un número suficientemente grande todos los elementos de la órbita son iguales a dicho número, pero esto se puede refutar considerando el valor inicial cero, ya que la órbita consiste exclusivamente de cero.

[1] Consulte en la ayuda de *MathematicalFunctions*.

Una forma de determinar estos valores es encontrar los puntos de corte de la función y la recta identidad, como puede verse en el gráfico:

```
Plot[{√x, x}, {x, 0, 3}]
```



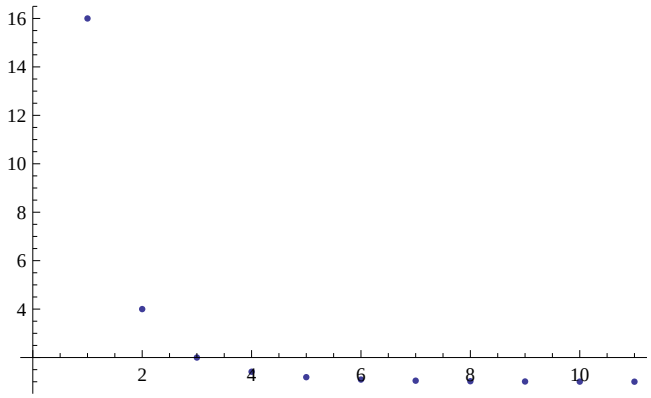
A cada una de las abscisas de los puntos de intersección de las curvas anteriores se denominan punto fijo de la función raíz cuadrada

Actividad 2:

- **a.** Realice nuevamente la actividad uno, para la función elevar al cuadrado.
- **b.** Superponga las gráficas de la función elevar al cuadrado y la función identidad.
- **c.** Determine los puntos fijos de la función elevar al cuadrado.

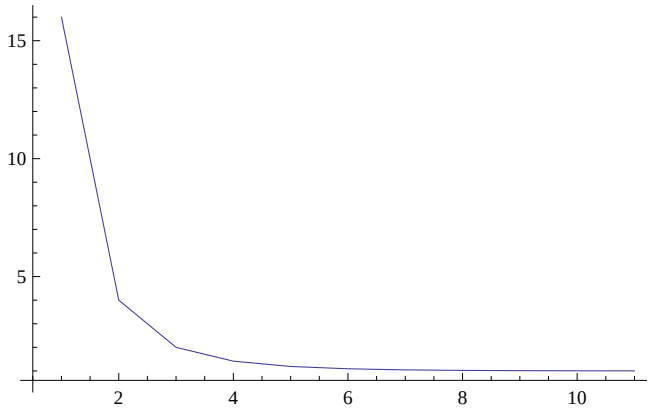
Otra manera de estudiar las órbitas de la función raíz cuadrada es graficar la sucesión de puntos que se genera al iterar una función dada mediante la función *ListPlot*.

```
ListPlot[NestList[Sqrt, N[16], 10]]
```



Aunque la órbita sólo es una sucesión de puntos, se puede utilizar el atributo *Joined* de la función *ListPlot*, con el fin de visualizar el comportamiento de dicha órbita.

```
ListPlot[NestList[Sqrt, 16., 10], Joined → True, AxesOrigin → {.5, .6}]
```



Actividad 3:

Verifique que las conclusiones resultantes de las dos primeras actividades son válidas, graficando cada una de las órbitas para los valores iniciales dados en cada una de ellas.

1.1.1. Puntos fijos, atractivos y repulsivos

Un punto x_0 es punto fijo de la función f si $f(x_0) = x_0$. También se puede describir como un punto para el cual su órbita se reduce a dicho punto. Un punto fijo x_0 de la función f será **atractivo** o **atrayente** si puntos cercanos a x_0 tienen órbitas que se le aproximan. Un punto fijo x_0 de la función f será **repulsivo** o **repelente** si puntos cercanos a x_0 tienen órbitas que se alejan de él.

Actividad 4:

Utilice la función *Solve* para encontrar los puntos fijos de las funciones raíz cuadrada y elevar al cuadrado.

Sugerencia: Para cada función f debe resolverse la ecuación $f(x) = x$, ya que esto significa que x es la intersección de la curva de f y la función identidad.

Actividad 5:

Utilice la función *FindRoot* para encontrar los puntos fijos de las funciones raíz cuadrada y elevar al cuadrado. Además, utilice la función *Manipulate* junto con la función *FindRoot*, para comprobar que el punto fijo que se calcula depende del valor inicial que se considere.

Sugerencia: Dado que la función *FindRoot* devuelve aproximaciones a las raíces de la ecuación o función que se analiza, puede utilizarse la función *Chop*.

1.1.2. Importancia del número de iteraciones

Con el siguiente ejemplo se pretende destacar la importancia del número de iteraciones que se deben hacer de acuerdo con la función, el valor inicial, entre otros aspectos, ya que se pueden hacer afirmaciones incorrectas, además se debe tener en cuenta las limitaciones de la computadora, etc.

Una forma de definir funciones en *Mathematica* es la siguiente:

```
f[x_] := 1/3. (2 - e^x + x^2)
```

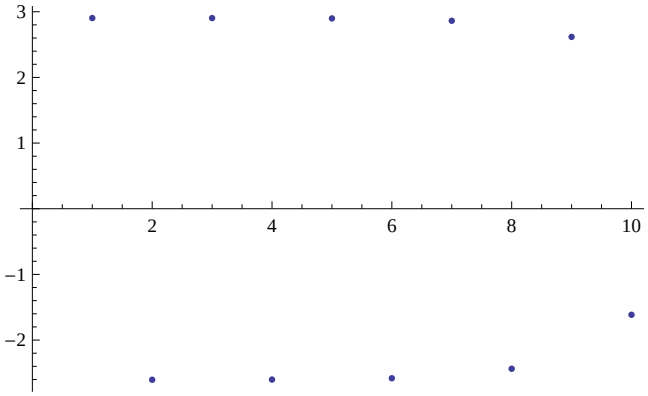
Observe que basta con un valor aproximado en el argumento de la función para que ninguna imagen sea un valor exacto, como se comprueba a continuación:

```
f[5]
f[1/7]
```

```
-40.4711

0.288948
```

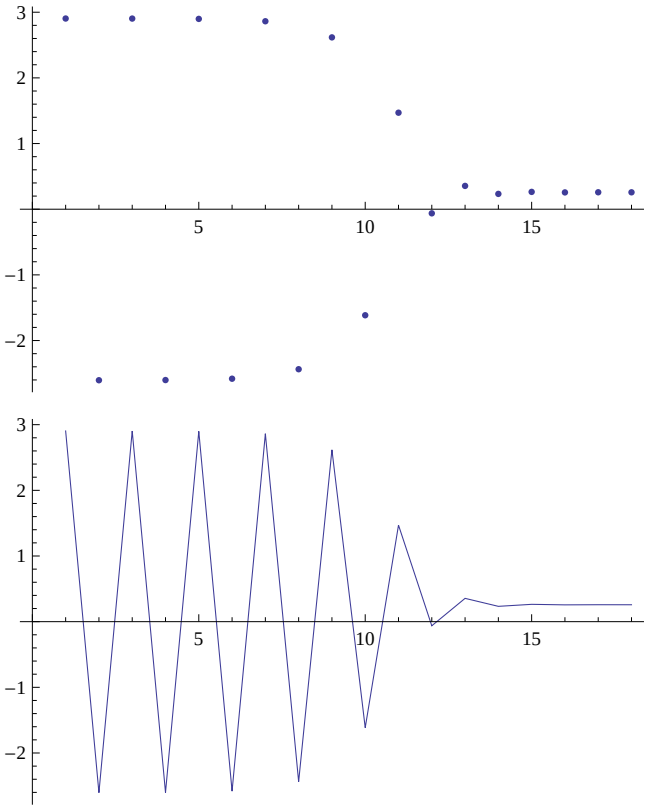
A continuación se construyen los primeros diez elementos de la órbita del valor inicial $x_0 = 2.904$.

```
x0 =.; x0 = 2.904; ListPlot[NestList[f, x0, 9], Joined → False]
```



Al observar el gráfico parecerá que los valores se aproximan a dos puntos distintos, pero al aumentar el número de iteraciones se comprueba que realmente la órbita converge a un punto fijo.

```
ListPlot[NestList[f, x0, 17], Joined → False]
ListPlot[NestList[f, x0, 17], Joined → True]
```



Dicho punto fijo se puede calcular utilizando la función *FixedPoint*:

```
FixedPoint[f, x0]
```

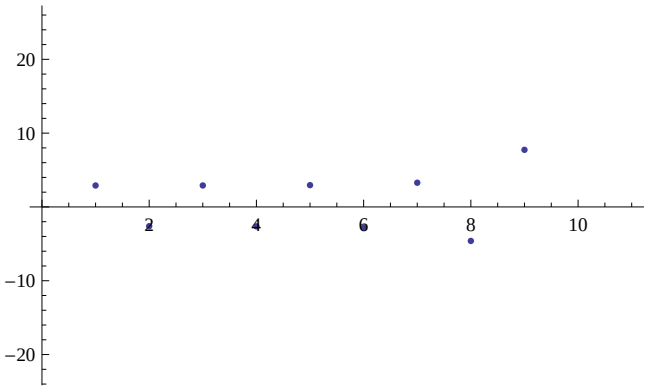
```
0.25753
```

Empero, algunas veces no se pueden hacer muchas iteraciones, por ejemplo, si se construye la órbita para el valor inicial anterior incrementado en 0.001, *Mathematica* graficará solamente algunos elementos de la misma, junto con un mensaje de error.

```
x0 = .; x0 = 2.905; ListPlot[NestList[f, x0, 17], Joined → False]
```

General::unfl : Underflow occurred in computation. ≫

General::ovfl : Overflow occurred in computation. ≫



El error de desbordamientotambién lo puede dar la función *FixedPoint* y para evitarlo debe buscarse una vecindad del punto fijo que se está buscando, como se comprueba a continuación:

```
FixedPoint[f, x0]
```

General::unfl : Underflow occurred in computation. ≫

General::ovfl : Overflow occurred in computation. ≫

```
Indeterminate
```

Así, una mejor manera de buscar los puntos fijos puede ser utilizar la función *FindRoot,* ya que el valor inicial no necesariamentedebe estar próximo al punto fijo:

```
FindRoot[f[x] == x, {x, 20 x0}]
```

```
{x → 0.25753}
```

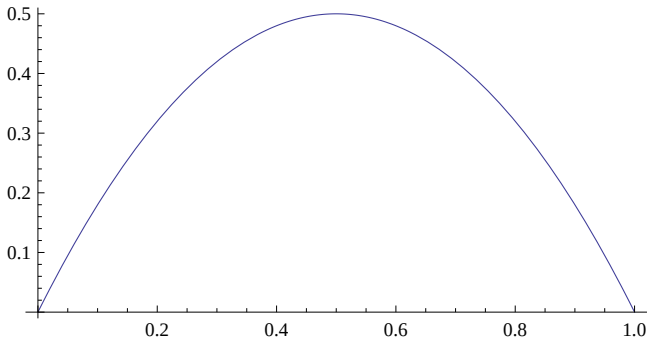
Actividad 6:

- **a.** Determine los puntos fijos de las función $g(x) = \sqrt[3]{x+1}$. Luego, construya la órbita (y grafíquela) de puntos iniciales próximos al punto fijo, para determinar si es atractivo o repulsivo.
- **b.** Realice el ejercicio anterior para la función $h(x) = \pi + \frac{1}{2}\,\text{Sen}[\frac{x}{2}]$.

1.2. Análisis de la ecuación logística

Suponiendo que la constante c es positiva se verifica que la función $f(x) = c\,x(1-x)$ es una parábola.

```
Clear[f, c]
c = 2.;
f[x_] := c x (1 - x)
Plot[f[x], {x, 0, 1}, AspectRatio → Automatic]
```



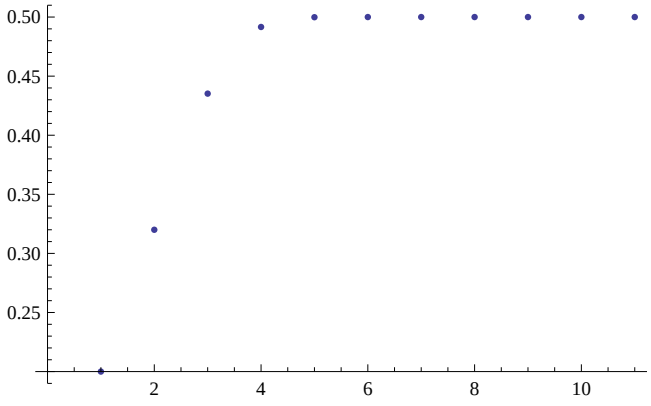
Teniendo en cuenta su interpretación como fenómeno biológico, lo que nos interesa es tomar un valor de c positivo, un punto x_0 entre 0 y 1, y ver que pasa con el conjunto de las sucesivas iteraciones, que no es más que la órbita de x_0 bajo la función f; el comportamiento de todas las órbitas es llamado *comportamiento dinámico* de f.
Tomando por ejemplo $c = 2$ y $x_0 = 0.2$, los primeros 10 valores aproximados de la órbita, incluido el inicial, están dados en la lista:

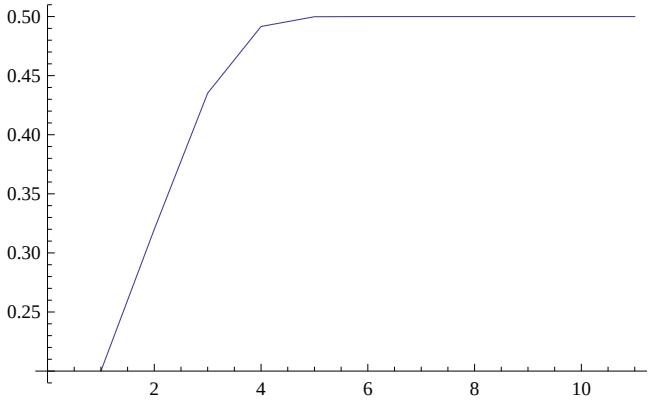
```
NestList[f, .2, 10]
```

```
{0.2, 0.32, 0.4352, 0.491602, 0.499859, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5}
```

Se ve que después de muy poco, el valor se mantiene estacionariamente en $\frac{1}{2}$. Esto se puede verificar gráficamente como sigue:

```
ListPlot[NestList[f, 0.2, 10], Joined → False]
ListPlot[NestList[f, 0.2, 10], Joined → True]
```





Empleando *Mathematica* se verifica que $\frac{1}{2}$ es punto fijo.

```
f[1/2]
```

```
0.5
```

Por supuesto $\frac{1}{2}$ no es el único punto fijo, como se puede comprobar la evaluar f en cero.

```
f[0]
```

```
0
```

Al tomar otro punto inicial, por ejemplo $x_0 = 0.7$, se obtine

```
NestList[f, .7, 10]
```

```
{0.7, 0.42, 0.4872, 0.499672, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5}
```

Se observa que también al poco tiempo los valores se parecen a $\frac{1}{2}$.

Ciertamente para cualquier parámetro $c \neq 0$ siempre se obtienen dos puntos fijos, estos son:

```
c =.; Solve[f[x] == x, x]
```

$$\left\{\{x \to 0\}, \left\{x \to \frac{-1+c}{c}\right\}\right\}$$

Así, cuando $c = 2$ las raíces son:

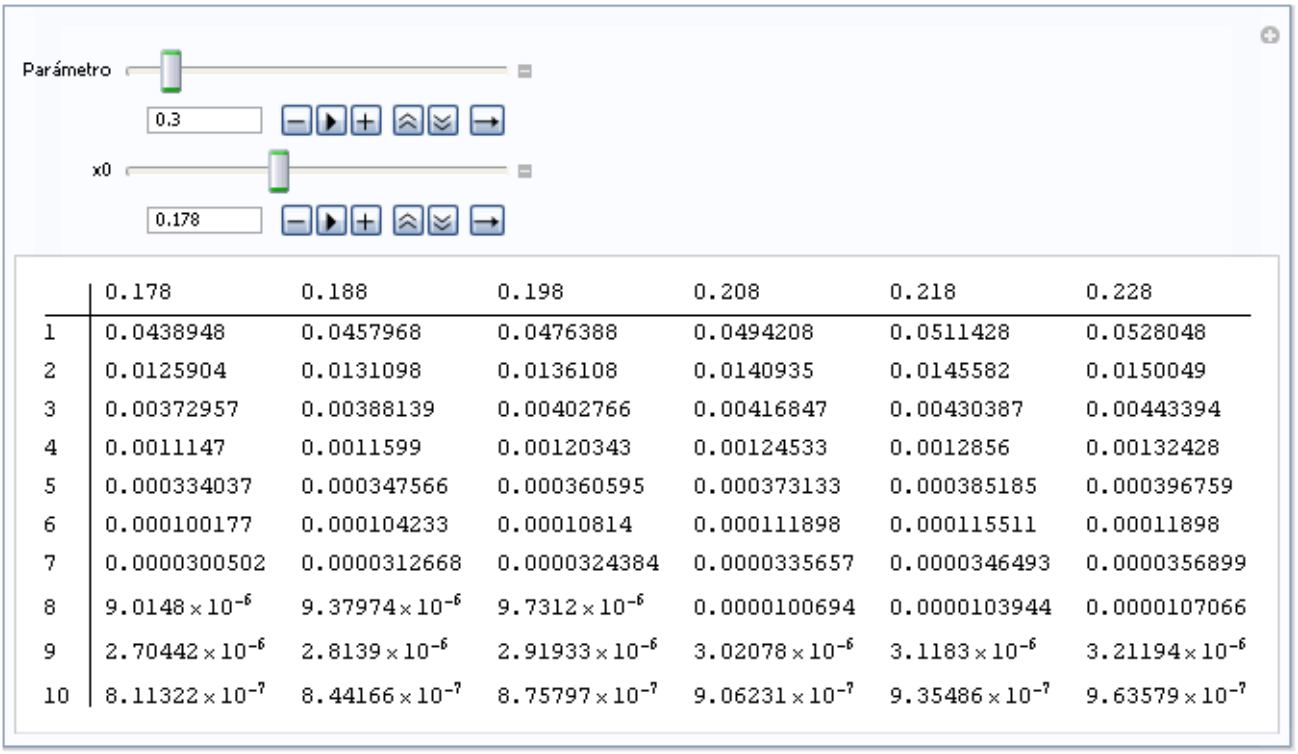
```
% /. c → 2 // Flatten
```

$$\left\{x \to 0, x \to \frac{1}{2}\right\}$$

1.2.1. Buscando puntos fijos atractivos y repulsivos.

En general, se puede preguntar si al iterar la función f los valores se van acercando o alejando de algún punto fijo. Hágase la prueba con distintos valores de c y x. En cuyas pruebas, no hará falta considerar $x = 0$ ó $x = \frac{1}{2}$ los cuales son puntos fijos, ni tampoco valores mayores que $\frac{1}{2}$, puesto que si $\frac{1}{2} < x < 1$, definiendo $y = 1 - x$, ya en la primera iteración tenemos $f(x) = c\,x\,(1 - x) = c\,(1 - y)\,y = f(y)$, donde y varia entre 0 y $\frac{1}{2}$ (Esto, significa que f es simétrica respecto de la recta $x = \frac{1}{2}$). Por ejemplo, al elaborar una tabla para las primeras 10 iteraciones para $c = 0.3$ y para distintos valores iniciales, parece claro que los valores se aproximan a 0, como se observa en la siguiente tabla.

```
Manipulate[
 TableForm[
  Rest[
    Transpose[
      Map[NestList[c # (1 - #) &, #, 10] &, V_inicial = Table[x0 + Δx, {Δx, 0, .05, .01}]]]],
  TableHeadings → {Automatic, V_inicial}, TableSpacing → {2, 2}],
 {{c, .3, "Parámetro"}, 0.001, 3, .1, Appearance → "Open"},
 {x0, 0, .45, Appearance → "Open"}, SaveDefinitions → True]
```



	0.178	0.188	0.198	0.208	0.218	0.228
1	0.0438948	0.0457968	0.0476388	0.0494208	0.0511428	0.0528048
2	0.0125904	0.0131098	0.0136108	0.0140935	0.0145582	0.0150049
3	0.00372957	0.00388139	0.00402766	0.00416847	0.00430387	0.00443394
4	0.0011147	0.0011599	0.00120343	0.00124533	0.0012856	0.00132428
5	0.000334037	0.000347566	0.000360595	0.000373133	0.000385185	0.000396759
6	0.000100177	0.000104233	0.00010814	0.000111898	0.000115511	0.00011898
7	0.0000300502	0.0000312668	0.0000324384	0.0000335657	0.0000346493	0.0000356899
8	9.0148×10^{-6}	9.37974×10^{-6}	9.7312×10^{-6}	0.0000100694	0.0000103944	0.0000107066
9	2.70442×10^{-6}	2.8139×10^{-6}	2.91933×10^{-6}	3.02078×10^{-6}	3.1183×10^{-6}	3.21194×10^{-6}
10	8.11322×10^{-7}	8.44166×10^{-7}	8.75797×10^{-7}	9.06231×10^{-7}	9.35486×10^{-7}	9.63579×10^{-7}

Actividad 1:

Observe el comportamiento de la ecuación logística para los siguientes valores de *c*: 0.8; 1; 1.2; 1.5; 2 y 3.

a. ¿Se aproxima a algún punto fijo la órbita?

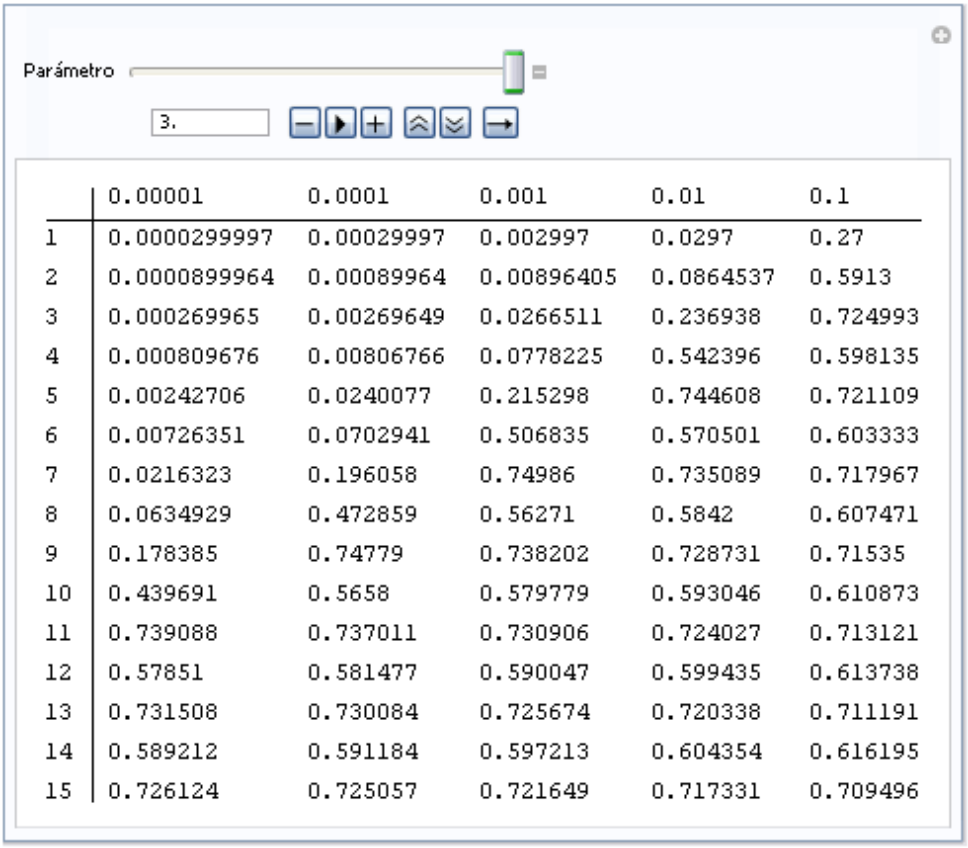
b. ¿Si aumenta a 50 el número de iteraciones se modifica el comportamiento de cada órbita?

c. ¿Depende este comportamiento del valor inicial?

d. Realizando más experimentos infiera el intervalo al cual pertenece el parámetro *c* para que la órbita se aproxime a cada uno de los puntos fijos.

Con respecto a la primera pregunta tómese por ejemplo $c = 3$, con 15 iteraciones se observa que efectivamente los valores (de la órbita) se van alejando del punto fijo cero, aunque los valores iniciales estén muy próximos al mismo, como se comprueba en la tabla siguiente.

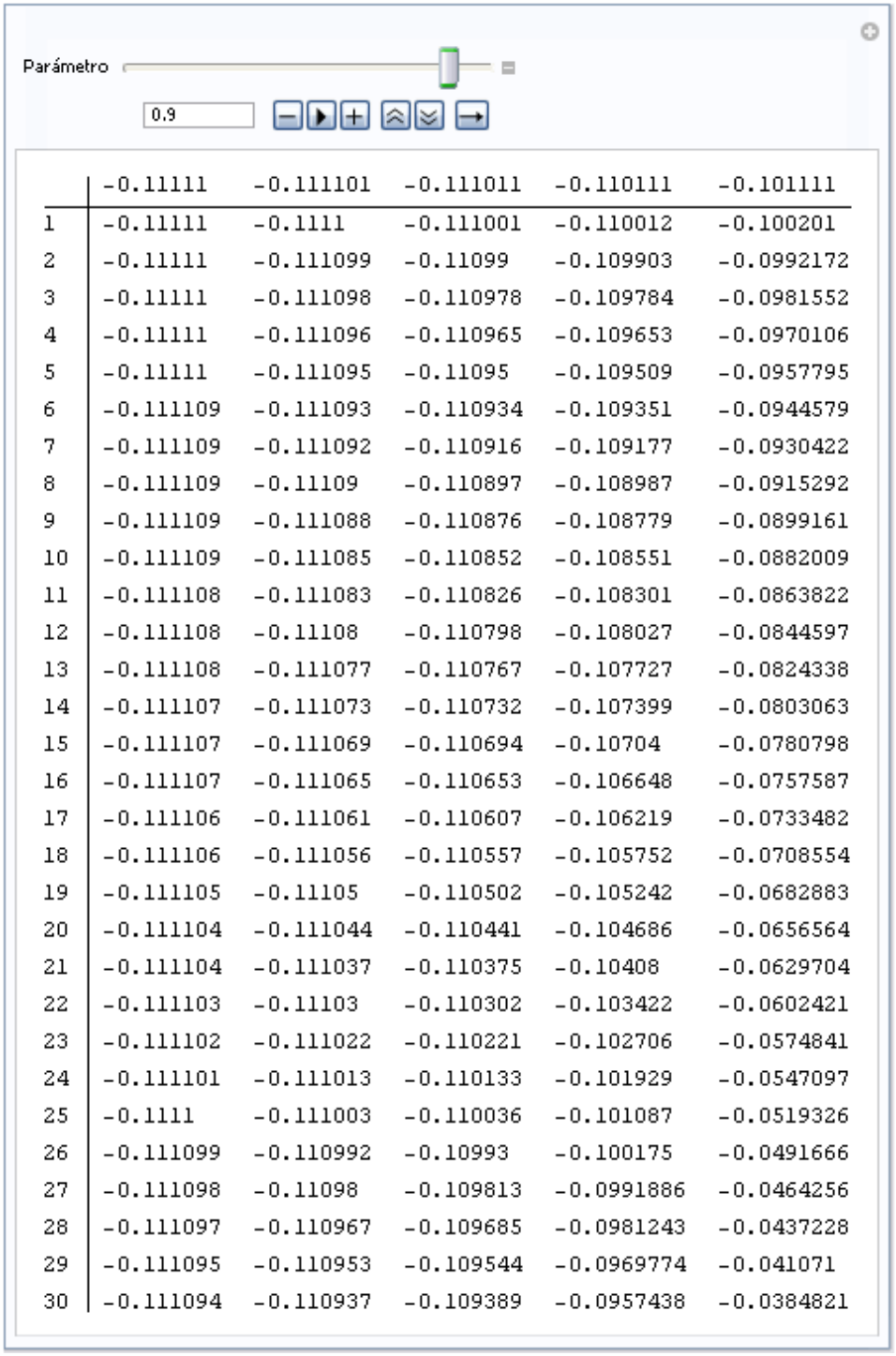
```
Manipulate[
 TableForm[
  Rest[
    Transpose[
      Map[NestList[c # (1 - #) &, #, 15] &, V_inicial = Table[N[10^(k-5)], {k, 0, 4}]]]],
  TableHeadings → {Automatic, V_inicial}, TableSpacing → {2, 2}],
 {{c, 3, "Parámetro"}, 1.001, 3, .1, Appearance → "Open"}, SaveDefinitions → True]
```



	0.00001	0.0001	0.001	0.01	0.1
1	0.0000299997	0.00029997	0.002997	0.0297	0.27
2	0.0000899964	0.00089964	0.00896405	0.0864537	0.5913
3	0.000269965	0.00269649	0.0266511	0.236938	0.724993
4	0.000809676	0.00806766	0.0778225	0.542396	0.598135
5	0.00242706	0.0240077	0.215298	0.744608	0.721109
6	0.00726351	0.0702941	0.506835	0.570501	0.603333
7	0.0216323	0.196058	0.74986	0.735089	0.717967
8	0.0634929	0.472859	0.56271	0.5842	0.607471
9	0.178385	0.74779	0.738202	0.728731	0.71535
10	0.439691	0.5658	0.579779	0.593046	0.610873
11	0.739088	0.737011	0.730906	0.724027	0.713121
12	0.57851	0.581477	0.590047	0.599435	0.613738
13	0.731508	0.730084	0.725674	0.720338	0.711191
14	0.589212	0.591184	0.597213	0.604354	0.616195
15	0.726124	0.725057	0.721649	0.717331	0.709496

Como resultado del ejercicio anterior, puede suponerse que siempre se converge a un punto fijo, que es 0 cuando $c \leqslant 1$ y es $1 - 1/c$ para $1 < c \leqslant 3$. Pero aún no hemos respondido completamente la pregunta de si los puntos fijos son atractivos o repulsivos. Surgen pues las dos preguntas siguientes: ¿Para $1 < c \leqslant 3$, $x = 0$ es atractivo o repulsivo? y ¿Para $0 < c \leqslant 1$, $x = 1 - 1/c$ es atractivo o repulsivo?

```
Manipulate[
 TableForm[
  Rest[
   Transpose[
    Map[NestList[c # (1 - #) &, #, 30] &, V_inicial = Table[1. - 1/c + 10^(k-6), {k, 0, 4}]]]],
  TableHeadings → {Automatic, V_inicial}, TableSpacing → {2, 2}],
 {{c, .9, "Parámetro"}, 0.001, 1, .1, Appearance → "Open"}, SaveDefinitions → True]
```



	-0.11111	-0.111101	-0.111011	-0.110111	-0.101111
1	-0.11111	-0.1111	-0.111001	-0.110012	-0.100201
2	-0.11111	-0.111099	-0.11099	-0.109903	-0.0992172
3	-0.11111	-0.111098	-0.110978	-0.109784	-0.0981552
4	-0.11111	-0.111096	-0.110965	-0.109653	-0.0970106
5	-0.11111	-0.111095	-0.11095	-0.109509	-0.0957795
6	-0.111109	-0.111093	-0.110934	-0.109351	-0.0944579
7	-0.111109	-0.111092	-0.110916	-0.109177	-0.0930422
8	-0.111109	-0.11109	-0.110897	-0.108987	-0.0915292
9	-0.111109	-0.111088	-0.110876	-0.108779	-0.0899161
10	-0.111109	-0.111085	-0.110852	-0.108551	-0.0882009
11	-0.111108	-0.111083	-0.110826	-0.108301	-0.0863822
12	-0.111108	-0.11108	-0.110798	-0.108027	-0.0844597
13	-0.111108	-0.111077	-0.110767	-0.107727	-0.0824338
14	-0.111107	-0.111073	-0.110732	-0.107399	-0.0803063
15	-0.111107	-0.111069	-0.110694	-0.10704	-0.0780798
16	-0.111107	-0.111065	-0.110653	-0.106648	-0.0757587
17	-0.111106	-0.111061	-0.110607	-0.106219	-0.0733482
18	-0.111106	-0.111056	-0.110557	-0.105752	-0.0708554
19	-0.111105	-0.11105	-0.110502	-0.105242	-0.0682883
20	-0.111104	-0.111044	-0.110441	-0.104686	-0.0656564
21	-0.111104	-0.111037	-0.110375	-0.10408	-0.0629704
22	-0.111103	-0.11103	-0.110302	-0.103422	-0.0602421
23	-0.111102	-0.111022	-0.110221	-0.102706	-0.0574841
24	-0.111101	-0.111013	-0.110133	-0.101929	-0.0547097
25	-0.1111	-0.111003	-0.110036	-0.101087	-0.0519326
26	-0.111099	-0.110992	-0.10993	-0.100175	-0.0491666
27	-0.111098	-0.11098	-0.109813	-0.0991886	-0.0464256
28	-0.111097	-0.110967	-0.109685	-0.0981243	-0.0437228
29	-0.111095	-0.110953	-0.109544	-0.0969774	-0.041071
30	-0.111094	-0.110937	-0.109389	-0.0957438	-0.0384821

Observación: Si $0 < c < 1$ entonces $1 - 1/c < 0$, y la órbita de $x_0 > 0$ siempre toma valores positivos, con lo que nunca se aproximará al punto fijo $1 - 1/c$. Empero, para determinar si dicho punto fijo es repulsivo es preciso analizar las órbitas de valores iniciales cercanos a éste.

En la tabla anterior se puede apreciar que después de 30 iteraciones, la imagen del punto −0.11111 es aproximadamente igual (debido a errores de aproximación) a la imagen obtenida después de 6 iteraciones para el punto −0.111101, por lo que al iterar 24 veces más la imagen del punto −0.11111 será aproximadamente −0.110937, la cual a su vez es aproximadamente igual a la sexta imagen del punto −0.111011, continuando de esta manera se infiere que los valores de la órbita del punto −0.11111, se alejan del punto fijo −0.111 ... al aumentar el número de iteraciones, por lo que este es repulsivo.

Actividad 2:

Al observar detenidamente el trabajo realizado previamente, puede percatarse que los valores del parámetro son menores o igual a tres. Surge pues la pregunta siguiente ¿Cuándo el parámetro c aumente se obtienen resultados similares u ocurre un nuevo fenómeno?
Observe el comportamiento de la órbita, con al menos 35 iteraciones, para cada uno de los siguientes valores del parámetro c : 3.01; 3.02; 3.18; 3.2; 3.32; 3.43; 3.5; 3.67 y 3.9.

a. ¿La órbita se aproxima a algún punto fijo?

b. ¿Existe alguna regularidad en el comportamiento?

Recordatorio: Los valores iniciales deben variar entre cero y un medio.

1.2.2. Ciclo de longitud dos.

Se observa en las tablas construidas anteriormente que ocurre un nuevo fenómeno. Por ejemplo para $c = 3.2$, la tabla muestra que las órbitas se acercan alternadamente a 0.513045 y 0.799455 y no se aproximan a un punto fijo, que en este caso son $x = 0$ y $x = 1 - 1/3.2 \approx 0.6875$. En este caso se dice que la órbita es periódica (atractiva) de período dos, o que se tiene un ciclo (atractivo) de longitud dos.

Por definición, para tener un ciclo de longitud dos, cada uno de los puntos u en el ciclo tiene que satisfacer la ecuación

$$f^2(u) = f(f(u)) = u \tag{1.1}$$

y no ser un punto fijo, es decir no ser raíz de la ecuación

$$f(u) = u \tag{1.2}$$

Utilizando *Mathematica* se pueden encontrar las raíces de la ecuación 1.1, cuando f es la función logística:

```
(*Clear[f,c];f[x_]:=c x(1-x)*)
Flatten[Simplify[Solve[(f[f[x]] - x)/(x (x - (c-1)/c)) == 0, {x}]]]
```

$$\left\{x \to \frac{1 + c - \sqrt{-3 - 2c + c^2}}{2c},\ x \to \frac{1 + c + \sqrt{-3 - 2c + c^2}}{2c}\right\}$$

Cuando $c = 3.2$ estos valores son aproximadamente

```
% /. c → 3.2
```

```
{x → 0.513045, x → 0.799455}
```

que son los valores que ya se detectaron anteriormente.

Al analizar la cantidad subradical de las raíces de la ecuación antes resuelta, suponiendo que c es real, se obtienen dos puntos reales y distintos que forman un ciclo de longitud dos cuando

$$-3 - 2c + c^2 > 0 \Leftrightarrow (c-1)^2 > 4 \Leftrightarrow |c-1| > 2 \Leftrightarrow c > 3 \vee c < -1$$

De forma tal que, si $c > 0$ sólo puede haber un ciclo de longitud dos cuando $c > 3$.

De igual manera se encuentra ciclos de mayor longitud. Por ejemplo, para $c = 3.5$ observará en la tabla elaborada anticipadamente que los sucesivos valores se aproximan alternadamente a 0.38282; 0.826941; 0.500884 y 0.874997, formando un ciclo atractivo que es de longitud cuatro, aún cuando los puntos fijos son $x = 0$ y $x = 5/7 \approx 0.71428$ en este caso.

Es importante señalar que no es factible realizar un análisis como el elaborado para los ciclos de longitud dos a fin de encontrar los puntos de ciclos de longitud *"n"* arbitraria, pero las ecuaciones resultantes, que serán en general de grado $n - 2$ después de simplificar, no se pueden resolver en forma cerrada salvo casos muy particulares, por lo que es imprescindible elaborar un análisis más cualitativo.

Nuevamente surge la pregunta si aparecen los mismos o nuevos fenómenos al aumentan el valor del parámetro, razón por la cual se propone realizar la siguiente actividad.

Actividad 3:

Elabore tablas y estudie el comportamiento de las órbitas para distintos valores iniciales entre cero y un medio, y para $c =$ 4; 4.01 y 5.

a. Para cada valor del parámetro calcule los puntos fijos.

b. Determine los elementos del ciclo (si existen) y cual es su longitud, o en el caso en el que se presente un nuevo fenómeno descríbalo.

Los resultados de la anterior actividad para el caso de $c = 4$, sugieren un análisis más detallado, por lo cual se debe recurrir a elaborar gráficas (lo que motiva la próxima actividad). Mientras que los resultados obtenidos para $c = 4.01$, sirven para confirmar la importancia del valor $c = 4$, pues con tan solo una variación en el parámetro de 0.01 se puede afirmar que si el valor absoluto de c es suficientementegrande, "casi" todos los puntos tiene órbitas cuyos valores son cada vez más negativos (aseveración confirmada por los resultados para $c = 5$), donde el "casi" es debido al caso $c = 5$ y $x_0 = 0.2$, por ejemplo, en el cual se observa que las preimágenes de puntos fijos tienen órbitas que terminan en el punto fijo.

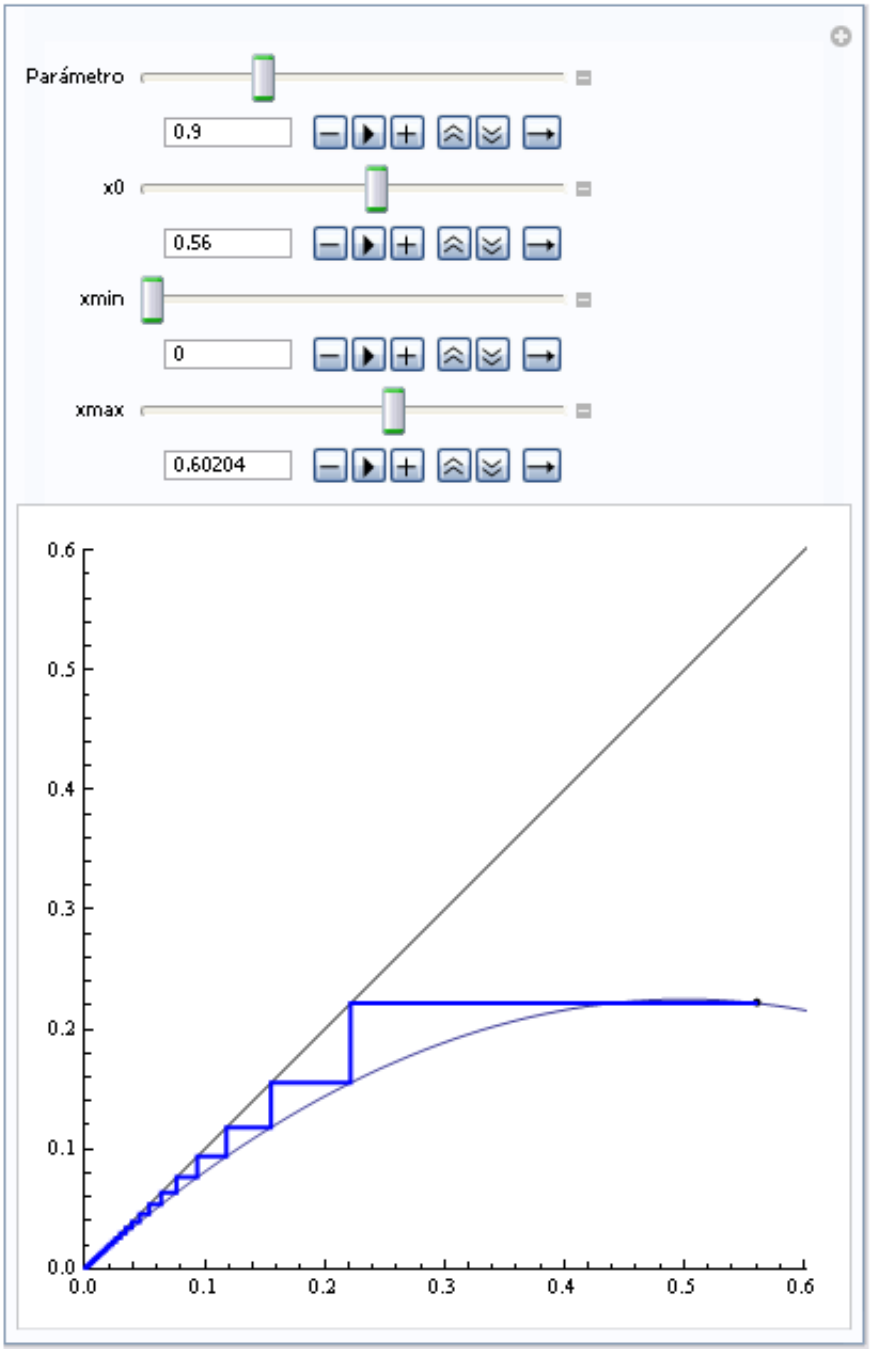
1.2.3. Análisis Gráfico.

Para continuar con el estudio del comportamiento de las órbitas es útil y convenientes hacer un *análisis gráfico.* Para ello se sigue el proceso que a continuación se describe: Un punto del gráfico de f se mueve horizontalmentehasta intersecar la diagonal $y = x$, seguidamentese mueve verticalmentehasta intersecar el gráfico de f, y continúa así moviéndose horizontal y verticalmente en forma alternada.

De manera más formal este proceso es como sigue: Dado el punto x_0, localizado en el eje horizontal, el siguiente punto x_1 en la órbita se obtiene como $x_1 = f(x_0)$, y por lo tanto se dibuja en el eje vertical. Después se grafica x_1 en el eje horizontal, puesto que $x_2 = f(x_1)$. Es necesario entonces pasar x_1 de un eje a otro, y para ello se dibuja una diagonal a 45º en el primer cuadrante, luego, trazando una línea horizontal se une el punto x_1 que está en el eje vertical con un punto en la diagonal, finalmentese hace la proyección ortogonal de dicho punto sobre el eje horizontal para obtener x_1 sobre dicho eje, y se repite el proceso.

La siguiente función se basa en la descripción del proceso antes descrito, analizando gráficamente la órbita de x_0 bajo la función f para las primeras iteraciones y haciendo el gráfico en el intervalo delimitado por x_{min} y x_{max}.

```
trayectoria = .
trayectoria[función_, x0_, iteraciones_, xmin_, xmax_] :=
  Block[{orbita, lineas},
     orbita = NestList[función, x0, iteraciones];
     lineas = Line[Rest[Partition[Flatten[Transpose[{orbita, orbita}]], 2, 1]]];
       Plot[función[x], {x, xmin, xmax},
            PlotRange → {{xmin, xmax}, {xmin, xmax}},
            AspectRatio → Automatic,
            Epilog → {Line[{{xmin, xmin}, {xmax, xmax}}],
                      {AbsolutePointSize[4], Point[{x0, función[x0]}]},
                      {AbsoluteThickness[2], Blue, lineas}}]]

Manipulate[trayectoria[c # (1 - #) &, x0, 50, xmin, xmax],
         {{c, 0.6, "Parámetro"}, 0.1, 3, .1}, {x0, 0, 1.}, {xmin, 0, .99},
         {{xmax, 1.}, xmin + .0001, 1}, SaveDefinitions → True]
```



Actividad 4:

Realice un análisis gráfico para cada uno de los valores de *c* que se presenta en las actividades 1, 2 y 3. En cada caso varíe el valor inicial y contraste con los resultados obtenidos mediante elaboración de tablas al estudiar las distintas órbitas.
Sugerencia: Aumente o disminuya los valores de *xmin* y *xman* para obtener un acercamiento o alejamiento del gráfico.

El análisis gráfico también sirve de ayuda para comprender qué pasa cuando el valor inicial de *x* está fuera del intervalo [0, 1] , por lo que se pide: Hacer un análisis gráfico cuando *c* toma los valores 1, 2 y 3, con *x* negativo y *x* mayor que uno.
Aclaración: Esta actividad está diseñada para resaltar el hecho de que en cualquiera de estos casos las órbitas toman valores cada vez más negativos cuando el valor inicial de *x* está fuera del intervalo [0, 1] .

Puesto que la detección de puntos fijos y ciclos atractivos es importante, se presenta a continuación el código para encontrarlos utilizando *Mathematica*. La idea es la siguiente: Se genera una lista de las iteraciones sucesivas de f y otra de las de f^2, ambas a partir de un punto x_0 inicial, es decir, la segunda lista "va dos veces más rápido que la primera". Se van comparando los correspondientes valores hasta encontrar dos valores iguales (uno en cada lista) o exceder un número límite de iteraciones máximas. En caso de obtener dos valores iguales (según la precisión predefinida), por ejemplo en la posición n, se habrá encontrado un ciclo de longitud n, y se prosigue a buscar un ciclo de longitud menor. El resultado de dicho programa será el ciclo de menor longitud o la lista vacía si no se han detectado ciclos.

```
Clear[cicloatractivo, precision]; precision = N[10^-6];
cicloatractivo[función_, x0_, itermax_] :=
        Module[{n, x1, x2, lista}, n = 1; x1 = función[x0]; x2 = función[x1];
                While[(Abs[x1 - x2] > precision) && (n ≤ itermax),
                  x1 = función[x1]; x2 = función[función[x2]]; n++];
                If[n ≤ itermax, (*ciclo de longitud≤n*) lista = {x1}; x2 = función[x1];
                  While[Abs[x1 - x2] > precision, AppendTo[lista, x2]; x2 = función[x2]];
                  lista, {}]];
```

Actividad 5:

Emplee la función *cicloatractivo* para determinar (si es que existen) puntos fijos o ciclos atractivos, para cada uno de los valores de c que se presenta en las actividades 1, 2 y 3, y distintos valores iniciales.

Importante: Para $c = 4$, utilizando la función *cicloatractivo* se comprueba que después de mil iteraciones no se encuentran ciclos atractivos de longitud menor que cien, para diversos valores iniciales.

```
(*Clear[f,c];f[x_]:=c x(1-x)*)
c = 4; valoresini = Map[Nest[f, #, 1000] &, {.1, .2, .3, .4}];
Map[cicloatractivo[f, #, 100] &, valoresini]
```

```
{{}, {}, {}, {}}
```

1.3. Soluciones a actividades de la sección 1.1

Solución actividad 1:

La solución del literal a del ejercicio anterior es:

```
NestList[Sqrt, N[10], 15]
NestList[Sqrt, N[100], 15]
NestList[Sqrt, N[1000], 15]
NestList[Sqrt, N[1 000 000], 15]
```

```
{10., 3.16228, 1.77828, 1.33352, 1.15478, 1.07461, 1.03663, 1.01815,
 1.00904, 1.00451, 1.00225, 1.00112, 1.00056, 1.00028, 1.00014, 1.00007}
```

```
{100., 10., 3.16228, 1.77828, 1.33352, 1.15478, 1.07461, 1.03663,
 1.01815, 1.00904, 1.00451, 1.00225, 1.00112, 1.00056, 1.00028, 1.00014}
```

```
{1000., 31.6228, 5.62341, 2.37137, 1.53993, 1.24094, 1.11397, 1.05545,
 1.02735, 1.01358, 1.00677, 1.00338, 1.00169, 1.00084, 1.00042, 1.00021}
```

```
{1. × 10^6, 1000., 31.6228, 5.62341, 2.37137, 1.53993, 1.24094, 1.11397,
 1.05545, 1.02735, 1.01358, 1.00677, 1.00338, 1.00169, 1.00084, 1.00042}
```

A continuación se presenta la solución del literal b del ejercicio anterior:

```
NestList[√# &, 0.1, 15]
NestList[√# &, 0.01, 15]
NestList[√# &, 0.0001, 15]
NestList[√# &, 0.00001, 15]
```

```
{0.1, 0.316228, 0.562341, 0.749894, 0.865964, 0.930572, 0.964662, 0.982172,
 0.991046, 0.995513, 0.997754, 0.998876, 0.999438, 0.999719, 0.999859, 0.99993}
```

```
{0.01, 0.1, 0.316228, 0.562341, 0.749894, 0.865964, 0.930572, 0.964662,
 0.982172, 0.991046, 0.995513, 0.997754, 0.998876, 0.999438, 0.999719, 0.999859}
```

```
{0.0001, 0.01, 0.1, 0.316228, 0.562341, 0.749894, 0.865964, 0.930572,
 0.964662, 0.982172, 0.991046, 0.995513, 0.997754, 0.998876, 0.999438, 0.999719}
```

```
{0.00001, 0.00316228, 0.0562341, 0.237137, 0.486968, 0.697831, 0.835363, 0.913982,
 0.956024, 0.977765, 0.98882, 0.994394, 0.997193, 0.998596, 0.999298, 0.999649}
```

Basándose en los resultados anteriores se puede inferir que los valores de cada una de las órbitas se aproximan a uno, sin importar si son próximos a cero o muy grandes.

Con el propósito de confirmar la observación hecha previamente, se aumenta el número de elementos de cada órbita:

```
NestList[√# &, N[10], 25]
NestList[√# &, N[100], 25]
NestList[√# &, N[1000], 25]
NestList[√# &, N[1000000], 25]
```

```
{10., 3.16228, 1.77828, 1.33352, 1.15478, 1.07461, 1.03663,
 1.01815, 1.00904, 1.00451, 1.00225, 1.00112, 1.00056, 1.00028,
 1.00014, 1.00007, 1.00004, 1.00002, 1.00001, 1., 1., 1., 1., 1., 1.}
```

```
{100., 10., 3.16228, 1.77828, 1.33352, 1.15478, 1.07461,
 1.03663, 1.01815, 1.00904, 1.00451, 1.00225, 1.00112, 1.00056, 1.00028,
 1.00014, 1.00007, 1.00004, 1.00002, 1.00001, 1., 1., 1., 1., 1., 1.}
```

```
{1000., 31.6228, 5.62341, 2.37137, 1.53993, 1.24094, 1.11397,
 1.05545, 1.02735, 1.01358, 1.00677, 1.00338, 1.00169, 1.00084, 1.00042,
 1.00021, 1.00011, 1.00005, 1.00003, 1.00001, 1.00001, 1., 1., 1., 1., 1.}
```

```
{1. × 10^6, 1000., 31.6228, 5.62341, 2.37137, 1.53993, 1.24094, 1.11397,
 1.05545, 1.02735, 1.01358, 1.00677, 1.00338, 1.00169, 1.00084, 1.00042,
 1.00021, 1.00011, 1.00005, 1.00003, 1.00001, 1.00001, 1., 1., 1., 1.}
```

```
NestList[Sqrt, 0.1, 27]
NestList[Sqrt, 0.01, 27]
NestList[Sqrt, 0.0001, 27]
NestList[Sqrt, 0.00001, 27]
```

```
{0.1, 0.316228, 0.562341, 0.749894, 0.865964, 0.930572, 0.964662, 0.982172, 0.991046,
 0.995513, 0.997754, 0.998876, 0.999438, 0.999719, 0.999859, 0.99993, 0.999965,
 0.999982, 0.999991, 0.999996, 0.999998, 0.999999, 0.999999, 1., 1., 1., 1., 1.}
```

```
{0.01, 0.1, 0.316228, 0.562341, 0.749894, 0.865964, 0.930572, 0.964662, 0.982172,
 0.991046, 0.995513, 0.997754, 0.998876, 0.999438, 0.999719, 0.999859, 0.99993,
 0.999965, 0.999982, 0.999991, 0.999996, 0.999998, 0.999999, 0.999999, 1., 1., 1., 1.}
```

```
{0.0001, 0.01, 0.1, 0.316228, 0.562341, 0.749894, 0.865964, 0.930572, 0.964662, 0.982172,
 0.991046, 0.995513, 0.997754, 0.998876, 0.999438, 0.999719, 0.999859, 0.99993,
 0.999965, 0.999982, 0.999991, 0.999996, 0.999998, 0.999999, 0.999999, 1., 1., 1.}
```

```
{0.00001, 0.00316228, 0.0562341, 0.237137, 0.486968, 0.697831, 0.835363, 0.913982, 0.956024,
 0.977765, 0.98882, 0.994394, 0.997193, 0.998596, 0.999298, 0.999649, 0.999824, 0.999912,
 0.999956, 0.999978, 0.999989, 0.999995, 0.999997, 0.999999, 0.999999, 1., 1., 1.}
```

Para responder la pregunta planteada en el último literal del ejercicio anterior se construye la órbita para algunos valores iniciales cercanos a uno, lo que permite afirmar que las órbitas se aproximan a uno.

```
k = .; k = 20;
NestList[Sqrt, 0.9, k]
NestList[Sqrt, 0.99, k]
NestList[Sqrt, 1.001, k]
NestList[Sqrt, 1.0001, k]
```

```
{0.9, 0.948683, 0.974004, 0.986916, 0.993437, 0.996713, 0.998355, 0.999177, 0.999589, 0.999794,
 0.999897, 0.999949, 0.999974, 0.999987, 0.999994, 0.999997, 0.999998, 0.999999, 1., 1., 1.}
```

```
{0.99, 0.994987, 0.997491, 0.998744, 0.999372, 0.999686, 0.999843, 0.999921, 0.999961,
 0.99998, 0.99999, 0.999995, 0.999998, 0.999999, 0.999999, 1., 1., 1., 1., 1., 1.}
```

```
{1.001, 1.0005, 1.00025, 1.00012, 1.00006, 1.00003,
 1.00002, 1.00001, 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1.}
```

```
{1.0001, 1.00005, 1.00002, 1.00001, 1.00001,
 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1.}
```

Solución actividad 2:

La solución del literal a del ejercicio uno para la función elevar al cuadrado es:

```
k =.; k = 12;
NestList[#^2 &, N[10], k]
NestList[#^2 &, N[100], k]
NestList[#^2 &, N[1000], k]
NestList[#^2 &, N[1 000 000], k]
```

$$\{10.,\ 100.,\ 10\,000.,\ 1.\times 10^{8},\ 1.\times 10^{16},\ 1.\times 10^{32},\ 1.\times 10^{64},\ 1.\times 10^{128},\ 1.\times 10^{256},\ 1.000000000000001\times 10^{512},\ 1.000000000000002\times 10^{1024},\ 1.000000000000004\times 10^{2048},\ 1.00000000000001\times 10^{4096}\}$$

$$\{100.,\ 10\,000.,\ 1.\times 10^{8},\ 1.\times 10^{16},\ 1.\times 10^{32},\ 1.\times 10^{64},\ 1.\times 10^{128},\ 1.\times 10^{256},\ 1.000000000000001\times 10^{512},\ 1.000000000000002\times 10^{1024},\ 1.000000000000004\times 10^{2048},\ 1.00000000000001\times 10^{4096},\ 1.00000000000002\times 10^{8192}\}$$

$$\{1000.,\ 1.\times 10^{6},\ 1.\times 10^{12},\ 1.\times 10^{24},\ 1.\times 10^{48},\ 1.\times 10^{96},\ 1.\times 10^{192},\ 1.000000000000000\times 10^{384},\ 1.000000000000000\times 10^{768},\ 1.000000000000000\times 10^{1536},\ 1.00000000000000\times 10^{3072},\ 1.00000000000000\times 10^{6144},\ 1.00000000000000\times 10^{12\,288}\}$$

$$\{1.\times 10^{6},\ 1.\times 10^{12},\ 1.\times 10^{24},\ 1.\times 10^{48},\ 1.\times 10^{96},\ 1.\times 10^{192},\ 1.000000000000000\times 10^{384},\ 1.000000000000000\times 10^{768},\ 1.000000000000000\times 10^{1536},\ 1.00000000000000\times 10^{3072},\ 1.00000000000000\times 10^{6144},\ 1.00000000000000\times 10^{12\,288},\ 1.0000000000000\times 10^{24\,576}\}$$

A continuación se presenta la solución del literal b del ejercicio uno para la función elevar al cuadrado:

```
NestList[#^2 &, 0.1, k]
NestList[#^2 &, 0.01, k]
NestList[#^2 &, 0.0001, k]
NestList[#^2 &, 0.00001, k]
```

$$\{0.1,\ 0.01,\ 0.0001,\ 1.\times 10^{-8},\ 1.\times 10^{-16},\ 1.\times 10^{-32},\ 1.\times 10^{-64},\ 1.\times 10^{-128},\ 1.\times 10^{-256},\ 1.000000000000055\times 10^{-512},\ 1.000000000000110\times 10^{-1024},\ 1.000000000000220\times 10^{-2048},\ 1.00000000000044\times 10^{-4096}\}$$

$$\{0.01,\ 0.0001,\ 1.\times 10^{-8},\ 1.\times 10^{-16},\ 1.\times 10^{-32},\ 1.\times 10^{-64},\ 1.\times 10^{-128},\ 1.\times 10^{-256},\ 1.000000000000002\times 10^{-512},\ 1.000000000000005\times 10^{-1024},\ 1.000000000000009\times 10^{-2048},\ 1.00000000000002\times 10^{-4096},\ 1.00000000000004\times 10^{-8192}\}$$

$$\{0.0001,\ 1.\times 10^{-8},\ 1.\times 10^{-16},\ 1.\times 10^{-32},\ 1.\times 10^{-64},\ 1.\times 10^{-128},\ 1.\times 10^{-256},\ 1.000000000000002\times 10^{-512},\ 1.000000000000005\times 10^{-1024},\ 1.000000000000009\times 10^{-2048},\ 1.00000000000002\times 10^{-4096},\ 1.00000000000004\times 10^{-8192},\ 1.00000000000007\times 10^{-16\,384}\}$$

$$\{0.00001,\ 1.\times 10^{-10},\ 1.\times 10^{-20},\ 1.\times 10^{-40},\ 1.\times 10^{-80},\ 1.\times 10^{-160},\ 1.000000000000006\times 10^{-320},\ 1.000000000000012\times 10^{-640},\ 1.000000000000024\times 10^{-1280},\ 1.00000000000005\times 10^{-2560},\ 1.00000000000010\times 10^{-5120},\ 1.00000000000019\times 10^{-10\,240},\ 1.0000000000004\times 10^{-20\,480}\}$$

Basándose en los resultados anteriores se puede inferir que los valores de cada una de las órbitas son cada vez más grandes si los valores iniciales son grandes y se aproximan a cero, si los valores iniciales son próximos al mismo.

Con el propósito de confirmar la observación hecha previamente, se aumenta el número de elementos de cada órbita:

```
k =.; k = 20;
NestList[#^2 2 &, 10., k]
NestList[#^2 &, 100., k]
NestList[#^2 &, 1000., k]
NestList[#^2 &, 1 000 000., k]
```

$$\{10.,\ 200.,\ 80\,000.,\ 1.28\times10^{10},\ 3.2768\times10^{20},$$
$$2.14748\times10^{41},\ 9.22337\times10^{82},\ 1.70141\times10^{166},\ 5.78960446186581\times10^{332},$$
$$6.70390396497131\times10^{665},\ 8.98846567431160\times10^{1331},\ 1.61585030356556\times10^{2664},$$
$$5.2219444070658\times10^{5328},\ 5.4537406780972\times10^{10\,657},\ 5.9486574767864\times10^{21\,315},$$
$$7.077305155225\times10^{42\,631},\ 1.0017649652036\times10^{85\,264},\ 2.007066091019\times10^{170\,528},$$
$$8.05662858743\times10^{341\,056},\ 1.298185283917\times10^{682\,114},\ 3.37057006276\times10^{1\,364\,228}\}$$

$$\{100.,\ 10\,000.,\ 1.\times10^{8},\ 1.\times10^{16},\ 1.\times10^{32},\ 1.\times10^{64},\ 1.\times10^{128},\ 1.\times10^{256},\ 1.000000000000001\times10^{512},$$
$$1.000000000000002\times10^{1024},\ 1.000000000000004\times10^{2048},\ 1.00000000000001\times10^{4096},$$
$$1.00000000000002\times10^{8192},\ 1.00000000000003\times10^{16\,384},\ 1.0000000000001\times10^{32\,768},$$
$$1.0000000000001\times10^{65\,536},\ 1.0000000000003\times10^{131\,072},\ 1.000000000001\times10^{262\,144},$$
$$1.000000000001\times10^{524\,288},\ 1.000000000002\times10^{1\,048\,576},\ 1.000000000004\times10^{2\,097\,152}\}$$

$$\{1000.,\ 1.\times10^{6},\ 1.\times10^{12},\ 1.\times10^{24},\ 1.\times10^{48},\ 1.\times10^{96},$$
$$1.\times10^{192},\ 1.000000000000000\times10^{384},\ 1.000000000000000\times10^{768},$$
$$1.000000000000000\times10^{1536},\ 1.00000000000000\times10^{3072},\ 1.00000000000000\times10^{6144},$$
$$1.00000000000000\times10^{12\,288},\ 1.0000000000000\times10^{24\,576},\ 1.0000000000000\times10^{49\,152},$$
$$1.0000000000000\times10^{98\,304},\ 1.000000000000\times10^{196\,608},\ 1.000000000000\times10^{393\,216},$$
$$1.000000000000\times10^{786\,432},\ 1.000000000000\times10^{1\,572\,864},\ 1.00000000000\times10^{3\,145\,728}\}$$

$$\{1.\times10^{6},\ 1.\times10^{12},\ 1.\times10^{24},\ 1.\times10^{48},\ 1.\times10^{96},\ 1.\times10^{192},$$
$$1.000000000000000\times10^{384},\ 1.000000000000000\times10^{768},\ 1.000000000000000\times10^{1536},$$
$$1.00000000000000\times10^{3072},\ 1.00000000000000\times10^{6144},\ 1.00000000000000\times10^{12\,288},$$
$$1.0000000000000\times10^{24\,576},\ 1.0000000000000\times10^{49\,152},\ 1.0000000000000\times10^{98\,304},$$
$$1.000000000000\times10^{196\,608},\ 1.000000000000\times10^{393\,216},\ 1.000000000000\times10^{786\,432},$$
$$1.000000000000\times10^{1\,572\,864},\ 1.00000000000\times10^{3\,145\,728},\ 1.00000000000\times10^{6\,291\,456}\}$$

```
NestList[#^2 &, 0.1, k]
NestList[#^2 &, 0.01, k]
NestList[#^2 &, 0.0001, k]
NestList[#^2 &, 0.00001, k]
```

$$\{0.1,\ 0.01,\ 0.0001,\ 1.\times10^{-8},\ 1.\times10^{-16},\ 1.\times10^{-32},\ 1.\times10^{-64},\ 1.\times10^{-128},\ 1.\times10^{-256},$$
$$1.000000000000055\times10^{-512},\ 1.000000000000110\times10^{-1024},\ 1.000000000000220\times10^{-2048},$$
$$1.00000000000044\times10^{-4096},\ 1.00000000000088\times10^{-8192},\ 1.00000000000176\times10^{-16\,384},$$
$$1.0000000000035\times10^{-32\,768},\ 1.0000000000070\times10^{-65\,536},\ 1.0000000000141\times10^{-131\,072},$$
$$1.000000000028\times10^{-262\,144},\ 1.000000000056\times10^{-524\,288},\ 1.000000000113\times10^{-1\,048\,576}\}$$

$$\{0.01, 0.0001, 1.\times 10^{-8}, 1.\times 10^{-16}, 1.\times 10^{-32}, 1.\times 10^{-64}, 1.\times 10^{-128},$$
$$1.\times 10^{-256}, 1.000000000000002\times 10^{-512}, 1.000000000000005\times 10^{-1024},$$
$$1.000000000000009\times 10^{-2048}, 1.00000000000002\times 10^{-4096},$$
$$1.00000000000004\times 10^{-8192}, 1.00000000000007\times 10^{-16384}, 1.0000000000001\times 10^{-32768},$$
$$1.0000000000003\times 10^{-65536}, 1.0000000000006\times 10^{-131072}, 1.000000000001\times 10^{-262144},$$
$$1.000000000002\times 10^{-524288}, 1.000000000005\times 10^{-1048576}, 1.000000000010\times 10^{-2097152}\}$$

$$\{0.0001, 1.\times 10^{-8}, 1.\times 10^{-16}, 1.\times 10^{-32}, 1.\times 10^{-64}, 1.\times 10^{-128},$$
$$1.\times 10^{-256}, 1.000000000000002\times 10^{-512}, 1.000000000000005\times 10^{-1024},$$
$$1.000000000000009\times 10^{-2048}, 1.00000000000002\times 10^{-4096}, 1.00000000000004\times 10^{-8192},$$
$$1.00000000000007\times 10^{-16384}, 1.0000000000001\times 10^{-32768}, 1.0000000000003\times 10^{-65536},$$
$$1.0000000000006\times 10^{-131072}, 1.000000000001\times 10^{-262144}, 1.000000000002\times 10^{-524288},$$
$$1.000000000005\times 10^{-1048576}, 1.000000000010\times 10^{-2097152}, 1.00000000002\times 10^{-4194304}\}$$

$$\{0.00001, 1.\times 10^{-10}, 1.\times 10^{-20}, 1.\times 10^{-40}, 1.\times 10^{-80}, 1.\times 10^{-160},$$
$$1.000000000000006\times 10^{-320}, 1.000000000000012\times 10^{-640}, 1.000000000000024\times 10^{-1280},$$
$$1.00000000000005\times 10^{-2560}, 1.00000000000010\times 10^{-5120}, 1.00000000000019\times 10^{-10240},$$
$$1.0000000000004\times 10^{-20480}, 1.0000000000008\times 10^{-40960}, 1.0000000000015\times 10^{-81920},$$
$$1.000000000003\times 10^{-163840}, 1.000000000006\times 10^{-327680}, 1.000000000012\times 10^{-655360},$$
$$1.000000000024\times 10^{-1310720}, 1.00000000005\times 10^{-2621440}, 1.00000000010\times 10^{-5242880}\}$$

Para responder la pregunta planteada en el último literal del ejercicio uno para la función elevar al cuadrado se construye la órbita para algunos valores iniciales cercanos a uno, lo que permite afirmar que las órbitas se alejan de uno si los valores iniciales están a la derecha de éste y se aproximan a cero, si los valores iniciales son menores que uno y mayores que cero.

```
k =.; k = 23;
NestList[#^2 &, 0.9, k]
NestList[#^2 &, 0.99, k]
NestList[#^2 &, 1.001, k]
NestList[#^2 &, 1.0001, k]
```

$$\{0.9, 0.81, 0.6561, 0.430467, 0.185302, 0.0343368, 0.00117902,$$
$$1.39008\times 10^{-6}, 1.93233\times 10^{-12}, 3.73392\times 10^{-24}, 1.39421\times 10^{-47}, 1.94383\times 10^{-94},$$
$$3.77849\times 10^{-188}, 1.427701207893856\times 10^{-375}, 2.038330739021576\times 10^{-750},$$
$$4.15479220164025\times 10^{-1500}, 1.72622982388106\times 10^{-2999}, 2.9798694048564\times 10^{-5998},$$
$$8.8796216699994\times 10^{-11996}, 7.884768100232\times 10^{-23991}, 6.216956799444\times 10^{-47981},$$
$$3.865055184616\times 10^{-95961}, 1.493865158012\times 10^{-191921}, 2.231633110323\times 10^{-383842}\}$$

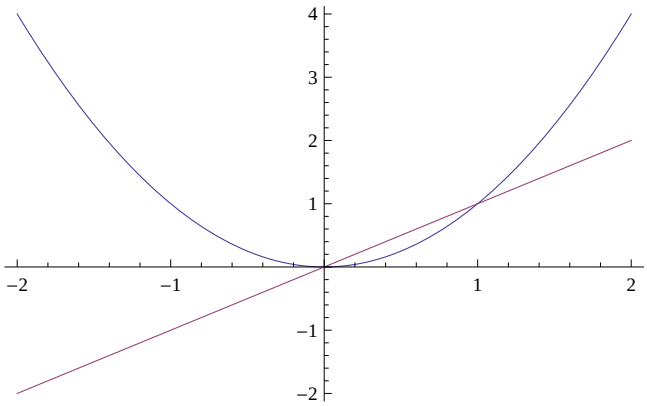
$$\{0.99, 0.9801, 0.960596, 0.922745, 0.851458, 0.72498, 0.525596, 0.276252,$$
$$0.076315, 0.00582398, 0.0000339187, 1.15048\times 10^{-9}, 1.3236\times 10^{-18}, 1.75192\times 10^{-36},$$
$$3.06922\times 10^{-72}, 9.42012\times 10^{-144}, 8.87387\times 10^{-287}, 7.87455988432386\times 10^{-573},$$
$$6.20086933718026\times 10^{-1145}, 3.84507805367823\times 10^{-2289}, 1.47846252388780\times 10^{-4577},$$
$$2.18585143454068\times 10^{-9154}, 4.7779464938836\times 10^{-18308}, 2.2828772698414\times 10^{-36615}\}$$

$$\{1.001, 1.002, 1.00401, 1.00803, 1.01612, 1.0325, 1.06606, 1.13648,$$
$$1.29159, 1.6682, 2.78289, 7.74445, 59.9765, 3597.18, 1.29397\times 10^{7}, 1.67437\times 10^{14},$$
$$2.80351\times 10^{28}, 7.85965\times 10^{56}, 6.17741\times 10^{113}, 3.81604\times 10^{227}, 1.456219874333072\times 10^{455},$$
$$2.120576322402629\times 10^{910}, 4.49684393913466\times 10^{1820}, 2.02216054129321\times 10^{3641}\}$$

$$\{1.0001, 1.0002, 1.0004, 1.0008, 1.0016, 1.0032, 1.00642, 1.01288, 1.02593, 1.05253,$$
$$1.10782, 1.22727, 1.50618, 2.26859, 5.14651, 26.4865, 701.536, 492153., 2.42214\times 10^{11},$$
$$5.86678\times 10^{22}, 3.44192\times 10^{45}, 1.18468\times 10^{91}, 1.40346\times 10^{182}, 1.969709176859194\times 10^{364}\}$$

La solución del literal b del ejercicio anterior es:

```
Plot[{x^2, x}, {x, -2, 2}]
```

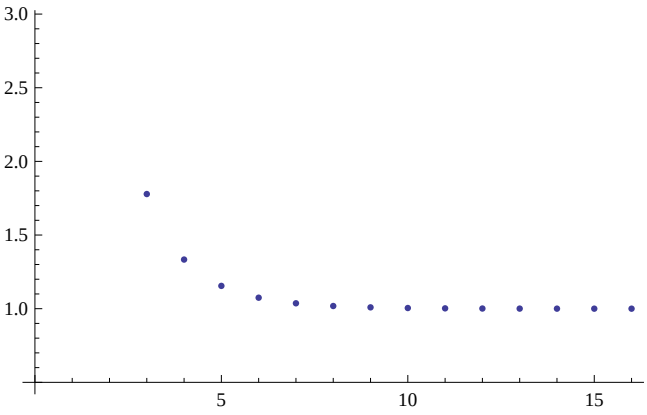


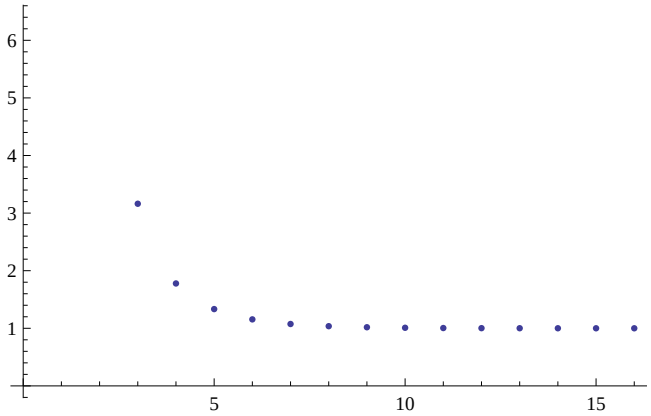
De la simetría de la gráfica de la función elevar al cuadrado respecto del origen se puede afirmar que las observaciones antes hechas, son válidas también para los simétricos de cada uno de los valores iniciales estudiados antes.

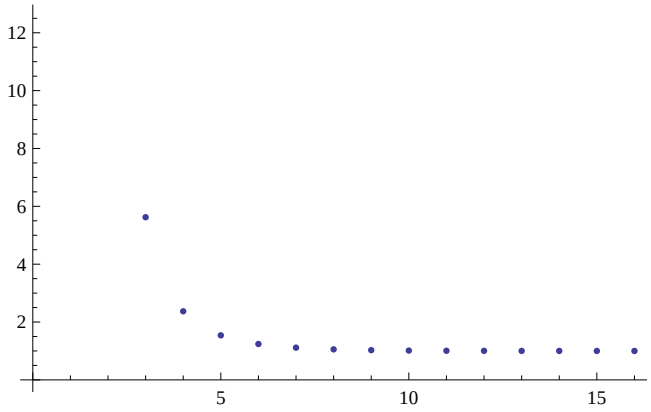
Solución actividad 3:

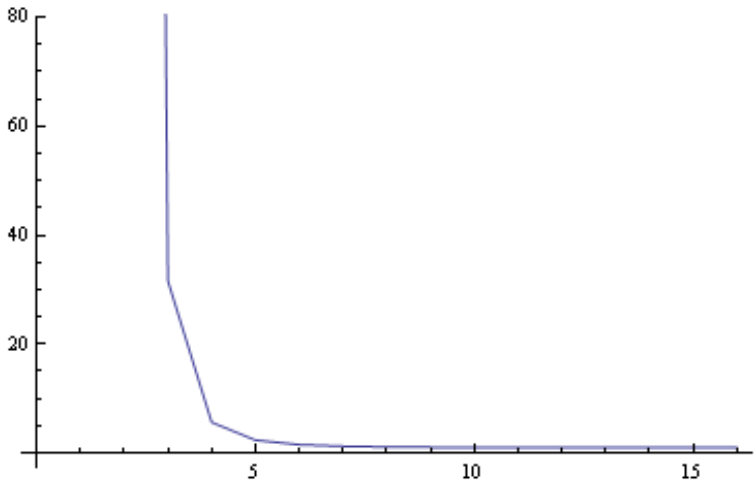
Para graficar las órbitas de los valores iniciales presentados en el primer ejercicio, basta anteponer la función *ListPlot* a los códigos elaborados para dicha actividad:

```
ListPlot[NestList[Sqrt, N[10], 15], AxesOrigin → {0, .5}]
ListPlot[NestList[Sqrt, N[100], 15]]
ListPlot[NestList[Sqrt, N[1000], 15], PlotRange → Automatic]
ListPlot[NestList[Sqrt, N[1 000 000], 15], Joined → True]
```



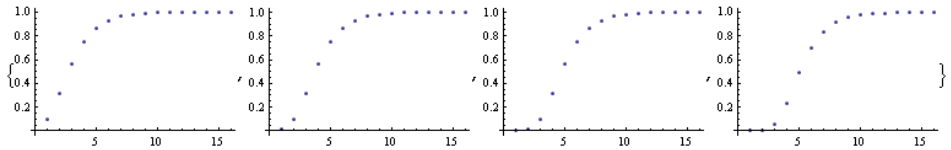






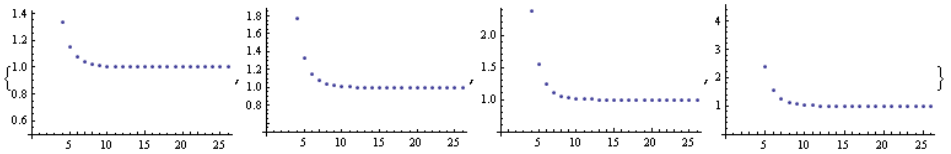
Una manera sencilla de graficar las órbitas requiere emplear la función *Map*:

```
ListPlot /@ {NestList[√# &, 0.1, 15], NestList[√# &, 0.01, 15], NestList[√# &, 0.0001, 15],
    NestList[√# &, 0.00001, 15]}
```



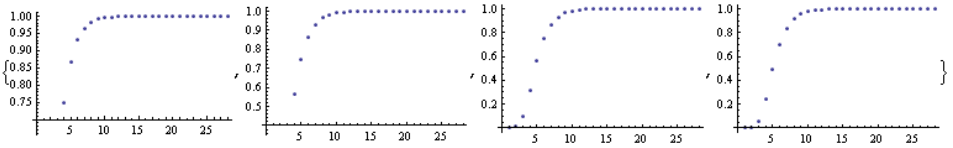
Una forma de variar el valor inicial es utilizar la función *Table*:

```
Flatten[{Table[ListPlot[NestList[√# &, 10^n, 25], AxesOrigin → {0, 0.5}], {n, 1, 3}],
    ListPlot[NestList[√# &, 1000000, 25], AxesOrigin → {0, 0}]}]
```



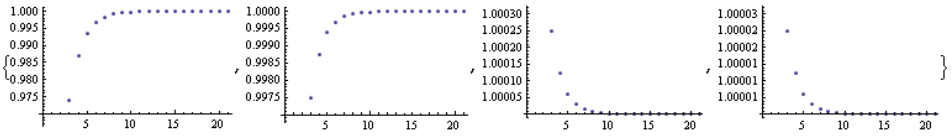
Si se desea omitir uno de los elementos de una lista se utiliza la función *Part*:

```
Map[ListPlot, Table[NestList[Sqrt, 10^-n, 27], {n, 1, 5}][[{1, 2, 4, 5}]]]
```



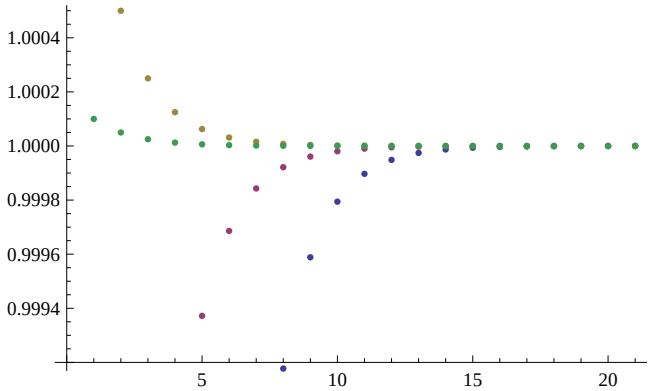
Para construir una lista de un conjunto se puede utilizar la función *List*:

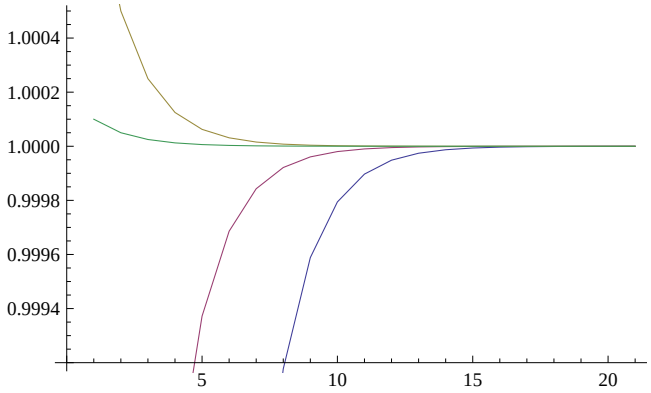
```
k = .; k = 20;
ListPlot /@ List[NestList[Sqrt, 0.9, k], NestList[Sqrt, 0.99, k], NestList[Sqrt, 1.001, k],
    NestList[Sqrt, 1.0001, k]]
```



Si se invierte el orden de aplicación de la función *Map* y *ListPlot* se grafican todas las órbitas en un mismo plano:

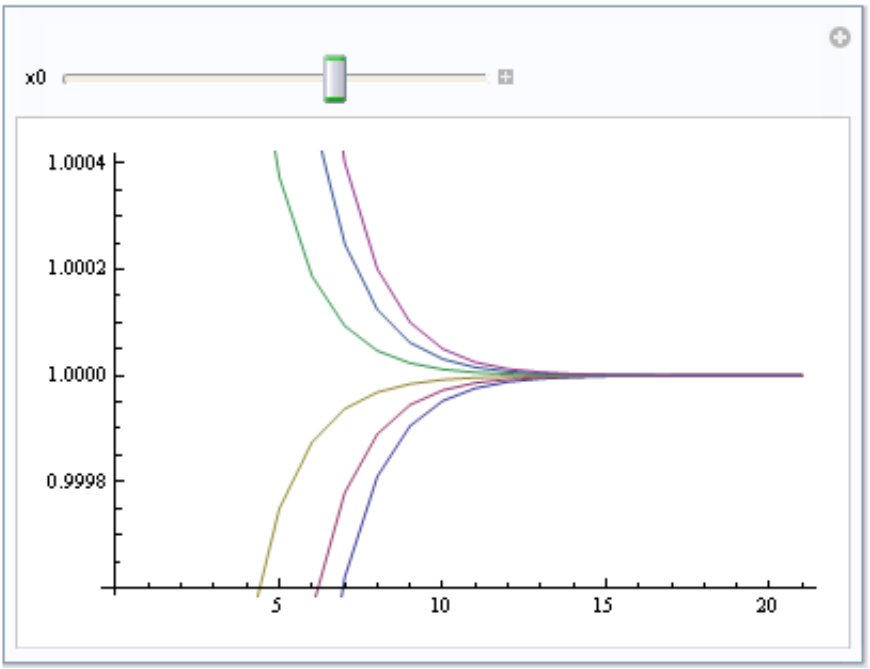
```
ListPlot[NestList[Sqrt, #, k] & /@ {.9, 0.99, 1.001, 1.0001}, Joined → False]
ListPlot[NestList[Sqrt, #, k] & /@ {.9, 0.99, 1.001, 1.0001}, Joined → True]
```





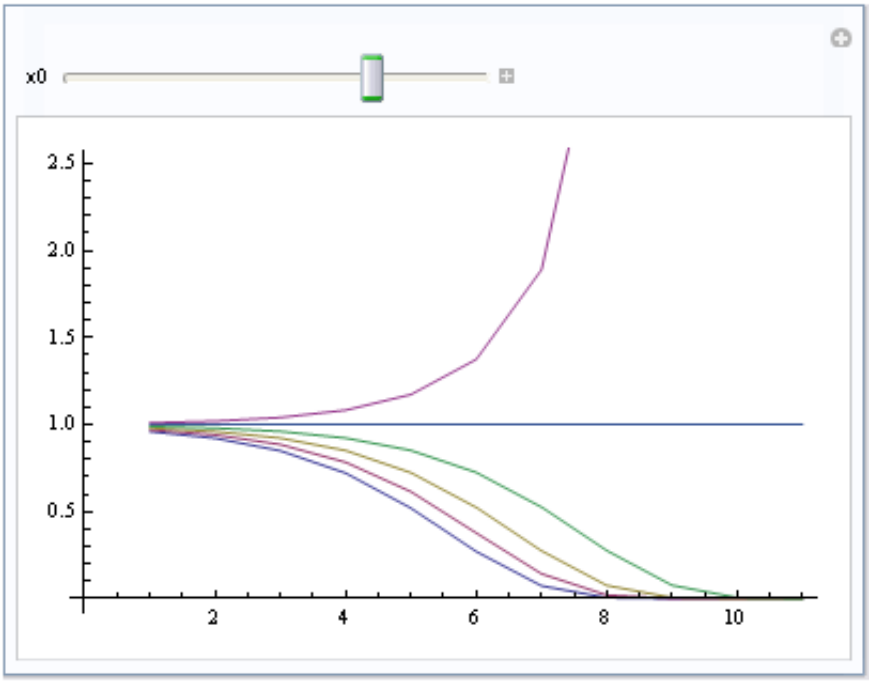
Si se agrega al código anterior la función *Manipulate* se puede variar los valores iniciales:

```
Manipulate[

 ListPlot[Map[NestList[Sqrt, #, 20] &, Table[x0 + Δx, {Δx, 0, .05, .01}]], Joined → True],
       {x0, 0, 1.5}, SaveDefinitions → True]
```



Si se sustituye la función raíz cuadrada por elevar al cuadrado se tiene:

```
Manipulate[

 ListPlot[Map[NestList[#^2 &, #, 10] &, Table[x0 + Δx, {Δx, 0, .05, .01}]], Joined → True],
       {x0, -2, 2, Appearance → "Open"}, SaveDefinitions → True]
```



Solución actividad 4:

```
Solve[√x == x, x]
```

```
{{x → 0}, {x → 1}}
```

```
Solve[x^2 == x, x]
```

```
{{x → 0}, {x → 1}}
```

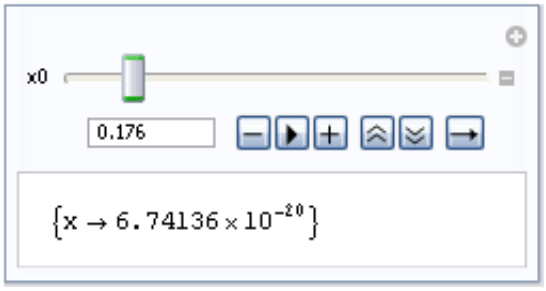
Solución actividad 5:

Para la función raíz cuadrada:

```
FindRoot[√x == x, {x, 0.6}]
```

```
{x → 1.}
```

```
Manipulate[FindRoot[√x == x, {x, x0}], {x0, 0, 1.2, Appearance → "Open"}]
```

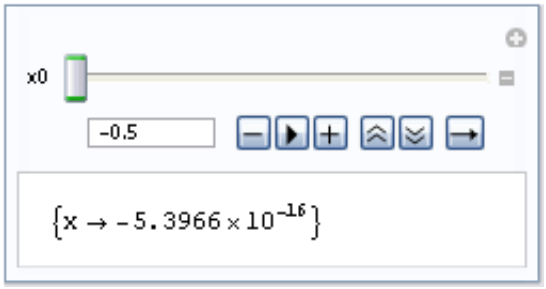


Para la función elevar al cuadrado:

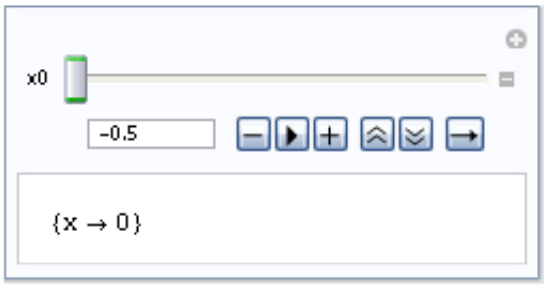
```
FindRoot[x^2 - x, {x, 0.6}]
```

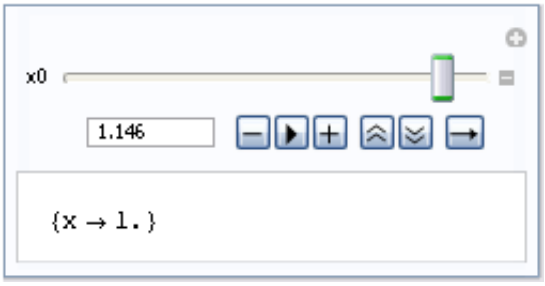
```
{x → 1.}
```

```
Manipulate[FindRoot[x^2 - x, {x, x0}], {x0, -.5, 1.3, Appearance → "Open"}]
```



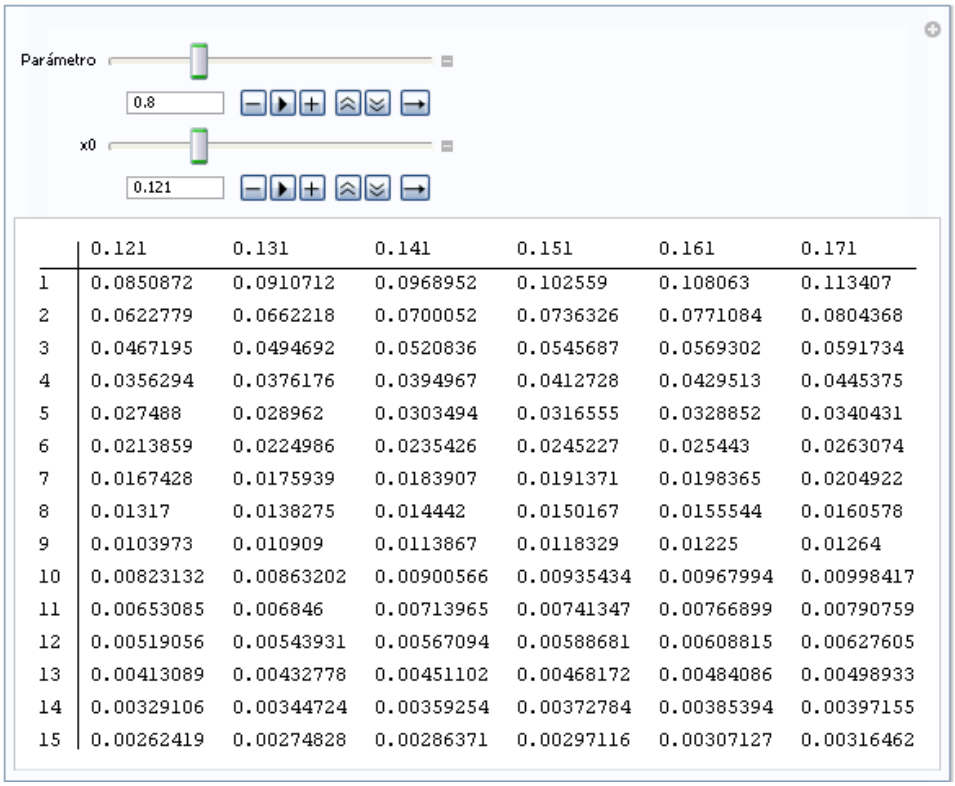

```
Manipulate[Chop[FindRoot[x^2 - x, {x, x0}]], {x0, -.5, 1.3, Appearance → "Open"}]
```



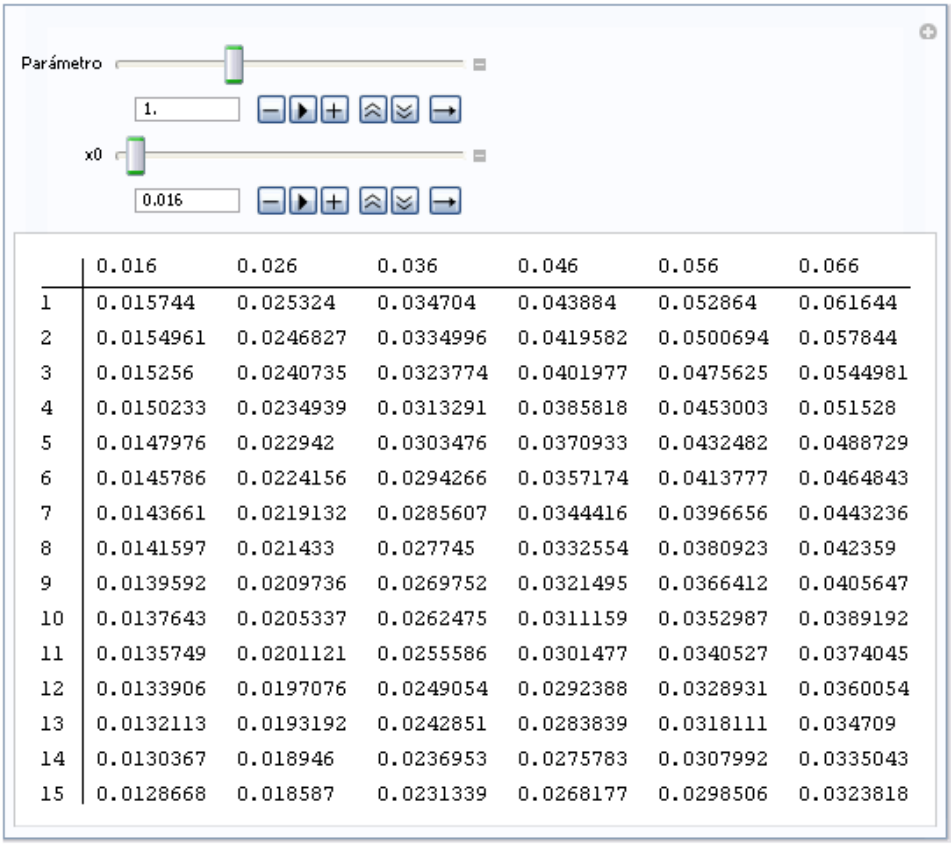

x0
1.146
{x → 1.}

1.4. Soluciones a actividades de la sección 1.2

Solución actividad 1:

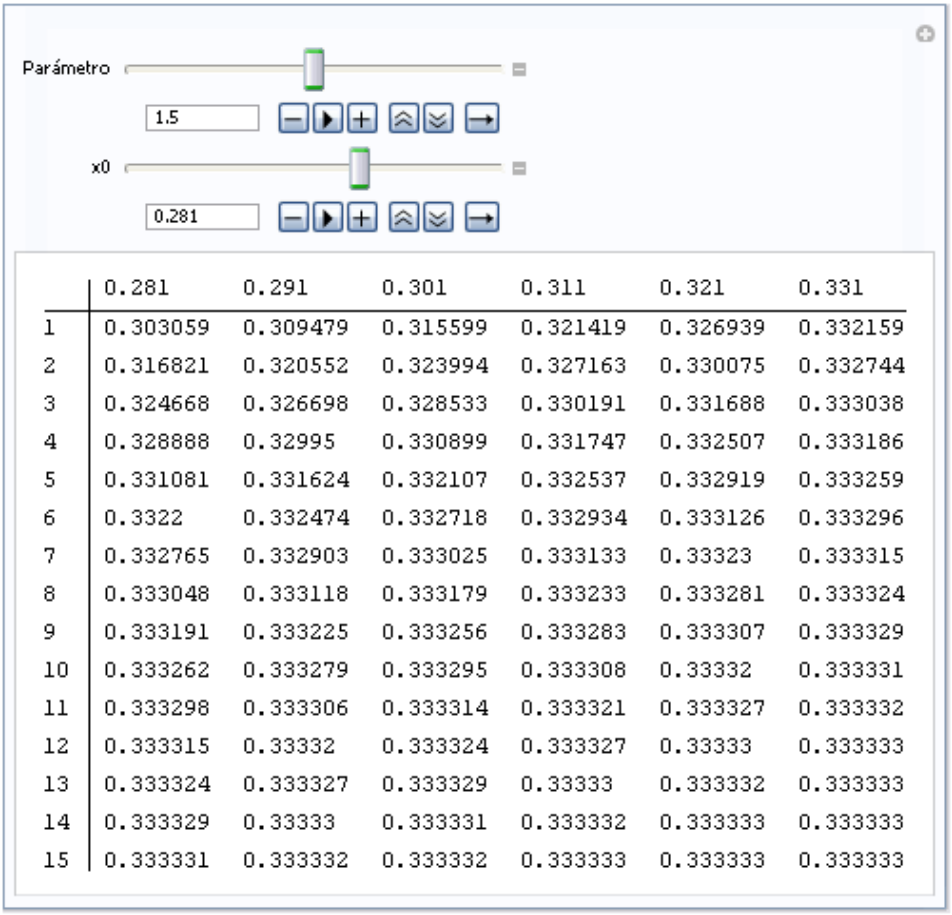


	0.121	0.131	0.141	0.151	0.161	0.171
1	0.0850872	0.0910712	0.0968952	0.102559	0.108063	0.113407
2	0.0622779	0.0662218	0.0700052	0.0736326	0.0771084	0.0804368
3	0.0467195	0.0494692	0.0520836	0.0545687	0.0569302	0.0591734
4	0.0356294	0.0376176	0.0394967	0.0412728	0.0429513	0.0445375
5	0.027488	0.028962	0.0303494	0.0316555	0.0328852	0.0340431
6	0.0213859	0.0224986	0.0235426	0.0245227	0.025443	0.0263074
7	0.0167428	0.0175939	0.0183907	0.0191371	0.0198365	0.0204922
8	0.01317	0.0138275	0.014442	0.0150167	0.0155544	0.0160578
9	0.0103973	0.010909	0.0113867	0.0118329	0.01225	0.01264
10	0.00823132	0.00863202	0.00900566	0.00935434	0.00967994	0.00998417
11	0.00653085	0.006846	0.00713965	0.00741347	0.00766899	0.00790759
12	0.00519056	0.00543931	0.00567094	0.00588681	0.00608815	0.00627605
13	0.00413089	0.00432778	0.00451102	0.00468172	0.00484086	0.00498933
14	0.00329106	0.00344724	0.00359254	0.00372784	0.00385394	0.00397155
15	0.00262419	0.00274828	0.00286371	0.00297116	0.00307127	0.00316462

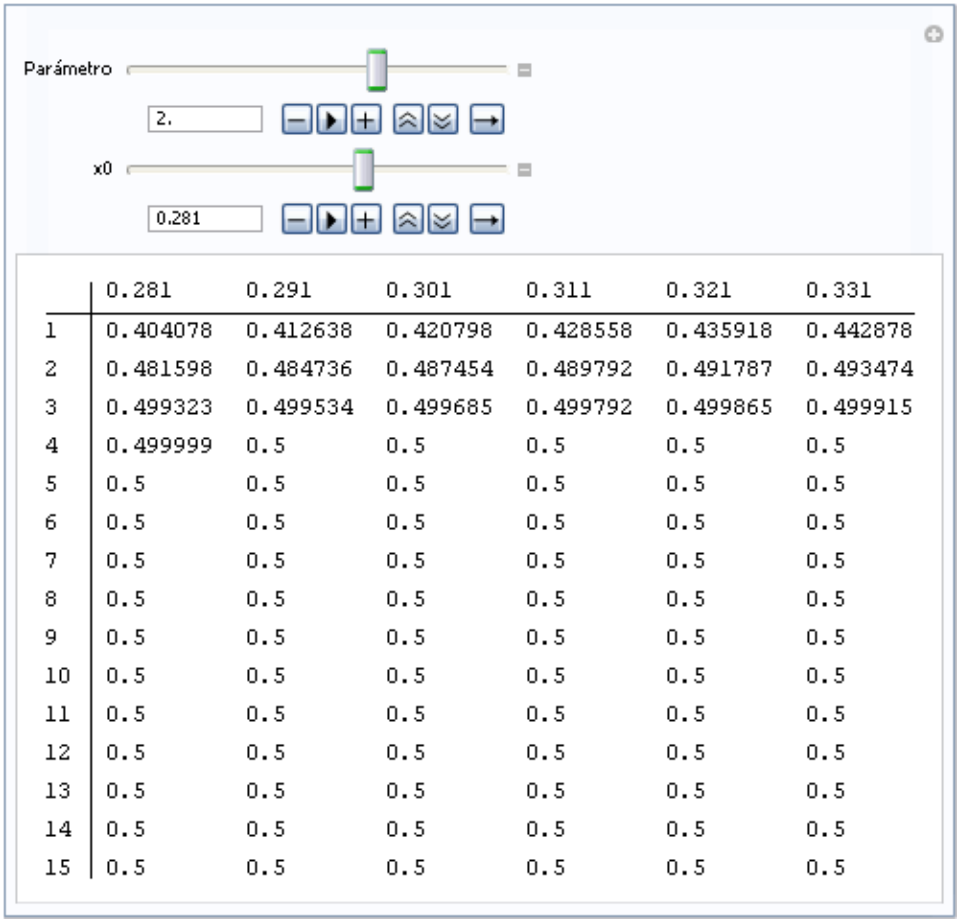


	0.016	0.026	0.036	0.046	0.056	0.066
1	0.015744	0.025324	0.034704	0.043884	0.052864	0.061644
2	0.0154961	0.0246827	0.0334996	0.0419582	0.0500694	0.057844
3	0.015256	0.0240735	0.0323774	0.0401977	0.0475625	0.0544981
4	0.0150233	0.0234939	0.0313291	0.0385818	0.0453003	0.051528
5	0.0147976	0.022942	0.0303476	0.0370933	0.0432482	0.0488729
6	0.0145786	0.0224156	0.0294266	0.0357174	0.0413777	0.0464843
7	0.0143661	0.0219132	0.0285607	0.0344416	0.0396656	0.0443236
8	0.0141597	0.021433	0.027745	0.0332554	0.0380923	0.042359
9	0.0139592	0.0209736	0.0269752	0.0321495	0.0366412	0.0405647
10	0.0137643	0.0205337	0.0262475	0.0311159	0.0352987	0.0389192
11	0.0135749	0.0201121	0.0255586	0.0301477	0.0340527	0.0374045
12	0.0133906	0.0197076	0.0249054	0.0292388	0.0328931	0.0360054
13	0.0132113	0.0193192	0.0242851	0.0283839	0.0318111	0.034709
14	0.0130367	0.018946	0.0236953	0.0275783	0.0307992	0.0335043
15	0.0128668	0.018587	0.0231339	0.0268177	0.0298506	0.0323818

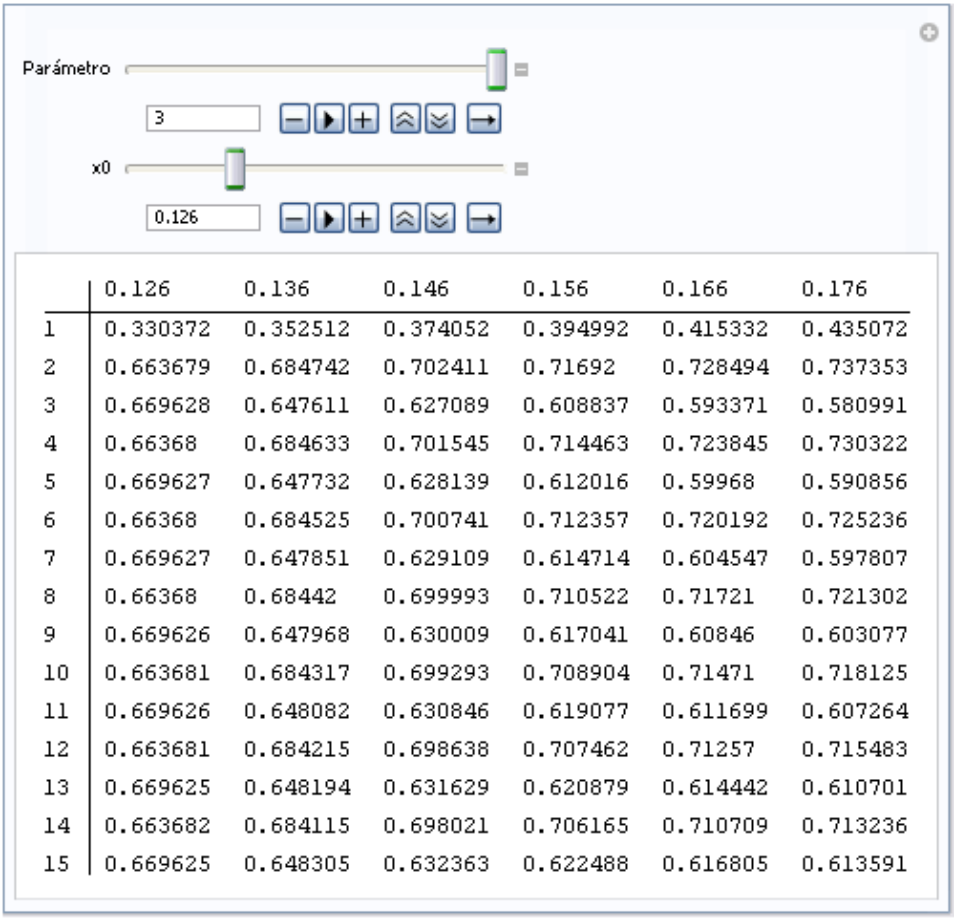
	0.233	0.243	0.253	0.263	0.273	0.283
1	0.214453	0.220741	0.226789	0.232597	0.238165	0.243493
2	0.202156	0.206417	0.210427	0.214195	0.217731	0.221045
3	0.193546	0.196571	0.199377	0.201979	0.204389	0.206621
4	0.187303	0.189517	0.191551	0.19342	0.195137	0.196715
5	0.182665	0.18432	0.185831	0.18721	0.18847	0.189622
6	0.179158	0.180416	0.181557	0.182595	0.183539	0.184398
7	0.176473	0.177439	0.178313	0.179105	0.179823	0.180475
8	0.174396	0.175145	0.175821	0.176432	0.176984	0.177484
9	0.172779	0.173363	0.17389	0.174364	0.174793	0.17518
10	0.171511	0.17197	0.172382	0.172754	0.173088	0.173391
11	0.170514	0.170876	0.1712	0.171492	0.171755	0.171992
12	0.169727	0.170013	0.170269	0.170499	0.170706	0.170893
13	0.169104	0.16933	0.169533	0.169715	0.169878	0.170026
14	0.168609	0.168789	0.16895	0.169094	0.169224	0.169341
15	0.168216	0.168359	0.168487	0.168601	0.168704	0.168797



	0.281	0.291	0.301	0.311	0.321	0.331
1	0.303059	0.309479	0.315599	0.321419	0.326939	0.332159
2	0.316821	0.320552	0.323994	0.327163	0.330075	0.332744
3	0.324668	0.326698	0.328533	0.330191	0.331688	0.333038
4	0.328888	0.32995	0.330899	0.331747	0.332507	0.333186
5	0.331081	0.331624	0.332107	0.332537	0.332919	0.333259
6	0.3322	0.332474	0.332718	0.332934	0.333126	0.333296
7	0.332765	0.332903	0.333025	0.333133	0.33323	0.333315
8	0.333048	0.333118	0.333179	0.333233	0.333281	0.333324
9	0.333191	0.333225	0.333256	0.333283	0.333307	0.333329
10	0.333262	0.333279	0.333295	0.333308	0.33332	0.333331
11	0.333298	0.333306	0.333314	0.333321	0.333327	0.333332
12	0.333315	0.33332	0.333324	0.333327	0.33333	0.333333
13	0.333324	0.333327	0.333329	0.33333	0.333332	0.333333
14	0.333329	0.33333	0.333331	0.333332	0.333333	0.333333
15	0.333331	0.333332	0.333332	0.333333	0.333333	0.333333

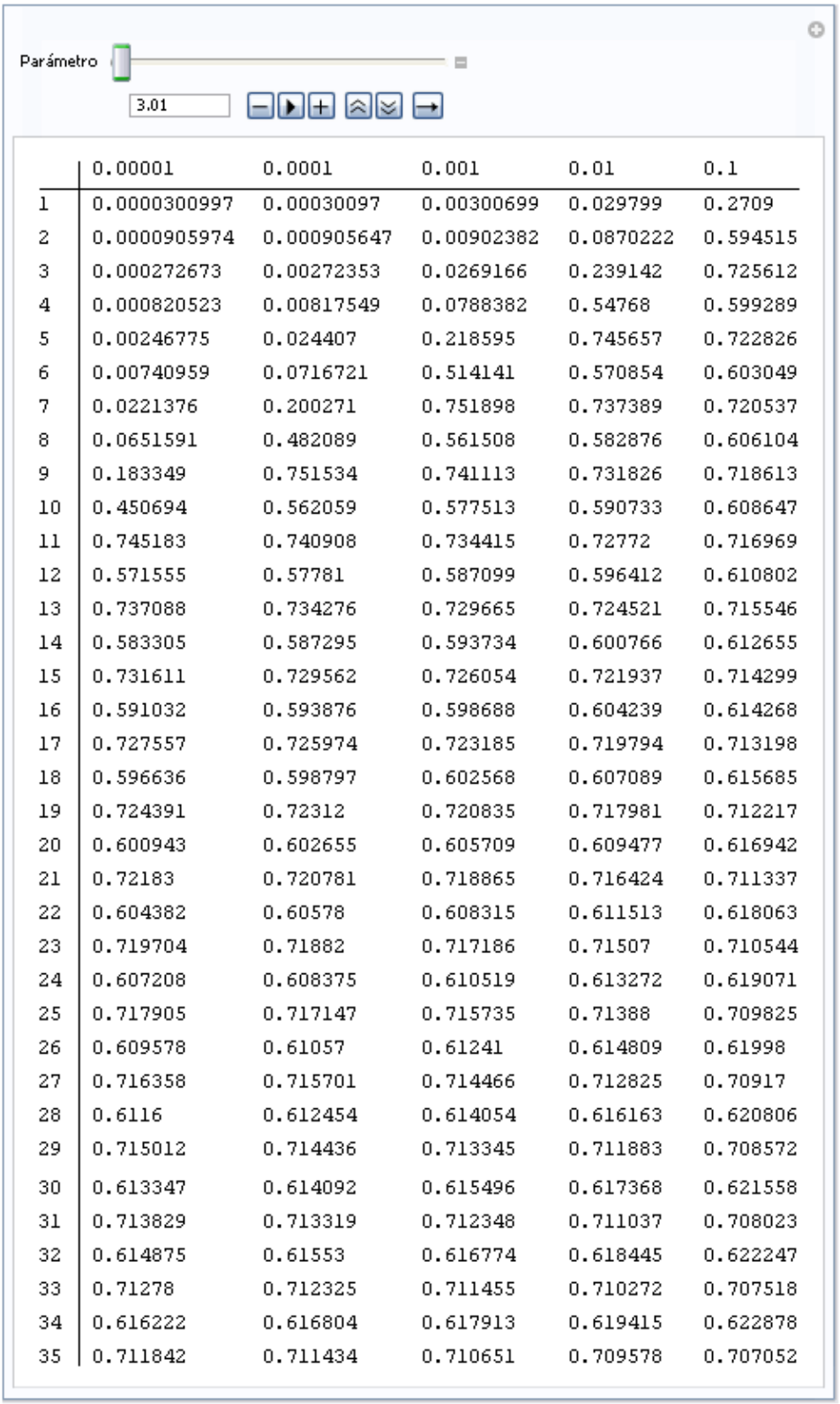


	0.281	0.291	0.301	0.311	0.321	0.331
1	0.404078	0.412638	0.420798	0.428558	0.435918	0.442878
2	0.481598	0.484736	0.487454	0.489792	0.491787	0.493474
3	0.499323	0.499534	0.499685	0.499792	0.499865	0.499915
4	0.499999	0.5	0.5	0.5	0.5	0.5
5	0.5	0.5	0.5	0.5	0.5	0.5
6	0.5	0.5	0.5	0.5	0.5	0.5
7	0.5	0.5	0.5	0.5	0.5	0.5
8	0.5	0.5	0.5	0.5	0.5	0.5
9	0.5	0.5	0.5	0.5	0.5	0.5
10	0.5	0.5	0.5	0.5	0.5	0.5
11	0.5	0.5	0.5	0.5	0.5	0.5
12	0.5	0.5	0.5	0.5	0.5	0.5
13	0.5	0.5	0.5	0.5	0.5	0.5
14	0.5	0.5	0.5	0.5	0.5	0.5
15	0.5	0.5	0.5	0.5	0.5	0.5

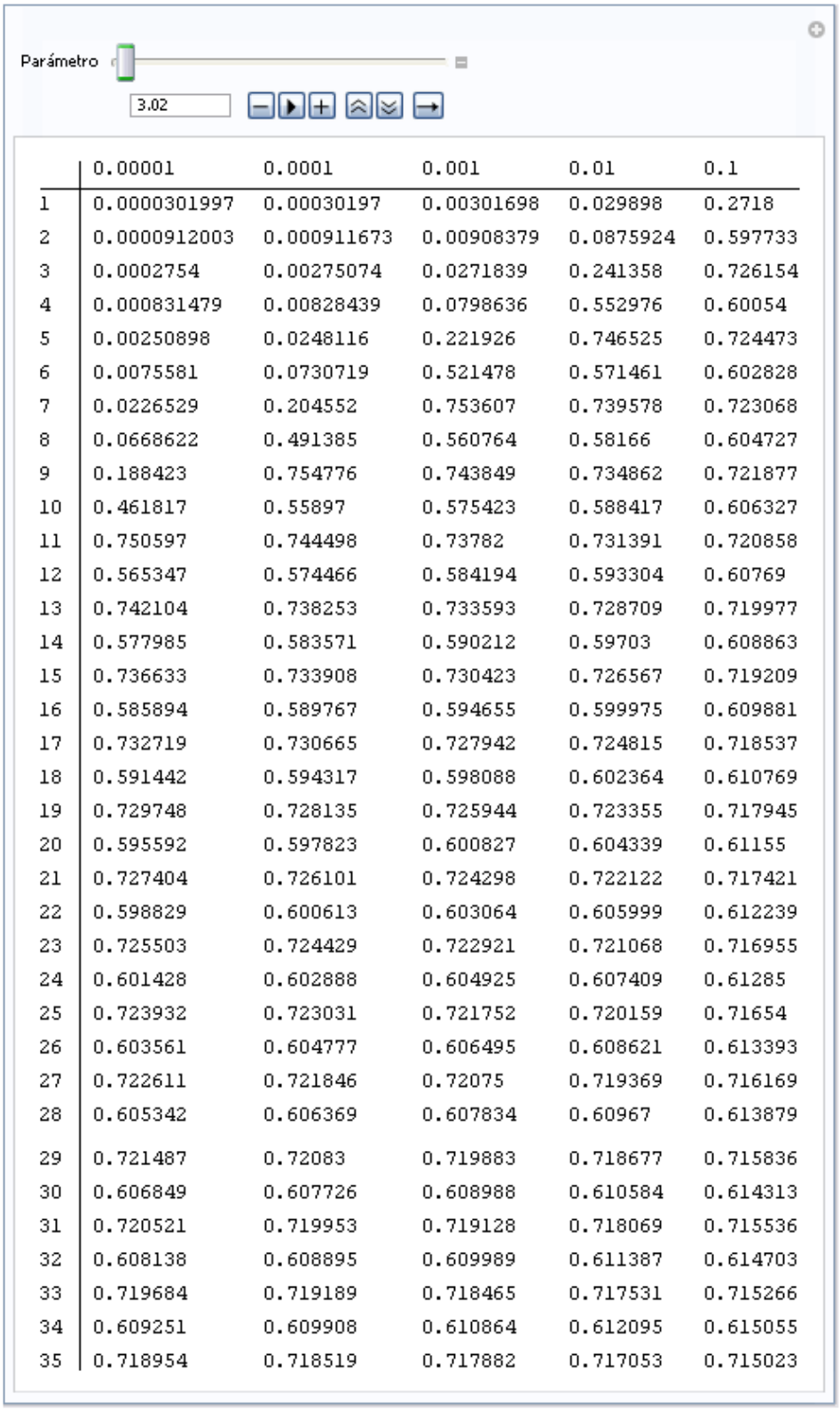


	0.126	0.136	0.146	0.156	0.166	0.176
1	0.330372	0.352512	0.374052	0.394992	0.415332	0.435072
2	0.663679	0.684742	0.702411	0.71692	0.728494	0.737353
3	0.669628	0.647611	0.627089	0.608837	0.593371	0.580991
4	0.66368	0.684633	0.701545	0.714463	0.723845	0.730322
5	0.669627	0.647732	0.628139	0.612016	0.59968	0.590856
6	0.66368	0.684525	0.700741	0.712357	0.720192	0.725236
7	0.669627	0.647851	0.629109	0.614714	0.604547	0.597807
8	0.66368	0.68442	0.699993	0.710522	0.71721	0.721302
9	0.669626	0.647968	0.630009	0.617041	0.60846	0.603077
10	0.663681	0.684317	0.699293	0.708904	0.71471	0.718125
11	0.669626	0.648082	0.630846	0.619077	0.611699	0.607264
12	0.663681	0.684215	0.698638	0.707462	0.71257	0.715483
13	0.669625	0.648194	0.631629	0.620879	0.614442	0.610701
14	0.663682	0.684115	0.698021	0.706165	0.710709	0.713236
15	0.669625	0.648305	0.632363	0.622488	0.616805	0.613591

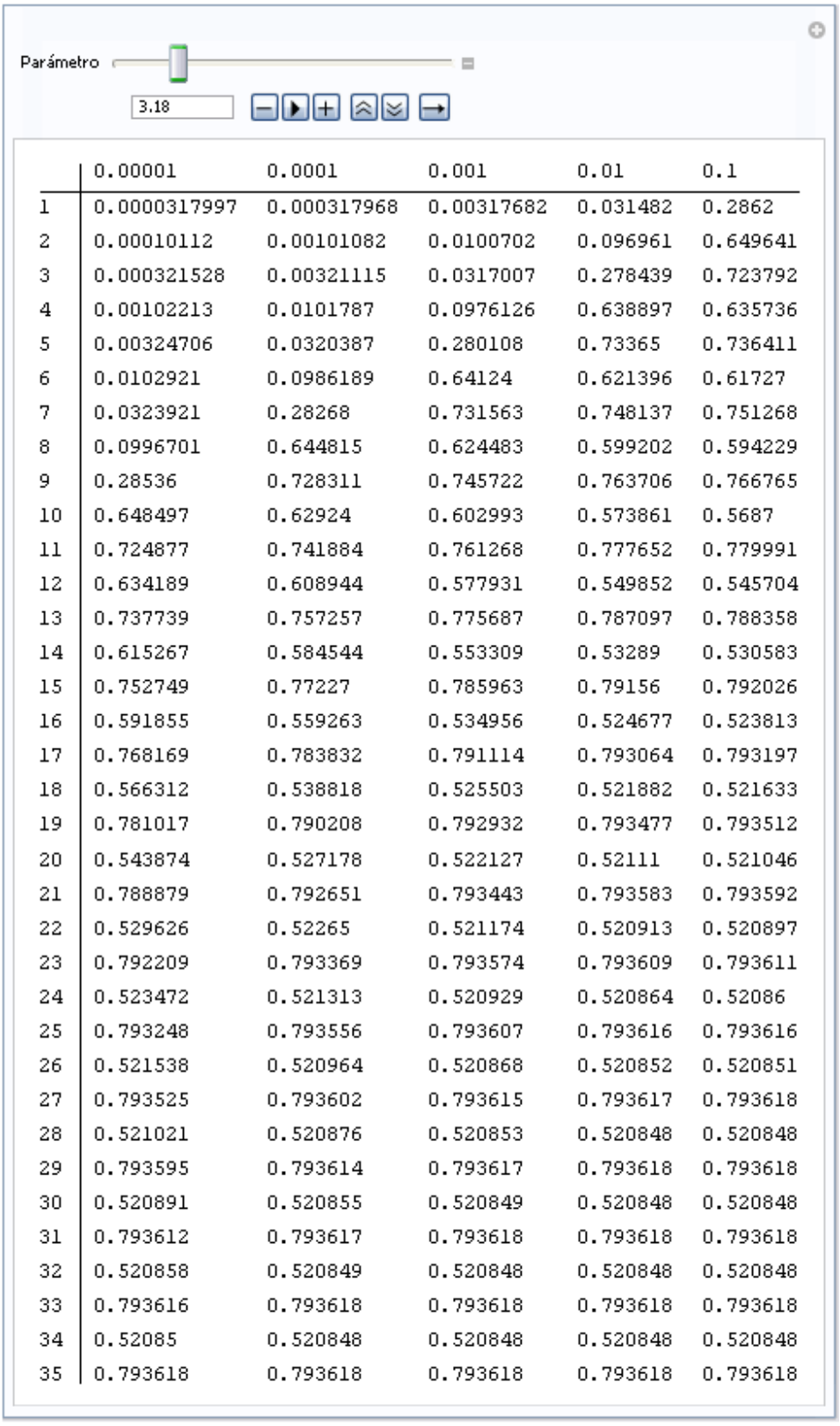
Solución actividad 2:



	0.00001	0.0001	0.001	0.01	0.1
1	0.0000300997	0.00030097	0.00300699	0.029799	0.2709
2	0.0000905974	0.000905647	0.00902382	0.0870222	0.594515
3	0.000272673	0.00272353	0.0269166	0.239142	0.725612
4	0.000820523	0.00817549	0.0788382	0.54768	0.599289
5	0.00246775	0.024407	0.218595	0.745657	0.722826
6	0.00740959	0.0716721	0.514141	0.570854	0.603049
7	0.0221376	0.200271	0.751898	0.737389	0.720537
8	0.0651591	0.482089	0.561508	0.582876	0.606104
9	0.183349	0.751534	0.741113	0.731826	0.718613
10	0.450694	0.562059	0.577513	0.590733	0.608647
11	0.745183	0.740908	0.734415	0.72772	0.716969
12	0.571555	0.57781	0.587099	0.596412	0.610802
13	0.737088	0.734276	0.729665	0.724521	0.715546
14	0.583305	0.587295	0.593734	0.600766	0.612655
15	0.731611	0.729562	0.726054	0.721937	0.714299
16	0.591032	0.593876	0.598688	0.604239	0.614268
17	0.727557	0.725974	0.723185	0.719794	0.713198
18	0.596636	0.598797	0.602568	0.607089	0.615685
19	0.724391	0.72312	0.720835	0.717981	0.712217
20	0.600943	0.602655	0.605709	0.609477	0.616942
21	0.72183	0.720781	0.718865	0.716424	0.711337
22	0.604382	0.60578	0.608315	0.611513	0.618063
23	0.719704	0.71882	0.717186	0.71507	0.710544
24	0.607208	0.608375	0.610519	0.613272	0.619071
25	0.717905	0.717147	0.715735	0.71388	0.709825
26	0.609578	0.61057	0.61241	0.614809	0.61998
27	0.716358	0.715701	0.714466	0.712825	0.70917
28	0.6116	0.612454	0.614054	0.616163	0.620806
29	0.715012	0.714436	0.713345	0.711883	0.708572
30	0.613347	0.614092	0.615496	0.617368	0.621558
31	0.713829	0.713319	0.712348	0.711037	0.708023
32	0.614875	0.61553	0.616774	0.618445	0.622247
33	0.71278	0.712325	0.711455	0.710272	0.707518
34	0.616222	0.616804	0.617913	0.619415	0.622878
35	0.711842	0.711434	0.710651	0.709578	0.707052



	0.00001	0.0001	0.001	0.01	0.1
1	0.0000301997	0.00030197	0.00301698	0.029898	0.2718
2	0.0000912003	0.000911673	0.00908379	0.0875924	0.597733
3	0.0002754	0.00275074	0.0271839	0.241358	0.726154
4	0.000831479	0.00828439	0.0798636	0.552976	0.60054
5	0.00250898	0.0248116	0.221926	0.746525	0.724473
6	0.0075581	0.0730719	0.521478	0.571461	0.602828
7	0.0226529	0.204552	0.753607	0.739578	0.723068
8	0.0668622	0.491385	0.560764	0.58166	0.604727
9	0.188423	0.754776	0.743849	0.734862	0.721877
10	0.461817	0.55897	0.575423	0.588417	0.606327
11	0.750597	0.744498	0.73782	0.731391	0.720858
12	0.565347	0.574466	0.584194	0.593304	0.60769
13	0.742104	0.738253	0.733593	0.728709	0.719977
14	0.577985	0.583571	0.590212	0.59703	0.608863
15	0.736633	0.733908	0.730423	0.726567	0.719209
16	0.585894	0.589767	0.594655	0.599975	0.609881
17	0.732719	0.730665	0.727942	0.724815	0.718537
18	0.591442	0.594317	0.598088	0.602364	0.610769
19	0.729748	0.728135	0.725944	0.723355	0.717945
20	0.595592	0.597823	0.600827	0.604339	0.61155
21	0.727404	0.726101	0.724298	0.722122	0.717421
22	0.598829	0.600613	0.603064	0.605999	0.612239
23	0.725503	0.724429	0.722921	0.721068	0.716955
24	0.601428	0.602888	0.604925	0.607409	0.61285
25	0.723932	0.723031	0.721752	0.720159	0.71654
26	0.603561	0.604777	0.606495	0.608621	0.613393
27	0.722611	0.721846	0.72075	0.719369	0.716169
28	0.605342	0.606369	0.607834	0.60967	0.613879
29	0.721487	0.72083	0.719883	0.718677	0.715836
30	0.606849	0.607726	0.608988	0.610584	0.614313
31	0.720521	0.719953	0.719128	0.718069	0.715536
32	0.608138	0.608895	0.609989	0.611387	0.614703
33	0.719684	0.719189	0.718465	0.717531	0.715266
34	0.609251	0.609908	0.610864	0.612095	0.615055
35	0.718954	0.718519	0.717882	0.717053	0.715023

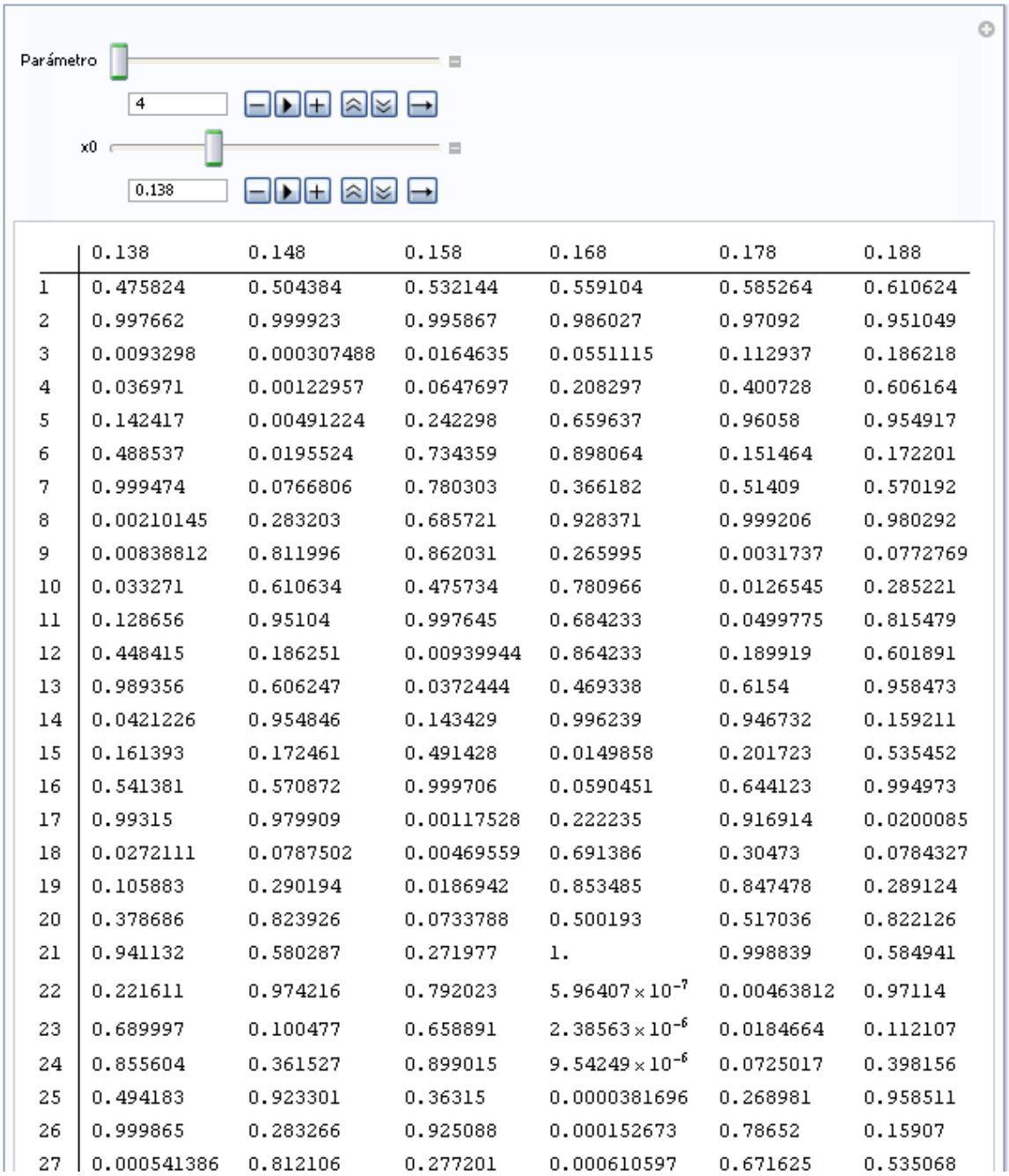


	0.00001	0.0001	0.001	0.01	0.1
1	0.0000317997	0.000317968	0.00317682	0.031482	0.2862
2	0.00010112	0.00101082	0.0100702	0.096961	0.649641
3	0.000321528	0.00321115	0.0317007	0.278439	0.723792
4	0.00102213	0.0101787	0.0976126	0.638897	0.635736
5	0.00324706	0.0320387	0.280108	0.73365	0.736411
6	0.0102921	0.0986189	0.64124	0.621396	0.61727
7	0.0323921	0.28268	0.731563	0.748137	0.751268
8	0.0996701	0.644815	0.624483	0.599202	0.594229
9	0.28536	0.728311	0.745722	0.763706	0.766765
10	0.648497	0.62924	0.602993	0.573861	0.5687
11	0.724877	0.741884	0.761268	0.777652	0.779991
12	0.634189	0.608944	0.577931	0.549852	0.545704
13	0.737739	0.757257	0.775687	0.787097	0.788358
14	0.615267	0.584544	0.553309	0.53289	0.530583
15	0.752749	0.77227	0.785963	0.79156	0.792026
16	0.591855	0.559263	0.534956	0.524677	0.523813
17	0.768169	0.783832	0.791114	0.793064	0.793197
18	0.566312	0.538818	0.525503	0.521882	0.521633
19	0.781017	0.790208	0.792932	0.793477	0.793512
20	0.543874	0.527178	0.522127	0.52111	0.521046
21	0.788879	0.792651	0.793443	0.793583	0.793592
22	0.529626	0.52265	0.521174	0.520913	0.520897
23	0.792209	0.793369	0.793574	0.793609	0.793611
24	0.523472	0.521313	0.520929	0.520864	0.52086
25	0.793248	0.793556	0.793607	0.793616	0.793616
26	0.521538	0.520964	0.520868	0.520852	0.520851
27	0.793525	0.793602	0.793615	0.793617	0.793618
28	0.521021	0.520876	0.520853	0.520848	0.520848
29	0.793595	0.793614	0.793617	0.793618	0.793618
30	0.520891	0.520855	0.520849	0.520848	0.520848
31	0.793612	0.793617	0.793618	0.793618	0.793618
32	0.520858	0.520849	0.520848	0.520848	0.520848
33	0.793616	0.793618	0.793618	0.793618	0.793618
34	0.52085	0.520848	0.520848	0.520848	0.520848
35	0.793618	0.793618	0.793618	0.793618	0.793618

	0.112	0.122	0.132	0.142	0.152	0.162
1	0.348096	0.374906	0.401016	0.426426	0.451136	0.475146
2	0.794238	0.82023	0.840708	0.856054	0.866643	0.872838
3	0.571984	0.516084	0.468714	0.431289	0.404505	0.388471
4	0.856864	0.874095	0.871574	0.858476	0.843082	0.831465
5	0.429268	0.385186	0.391764	0.425232	0.463031	0.490459
6	0.857489	0.828862	0.833998	0.855434	0.870216	0.874681
7	0.427705	0.496473	0.48456	0.432832	0.395289	0.383649
8	0.856707	0.874956	0.874166	0.85921	0.836625	0.827618
9	0.429661	0.382927	0.385	0.423389	0.478393	0.499332
10	0.857683	0.827029	0.828713	0.854458	0.873366	0.874998
11	0.427219	0.500683	0.496818	0.435259	0.387092	0.382817
12	0.85646	0.874998	0.874965	0.86033	0.830382	0.826938
13	0.430276	0.382817	0.382906	0.420568	0.492968	0.50089
14	0.857985	0.826938	0.827011	0.852917	0.874827	0.874997
15	0.426464	0.50089	0.500723	0.439074	0.383267	0.38282
16	0.856073	0.874997	0.874998	0.862008	0.827307	0.826941
17	0.431241	0.38282	0.382817	0.416325	0.500046	0.500884
18	0.858453	0.826941	0.826939	0.850495	0.875	0.874997
19	0.425291	0.500884	0.500889	0.445037	0.382813	0.38282
20	0.855465	0.874997	0.874997	0.864427	0.826935	0.826941
21	0.432757	0.38282	0.38282	0.410176	0.500898	0.500884
22	0.859174	0.826941	0.826941	0.846761	0.874997	0.874997
23	0.423479	0.500884	0.500884	0.454149	0.38282	0.38282
24	0.854506	0.874997	0.874997	0.867642	0.826941	0.826941
25	0.43514	0.38282	0.38282	0.401938	0.500884	0.500884
26	0.860276	0.826941	0.826941	0.841343	0.874997	0.874997
27	0.420704	0.500884	0.500884	0.467196	0.38282	0.38282

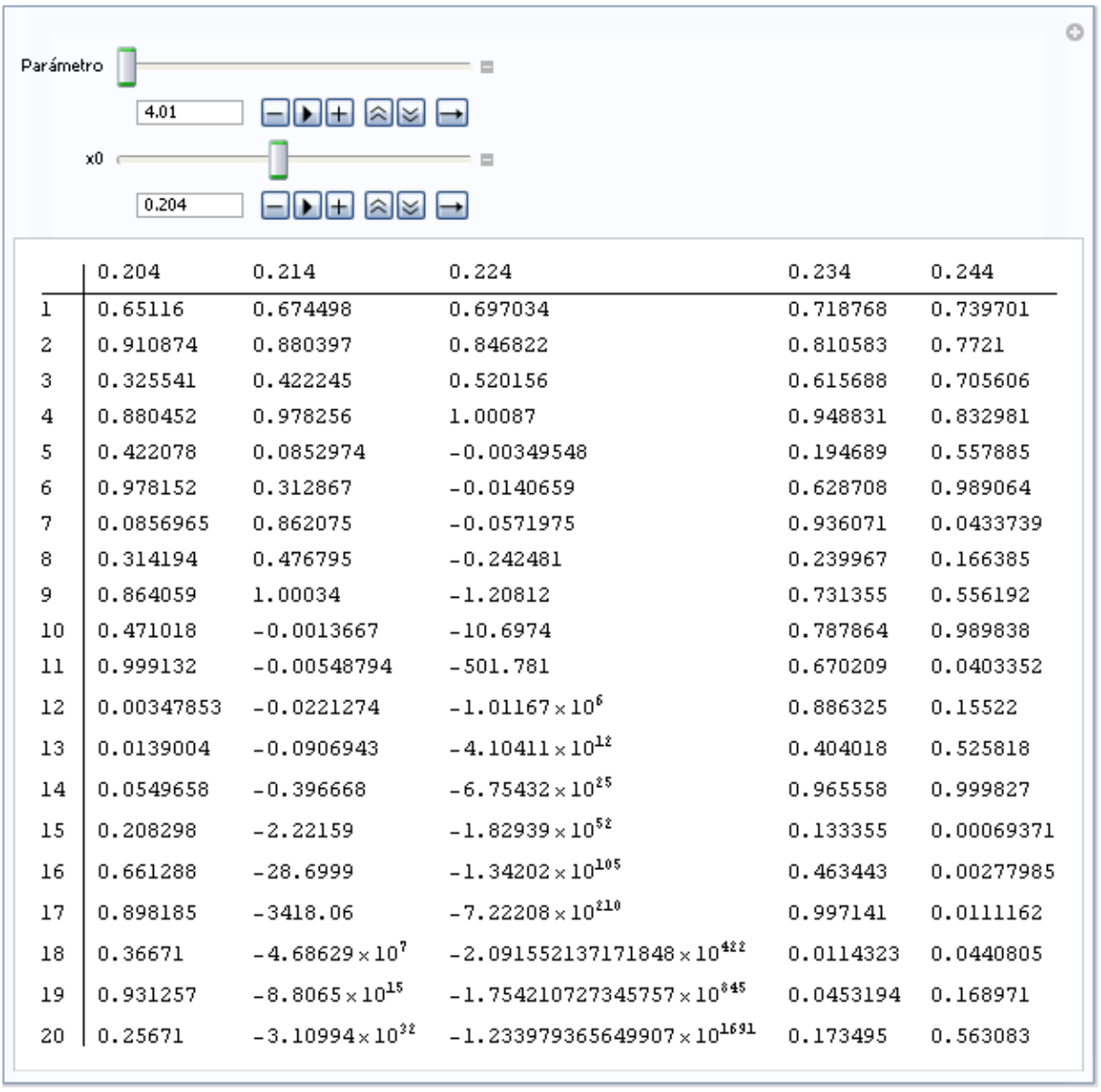
28	0.852992	0.874997	0.874997	0.871234	0.826941	0.826941
29	0.438887	0.38282	0.38282	0.392649	0.500884	0.500884
30	0.861928	0.826941	0.826941	0.834665	0.874997	0.874997
31	0.416527	0.500884	0.500884	0.482997	0.38282	0.38282
32	0.850613	0.874997	0.874997	0.873988	0.826941	0.826941
33	0.444746	0.38282	0.38282	0.385465	0.500884	0.500884
34	0.864315	0.826941	0.826941	0.829086	0.874997	0.874997
35	0.410462	0.500884	0.500884	0.495958	0.38282	0.38282

Solución actividad 3:

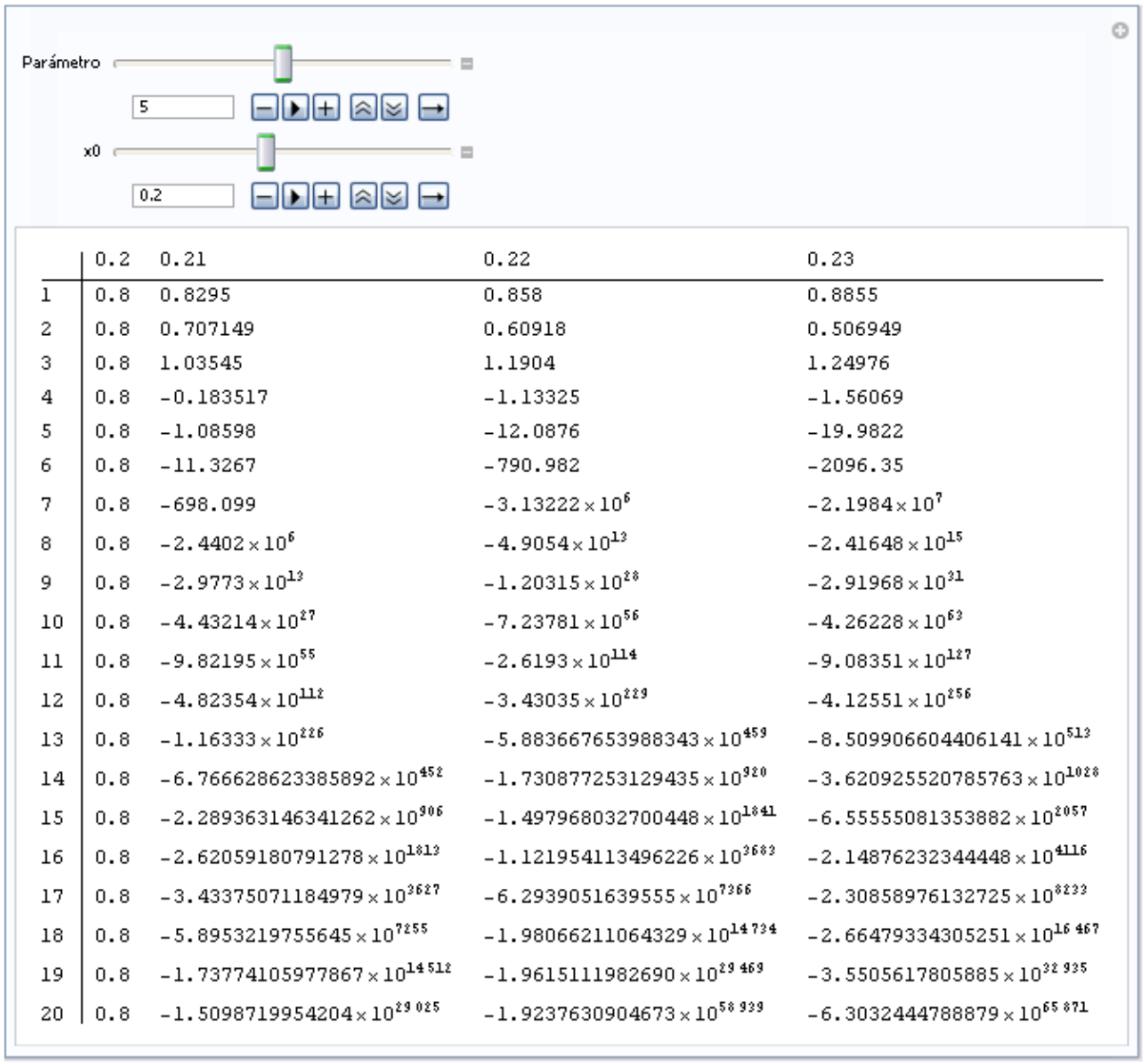


	0.138	0.148	0.158	0.168	0.178	0.188
1	0.475824	0.504384	0.532144	0.559104	0.585264	0.610624
2	0.997662	0.999923	0.995867	0.986027	0.97092	0.951049
3	0.0093298	0.000307488	0.0164635	0.0551115	0.112937	0.186218
4	0.036971	0.00122957	0.0647697	0.208297	0.400728	0.606164
5	0.142417	0.00491224	0.242298	0.659637	0.96058	0.954917
6	0.488537	0.0195524	0.734359	0.898064	0.151464	0.172201
7	0.999474	0.0766806	0.780303	0.366182	0.51409	0.570192
8	0.00210145	0.283203	0.685721	0.928371	0.999206	0.980292
9	0.00838812	0.811996	0.862031	0.265995	0.0031737	0.0772769
10	0.033271	0.610634	0.475734	0.780966	0.0126545	0.285221
11	0.128656	0.95104	0.997645	0.684233	0.0499775	0.815479
12	0.448415	0.186251	0.00939944	0.864233	0.189919	0.601891
13	0.989356	0.606247	0.0372444	0.469338	0.6154	0.958473
14	0.0421226	0.954846	0.143429	0.996239	0.946732	0.159211
15	0.161393	0.172461	0.491428	0.0149858	0.201723	0.535452
16	0.541381	0.570872	0.999706	0.0590451	0.644123	0.994973
17	0.99315	0.979909	0.00117528	0.222235	0.916914	0.0200085
18	0.0272111	0.0787502	0.00469559	0.691386	0.30473	0.0784327
19	0.105883	0.290194	0.0186942	0.853485	0.847478	0.289124
20	0.378686	0.823926	0.0733788	0.500193	0.517036	0.822126
21	0.941132	0.580287	0.271977	1.	0.998839	0.584941
22	0.221611	0.974216	0.792023	5.96407×10^{-7}	0.00463812	0.97114
23	0.689997	0.100477	0.658891	2.38563×10^{-6}	0.0184664	0.112107
24	0.855604	0.361527	0.899015	9.54249×10^{-6}	0.0725017	0.398156
25	0.494183	0.923301	0.36315	0.0000381696	0.268981	0.958511
26	0.999865	0.283266	0.925088	0.000152673	0.78652	0.15907
27	0.000541386	0.812106	0.277201	0.000610597	0.671625	0.535068

28	0.00216437	0.61036	0.801443	0.0024409	0.88218	0.995081
29	0.00863875	0.951282	0.636529	0.00973975	0.415753	0.0195794
30	0.0342565	0.185377	0.92544	0.0385796	0.97161	0.0767842
31	0.132332	0.60405	0.276005	0.148365	0.110336	0.283553
32	0.459281	0.956695	0.799305	0.505411	0.392647	0.812604
33	0.993368	0.16572	0.641667	0.999883	0.953901	0.609116
34	0.0263532	0.553027	0.919722	0.000468329	0.175895	0.952375
35	0.102635	0.988753	0.295333	0.00187244	0.579824	0.181428
36	0.368404	0.0444834	0.832446	0.00747572	0.974512	0.594048
37	0.930729	0.170019	0.55792	0.0296793	0.0993519	0.96462
38	0.257888	0.564449	0.986581	0.115194	0.357924	0.136512
39	0.765528	0.983385	0.0529546	0.407697	0.919258	0.471506
40	0.71798	0.0653547	0.200602	0.965921	0.296891	0.996752
41	0.80994	0.244334	0.641443	0.131671	0.834986	0.0129482
42	0.61575	0.738539	0.919976	0.457336	0.551137	0.051122
43	0.946408	0.772397	0.294481	0.992719	0.98954	0.194034
44	0.20288	0.7032	0.831048	0.0289111	0.0414022	0.62554
45	0.64688	0.834838	0.56163	0.112301	0.158752	0.936959
46	0.913706	0.551533	0.984807	0.398758	0.5342	0.236268
47	0.315391	0.989377	0.0598479	0.959	0.995321	0.721781
48	0.863678	0.0420392	0.225065	0.157276	0.018627	0.803252
49	0.470953	0.161088	0.697642	0.530162	0.07312	0.632152
50	0.996625	0.540553	0.843751	0.996361	0.271094	0.930143

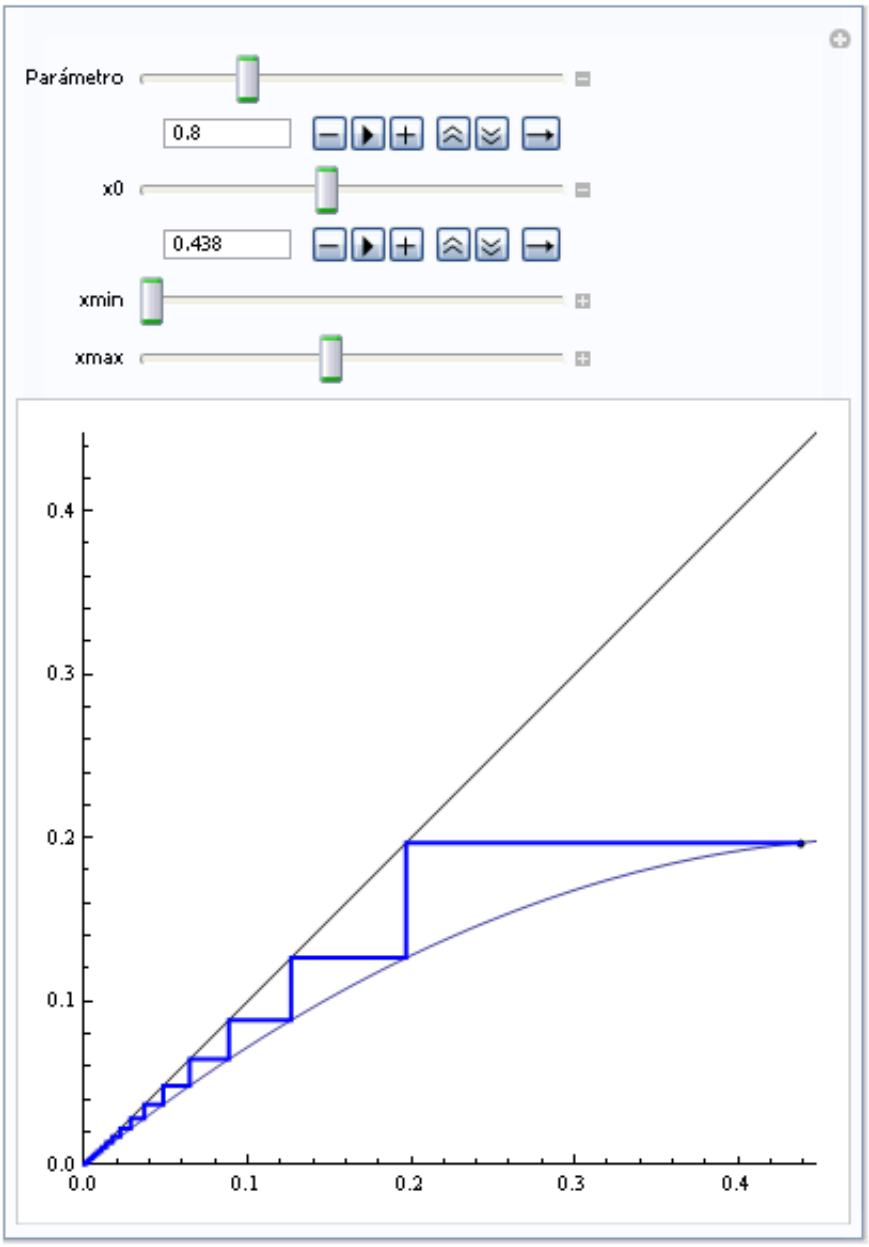


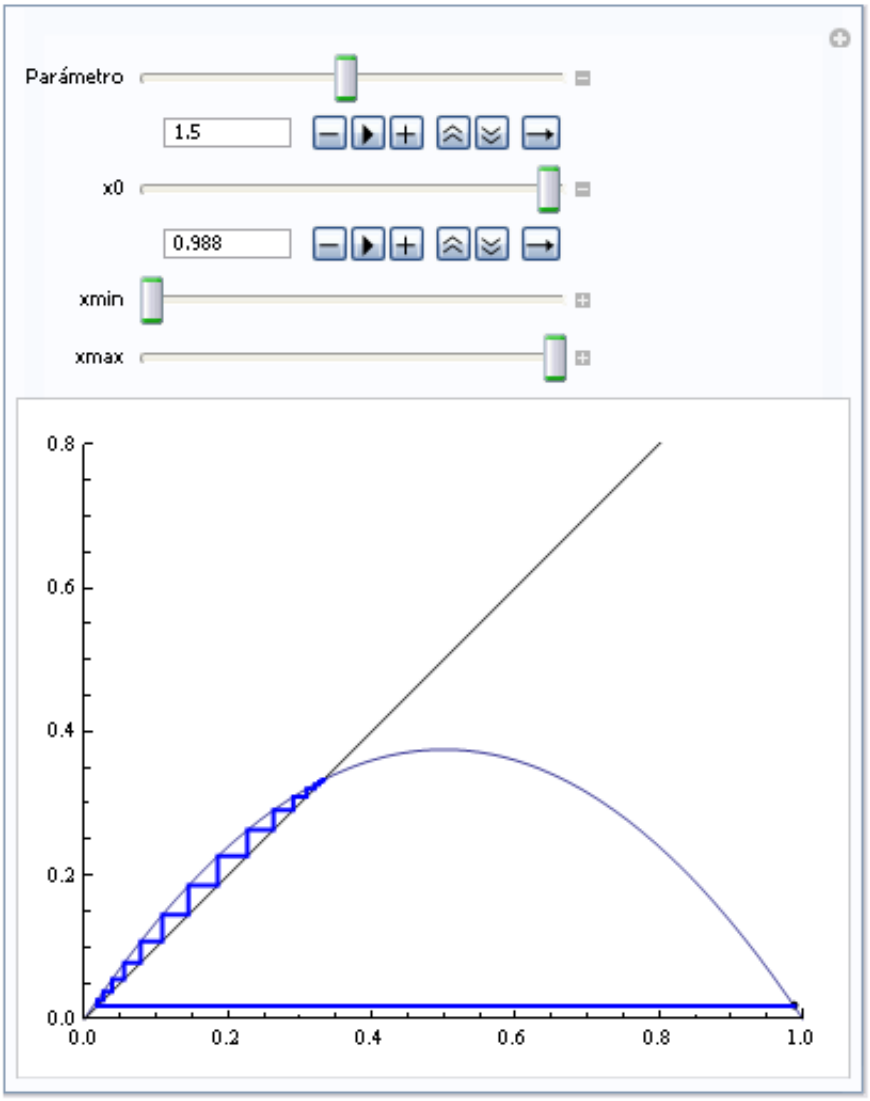
	0.204	0.214	0.224	0.234	0.244
1	0.65116	0.674498	0.697034	0.718768	0.739701
2	0.910874	0.880397	0.846822	0.810583	0.7721
3	0.325541	0.422245	0.520156	0.615688	0.705606
4	0.880452	0.978256	1.00087	0.948831	0.832981
5	0.422078	0.0852974	-0.00349548	0.194689	0.557885
6	0.978152	0.312867	-0.0140659	0.628708	0.989064
7	0.0856965	0.862075	-0.0571975	0.936071	0.0433739
8	0.314194	0.476795	-0.242481	0.239967	0.166385
9	0.864059	1.00034	-1.20812	0.731355	0.556192
10	0.471018	-0.0013667	-10.6974	0.787864	0.989838
11	0.999132	-0.00548794	-501.781	0.670209	0.0403352
12	0.00347853	-0.0221274	-1.01167×10^{6}	0.886325	0.15522
13	0.0139004	-0.0906943	-4.10411×10^{12}	0.404018	0.525818
14	0.0549658	-0.396668	-6.75432×10^{25}	0.965558	0.999827
15	0.208298	-2.22159	-1.82939×10^{52}	0.133355	0.00069371
16	0.661288	-28.6999	-1.34202×10^{105}	0.463443	0.00277985
17	0.898185	-3418.06	-7.22208×10^{210}	0.997141	0.0111162
18	0.36671	-4.68629×10^{7}	$-2.091552137171848\times10^{422}$	0.0114323	0.0440805
19	0.931257	-8.8065×10^{15}	$-1.754210727345757\times10^{845}$	0.0453194	0.168971
20	0.25671	-3.10994×10^{32}	$-1.233979365649907\times10^{1691}$	0.173495	0.563083

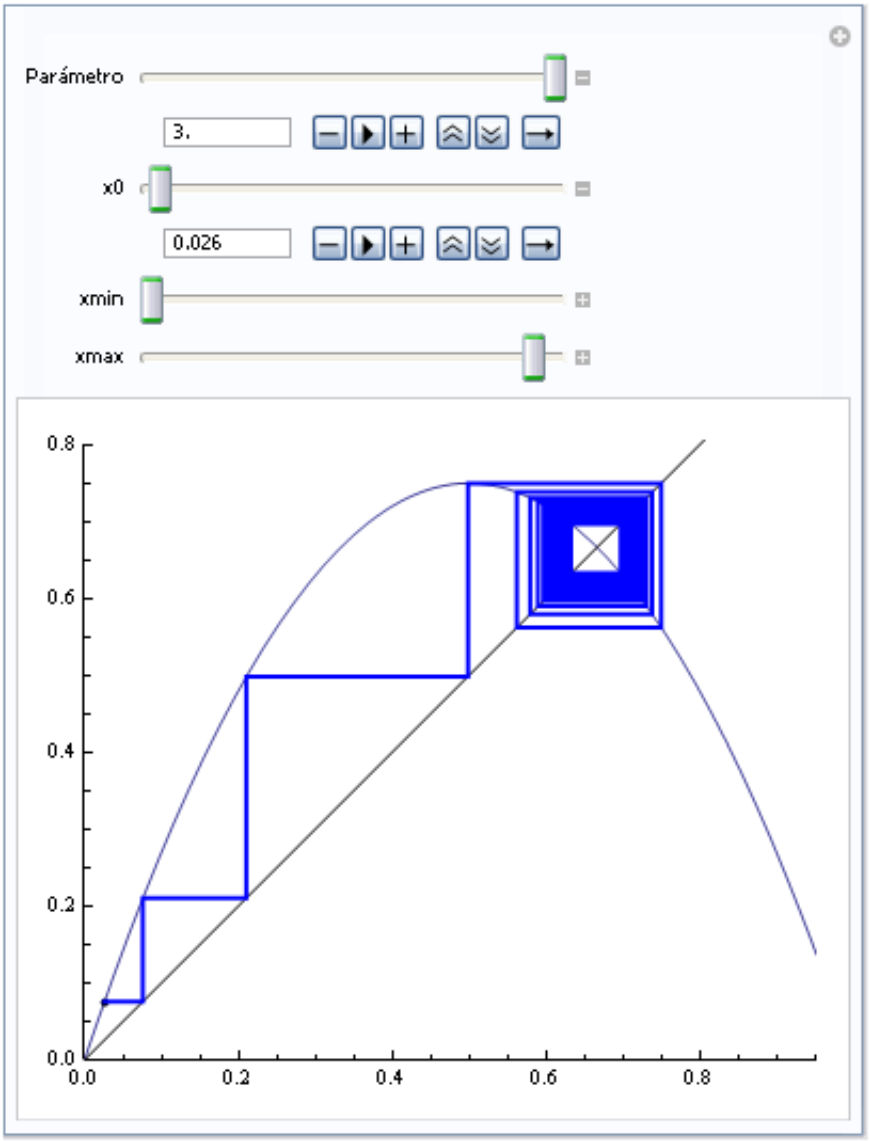


	0.2	0.21	0.22	0.23
1	0.8	0.8295	0.858	0.8855
2	0.8	0.707149	0.60918	0.506949
3	0.8	1.03545	1.1904	1.24976
4	0.8	-0.183517	-1.13325	-1.56069
5	0.8	-1.08598	-12.0876	-19.9822
6	0.8	-11.3267	-790.982	-2096.35
7	0.8	-698.099	-3.13222×10^{6}	-2.1984×10^{7}
8	0.8	-2.4402×10^{6}	-4.9054×10^{13}	-2.41648×10^{15}
9	0.8	-2.9773×10^{13}	-1.20315×10^{28}	-2.91968×10^{31}
10	0.8	-4.43214×10^{27}	-7.23781×10^{56}	-4.26228×10^{63}
11	0.8	-9.82195×10^{55}	-2.6193×10^{114}	-9.08351×10^{127}
12	0.8	-4.82354×10^{112}	-3.43035×10^{229}	-4.12551×10^{256}
13	0.8	-1.16333×10^{226}	$-5.883667653988343\times10^{459}$	$-8.509906604406141\times10^{513}$
14	0.8	$-6.766628623385892\times10^{452}$	$-1.730877253129435\times10^{920}$	$-3.620925520785763\times10^{1028}$
15	0.8	$-2.289363146341262\times10^{906}$	$-1.497968032700448\times10^{1841}$	$-6.55555081353882\times10^{2057}$
16	0.8	$-2.62059180791278\times10^{1813}$	$-1.121954113496226\times10^{3683}$	$-2.14876232344448\times10^{4116}$
17	0.8	$-3.43375071184979\times10^{3627}$	$-6.2939051639555\times10^{7366}$	$-2.30858976132725\times10^{8233}$
18	0.8	$-5.8953219755645\times10^{7255}$	$-1.98066211064329\times10^{14734}$	$-2.66479334305251\times10^{16467}$
19	0.8	$-1.73774105977867\times10^{14512}$	$-1.9615111982690\times10^{29469}$	$-3.5505617805885\times10^{32935}$
20	0.8	$-1.5098719954204\times10^{29025}$	$-1.9237630904673\times10^{58939}$	$-6.3032444788879\times10^{65871}$

Solución actividad 4:




Parámetro
1.5
x0
0.988
xmin
xmax
0.8
0.6
0.4
0.2
0.0
0.0
0.2
0.4
0.6
0.8
1.0


Parámetro
3.
x0
0.026
xmin
xmax
0.8
0.6
0.4
0.2
0.0
0.0
0.2
0.4
0.6
0.8

2

Introducción a los sistemas dinámicos caóticos

2.1. El diagrama de órbitas

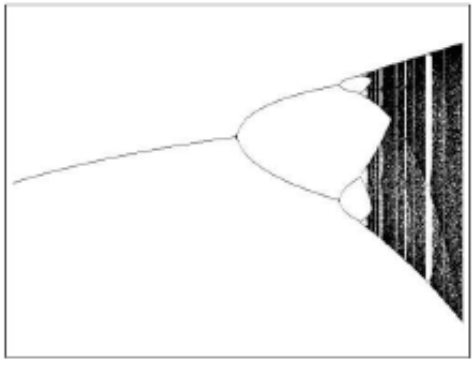
La ecuación logística se ha convertido en la manera usual de introducir las características del caos. Se trata de una ecuación en diferencias que fue formulada por Verhulst en el siglo pasado para explicar el crecimiento de una población perteneciente a la misma especie y que se reproduce en un entorno cerrado sin ningún tipo de influencia externa. Pese a su aparente sencillez, constituye un buen ejemplo para mostrar el comportamiento de los sistemas caóticos. La ecuación se puede escribir como

$$x_{n+1} = r\, x_n(1 - x_n)$$

donde el parámetro r es una constante denominada parámetro de crecimiento (generalmente entre 0 y 4) y la variable x_n puede verse como la fracción máxima de población que el ambiente puede soportar en el instante t_n. Considerando que la población límite $x_\infty = \lim_{n\to\infty} x_n$ existe, queremos investigar la forma en la cual x_n depende del parámetro de crecimiento r. Si estudiamos experimentalmente el sistema, observaremos que para valores de $r < 3$ el sistema converge a un punto fijo estable, que es cero cuando $r < 1$. Cuando $r > 3$, el punto fijo se hace inestable y el valor de x_∞ oscila entre dos valores. Este fenómeno se conoce como una ***duplicación de periodo***.

Si se aumenta r ligeramente, por ejemplo $r = 3.44$ ..., el número de puntos sobre los que oscila x_∞ es cuatro. Si se sigue aumentando el valor de r, aparece una nueva duplicación de periodo para $r = 3.544$ ..., obteniendo un periodo de ocho. Y así sucesivamente hasta llegar a obtener una sucesión de infinitos valores para x_∞ correspondiente al caos. Nótese cómo los valores de r para los que se producen las sucesivas duplicaciones de periodo están cada vez más cerca unos de otros.

El comportamiento de la ecuación logística en función de r puede observarse visualmente a través de un ***diagrama de órbitas*** o ***diagrama de bifurcación***. En el eje horizontal se representa un cierto intervalo de valores de r y entonces se dibujan los valores de x generados por la iteración en el eje vertical. La figura siguiente muestra el diagrama de bifurcación de la ecuación logística en el rango $2 \le r \le 4$.



La duplicación de periodo es un signo ineludible del comportamiento caótico de un sistema. Son muchos, y cada día más, los sistemas dinámicos en los que se observa este fenómeno y que desembocan, variando alguno de sus parámetros, en caos. Es más, un famoso artículo publicado en los inicios de la teoría del caos demostró que cualquier sistema en el que, para algún valor de sus parámetros, se registrara una periodicidad de periodo tres, desembocará para otros valores de sus parámetros en comportamiento caótico.
Lo anterior hace pensar en una universalidad del caos todavía no muy bien conocida que hace que sistemas muy diferentes muestren pautas similares de comportamiento. Un hecho que vino a corroborar esto y a mostrarnos que existe un cierto orden en el caos fue el descubrimiento por parte de Feigenbaum alrededor de 1975 de la constante que lleva su nombre. Una vez obtenido el diagrama de bifurcación de la ecuación logística, se puede calcular el incremento del parámetro entre dos bifurcaciones contiguas

$$\Delta i = r_i - r_{i-1}$$

y dividiendo por el incremento en el siguiente intervalo

$$\frac{\Delta i}{\Delta (i+1)} = \frac{r_i - r_{i-1}}{r_{i+1} - r_i}$$

Feigenbaum encontró que la fracción anterior convergía hacia un valor determinado al ir haciendo i mayor, de modo que en el límite se obtenía

$$\delta = \lim_{i \to \infty} \frac{\Delta i}{\Delta (i+1)} = 4.669201609102990671853203820466 2 \ ...$$

Feigenbaum calculó el límite anterior para otras ecuaciones en diferencias y obtuvo el mismo valor para δ. Posteriormente se ha encontrado el mismo valor de δ en algunos sistemas continuos e incluso en sistemas experimentales, todos de muy diversa procedencia. Hoy sabemos que δ, conocida como constante de Feigenbaum, es una constante universal tan fundamental como π o e y que ha provocado una nueva forma de ver el mundo.

En una dimensión, las funciones "suaves" y no lineales más sencillas son polinomios de grado dos, que se representan como: $P(x) = a\,x^2 + b\,x + c$, a, b y c son números reales con $a \neq 0$. Por ejemplo, si $b = -a$ y $c = 0$, se obtiene la función que se presenta en la ecuación logística.
Más aun, un poco de reflexión conduce a pensar que al trasladar el origen a un punto cualquiera, esta traslación no tendrá efecto sobre el comportamiento de las órbitas. Lo mismo sucede al expandir o contraer mediante una homotecia (como se aprecia al realizar el análisis gráfico). Así, llamando u a las coordenadas resultantes de una traslación, la transformación original $x \to x_1 = a\,x^2 + b\,x + c$ se escribirá mediante el cambio $u = x + d$, $u_1 = x_1 + d$.

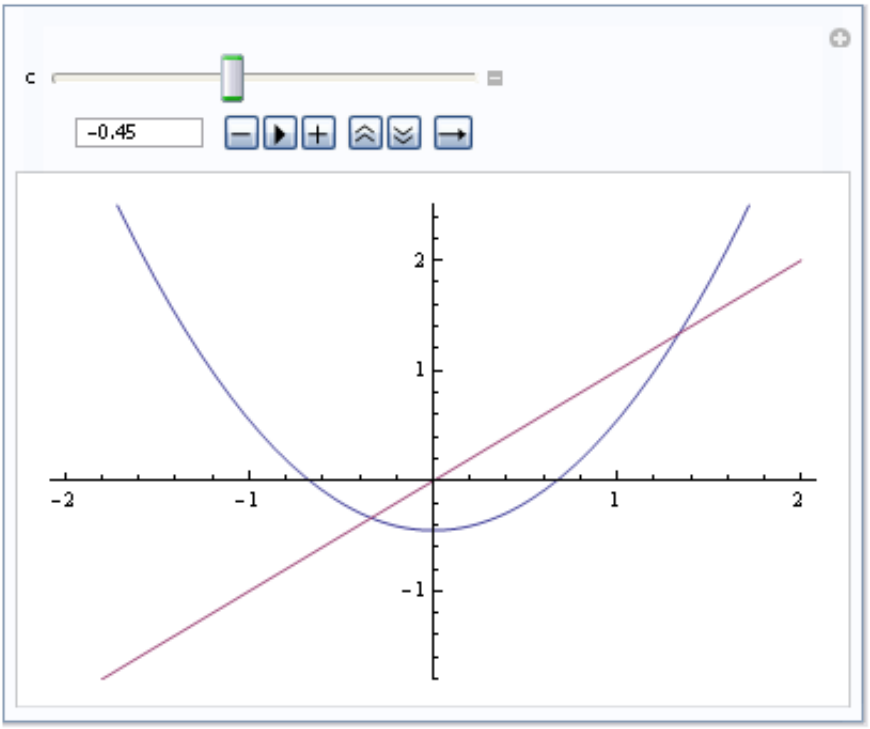
Actividad 1:

- **a.** Exprese u_1 en términos de u y desarrolle dicha expresión.
- **b.** Hágase la sustitución $d = \frac{b}{2\,a}$ y después de simplificar, haga $e = c + \frac{b(2-b)}{4\,a}$.
- **c.** Puesto que $a \neq 0$, se puede realizar la transformación $v = a\,u$, $v_1 = a\,u_1$ (que es homotecia si $a > 0$ o una homotecia y rotación si $a < 0$) .
- **d.** Finalmente, haciendo $k = a\,e$ se obtiene la función $v \to v_1 = v^2 + k$ equivalente a la original.

El ejercicio anterior permite afirmar que basta estudiar las órbitas en el caso en que $a = 1$ y $b = 0$, para el cual la función es $x \to x_1 = x^2 + c$ donde c es un número real arbitrario.

Dado que c es el único parámetro surge la siguiente pregunta: ¿cómo cambia el comportamiento de las órbitas a medida que varia el parámetro c en la función $f(x) = x^2 + c$?
Para ayudar a contestar la pregunta anterior se presenta el código siguiente:

```
Manipulate[Plot[{x^2 + c, x}, {x, -2, 2}, PlotRange → {-1.8, 2.5}],
 {c, -1.5, 1, .05, Appearance → "Open"},
          SaveDefinitions → True]
```



Actividad 2:

Varíe el parámetro c y estime los valores del mismo, para los cuales se observa un cambio de comportamiento.

Actividad 3:

Encontrar algebraicamente los puntos fijos de la función $f(x) = x^2 + c$ y demuestre que:

a. Si $c < \frac{1}{4}$ existen dos puntos fijos distintos.

b. Si $c = \frac{1}{4}$ el punto fijo es único.

c. Si $c > \frac{1}{4}$ no existen dos puntos fijos (reales).

El hecho de que al disminuir c desde $\frac{1}{4}$ "aparecen" dos puntos fijos es un ejemplo de un fenómeno llamado **bifurcación**: un cambio repentino de las condiciones del problema. Un aspecto que es importante estudiar para cada valor del parámetro c es determinar el conjunto de puntos x's cuyas órbitas "se escapan a infinito", es decir, aquellos x's cuyas órbitas divergen.

Actividad 4:

Modificando apropiadamente lo realizado para la ecuación logística, establezca para cada valor del parámetro c el conjunto de x's cuyas órbitas son no acotadas y el conjunto de órbitas no acotadas buscando mediante el uso de tablas, análisis gráfico y la función *cicloatractivo*.

Tomando en cuenta que para $c < \frac{1}{4}$, se pueden obtener dos puntos fijos mediante la fórmula cuadrática, denomine con p y q a los mismos y pruebe que: $-p < q < p$. Esto motiva la actividad siguiente:

Actividad 5:

Utilice el análisis gráfico para determinar que cuando $c < \frac{1}{4}$, la órbita de cada valor inicial es divergente si $x > p$ o si $x < -p$.

Actividad 6:

Haciendo el gráfico de la función f, y las rectas $y = x$ y $y = -p$, determine (si existe) el valor del parámetro c para que la parábola sea tangente a la recta $y = -p$, en caso de que su respuesta sea afirmativa determine si es único.

Corrobore sus resultados analíticamente, de la manera siguiente:

a. Empleando desigualdades.

b. Utilizando la derivada.

Una forma de ver cómo varían las órbitas a medida que varía el parámetro c es considerar las órbitas para un mismo punto x para todas ellas. Por ejemplo tomando como valor inicial a cero bajo las funciones de la forma $x \to x^2 + c$, se elabora el correspondiente gráfico para el cual se ubican sobre el eje horizontal los valores del parámetro c entre -2 y $\frac{1}{4}$, mientras que en el eje vertical se colocan los distintos valores de la órbita iniciada en cero, se obtiene así el denominado

Diagrama de Órbitas.

A continuación se presenta un código para hacer el diagrama de órbitas. Primeramente se define la función *orbitaciclo*, que para cada valor de c devuelve un conjunto de puntos en la órbita inicializada en $x = 0$. Puesto que sólo interesa estudiar el comportamiento asintótico de cada órbita, primero se realizan de *itermin* iteraciones para obtener un nuevo valor inicial a partir del cual se realizará el análisis. Por supuesto, dado que sólo es posible realizar un número finito de iteraciones, no se continuará el proceso si el número de iteraciones es mayor que *mostrar*. Además, dado que *orbitaciclo* es una función que no asigna a cada c un valor, sino un conjunto de puntos, no se utiliza *Plot* para graficar. Es pues necesario construir una tabla con los resultados de *orbitaciclo* para varios valores de c, para luego graficar los valores tabulados. Empero, para no dibujar tantos puntos innecesarios se utilizará una variante de *cicloatractivo*, la cual es llamada desde la función *diagramar*.

```
Clear[orbitaociclo, itermin, mostrar, precision]
orbitaociclo[c_] :=
    Module[{n = 1, x1, x2, x3, lista}, (*Todas las órbitas se inician en cero*)
        x1 = x3 = Nest[(f[#]) &, 0., itermin]; x2 = f[x1];
        While[(Abs[x1 - x2] > precision) ⋀ (n++ ≤ mostrar), x1 = f[x1]; x2 = f[f[x2]]];
              If[n ≤ mostrar, (*ciclo de longitud≤n*)lista = {x1}; x2 = f[x1];
                n = 1;
   While[Abs[x1 - x2] > precision, AppendTo[lista, x2]; x2 = f[x2]; n++]; lista,
        (*else*)NestList[(f[#]) &, x3, mostrar - 1]]]

diagramar[c1_, c2_, divisiones_] :=
        Flatten[Table[Map[{c, #} &, orbitaociclo[c]], {c, c1, c2, (c2 - c1) / divisiones}], 1]
```

Ejemplo:

Construir el diagrama de órbita para la función $x \to x^2 + c$, con un número de iteraciones mínima de 600, una precisión de .001, en el cual se muestre no más de 350 elementos de cada órbita, para distintos valores de c en el intervalo $\left[-2, \frac{1}{4}\right]$.

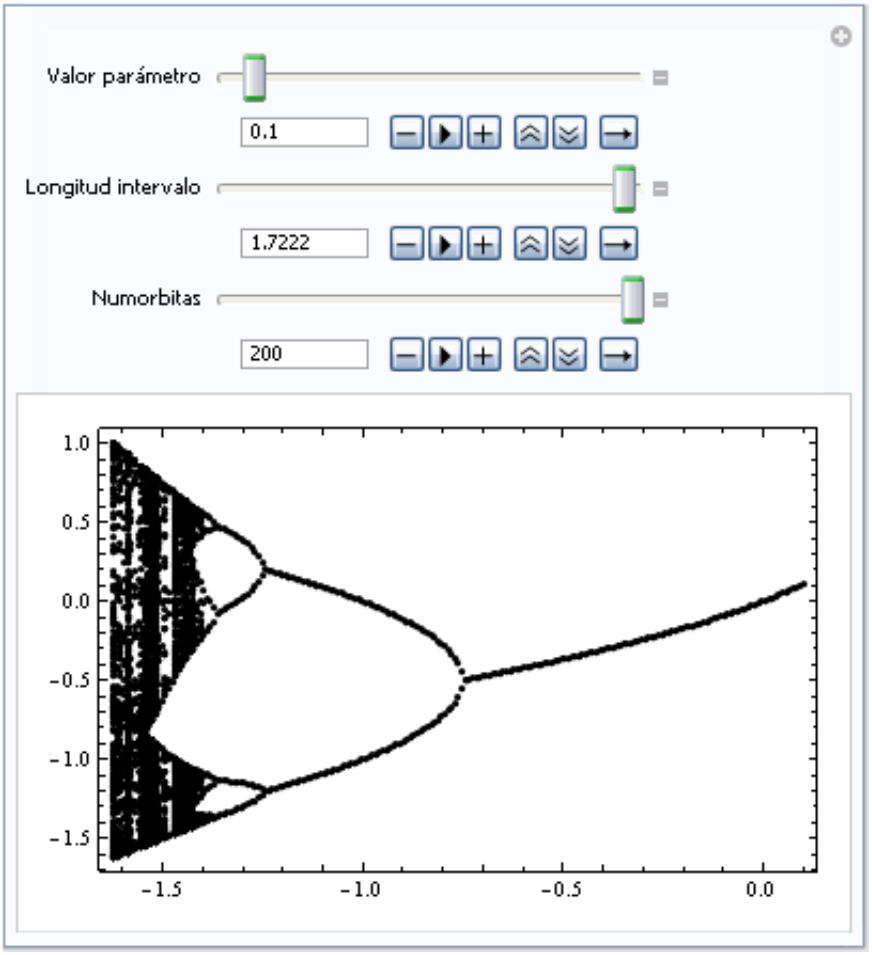
Solución:

Primero se define la aplicación.

```
Clear[f, c]; f[x_] := x^2 + c
```

Después, con la función *diagramar* se construye la tabla de valores cada órbita inicializada en cero y los distintos valores del parámetro c. Finalmente, se grafica cada par ordenado de la tabla antes construida.

```
Manipulate[itermin = 600; mostrar = 350; precision = N[10^-3];
           Graphics[{PointSize[0.01], Point /@ diagramar[c0, Max[c0 - l, -2], Numorbitas]},
  AspectRatio → 1 / GoldenRatio,
                    Frame → True, PlotRange → All],
           {{c0, .25, "Valor parámetro"}, .25, -2., -.01, Appearance → "Open"},
            {{l, 1.75, "Longitud intervalo"}, 0.001, 1.75, .0001, Appearance → "Open"},
           {{Numorbitas, 35}, 30, 200, 20, Appearance → "Open"}, SaveDefinitions → True]
```



Actividad 7:

Observe detenidamente el gráfico anteriormente elaborado e interprételo.

Observación: Para construir una mejor aproximación al diagrama de Órbitas, aumente el valor de **Numorbitas**.

Actividad 8:

Con respecto al trabajo realizado anteriormente, aumente el número de iteraciones mínimas y cambie la precisión.

a. ¿Depende sustancialmente del valor de la precisión el nivel de detalle de las gráficas generadas?

b. ¿Surgen nuevas bifurcaciones?, si es así estime el valor del parámetro y describa el fenómeno.

Otra manera de ver los detalles en el comportamiento de las órbitas en el diagrama de órbitas es cambiar los extremos del intervalo (hacer un "zoom") al que pertenece el parámetro c.

Dado que los valores de $c = -1.25$ y $c = -1.75$, son los valores en los que se han detectado previamente bifurcaciones es necesario hacer un estudio más detallado, por lo que se propone la siguiente actividad:

Actividad 9:

a. Construir el diagrama de órbita para la función $x \to x^2 + c$, con un número 500 iteraciones como mínimo, y un máximo en la longitud de cada ciclo de 300, para 300 valores de c equiespaciados en el intervalo $[-1.45, -1.2]$.

b. Construir el diagrama de órbita para la función $x \to x^2 + c$, con un número 700 iteraciones como mínimo, y un máximo en la longitud de cada ciclo de 300, para 300 valores de c equiespaciados en el intervalo $[-1.8, -1.74]$.

El primero de los gráficos elaborados en la actividad anterior muestra que efectivamente, cerca de aproximadamente $c = -1.37$ se pasa de una órbita de longitud cuatro a una órbita de longitud ocho, y que después de aproximadamente -1.39 se genera una órbita más complicada.

Del segundo gráfico construido en la actividad anterior permite afirmar que después de un comportamiento "caótico" poco antes de $c = -1.75$, la órbita inicializada en cero es atraída a un ciclo de longitud tres y no una potencia de dos (como podría suponerse). Además, cada uno de los tres grandes grupos en que se divide el gráfico por debajo de $c = -1.76$ presenta características similares al diagrama de órbitas original.

Para verificar las observaciones hechas anteriormente será necesario fabricar una función que permita hacer tablas de órbitas en cualquier lista del parámetro *c*:

```
Clear[función, orbita, tablaorbitas]
función[x_, c_] := x^2 + c
orbita[c_] :=
    Module[{x = Nest[función[#, c] &, 0, itermin]}, NestList[función[#, c] &, x, mostrar - 1]]
tablaorbitas[valoresdec_List] :=
     TableForm[
              Transpose[
                    Map[orbita, valoresdec]],
              TableHeadings → {Automatic, valoresdec},
              TableSpacing → {1, 2}]
```

Actividad 10:

a. Elabore una tabla para valores de $c = -1.41, -1.401, -1.40, -1.39$; comenzando con 2000 iteraciones y mostrando las 70 siguientes.

b. Construya una tabla para valores de $c = -1.7501, -1.75, -1.7499, -1.7498, -1.7497$; comenzando con 2000 iteraciones y mostrando las 20 siguientes.

Observación: Adviértase que la función *tablaorbitas* recibe una lista de valores.

La primera de las tablas anteriores muestra que para $c = -1.39$ la órbita de $x_0 = 0$ es atraída hacia un ciclo de longitud 8, para $c = -1.40$, la órbita se acerca a un ciclo de período 32, por lo que para algún $c \in (-1.4, -1.39)$, se puede suponer que se produce una bifurcación hacia un ciclo atractivo de longitud 16 y por supuesto habrá otro valor del parámetro *c* intermedio donde se producirá una bifurcación hacia un ciclo de longitud 32. Mientras que la información obtenida con $c = -1.401, -1.41$ no es concluyente. Esto se debe a que los valores de *c* se aproximan a un valor importante: el punto de Myrberg-Feigenbaum, localizado en -1.401 ... que describe el límite de un comportamiento regular.

En la segunda tabla se puede observar que para valores de *c* próximos a -1.75, efectivamente hay un cambio repentino, pues la órbita pasa a ser atraída hacia un ciclo de longitud tres.

Otra herramienta que ayuda a confirmar las observaciones hechas con anterioridad es la función *longitudcicloatractivo*, que no es más que una variante de la función *cicloatractivo* y que permite calcular la longitud mínima del ciclo o devuelve el valor cero si no hay ciclos atractivos.

```
Clear[longitudcicloatractivo, itermin, mostrar, precision]
longitudcicloatractivo[función_, c_] :=
     Module[
  {x1 = Nest[función[#, c] &, 0., itermin], n = 1, longitud = 1, x2}, x2 = función[x1, c];
       While[(Abs[x1 - x2] > precision) ⋀ (n++ ≤ mostrar),
                  x1 = función[x1, c]; x2 = función[función[x2, c], c]];
         If[n ≤ mostrar, (*ciclo de longitud≤n*)
              x2 = función[x1, c];
           While[(Abs[x1 - x2] > precision), longitud++; x2 = función[x2, c]]; longitud,
              (*else*)0]]
```

Actividad 11:

Con ayuda de la función *longitudcicloatractivo* corrobore las aseveraciones hechas en la actividad 10.

Una observación importante sobre la función *longitudcicloatractivo* es que se puede utilizar como alternativa al diagrama de órbitas al hacer gráficos mediante *Plot.* Es por esto que se plantea la actividad siguiente:

Actividad 12:

Realice los gráficos correspondientes a los diagramas de órbitas que se han estudiado hasta el momento e interprete los resultados.

2.2. Constantes de Feigenbaum

Investiguemosun poco más sobre las bifurcaciones y las constantes de Feigenbaum. Para ello vamos a tratar de conocer los puntos en el diagrama de órbitas en donde aparecen las bifurcaciones en las que se duplica el período y determinar a partir de ellos las constantes de Feigenbaum. Pero antes es necesario definir un parámetro llamado *multiplicador* que permite determinar cuando un punto fijo o un punto periódico es atractivo o repulsivo.

Si hay un ciclo $\{x_1, x_2, \ldots, x_n\}$ de longitud n al iterar la función f, entonces $f^n(x_n) = f(f(\ldots(f(x_k))\ldots)) = x_k$, para $1 \le k \le n$. Definiendo la función g como la n-ésima iterada de f, $g = f^n$, se tiene $g(x_k) = x_k$ para $1 \le k \le n$ y usando la regla de la cadena se obtiene la derivada de g que es la pendiente de la recta tangente en cada x_k:

$$g'(x_k) = f'(x_{k-1})\, f'(x_{k-2}) \ldots f'(x_1)\, f'(x_n)\, f'(x_{n-1}) \ldots f'(x_k)$$

es decir, g' toma el mismo valor, llamado *multiplicador*, en todos los puntos del ciclo. Puesto que $g'(x)$ es aproximadamente $\frac{g(z)-g(x)}{z-x}$ para z suficientemente cerca de x, y $g(x) = x$ para los puntos del ciclo, se deduce que el multiplicador es aproximadamente $\frac{g(z)-g(x)}{z-x}$ para z suficientementecerca de un punto x del ciclo. Si el multiplicador es en valor absoluto menor que uno y z está suficientementecerca de x, $g(z)$ está más próximo a x que lo que estaba z y el ciclo será atractivo, mientras que si el valor absoluto del multiplicador es mayor que uno, $g(z)$ estará más alejado que lo que estaba z y el ciclo será repulsivo.

De la discusión anterior se puede afirmar que los ciclos atractivos de longitud n están caracterizados porque cada punto del ciclo es fijo para la función $g = f^n$, y el multiplicador tiene valor absoluto menor que uno. Al dejar n fijo y variar la constante c, los puntos fijos de g, y por lo tanto los puntos de los ciclos de longitud n y el multiplicador correspondiente,varían suavemente y ciclos de longitud n dejan de ser atractivos cuando el multiplicador correspondiente pasa de valor absoluto menor que uno a mayor que uno. Entonces debemos buscar valores de n, c y x para los que la función g satisface $g(x) = x$ y $g'(x) = \pm 1$.

La primera tarea es encontrar los puntos fijos de las iteradas f^n. Por ejemplo si f es la función $x \to x^2 + c$, la función iterada se obtiene simplemente mediante **Nest**:

```
Clear[f, fn]
f[x_, c_] := x^2 + c (*la función cuadrática*)
fn[x_, c_, n_] := Nest[f[#, c] &, x, n] (*y n iteraciones de f*)
```

Dado que la composición duplica el exponente, g tendrá 2^n raíces (tal vez algunas complejas, tal vez otras con multiplicidad mayor que uno), por lo que elegiremos los puntos iniciales de acuerdo a cómo atraen los ciclos:

```
Clear[puntofijodefn, x]
puntofijodefn[c_, n_] :=
 x /. FindRoot[fn[x, c, n] == x, {x, fn[0., c, 200 + n]}](*Se hacen 200 iteraciones como mínimo*)
```

Veamos algunos resultados. Por ejemplo si $c = \frac{1}{4}$, se espera un punto fijo en $\frac{1}{2}$:

```
c = .25; puntofijodefn[c, 1]
```

```
0.5
```

y también esperamos un ciclo de longitud tres para el valor de $c = -1.75$:

```
c = -1.75; x = puntofijodefn[c, 3]
```

```
1.30194
```

La solución obtenida aproximadamente es correcta:

```
fn[%, c, 3] - %
```

$$-7.10543 \times 10^{-15}$$

Con diez iteraciones después del punto fijo vemos que los valores se repiten cada tres.

```
NestList[f[#, c] &, x, 10]
```

```
{1.30194, -0.0549581, -1.74698, 1.30194, -0.0549581,
 -1.74698, 1.30194, -0.0549581, -1.74698, 1.30194, -0.0549581}
```

Ahora que ya se sabe encontrar los puntos fijos dependiendode c, vamos a hacer hacer variar c para obtener aquellos donde el multiplicador tiene valor absoluto uno. Para esto se necesita una expresión para la derivada de $g = f^n$ en cada punto fijo (valor que no depende del punto del ciclo en que se evalúe), usando la regla de la cadena y dado que $f'(x) = 2x$ en este caso, se tiene:

```
multiplicador=.;
multiplicador[c_, n_] := 2^n Product[Nest[f[#, c] &, puntofijodefn[c, n], i], {i, n}]
```

Así, el multiplicador correspondienteal punto fijo antes encontrado cuando $c = -1.75$ y $n = 3$ es:

```
multiplicador[-1.75, 3]
```

```
0.999999
```

Mientras que cambiando un poco el valor de c:

```
multiplicador[-1.749, 3]
```

```
-6.10561
```

```
multiplicador[-1.7501, 3]
```

```
0.859592
```

Actividad 13:

Determine valores del parámetro c, para que el multiplicador valor absoluto igual a la unidad, para los ciclos de longitud: 1, 2, 4, 8, 16, 32, 64.

Para encontrar ahora los c's en los que el multiplicador para un ciclo de longitud n tiene un valor dado, usamos **FindRoot**:

```
Clear[c, bifurcacion]
bifurcacion[valor_, n_, c0_] := c /. FindRoot[multiplicador[c, n] == valor, {c, c0}]
```

```
(*Las versiones de Mathematica 5,
6 y 7 dan mensajes de error, pero los resultados son correctos.
 Se eliminan los mensajes con las siguientes instrucciones,
para volver a habilitarlos cambiar
 «Off» por «On»*)
Off[FindRoot::"srect"]
Off[ReplaceAll::"reps"]
```

Actividad 14:

Utilice la función *bifurcacion* para demostrar que: Si c_k es la constante donde se pasa del período de longitud 2^k a 2^{k+1}, los cocientes $\frac{c_{k+1}-c_k}{c_{k+2}-c_{k+1}}$ se aproximan a la constante 4.66920 ...

Actividad 15:

Observar que por debajo de -1.75 en los diagramas de órbitas el período pasa de 3, a 6, a 12, etc. Estudiar los puntos donde ocurren estos cambios de período y ver si la constante de Feigenbaum está presente.
Verificar que la constante de Feigenbaum es universal, o sea válida para toda función suave en la que se dupliquen los períodos analizando la función logística.

Actividad 16:

Realizar un estudio análogo al realizado para la familia de funciones $x \to d\,\mathrm{Sin}[\pi x]$, donde $x \in [0, 1]$, y $d \in [0, 1]$, construyendo un gráfico del diagrama de órbitas del punto $x_0 = 0.5$ cuando varía el parámetro d. ¿Esta presente la constante de Feigenbaum para este sistema?

2.3. Teoremas de Brouwer, Li y Yorke y Sarkovsky

En esta sección se presentan algunos de los teoremas más importantes de la Teoría de Sistemas Dinámicos, entre los cuales se encuentra un teorema fundamental en dicha teoría debido a A. N. Sarkovsky, el cual trata sobre la existencia de puntos periódicos de diferentes periodos e implicaciones entre ellos.

Teorema de Brouwer en una dimensión

Si $f : [a, b] \to [a, b]$ es una función continua entonces existe un x tal que $f(x) = x$.

Sugerencia: Para la demostración del teorema de Brouwer, considerese la función $g(x) = f(x) - x$, y aplicar el teorema de Bolzano.

Antes de demostrar el teorema de Li y Yorke es necesario demostrar varios lemas, los cuales se enuncian a continuación.

Lema 1

Sea $X \subset \mathbb{R}$ un intervalo y $f : X \to X$. Si el sistema dinámico (X, f) tiene un punto periódico de periodo tres, entonces existe un punto $a \in X$ para el que se cumple que $f^3(a) = a < f(a) < f^2(a)$ ó $f^2(a) < f(a) < a = f^3(a)$.

Lema 2

Sea $I \subset \mathbb{R}$ un intervalo y $g : I \to \mathbb{R}$ una función continua. Para cualquier intervalo compacto $I_1 \subset g(I)$ existe un intervalo compacto $Q \subset I$ tal que $g(Q) = I_1$.

Lema 3

Sea J un intervalo, $f : J \to J$ una aplicación continua, e $\{I_n\}_{n=0}^{\infty}$ una sucesión de intervalos compactos con $I_n \subset J$ e $I_{n+1} \subset f(I_n)$, para todo $n \geqslant 0$. Entonces existe una sucesión de intervalos compactos $\{Q_n\}_{n=0}^{\infty}$ tal que, para todo $n \geqslant 0$, se cumple que

- **a.** $Q_{n+1} \subset Q_n \subset I_0$.
- **b.** $f^n(Q_n) = I_n$.
- **c.** $f^n(Q) \subset I_n$, donde $Q = \bigcap_{n=0}^{\infty} Q_n$.

Lema 4

Sean $J \subset \mathbb{R}$ un intervalo, $g : J \to \mathbb{R}$ una función continua, e $I \subset J$ un intervalo compacto tal que $I \subset g(I)$. Entonces existe $p \in I$ tal que $g(p) = p$.

Una vez vistos estos cuatro lemas, ya se tienen las condiciones para demostrar el teorema de Li & Yorke, el cual se enuncia a continuación.

Teorema de Li & Yorke.

Sea (X, f) un sistema dinámico, $X \subset \mathbb{R}$ un intervalo, que tiene un punto periódico de periodo tres. Entonces, el sistema dinámico (X, f) tiene puntos periódicos de todos los periodos posibles; es decir, (X, f) tiene algún punto periódico de periodo k, para todo $k \geqslant 1$.

Actividad 1:

Considerese el sistema dinámico $([1, 5], f)$ tal que la imagen de f es una linea poligonal, mientras que $f(1) = 3$, $f(2) = 5$, $f(3) = 4$, $f(4) = 2$, $f(5) = 1$. Compruebe con las indicaciones que se sugieren, lo siguiente:

1. f tiene puntos periódicos de periodo 5 (por ejemplo $x = 1$).

2. f no tiene puntos periódicos de periodo 3, para lo que debe que probarse que f^3 no tiene puntos fijos que no lo sean de f o f^2. Para ello pruebe lo siguiente:

- **a.** Viendo las sucesivas transformadas de cada uno de los subintervalos ($[1, 2]$, $[2, 3]$, $[3, 4]$ y $[4, 5]$), probar que f^3 no puede tener puntos fijos salvo, quizás, en $[3, 4]$.
- **b.** Probar que f^3 puede tener un punto fijo $p \in [3, 4]$, pero para ello es necesario que p, $f(p)$ y $f^2(p)$ estén todos en $[3, 4]$.
- **c.** Encontrar la fórmula de $f^3(x)$ en el caso de que x, $f(x)$ y $f^2(x)$ estén todos en $[3, 4]$. Comprobar analíticamente que la función obtenida tiene un único punto fijo en dicho intervalo.
- **d.** Comprobar que el único punto fijo de f^3, obtenido en el literal anterior, lo es también de f; luego, en realidad, no es un punto periódico de periodo tres sino un punto fijo de f.

Teorema de Sarkovsky

En el año 1964, el matemático ruso Alexander N. Sarkovsky hizo pública la siguiente ordenación de los números positivos:

$$
\begin{array}{l}
\quad 3 \rightarrow \quad 5 \rightarrow \quad 7 \rightarrow \quad 9 \rightarrow \quad 11 \rightarrow \cdots \rightarrow \\
\rightarrow 2 \cdot 3 \rightarrow 2 \cdot 5 \rightarrow 2 \cdot 7 \rightarrow 2 \cdot 9 \rightarrow 2 \cdot 11 \rightarrow \cdots \rightarrow \\
\rightarrow 2^2 \cdot 3 \rightarrow 2^2 \cdot 5 \rightarrow 2^2 \cdot 7 \rightarrow 2^2 \cdot 9 \rightarrow 2^2 \cdot 11 \rightarrow \cdots \rightarrow \\
\rightarrow 2^3 \cdot 3 \rightarrow 2^3 \cdot 5 \rightarrow 2^3 \cdot 7 \rightarrow 2^3 \cdot 9 \rightarrow 2^3 \cdot 11 \rightarrow \cdots \rightarrow \\
\vdots \\
\rightarrow 2^n \cdot 3 \rightarrow 2^n \cdot 5 \rightarrow 2^n \cdot 7 \rightarrow 2^n \cdot 9 \rightarrow 2^n \cdot 11 \rightarrow \cdots \rightarrow \\
\vdots \\
\rightarrow \cdots \rightarrow \quad 2^5 \rightarrow \quad 2^4 \rightarrow \quad 2^3 \rightarrow \quad 2^2 \rightarrow \quad 2 \rightarrow \quad 1
\end{array}
$$

donde, en la primera fila están los números impares, en la segunda el producto de dos por los impares, en la tercera el producto del cuadrado de dos por los impares, y asi sucesivamentehasta la última fila que contiene todas las potencias de dos ordenadas en orden decreciente.

Además, Sarkovsky probó el siguiente resultado:

Sea $X \subset \mathbb{R}$ un intervalo y $f : X \to X$ una función continua. Si el sistema dinámico (X, f) tiene un punto periódico de periodo k, entonces también tiene puntos periódicos de periodo m, para cualquier m con $k \rightarrow m$.

Es claro que el teorema de Li & Yorke, es una consecuencia inmediata del teorema de Sarkovsky, empero, la demostración de este teorema es muy técnica y no se corresponde con el nivel de contenidos que aparecen en este trabajo de investigación, razón por la cual se propuso demostrar el teorema de Li & Yorke de manera independiente.

Asímismo, si un sistema dinámico, con las características del teorema de Sarkovsky, tiene un punto periódico de periodo $k \cdot 2^m$, con $k \neq 1$ impar y $m \geq 0$, entonces tiene puntos periódicos de infinitos periodos.

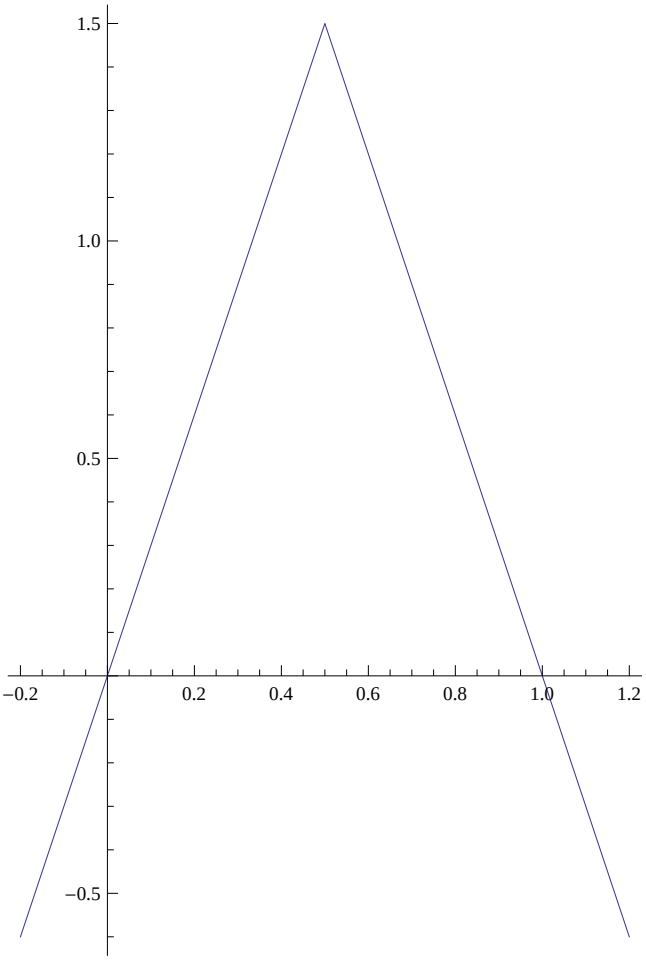
2.4. Sistemas Dinámicos y el Conjunto de Cantor

Para terminar este capítulo se pondrá de manifiesto, en esta sección la íntima relación que existe entre los sistemas dinámicos y los fractales. Esto se realizará por medio del estudio del comportamientodinámico de una función que no es suave, como por ejemplo la función *f* cuya regla de correspondenciaes

```
f = .
f[x_] := 3 x  /; x ≤ 1 / 2
f[x_] := 3 - 3 x  /; x > 1 / 2
```

Esta función tiene un gráfico en forma de "carpa":

```
Plot[f[x], {x, -.2, 1.2}, AspectRatio → Automatic]
```



El gráfico de *f* tiene similitudes con los gráficos de las funciones $x \to c\,x\,(1-x)$ para valores de $c > 0$, y no es de extrañar que las órbitas tengan comportamientos comparables, por lo que se propone la siguiente actividad:

Actividad 1:

Emplee la función *trayectoria* al realizar un análisis gráfico para distintos valores iniciales.

Del análisis gráfico surge que las órbitas que se inician con $x > 1$ o con $x < 0$ tomarán valores cada vez más negativos, mientras que cuando x está entre 0 y 1 a veces la órbita se mantiene acotada y a veces "se escapa".

Analizando un poco más la definición de f y los gráficos, se observa que si $\frac{1}{3} < x < \frac{2}{3}$ entonces $f(x) > 1$, y por lo tanto su órbita "se escapará". Esto hace pensar que se puede desestimar este intervalo, y lo que hara recordar a los que esten familiarizados con los Fractales, la construcción del conjunto ternario de Cantor: ¿puede seguirse sacando los intervalos intermedios?

En efecto. Observese primero que por la simetría basta estudiar el caso $x < \frac{1}{2}$. Ahora si $\frac{1}{9} < x < \frac{2}{9}$ entonces $\frac{1}{3} < f(x) = 3x < \frac{2}{3}$ por lo que las órbitas de x cuando $\frac{1}{9} < x < \frac{2}{9}$ o $\frac{7}{9} < x < \frac{8}{9}$ también "se escapan". En fin, para corroborar las afirmaciones hechas, si $\frac{1}{27} < x < \frac{2}{27}$ será $\frac{1}{9} < f(x) = 3x < \frac{2}{9}$ mientras que si $\frac{7}{27} < x < \frac{8}{27}$ será $\frac{7}{9} < f(x) = 3x < \frac{8}{9}$, y el análisis continúa de la misma forma.

En conclusión, aunque no se ha demostrado rigurosamente, se puede decir que salvo en el conjunto de Cantor las órbitas se escapan. Empero, ¿qué pasa en el conjunto de Cantor?, ¿cuáles son los puntos fijos?, ¿hay ciclos?

Estudiando puntos de la forma $\frac{1}{27} < x < \frac{2}{27}$ será $\frac{1}{3}$, $\frac{1}{9}$, etc., que son los puntos más sencillos en el ternario de Cantor, se observa que

```
NestList[f, 1 / 3, 3]
```

$$\left\{\frac{1}{3}, 1, 0, 0\right\}$$

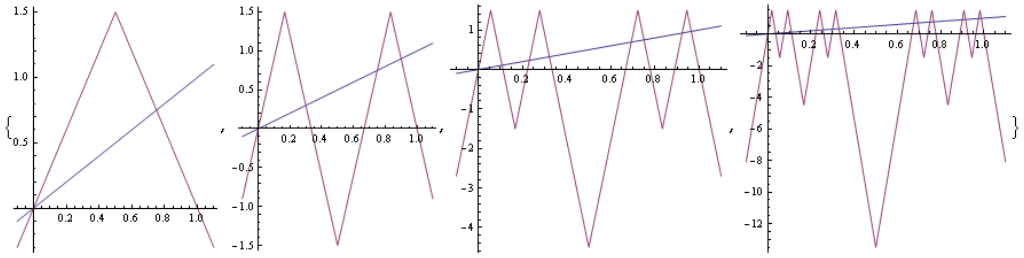
```
NestList[f, 1 / 9, 3]
```

$$\left\{\frac{1}{9}, \frac{1}{3}, 1, 0\right\}$$

y a partir de la definición de f se demuestra que si $x = 3^{-n}$, será $f^n(x) = 1$ y por lo tanto $f^m(x) = 0$ para $m > n$. De la misma forma, cualquier punto extremo de los intervalos que se van extrayendo para construir el ternario de Cantor tiene una órbita que eventualmente llega al punto fijo $x = 0$.

Para continuar con el estudio de otros puntos es conveniente analizar las iteradas f^2, f^3, f^4, etc. Para lo cual se elaboran los gráficos de cada una utilizando *Mathematica* como sigue:

```
Table[Plot[{x, Nest[f, x, n]}, {x, -.1, 1.1}, AspectRatio → Full], {n, 1, 4}]
```



En los gráficos anteriores se observa que hay dos puntos fijos para f (que son 0 y $\frac{3}{4}$), cuatro para f^2 (los cuales son 0, $\frac{3}{4}$, $\frac{3}{10}$ y $\frac{9}{10}$), ocho para f^3, y dieciséis para f^4. Así de la recurrencia anterior se infiere que en general habrá 2^n puntos fijos para f^n. Por supuesto, muchos de estos puntos (¡pero no todos!) podrán ser puntos fijos para f^m para algunos m con $m < n$, por ejemplo los puntos fijos de f y f^2 son puntos fijos de f^4 pero de cualquier manera habrán ciclos de longitud arbitrariamente grande. También puede observarse en los gráficos que las pendientes de las rectas tangentes en los puntos fijos tienen valores absolutos mayores que uno (recordar que los gráficos están distorsionados, la recta tiene pendiente uno), por lo que los ciclos resultantes son repulsivos.

Actividad 2:

Verifique que las trayectorias de los puntos $\frac{3}{28}$ y $\frac{3}{82}$ forman ciclos y determinar las longitudes mínimas de estos ciclos. Estudiar puntos cercanos (perturbaciones) para comprobar que los ciclos son repulsivos. Sugerencia: Utilice **NestList** o la función *cicloatractivo*.

Para terminar, es importante señalar que el comportamiento dinámico de esta función es similar al de la función $x \to 4x(1 - x)$: el conjunto de $x's$ cuya órbita está acotada es un conjunto con propiedades similares al conjunto de Cantor y tiene ciclos de longitud arbitrariamente grande, ciclos que también son repulsivos.

2.5. Soluciones a actividades de la sección 2.1

Solución actividad 2:

De los gráficos anteriores, se puede concluir que para un valor de c entre cero y uno, la parábola $y = x^2 + c$ será tangente a la recta identidad. Lo cual sucederá cuando haya un único punto fijo de la transformación $x \to x^2 + c$. Más aún, observando el gráfico anterior se concluye que dicho valor del parámetro es $c = 0.25$.

Solución actividad 4:

Como resultado de esta actividad es que para $c > \frac{1}{4}$, todas las órbitas parecen "escaparse a infinito", para $-2 \le c < \frac{1}{4}$, hay un intervalo de valores de x's que tienen órbitas acotadas, mientras que para $c < -2$, nuevamente todas las órbitas parecen "escaparse a infinito", aunque en este último caso se puede afirmar que al menos los puntos fijos y sus preimágenes tiene órbitas acotadas.

Solución actividad 5:

De acuerdo con el trabajo realizado hasta el momento se puede concluir que: basta estudiar las órbitas cuyos valores iniciales están entre $-p$ y p, y sólo para los valores del parámetro menores que un cuarto.

Solución actividad 6:

De los resultados de esta actividad se puede afirmar que el valor del parámetro $c = -2$ (al igual que $c = \frac{1}{4}$) también es especial puesto que ocurre una bifurcación y corresponde (vía los cambios de variables que se hicieron al inicio) al caso de la función $x \to 4\,x\,(1 - x)$, para el que ya se ha mencionado que el conjunto de valores iniciales cuyas órbitas convergen es el conjunto de Cantor. Por consiguiente, es necesario hacer un estudio más detallado cuando el parámetro c varía entre -2 y $\frac{1}{4}$.

Solución actividad 7:

Como resultado de la actividad anterior, se puede concluir que para los valores del parámetro entre 0.25 y -0.75 (aproximadamente)y para el valor inicial cero, cada órbita converge a un punto fijo, y por ende éste es atractivo. En el intervalo de -0.75 y -1.25, ocurre una bifurcación: ahora la órbita es atraída por un ciclo de longitud dos; a partir de -0.75 hasta -1.25, ocurre una nueva bifurcación, ya que la órbita es atraída a un ciclo de período cuatro. Finalmente, aunque no se puede distinguir claramente debido a la resolución con la que se está trabajando, se aprecia que poco después el comportamientoes complicado para llegar a un comportamientosencillo alrededor de -1.75.

Solución actividad 8:

Del trabajo realizado en esta actividad se puede concluir que después de la bifurcación hacia un ciclo de longitud cuatro cerca de -1.25, hay al menos otra bifurcación, para ello obsérvese que cerca de -1.37 surge un ciclo de longitud ocho. También, se confirma que alrededor de -1.75 se ve claramente como "laguna", y que surgen otras "lagunas" más reducidas en ancho dispersas por todo el diagrama de órbitas.

Solución actividad 10:

```
itermin = 2000; mostrar = 70;
tablaorbitas[{-1.41, -1.401, -1.40, -1.39}]
```

	−1.41	−1.401	−1.4	−1.39
1	−0.0605742	−0.033919	−0.0232501	0.0237125
2	−1.40633	−1.39985	−1.39946	−1.38944
3	0.567766	0.558579	0.558487	0.540537
4	−1.08764	−1.08899	−1.08809	−1.09782
5	−0.227036	−0.215101	−0.216055	−0.184792
6	−1.35845	−1.35473	−1.35332	−1.35585
7	0.435399	0.434298	0.431476	0.448334
8	−1.22043	−1.21239	−1.21383	−1.189
9	0.0794436	0.0688787	0.073379	0.0237125
10	−1.40369	−1.39626	−1.39462	−1.38944
11	0.560342	0.54853	0.544952	0.540537
12	−1.09602	−1.10011	−1.10303	−1.09782
13	−0.208747	−0.190747	−0.183332	−0.184792
14	−1.36642	−1.36462	−1.36639	−1.35585
15	0.457116	0.461175	0.46702	0.448334
16	−1.20104	−1.18832	−1.18189	−1.189
17	0.0325082	0.0110983	−0.00313082	0.0237125
18	−1.40894	−1.40088	−1.39999	−1.38944
19	0.575121	0.561456	0.559973	0.540537
20	−1.07924	−1.08577	−1.08643	−1.09782
21	−0.24525	−0.222109	−0.219668	−0.184792
22	−1.34985	−1.35167	−1.35175	−1.35585
23	0.412102	0.426005	0.427217	0.448334
24	−1.24017	−1.21952	−1.21749	−1.189
25	0.128027	0.0862288	0.0822715	0.0237125
26	−1.39361	−1.39356	−1.39323	−1.38944
27	0.532146	0.541022	0.541094	0.540537
28	−1.12682	−1.10829	−1.10722	−1.09782
29	−0.140276	−0.172682	−0.174069	−0.184792
30	−1.39032	−1.37118	−1.3697	−1.35585
31	0.522997	0.479137	0.476078	0.448334
32	−1.13647	−1.17143	−1.17335	−1.189
33	−0.118427	−0.0287565	−0.0232501	0.0237125
34	−1.39597	−1.40017	−1.39946	−1.38944
35	0.538746	0.559485	0.558487	0.540537
36	−1.11975	−1.08798	−1.08809	−1.09782
37	−0.156154	−0.217306	−0.216055	−0.184792
38	−1.38562	−1.35378	−1.35332	−1.35585
39	0.509931	0.431715	0.431476	0.448334
40	−1.14997	−1.21462	−1.21383	−1.189
41	−0.0875692	0.074307	0.073379	0.0237125
42	−1.40233	−1.39548	−1.39462	−1.38944
43	0.556534	0.54636	0.544952	0.540537
44	−1.10027	−1.10249	−1.10303	−1.09782

45	−0.199406	−0.185515	−0.183332	−0.184792
46	−1.37024	−1.36658	−1.36639	−1.35585
47	0.46755	0.466553	0.46702	0.448334
48	−1.1914	−1.18333	−1.18189	−1.189
49	0.00942689	−0.000733652	−0.00313082	0.0237125
50	−1.40991	−1.401	−1.39999	−1.38944
51	0.577849	0.561799	0.559973	0.540537
52	−1.07609	−1.08538	−1.08643	−1.09782
53	−0.25203	−0.222947	−0.219668	−0.184792
54	−1.34648	−1.35129	−1.35175	−1.35585
55	0.403011	0.424997	0.427217	0.448334
56	−1.24758	−1.22038	−1.21749	−1.189
57	0.146462	0.0883219	0.0822715	0.0237125
58	−1.38855	−1.3932	−1.39323	−1.38944
59	0.518068	0.540004	0.541094	0.540537
60	−1.14161	−1.1094	−1.10722	−1.09782
61	−0.106737	−0.170242	−0.174069	−0.184792
62	−1.39861	−1.37202	−1.3697	−1.35585
63	0.546102	0.481433	0.476078	0.448334
64	−1.11177	−1.16922	−1.17335	−1.189
65	−0.173963	−0.033919	−0.0232501	0.0237125
66	−1.37974	−1.39985	−1.39946	−1.38944
67	0.493674	0.558579	0.558487	0.540537
68	−1.16629	−1.08899	−1.08809	−1.09782
69	−0.0497773	−0.215101	−0.216055	−0.184792
70	−1.40752	−1.35473	−1.35332	−1.35585

```
itermin = 2000; mostrar = 20;
tablaorbitas[{-1.7501, -1.75, -1.7499, -1.7498, -1.7497}]
```

	−1.7501	−1.75	−1.7499	−1.7498	−1.7497
1	1.30499	1.302	−0.62204	−0.873582	−1.74706
2	−0.0471069	−0.0547922	−1.36297	−0.986655	1.30252
3	−1.74788	−1.747	0.107777	−0.776311	−0.0531457
4	1.30499	1.302	−1.73828	−1.14714	−1.74688
5	−0.0471069	−0.0547924	1.27173	−0.433869	1.30187
6	−1.74788	−1.747	−0.132599	−1.56156	−0.0548237
7	1.30499	1.302	−1.73232	0.688664	−1.74669
8	−0.0471069	−0.0547926	1.25102	−1.27554	1.30124
9	−1.74788	−1.747	−0.184838	−0.122792	−0.0564714
10	1.30499	1.302	−1.71573	−1.73472	−1.74651
11	−0.0471069	−0.0547929	1.19385	1.25946	1.3006
12	−1.74788	−1.747	−0.324632	−0.163558	−0.0581381
13	1.30499	1.302	−1.64451	−1.72305	−1.74632
14	−0.0471069	−0.0547931	0.954527	1.2191	1.29993
15	−1.74788	−1.747	−0.838779	−0.263602	−0.0598731
16	1.30499	1.302	−1.04635	−1.68031	−1.74612
17	−0.0471069	−0.0547934	−0.655051	1.07366	1.29922
18	−1.74788	−1.747	−1.32081	−0.597064	−0.0617317
19	1.30499	1.302	−0.0053657	−1.39331	−1.74589
20	−0.0471069	−0.0547936	−1.74987	0.191526	1.29843

2.6. Soluciones a actividades de la sección 2.2

Solución actividad 13:

Como ya se observó anteriormente (al estudiar el diagrama de órbitas), el valor del parámetro para el cual se pasa de un punto fijo a un ciclo de longitud dos es aproximadamente $c = -.75$

```
m = 1; c0 = -.76; cmax = -.74;
{multiplicador[c0, m], multiplicador[cmax, m]}
```

```
{-1.00998, -0.989975}
```

De lo cual se espera que el multiplicador sea -1 para $c = -.75$, como se comprueba a continuación:

```
m = 1; c0 = -.75;
multiplicador[c0, m]
```

```
-1.
```

Realizando un análisis similar para otras potencias de dos se obtiene:

```
m = 2; c0 = -1.26;
multiplicador[c0, m]
```

```
-1.04
```

```
m = 4; c0 = -1.37;
multiplicador[c0, m]
```

```
-1.03405
```

```
m = 8; c0 = -1.395;
multiplicador[c0, m]
```

```
-1.07927
```

```
m = 16; c0 = -1.3996;
multiplicador[c0, m]
```

```
-0.987939
```

```
m = 32; c0 = -1.4009;
multiplicador[c0, m]
```

```
-1.12906
```

```
m = 64; c0 = -1.40083;
multiplicador[c0, m]
```

```
0.990928
```

Así se puede inferir que los puntos en donde se duplica el período convergen hacia la constante -1.401 ...

Solución actividad 14:

Estudiando ahora las bifurcaciones en potencias de dos, para los valores del parámetro que se obtuvieron en la actividad anterior, se tiene:

```
m = 1; c0 = -.74;
c1a2 = bifurcacion[-1, m, c0]
```

```
-0.75
```

```
m = 2; c0 = -1.26;
c2a4 = bifurcacion[-1, m, c0]
```

```
-1.25
```

```
m = 4; c0 = -1.37;
c4a8 = bifurcacion[-1, m, c0]
```

```
-1.3681
```

```
m = 8; c0 = -1.395;
c8a16 = bifurcacion[-1, m, c0]
```

```
-1.39405
```

```
m = 16; c0 = -1.3996;
c16a32 = bifurcacion[-1, m, c0]
```

```
-1.39963
```

```
m = 32; c0 = -1.4009;
c32a64 = bifurcacion[-1, m, c0]
```

```
-1.40083
```

```
m = 64; c0 = -1.40083;
c64a128 = bifurcacion[-1, m, c0]
```

```
-1.40109
```

Finalmente se calculan los cocientes:

```
(c2a4 - c1a2) / (c4a8 - c2a4)
```

```
4.23374
```

```
(c4a8 - c2a4) / (c8a16 - c4a8)
```

```
4.55151
```

```
(c8a16 - c4a8) / (c16a32 - c8a16)
```

```
4.64581
```

```
(c16a32 - c8a16) / (c32a64 - c16a32)
```

```
4.66394
```

```
calculado = (c32a64 - c16a32) / (c64a128 - c32a64)
```

```
4.6681
```

Se comprueba pues que los cocientes convergen hacia la constante de Feigenbaum, la que se ha aproximado con un error menor que el 0.03%, como se observa a continuación:

```
valorreal = 4.669201660910; Abs[calculado - valorreal] / valorreal * 100 (*Error porcentual*)
```

```
0.0235104
```

3
Fractales Clásicos

3.1. Fractales Geométricos

Aquí se iniciará el estudio de los llamados Fractales Geométricos, obtenidos mediante "infinitas" iteraciones de construcciones geométricas. Históricamente, el primer fractal es el conjunto ternario de Cantor, seguido por las curvas de Peano, curvas que aunque no estrictamente fractales (puesto que tienen dimensión dos), son antecesoras a su vez de las construcciones de Sierpinski y *Koch*, que sí son fractales.

Un ejemplo importante:

Los razonamientos que se realizan cuando se estudian los Fractales deben realizarse con cuidado, ya que por ejemplo, en general no es cierto que si una sucesión de curvas converge hacia otra, entonces las longitudes converjan a la longitud de dicha curva. Considere el siguiente ejemplo:

```
f = .
f[x_] := x /; 0 ≤ x ≤ 1/2
f[x_] := 1 - x /; 1/2 < x ≤ 1
f[x_] := f[Mod[x, 1]]
Manipulate[Plot[f[2^n x] / 2^n, {x, 0, 1}, PlotRange → {{0, 1}, {0, .5}}],
          {{n, 0, "Iteraciones"}, 0, 7, 1, Appearance → "Open"}, SaveDefinitions → True]
```

Otra forma de definir una función seccionada es:

```
f = .; f[x_] := Piecewise[{{x, 0 ≤ x ≤ 1 / 2}, {1 - x, 1 / 2 < x ≤ 1}, {f[Mod[x, 1]], x < 0 || 1 < x}}]
```

Así aunque las curvas obtenidas son "autosemejantes" y siempre tienen longitud raíz cuadrada de dos, la curva límite es simplemente el segmento [0, 1], el cual tiene longitud uno y no es semejante en ningún sentido a las curvas anteriores.

Surge pues la interrogante referente a como determinar si un objeto dado es un fractal o no. Uno de los parámetros es la denominada Dimensión de Hausdorff.

3.1.1. Dibujando Árboles.

Una de las observaciones de Mandelbrot es que muchas de las formas en la Naturaleza muestran repeticiones en cada vez menor escala. Es lo que se puede ver, un tanto idealmente, en las ramificaciones de los árboles. No, los árboles no son fractales, pero las ideas que se desarrollan a continuación son una primera aproximación al concepto de fractal.

Para dibujar un árbol se debe dibujar cada segmento, así usando en *Mathematica* la función **Graphics** se obtiene

```
Clear[pt1, pt2]
pt1 = {0, 0}; pt2 = {1, 1};
Graphics[Line[{pt1, pt2}]]
```

A este segmento se la aplica una transformación lineal (en el plano) a fin de rotarlo hacia la derecha o izquierda, para lo cual se emplea la función **Rotation-Transform**.

```
pt3 = .; pt3 = RotationTransform[-π / 2][pt2];
Graphics[Line[{pt1, pt3}]]
```

A fin de imitar una rama que se bifurca, se deben combinar los efectos de rotar el segmento a la derecha y a la izquierda, desplazar los resultados de modo que coincidan las puntas con la del original, y después mostrar los tres segmentos juntos. Por ejemplo, considere el vector (1,0). Después de rotarlo 45° hacia la derecha e izquierda se obtienen vectores que todavía tienen como extremo inicial al origen.

```
Clear[a, b, izquierda, derecha]
a = {0, 0}; b = {1, 0};
izquierda = RotationTransform[N[Pi / 4]][b]
derecha = RotationTransform[N[-Pi / 4]][b]
```

Gráficamente se tiene:

```
Graphics[{Line[{a, b}], Line[{a, izquierda}], Line[{a, derecha}]}]
```

Por lo que es necesario desplazar los segmentos obtenidos mediante las rotaciones hacia el punto *b*.

```
Clear[seg1, seg2, seg3]
seg1 = Line[{a, b}];
seg2 = Line[{b, b + izquierda}];
seg3 = Line[{b, b + derecha}];
Graphics[{seg1, seg2, seg3}]
```

Para dibujar una rama vertical, debe comenzarse con el vector (0,1). Además, con el objeto de hacer las ramas más cortas que la original, se introduce un factor de contracción (una homotecia de razón menor que la unidad) inmediatamente después de la rotación y antes de la traslación:

```
Clear[a, b, contraccion, c, d, seg1, seg2, seg3]
a = {0, 0}; b = {0, 1};
contraccion = .6;
c = contraccion RotationTransform[N[Pi / 4]][b];
d = contraccion RotationTransform[-N[Pi / 4]][b];
seg1 = Line[{a, b}];
seg2 = Line[{b, b + c}];
seg3 = Line[{b, b + d}];
Graphics[{seg1, seg2, seg3}]
```

Ahora bien, para imitar un árbol es necesario combinar las transformaciones: rotación, homotecia y traslación, e iterar varias veces. Así, en *Mathematica* se comienza iterando la función **Line[a,b]** que simboliza al segmento en dos dimensiones que une los puntos *a* y *b* del plano.

El proceso de ramificación puede describirse como el de reemplazar el segmento con punto inicial en *a* y punto final en *b*, por dos segmentos como se indica a continuación:

```
Clear[ramificar, factor, angulo, v]
ramificar[Line[{a_, b_}]] :=
    {v = N[factor] (b - a); Line[{b, b + RotationTransform[-N[angulo]][v]}],
    Line[{b, b + RotationTransform[N[angulo]][v]}]}
ramificar[x_List] := Map[ramificar, Flatten[x]](*Se evalúa la función solo en listas*)
```

Así en el ejemplo anterior, se tiene que:

```
Clear[factor, angulo]
factor = 0.6;
angulo = 45 °;
ramificar[Line[{{0, 0}, {0, 1}}]]
```

Se tiene pues la función que genera una bifurcación de una rama, pero no elimina la base de la rama y sólo realiza una bifurcación, por lo que es necesario utilizar la función *NestList*. Por ejemplo, al aplicar dos veces la función *ramificar* a **Line[{{0, 0}, {0, 1}}]** se obtiene sucesivamente:

```
NestList[ramificar, Line[{{0, 0}, {1, 0}}], 0]
```

```
NestList[ramificar, Line[{{0, 0}, {1, 0}}], 1]
```

```
NestList[ramificar, Line[{{0, 0}, {1, 0}}], 2]
```

Por supuesto lo que se desea es dibujar un árbol hasta el "el nivel" deseado, por lo cual se deben usar las funciones *Graphics* y *Manipulate*.

```
Manipulate[angulo; factor;
          Graphics[
                    NestList[ramificar, Line[{{0, 0}, {0, 1}}], m]],
          {{m, 2, "Iteraciones"}, 1, 8, 1, Appearance → "Labeled"},
         {{angulo, 40 °, "Rotación"}, 0 °, 90 °, 5 °, Appearance → "Labeled"},
         {{factor, .5, "Contracción"}, 0, .9, .1, Appearance → "Labeled"},
          SaveDefinitions → True]
```

Otra solución es rotar puntos alrededor de puntos.

```
Clear[a, b, contraccion, c, d, seg1, seg2, seg3]
a = {0, 0}; b = {0, 1};
contraccion = .6;
c = RotationTransform[N[3 π / 4], b][(1 - contraccion) b];
d = RotationTransform[N[- 3 π / 4], b][(1 - contraccion) b];
seg1 = Line[{a, b}];
seg2 = Line[{b, c}];
seg3 = Line[{b, d}];
Graphics[{seg1, seg2, seg3}]
```

```
Clear[ramificar, factor, angulo, p]
ramificar[Line[{a_, b_}]] :=
    {p = b + N[factor] (a - b); Line[{b, RotationTransform[π - N[angulo], b][p]}],
    Line[{b, RotationTransform[N[angulo] - π, b][p]}]}
ramificar[x_List] := Map[ramificar, Flatten[x]](*Se evalúa la función solo en listas*)
```

```
Manipulate[angulo; factor;
          Graphics[
                    NestList[ramificar, Line[{{0, 0}, {0, 1}}], m]],
          {{m, 2, "Iteraciones"}, 1, 8, 1, Appearance → "Labeled"},
         {{angulo, 40 °, "Rotación"}, 0 °, 90 °, 5 °, Appearance → "Labeled"},
         {{factor, .5, "Contracción"}, 0, .9, .1, Appearance → "Labeled"},
          SaveDefinitions → True]
```

Actividad 1:

Modificar la función *ramificar* para obtener árboles que se ramifiquen en tres en vez de dos en cada iteración.

Observación: En adelante se aplicaran las ideas que se han desarrollado para obtener aproximaciones gráficas a objetos que sí son fractales.

Estructura básica de los códigos creados por el usuario.

Aunque para construir las funciones que permiten general los fractales geométricos que se estudian a continuación basta modificar la función *ramificar*, aquí se presenta la estructura general de los códigos que se deben crear, donde "*función*" es el nombre de la función diseñada por el usuario, "*vector*" es el vector director en la dirección del vector "*ab*", y "**Line**" , puede ser reemplazada por "*Polygon, Rectangle, Cuboid*, etc." , mientras que "*puntoinicial*" y "*puntofinal*" pueden ser pares ordenados o ternas ordenadas según el objeto geométrico en consideración.

La estructura del código para crear funciones es la siguiente:

```
función =.
función[Line[{a_, b_}]] :=

 (*Donde r (que representa el factor de contracción) y el ángulo de rotación son fijos*)
       {vector = r (b - a); c = Traslacióndelpuntoa + Rotacióndelvertor;
  d = Traslacióndelpuntoc + Rotacióndelvertor;
         Line[{a, c}], Line[{c, d}], Line[{d, b}]}
función[x_List] := función /@ Flatten[x]
```

Para poder visualizar los gráficos que se construyen con la función antes definida, se necesitara el código siguiente:

```
n =.
Manipulate[
    Graphics[
        Nest[función, Line[{puntoinicial, puntofinal}], n]],
    {{n, 2, "Rótulo del número de iteraciones"}, 0, 6, 1},
    SaveDefinitions → True]
```

3.1.2. Curva de Koch.

Al dibujar árboles se utilizó la idea de agregar a un segmento a otros dos formados a partir de él, o visto de otra forma, se reemplaza al segmento por otros dos obtenidos a partir del original. Una variante de esta idea es reemplazar un segmento por un conjunto de segmentos obtenidos a partir de él, y así "hasta el infinito"

Una de las curvas más elementales que se obtienen por este procedimiento es la denominada curva de Koch (en honor al Matemático sueco Niels Fabian Helge Von Koch) que consiste en dividir un segmento en tres partes iguales, quitar la del medio y reemplazarla por un triángulo equilátero (sin su base), quedarse con los dos tercios extremos, y repetir indefinidamente esta construcción.

Actividad 2:

Para visualizar la idea antes expuesta grafique el segmento que une en la recta real los números -1 y 1, utilizando *Mathematica.* Luego, genere la figura básica que permitirá formar la curva de Koch, dibujando una poligonal que con los puntos: $(-1, 0)$, $\left(-\frac{1}{3}, 0\right)$, $\left(0, \frac{\sqrt{3}}{3}\right)$, $\left(\frac{1}{3}, 0\right)$, $(1, 0)$.

Actividad 3:

Modifique la función *ramificar* para definir la función *koch*, e itérela hasta cinco veces. **Sugerencia:** Reemplace la función *NestList* por la función *Nest*.

Es necesario recalcar que solo se puede obtener una aproximación a la curva de Koch, ya que ésta se obtiene cuando el número de iteraciones tiende a infinito. De todas maneras la pantalla tiene limitaciones al igual que la vista humana, así que solo es posible observar una cantidad finita de iteraciones, con lo que se tiene una buena aproximación a la curva de Koch.

Con el propósito de descubrir algunas propiedades de la curva de Koch se presenta la siguiente actividad.

Actividad 4:

Basándose en los experimentos realizados sobre la curva de Koch, determine lo siguiente:

a. La sucesión que representa el número de segmentos de la curva de Koch.

b. La sucesión que representa la longitud de cada intervalo de la curva de Koch.

c. La sucesión que representa la longitud de la curva de la curva de Koch en cada iteración.

d. El límite de cada una de estas sucesiones.

Finalmente, describa los resultados obtenidos.

Copo de nieve de Koch.

Para dibujar el copo de nieve de Koch, es necesario dibujar la curva de Koch sobre los lados de un triángulo equilátero, esto se logra con el siguiente código:

```
Clear[n, p1, p2, p3]
p1 = {0, 0}; p2 = {1, 0}; p3 = RotationTransform[-60 °][p2];
Manipulate[
    Graphics[
        Nest[koch, List[Line[{p1, p2}], Line[{p2, p3}], Line[{p3, p1}]], n]],
    {n, 0, 6, 1}, SaveDefinitions → True]
 (*Si cambia el orden de los puntos iniciales invierte la orientación de la curva*)
```

Actividad 5:

A partir los gráficos anteriores, determine lo siguiente:

a. La sucesión que representa el número de segmentos del copo de nieve de Koch.

b. La sucesión que representa la longitud de cada intervalo del copo de nieve de Koch.

c. La sucesión que representa la longitud del copo de nieve de Koch en cada iteración.

d. El límite de cada una de estas sucesiones.

Finalmente, describa los resultados obtenidos.

Otra forma de generar la curva de Koch.

Una manera diferente de construir la curva de Koch es iniciar con un triángulo isósceles cuyos ángulos son: 30° y 120°. Luego, se elimina un triángulo equilátero cuyo lado es igual a un tercio de la longitud de la base del triángulo original, y se repite el proceso.

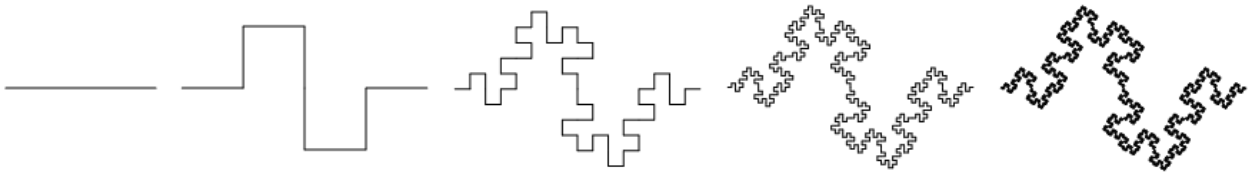
Actividad 6:

Observe la siguiente sucesión de transformaciones y construya una función para hacer los gráficos correspondientes:

Otra curva.

Existen muchas curvas que siguen la misma idea de la curva de Koch. Por ejemplo si la operación básica es ahora dividir un segmento en cuatro partes iguales, dejar los dos cuartos extremos fijos y cambiar los del medio cada uno por tres segmentos como se indica

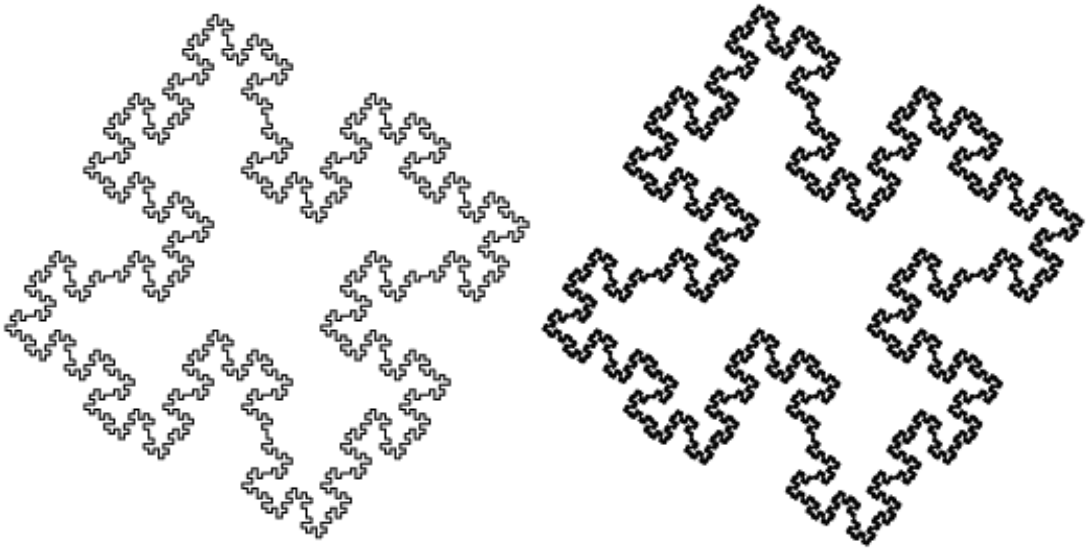
Así en las primeras tres iteraciones la curva toma las siguientes formas:



Actividad 7:

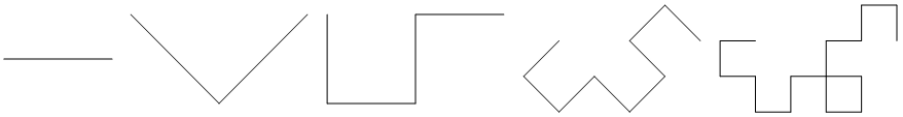
a. Modifique la función *koch* y defina una función *costado* que permita obtener los gráficos anteriores.

b. Con la ayuda de la función *costado* dibuje sobre un cuadrado la curva que esta genera, para obtener la siguiente figura (que se obtiene al iterar tres y cuatro veces respectivamente):



3.1.3. Curva Dragón.

Otra curva interesante es el dragón, en la cual se reemplaza al segmento por una "carpa" rectangular. La diferencia con la anterior es que se dibujan carpas alternadamentea derecha e izquierda. A continuación se ilustran las primeras iteraciones:



Observe que en cada iteración se construye un triángulo rectángulo e isósceles, cuya hipotenusa es el segmento original (la cual es removida). Además, debe tenerse en cuenta que se alterna de derecha a izquierda, una forma de hacer esto es incluir un signo como argumento en la función y otra más fácil es invertir la orientación de la curva, aprovechandoel hecho de que **Line** dibuja desde el punto *x* al punto *y* o de *y* hacia *x*, el mismo segmento.

Actividad 8:

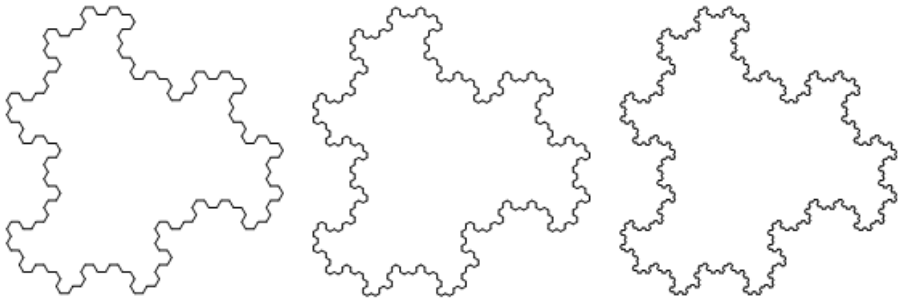
Observe detenidamentelos siguientes gráficos y construya la función *dragon2* que permita generarlos en *Mathematica*.

Otra curva dragón.

Actividad 9:

a. Modifique la función *koch* y defina una función *dragón2* que permita obtener los gráficos anteriores.

b. Con la ayuda de la función *dragón2* dibuje sobre un triángulo equilátero la curva que esta genera, para obtener la siguiente figura:



Actividad 10:

Observe la siguiente sucesión de transformacionesy construya una función para hacer los gráficos correspondientes:

3.1.4. El Ternario, la Función y Polvo de Cantor.

Hasta ahora se han dibujado las curvas en dos dimensiones,obteniendo conjuntos (los gráficos de las curvas) que son fractales. Pero, ¿no sería más fácil hacer primero construccionesen una dimensión? La respuesta claro está es afirmativa, aunque esto no significa necesariamenteque el estudio de sus propiedades sea más fácil. Así como en los casos anteriores se reemplazara un segmento por otro figura, para construir una aproximación al conjunto de Cantor se van quitando partes sin reemplazarlas. Fue Georg Cantor quien describió por primera vez un conjunto "límite" con las características de los fractales, formando los cimientos para las construccionesde Peano, Hilbert y otros.

Conjunto de Cantor.

Para describir el llamado Conjunto de Ternario de Cantor, se inicia dividiendo en tres partes iguales el intervalo [0, 1], empero en vez de reemplazar la parte del medio por un objeto en dos dimensiones, esta se elimina directamente. Para tomar "el límite" o realizar "infinitas iteraciones", es importante sacar sólo el intervalo abierto del medio, dejando dos intervalos cerrados. Sin embargo, gráficamente no se nota la diferencia.

Esta lista de números son los primeros once elementosde la **Órbita** para el **Valor Inicial** 16.

Actividad 11:

Para visualizar la idea antes expuesta grafique el segmento que une en la recta real los números 0 y 1, utilizando *Mathematica*. Luego, genere la figura básica que permitirá formar el conjunto de Cantor, dibujando los segmentos cuyos extremos son los puntos: (0, 0) y $\left(\frac{1}{3}, 0\right)$; $\left(\frac{2}{3}, 0\right)$ y (1, 0), respectivamente.

a. Modifique la función *ramificar* o *koch* y dibuje las primeras siete iteraciones del conjunto de Cantor. **Sugerencia:** No utilice la función **RotationTransform**, pues este es un conjunto en la recta real y observe que el factor de contracción es constante.

b. Calcule la medida del complementodel conjunto de cantor, y luego determine la medida del conjunto de cantor.

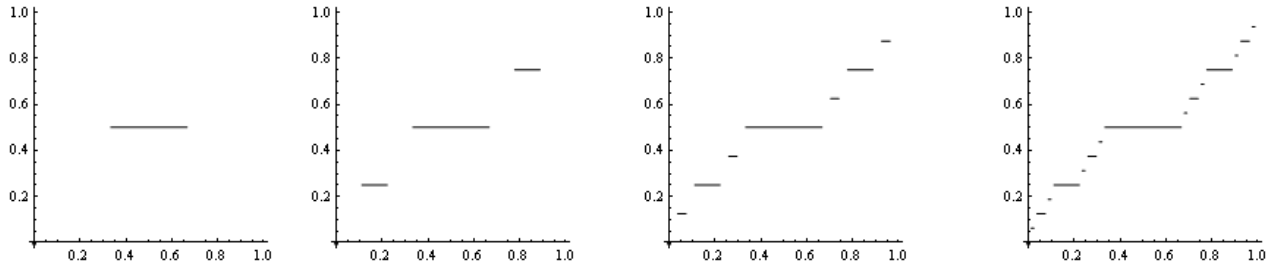
Función de Cantor.

Es importante señalar que a pesar que el conjunto de Cantor tiene medida nula, no es vacío, ya que al menos los puntos $\frac{1}{3}$, $\frac{2}{3}$, $\frac{1}{9}$, $\frac{2}{9}$, ... pertenecenal conjunto.

A partir del ternario de Cantor se puede construir la función de Cantor, cuyo dominio es el conjunto de Cantor y cuyo rango es el intervalo cerrado [0, 1], por lo que un conjunto de medida cero puede tener tantos puntos como un conjunto de medida uno, se tiene pues un ejemplo de que no es cierto que "el todo es mayor que las partes". Extendiendo esta función como constante en el complementodel ternario de Cantor, se obtiene una función del [0, 1] al [0, 1].

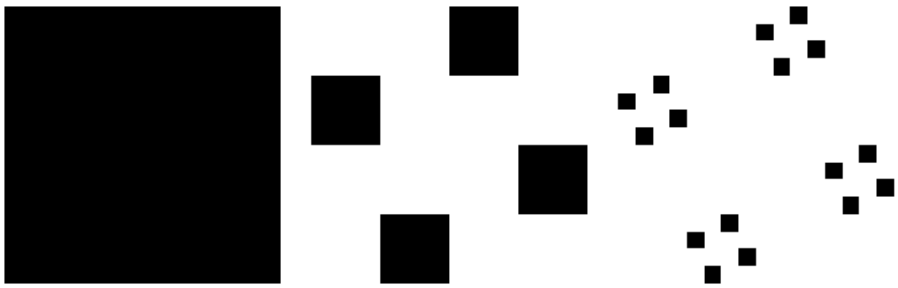
Actividad 12:

Observe detenidamentelos siguientes gráficos y construya la función fcantor que permita generarlos en *Mathematica*.



Polvo de Cantor.

Una variante del conjunto de cantor es considerar como figura base al cuadrado unidad, el cual se divide en dieciséis cuadrados idénticos cuyos lados miden un cuarto del lado original, de los cuales cuatro cuadrados son retenidos y el resto descartados, luego, se repite el proceso antes descrito para cada uno de los cuatro cuadrados y se continua de esta manera indefinidamente.Las figuras que se presenta a continuación ilustran es proceso:

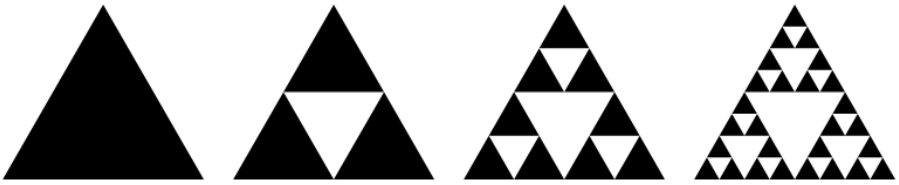


Actividad 13:

Observe detenidamentelos gráficos anteriores y construya la función cantor2D que permita generarlos en *Mathematica*.

3.1.5. El Tamiz y Tetraedro de Sierpinski.

Así como el ternario de Cantor se obtiene sacando intervalos intermedios, en dos dimensiones se comienza con una región del plano y se extrae una subregión, de manera que lo que queda sea una unión de figuras semejantes a la original. Considérese por ejemplo una región triangular o cuadrada en el plano.



Actividad 14:

a. Utilice la función *Polygon* para dibujar una región triangular delimitada por un triángulo equilátero cuyos vértices son: (0, 0) y (1, 0) y $\left(\frac{1}{2}, \frac{\sqrt{3}}{2}\right)$.

b. Determine los puntos medios de cada lado del triángulo y dibuje la región que se obtiene quitando al triángulo original la región delimitada por el triángulo medial.

c. Defina la función *sierpinski* e itérela.

Actividad 15:

A partir los gráficos anteriores, determine lo siguiente:

a. La sucesión que representa el número de triángulos que forma el Tamiz de Sierpinski.

b. La sucesión que representa el área de cada triángulo en cada iteración.

c. La sucesión que representa el área total en cada iteración.

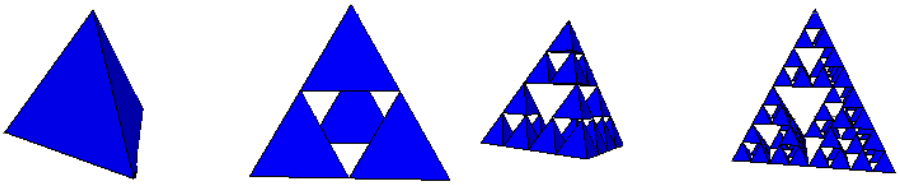
d. El límite de cada una de estas sucesiones.

Tetraedro de Sierpinski.

Actividad 16:

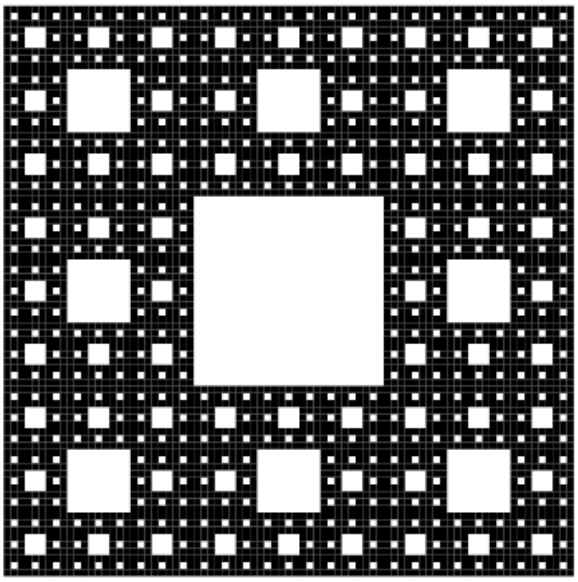
Realice nuevamente las actividades trece y catorce, tomando como figura base a un Tetraedro.

Sugerencia: Modifique la función *sierpinski*, reemplazando la función *Polygon* con *GraphicsComplex*. Además, tome en cuenta que para graficar en tres dimensiones es necesario emplear la función *Graphics3D*.



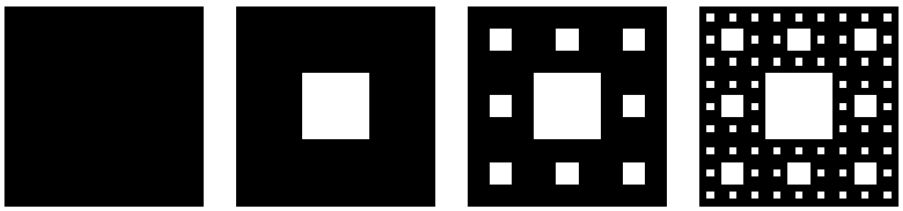
3.1.6. Alfombra de Sierpinski y Esponja de Menger.

Si en vez de comenzar con un triángulo se hace con un cuadrado y se va quitando sucesivamenteel cuadrado del centro, se obtiene la denominada Alfombra de Sierpinski, la cual se muestra a continuación después de cuatro iteraciones:



Actividad 17:

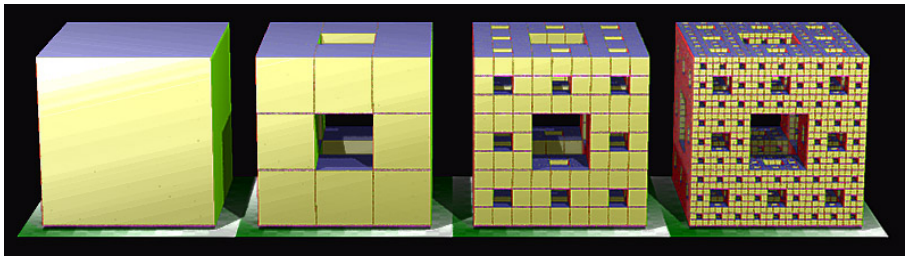
Observe detenidamentela secuencia de figuras siguiente y realice nuevamente las actividades catorce y quince.



Esponja de Menger.

Actividad 18:

Observe detenidamente la siguiente secuencia de figuras y realice nuevamente las actividades catorce y quince. Sugerencia: Utilice la función *Cuboid* en vez de *Rectangle*.



3.2. Dimensión de Hausdorff

Es una costumbre hablar de la dimensión de objetos geométricos, por ejemplo: un punto tiene dimensión cero, una recta tiene dimensión uno, un plano tiene dimensión dos, etc., pero ¿existen figuras geométricas que tengan dimensión no entera?

Basándose en las construcciones de Cantor, Peano y otros, Hausdorff a principios del siglo pasado ideó un concepto de dimensión y medida aplicable a estas curvas y superficies "raras" que aparecían. Posteriormente, Mandelbrot acuñó el término **fractal** para designar ciertos objetos geométricos cuya dimensión de Hausdorff no es entera, que son autosemejantes en el sentido de que no importa cuán grande sea el aumento de la lupa que se utilice al examinar el conjunto se verá un conjunto con formas muy parecidas al original.

Dado el interés (por la misma definición del término fractal), a continuación se presenta una fórmula que permite calcular la dimensión de las curvas y superficies que ya se han estudiado. Vale aclarar que estos son solo algunos casos muy especiales de la autosemejanza y que dicha fórmula solo brinda una aproximación a la dimensión de cada objeto.

Piense por ejemplo en un intervalo de longitud uno, si lo divide en n partes iguales, cada una de ellas medirá $\frac{1}{n}$. Así viendo cada una de esas partes con una lupa que aumente n veces, se obtiene una figura idéntica al intervalo original. De igual manera se puede razonar con un cuadrado de lado uno, que al dividir cada lado en n partes iguales genera n^2 cuadrados, todos de área $\frac{1}{n^2}$, que aumentados en n serán idénticos al original.

Se sabe además, que la dimensión del segmento es uno y la del cuadrado es dos, por lo que en el razonamiento anterior se desea rescatar precisamente los exponentes que aparecen. Una forma de hacerlo es utilizar una fórmula como la siguiente:

$$\text{dimensión} = \frac{\log(\text{número de pedazos})}{\log(\text{aumento})}$$

De modo que para el segmento se tiene:

$$\text{dimensión} = \frac{\log(n)}{\log(n)} = 1$$

y para el cuadrado

$$\text{dimensión} = \frac{\log(n^2)}{\log(n)} = 2$$

Fórmula para calcular la dimensión de Hausdorff

Una alternativa para el cálculo de la dimensión de Hausdorff, es observar que dado que todas las funciones estudiadas hasta el momento son contractivas el "aumento" es igual al "recíproco de la razón de homotecia" (la cual es siempre positiva y menor que uno), por lo que la fórmula es puede reescribir como:

$$\text{dimensión} = \frac{\log(\text{número de pedazos})}{\log(\text{aumento})} = \frac{\log(\text{número de pedazos})}{-\log(\text{razón de homotecia})}$$

Por ejemplo, dado un cubo de arista uno, se divide en n^3 cubos iguales, cada uno con arista $\frac{1}{n}$, así la razón entre los volúmenes es: $\frac{1}{n^3}$ y por consiguiente, la dimensión de Hausdorff es:

$$\text{dimensión} = \frac{\log(n^3)}{-\log\left(\frac{1}{n}\right)} = 3$$

Actividad 19:

Calcule la dimensión de Hausdorff de los fractales que se han estudiado previamente, e interprete los resultados.

3.3. Sistemas dinámicos de variable compleja.

3.3.1. Conjuntos de Julia.

Pasemos al estudio del comportamiento dinámico en dos dimensiones. En este caso es conveniente pensar en términos de variables complejas, o sea números "z" de la forma $a + b\,i$, donde a y b son reales e i es "la unidad imaginaria".

Con los cambios hechos en el caso real, se puede ver que basta estudiar la función $z \to z^2 + c$ donde c es un número complejo arbitrario. Como antes, se tienen conceptos como órbita, punto fijo y ciclo, y el propósito es estudiar las órbitas "que se escapan".

Como se hizo en el caso real, nos restringiremos a estudiar sólo los valores del parámetro c tales que $\| c \| \leq 2$, aunque se sabe (tomando por ejemplo c y z reales, $c < -2$ y z un punto fijo) que hay órbitas acotadas aún cuando $\| c \| > 2$. Usando la desigualdad triangular, no es difícil ver que si $\| c \| < 2 < \| z \|$, entonces la órbita de z "se escapará", por lo que el interés está en los $z's$ con valor absoluto menor que 2.

Los matemáticos franceses Pierre Fatou y Gaston Julia a principios del siglo pasado fueron quienes comenzaron el estudio de los puntos $z's$ que tienen una órbita acotada cuando el parámetro c se mantiene fijo. La frontera del conjunto de estos $z's$, que en general es un fractal, se conoce como conjunto de Julia. Así, si un punto z está en el conjunto de Julia, habrá puntos arbitrariamente cerca con órbitas no acotadas.

Por ejemplo, poniendo el caso más sencillo $c = 0$, es claro que la órbita que se inicia en z está acotada si y sólo si $\| z \| \leq 1$, dado que para z complejo vale la igualdad $\| z^2 \| = \| z \|^2$. Así, si $\| z \| < 1$, su órbita se aproximará a 0, mientras que si $\| z \| > 1$, su órbita se escapará. Por lo tanto, el conjunto de Julia correspondiente a $c = 0$ es la circunferencia unitaria.

Los puntos en la circunferencia tienen trayectorias siempre sobre la circunferencia (cuando $c = 0$), pero cuando se calcula, pequeños errores numéricos harán que la órbita se corra hacia uno u otro lado de la circunferencia y por lo tanto puede parecer que se aproxima a cero o se escapa. En otras palabras son *órbitas inestables*.

Ejercicio: Hacer una tabla de valores de la órbita de $z = 0.6 + 0.8\,i$ cuando $c = 0$, y verificar si estos valores tienen valor absoluto igual a uno. (Hacer por ejemplo 50 o 100 iteraciones).

Función Julia.

Un programa para graficar los puntos que mantienen una órbita acotada para cada valor de c fijo puede hacerse sencillamente construyendo una tabla para valores de z en una región rectangular en la que se anota la cantidad de iteraciones necesaria para obtener un valor absoluto mayor que dos. Por ejemplo,

```
(* Terminar si Abs[z] ≥ 2 *)
Clear[escape, julia]
escape = Compile[{{z, _Complex}, {c, _Complex}, {itermax, _Integer}},
               Module[{z1 = z, iter = 0},
                    While[(Abs[z1] < 2) && (++iter ≤ itermax), z1 = Chop[z1^2 + c]]; iter]];
julia[c_, itermax_, {{xmin_, xmax_}, {ymin_, ymax_}}, {ndivx_, ndivy_}] :=
     Table[escape[x + y ⅈ, c, itermax],
          {y, ymin, ymax, (ymax - ymin) / ndivy // N},
          {x, xmin, xmax, (xmax - xmin) / ndivx // N}]
```

Dibujando el círculo.

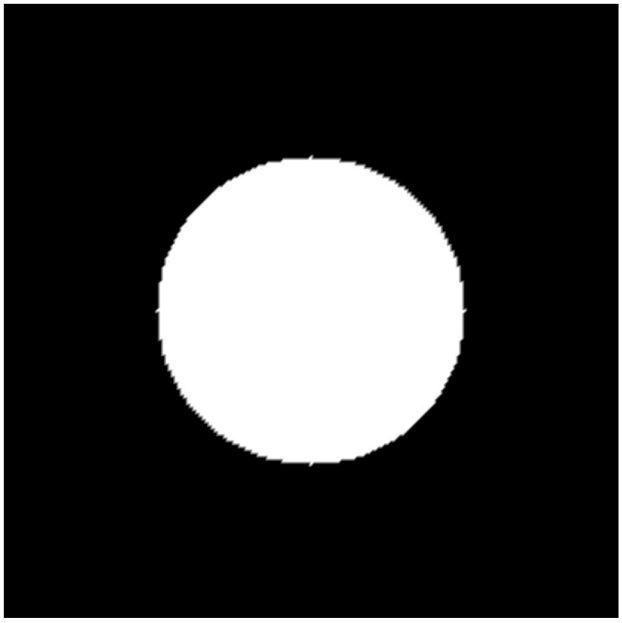
Una vez construida la tabla puede usarse **ListDensityPlot**, que muestra con distintos tonos la variación en el número de iteraciones.

Por ejemplo, consideremos la región definida por $-2 \leq x \leq 2,\ -2 \leq y \leq 2$ dividiéndola en cada dirección en 200 e iterando a lo sumo 100 veces:

```
c = 0.; iteraciones = 100; rango = {{-2, 2}, {-2, 2}}; puntos = 200;
Timing[tabla = julia[c, iteraciones, rango, {puntos, puntos}];]
```

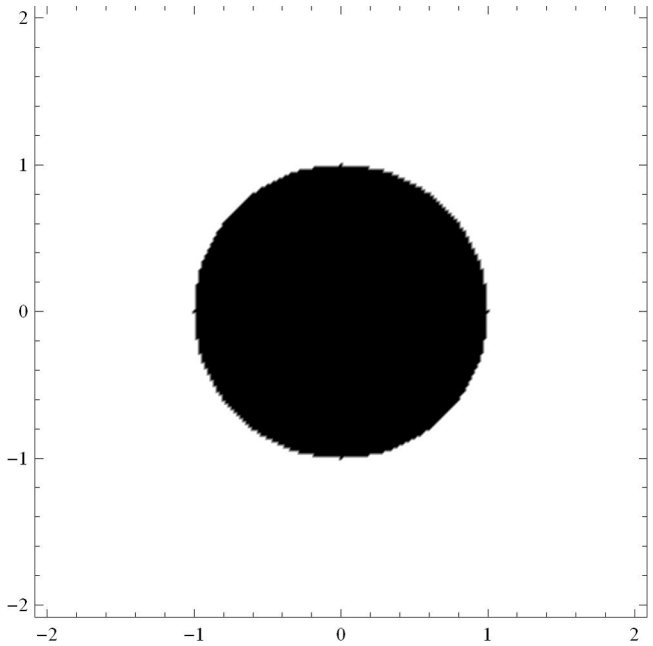
```
{0.75, Null}
```

```
ListDensityPlot[tabla, ColorFunction → (GrayLevel[Round[#]] &),
     Frame → False, Mesh → False, DataRange → rango]
```



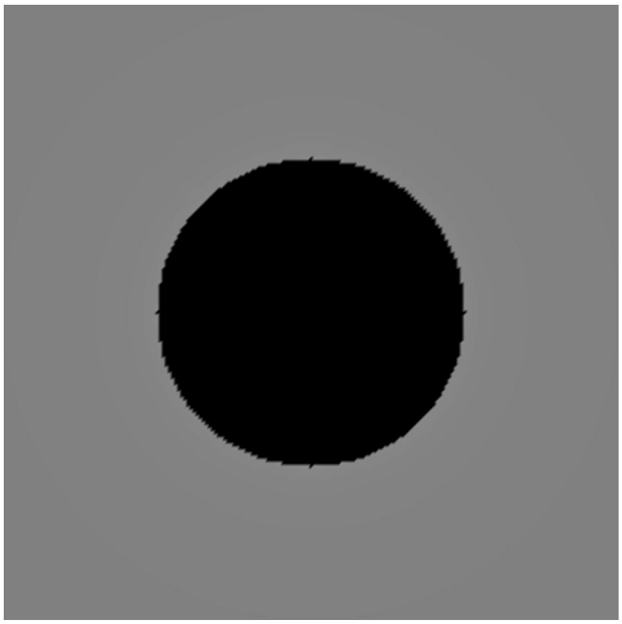
Si se desea que el círculo aparezca en negro y el resto en blanco:

```
ListDensityPlot[tabla, ColorFunction → (GrayLevel[1 – Round[#]] &),
    Frame → True, Mesh → False, DataRange → rango]
```



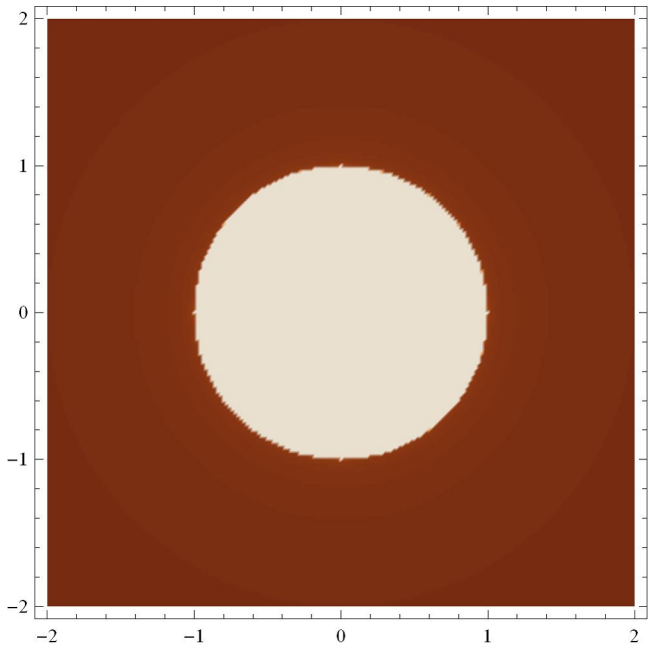
O puede definirse una función tono, que permita resaltar el conjunto poniéndolo en negro e ir aumentando en gris:

```
tono[x_] := If[x < .5, (x + 1)/2., 0];
ListDensityPlot[tabla, ColorFunction → (GrayLevel[tono[#]] &),
    Frame → False, Mesh → False, DataRange → rango]
```

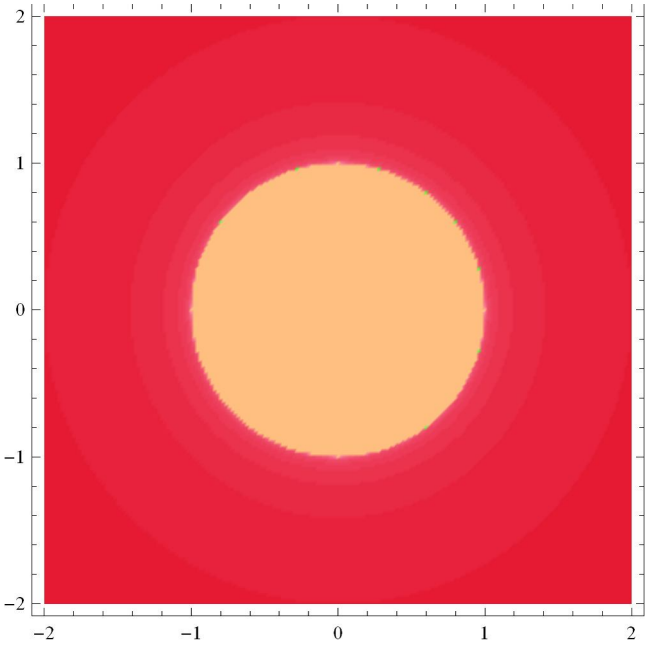


También puede utilizarse coloraciones predefinidas como:

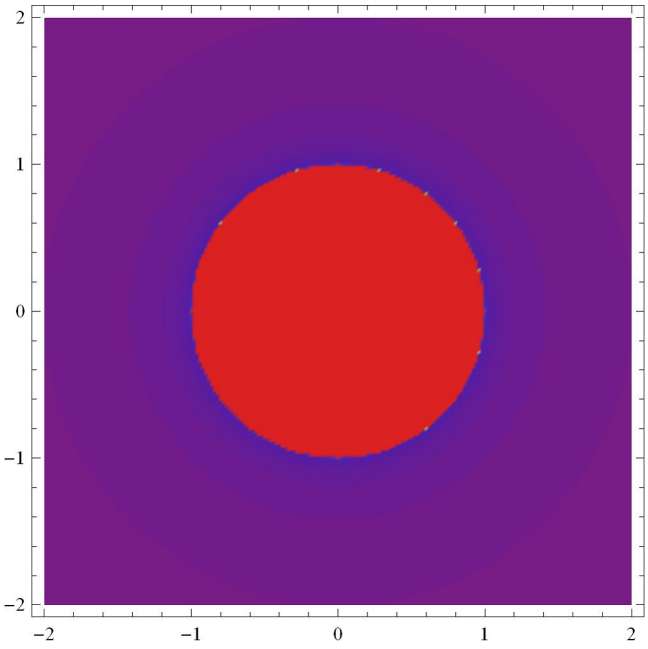
```
ListDensityPlot[tabla, ColorFunction → "SiennaTones",
    Frame → True, Mesh → False, DataRange → rango]
```



```
ListDensityPlot[tabla, ColorFunction → "BrightBands",
    Frame → True, Mesh → False, DataRange → rango]
```




```
ListDensityPlot[tabla, ColorFunction → "Rainbow",
    Frame → True, Mesh → False, DataRange → rango]
```



Actividad 1:

a. Obtener el gráfico para $c = -1$, en la región $\{(x, y) \mid x \in [-1.7, 1.7],\ y \in [-0.9, 0.9]\}$, dividiéndola en 200×200 puntos, con un máximo de 100 iteraciones, destacando en negro el conjunto y en blanco su complemento.

b. Usando la coloración por defecto de *Mathematica*, obtenga el respectivo gráfico para $c = i$ (la unidad imaginaria), en la región $\{(x, y) \mid x \in [-1.5, 1.5],\ y \in [-1.5, 1.5]\}$, dividiéndola en 200×200 puntos, con un máximo de 100 iteraciones.

Utilice la coloración por defecto de *Mathematica*, para obtener el gráfico correspondiente a $c = \frac{1}{2}$, en la región $\{(x, y) \mid x \in [-1, 1],\ y \in [-1.4, 1.4]\}$, dividiéndola en 200×200 puntos, con un máximo de 150 iteraciones.

d. Obtener el gráfico para $c = 0.360284 + 0.100376\, i$, en la región $\{(x, y) \mid x \in [-1, 1],\ y \in [-1.2, 1.2]\}$, dividiéndola en 200×200 puntos, con un máximo de 200 iteraciones, destacando en negro el conjunto y en blanco su complemento.

Actividad 2:

Genere los respectivos conjuntos de Julia para los siguientes valores del parámetro c:

a. La constelación: $c = -0.74543 + 0.11301\, i$

b. Conejo de Douady: $c = -0.122 + 0.745\, i$

c. Esponja: $c = -0.4 - 0.6\, i$

d. Remolinos: $c = -0.8 + 0.4\, i$

e. Cruces: $c = -1.5$

3.3.2. Conjunto de Mandelbrot.

El proceso para construir el conjunto de Mandelbrot es análogo al que se sigue para construir los conjuntos de Julia, empero, en este caso el valor inicial se mantiene fijo en cero y se varía el parámetro *c*. Dada la función $z \to z^2 + c$ donde *c* es un número complejo arbitrario, el conjunto de Mandelbrot es el conjunto de los puntos complejos *c* para los cuales la órbita inicializada en cero permanece acotada.

Dado que el código que permite construir el conjunto de Mandelbrot es muy similar al utilizado para construir los conjuntos de Julia, se presenta la siguiente actividad.

Actividad 3:

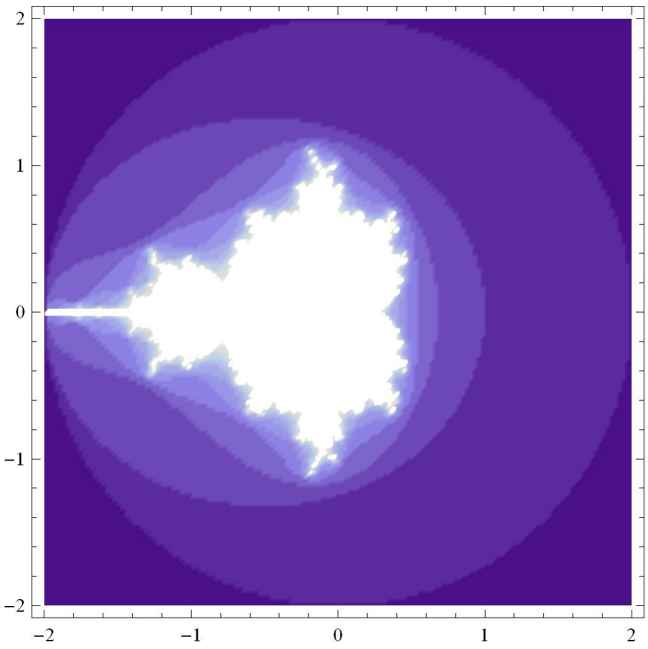
Tomando en cuenta que para graficar el conjunto de Mandelbrot se mantiene fijo el valor inicial en cero y se debe variar el parámetro *c*, defina la función mandelbrot que permita generar una tabla que contenga el número de iteraciones para que los elementos de la órbita inicializada en cero tenga módulo mayor que dos, y que no continúe si se ha superado el número de iteraciones máxima prefijado.

Como antes un aspecto importante que se debe considerar es el rango de estudio, para no esperar demasiado tiempo en calcular órbitas que divergen, por lo que es importante restringir la zona del gráfico lo más posible. Se ilustra este aspecto tomando el rango $\{(x, y) \mid x \in [-2, 2],\ y \in [-2, 2]\}$, 150 divisiones en cada eje y 125 iteraciones como máximo.

```
rango = {{-2., 2.}, {-2., 2.}};
Timing[man = mandelbrot[rango, 150, 125];]
```

```
{0.297, Null}
```

```
ListDensityPlot[man, Frame → True, Mesh → False, DataRange → rango]
```



Los resultados obtenidos hasta el momento permiten formular la actividad siguiente.

Actividad 4:

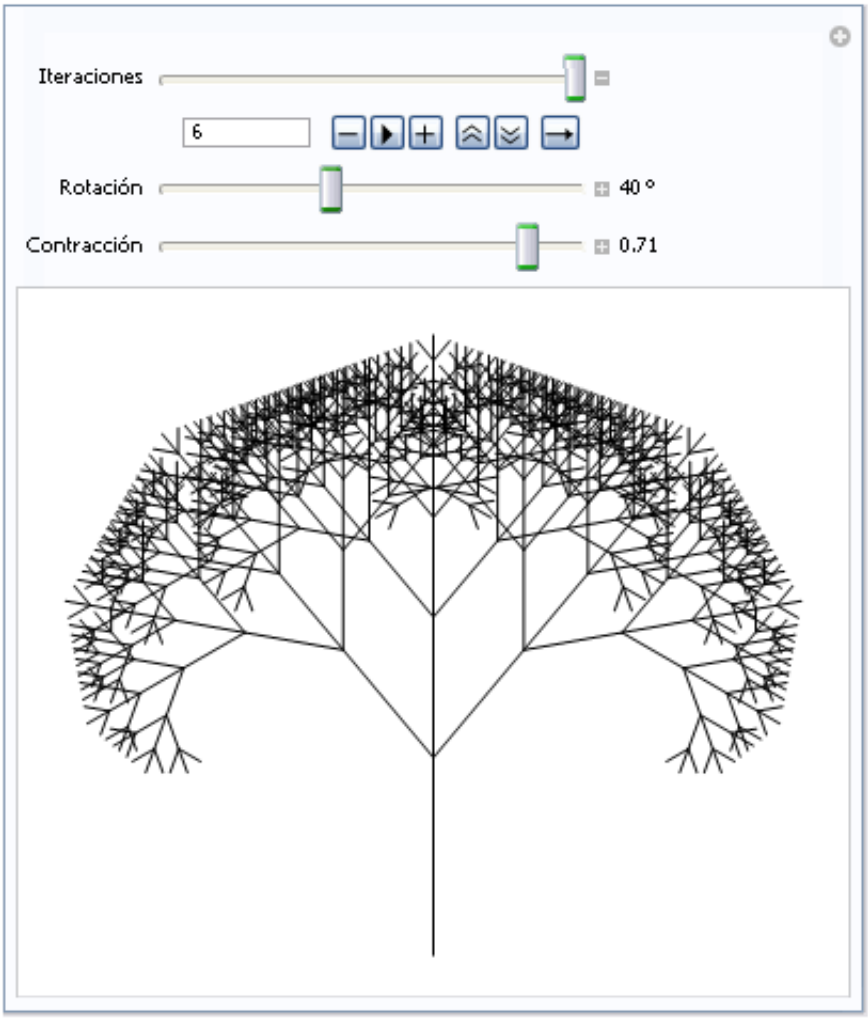
- **a.** Obtener un gráfico del conjunto de Mandelbrot, en la región $\{(x, y) \mid x \in [-2, 0.5],\ y \in [-1.25, 1.25]\}$, dividiéndola en 400×400 puntos, con un máximo de 100 iteraciones, utilizando la coloración por defecto de *Mathematica* y también, destacando en negro el conjunto y en blanco su complemento.
- **b.** Obtener un gráfico del conjunto de Mandelbrot, en la región $\{(x, y) \mid x \in [-1.2, -1.1],\ y \in [0.24, 0.34]\}$, dividiéndola en 375×375 puntos, con un máximo de 100 iteraciones, destacando en negro el conjunto y en blanco su complemento.
- **c.** Obtener un gráfico del conjunto de Mandelbrot, en la región $\{(x, y) \mid x \in [-1.192, -1.183],\ y \in [0.298, 0.307]\}$.
- **d.** Comparece el gráfico del conjunto de Mandelbrot con el Diagrama de Órbitas y establezca la relación entre ambos.
- **e.** Basándose en los resultados del literal anterior, determine las regiones en el conjunto de Mandelbrot en las que las órbitas son atraídas a puntos periódicos ó ciclos de longitud dos. Finalmente, corrobore los resultados algebraicamente.

3.4. Soluciones a actividades de la sección 3.1

Solución actividad 1:

```
Clear[ramificar2, angulo, contraccion, v]
ramificar2[Line[{a_, b_}]] :=
    {v = contraccion (b - a); Line[{b, b + RotationTransform[-angulo][v]}],
    Line[{b, b + v}], Line[{b, b + RotationTransform[angulo][v]}]}
ramificar2[x_List] := ramificar2 /@ Flatten[x]
```

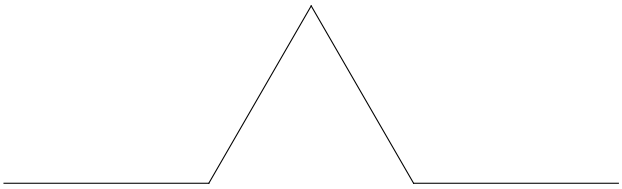
```
Manipulate[angulo; contraccion;
    Graphics[NestList[ramificar2, Line[{{0, 0}, {0, 1}}], n]],
    {{n, 1, "Iteraciones"}, 1, 6, 1, Appearance → "Open"},
    {{angulo, 40 °, "Rotación"}, 0 °, 100 °, 10 °, Appearance → "Labeled"},
    {{contraccion, .5, "Contracción"}, 0, .8, .01, Appearance → "Labeled"},
    SaveDefinitions → True]
```



```
Clear[a, b]
a = {-1, 0}; b = {1, 0};
Graphics[Line[{a, b}]]
```

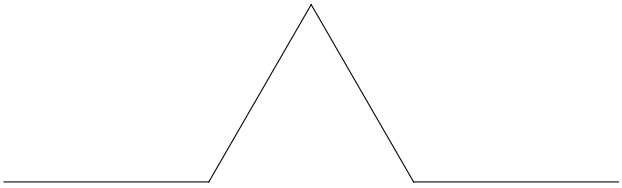


```
Clear[c, d, e, v]
v = (b - a)/3.; c = a + v; d = c + RotationTransform[60 °][v]; e = b - v;
Graphics[{Line[{a, c}], Line[{c, d}], Line[{d, e}], Line[{e, b}]}]
```

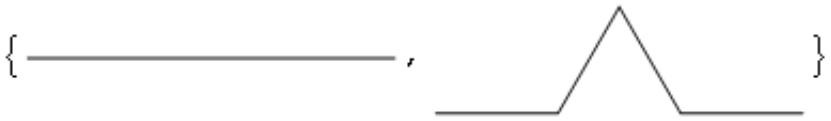


Otras formas de definir los puntos d y e son:

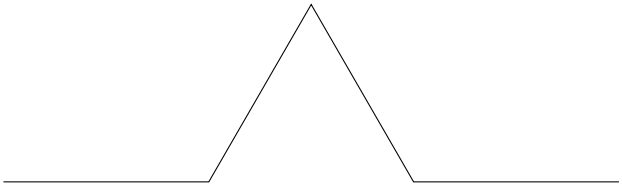
```
Clear[d, e]
d = c + RotationTransform[60 °][v]; e = d + RotationTransform[-60 °][v];
Graphics[{Line[{a, c}], Line[{c, d}], Line[{d, e}], Line[{e, b}]}]
```



```
Clear[d, e]
d = a + 2 v; e = RotationTransform[60 °, c][d];
List[Graphics[Line[{a, b}]],
 Graphics[{Line[{a, c}], Line[{c, e}], Line[{d, e}], Line[{d, b}]}]]
```



```
Clear[d, e]
d = RotationTransform[-120 °, c][a]; e = RotationTransform[60 °, d][c];
Graphics[Line[{a, c, d, e, b}]]
```

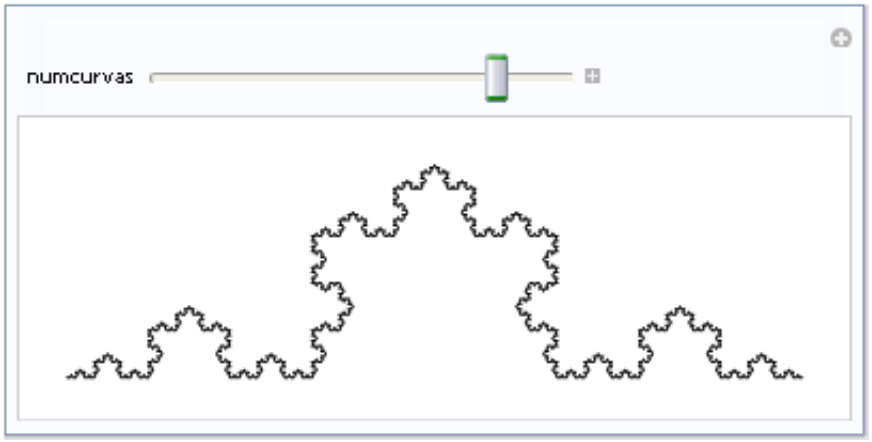


Solución actividad 3:

```
koch =.
koch[Line[{a_, b_}]] :=
        {v = (b - a)/3.; c = a + v; d = c + RotationTransform[π/3.][v];
  e = b - v; Line[{a, c}], Line[{c, d}], Line[{d, e}],
        Line[{e, b}]}(*La razón de semejanza es 1/3 y el ángulo de rotación es 60º*)
koch[x_List] := koch /@ Flatten[x]
```

En la función *Manipulate*, se debe utilizar *Nest* en vez de *NestList*, puesto que no debe graficarse la figura base.

```
n =.
Manipulate[
    Graphics[
        Nest[koch, Line[{{0, 0}, {1, 0}}], n]],
    {{n, 2, "numcurvas"}, 0, 6, 1},
    SaveDefinitions → True]
```



Solución actividad 4:

Las sucesiones son: $s_n = \{4^n\}_{n=0}^{\infty}$, $l_n = \left\{\frac{1}{3^n}\right\}_{n=0}^{\infty}$, $t_n = \left\{\left(\frac{4}{3}\right)^n\right\}_{n=0}^{\infty}$. Y el límite de cada una es:

```
Limit[4^n, n → ∞]
Limit[1/3^n, n → ∞]
Limit[(4/3)^n, n → ∞]
```

```
∞
```

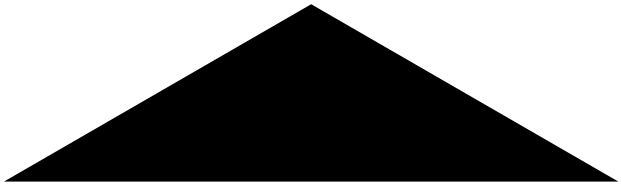
```
0
```

```
∞
```

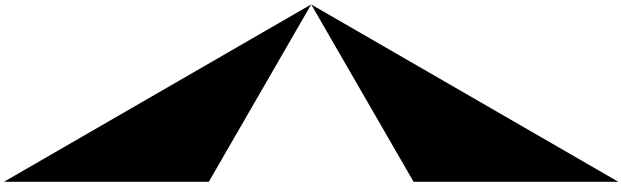
Los resultados anteriores presentan cualidades paradójicas de la curva de Koch, pues cuando el número de iteraciones tiende a infinito su longitud es infinita, aunque la curva sea acotada; además, esta tiene área cero, así que ni la longitud ni el área proporcionan una descripción adecuada de la misma. Pese a esto, la curva es autosemenjante, ya que, la región comprendida entre 0 y $\frac{1}{3}$ (en el eje horizontal) en la n-ésima iteración se corresponde exactamente a la curva entre 0 y 1 en la $n-1$-ésima iteración.

Solución actividad 6:

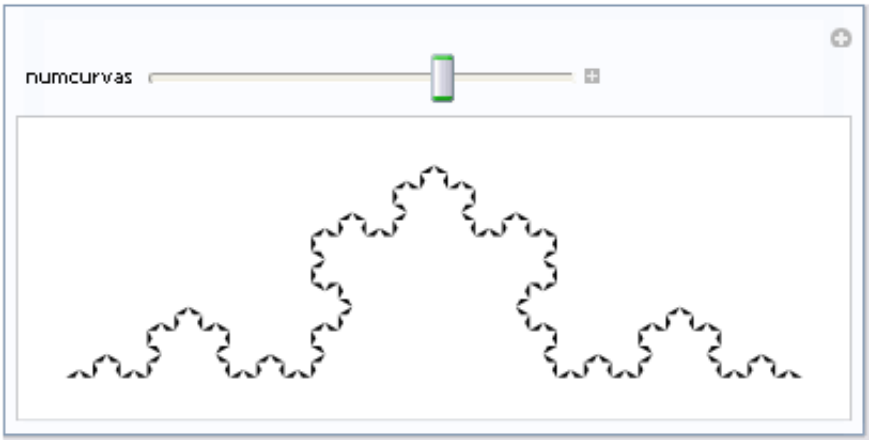
```
Clear[a, b]
a = {0, 0}; b = {1, 0}; c = {1 / 2, Sqrt[3] / 6};
Graphics[Polygon[{a, b, c}]]
```



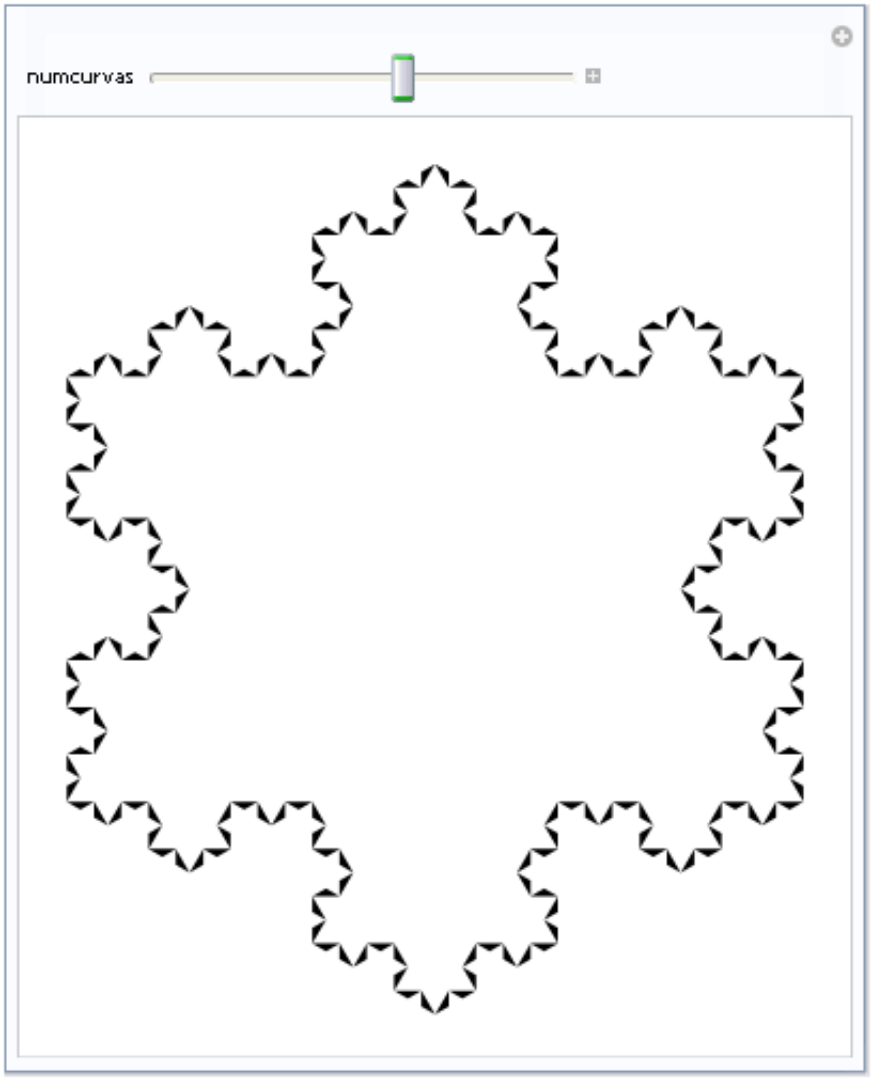
```
Clear[d, e, v]
v = (b - a)/3.; d = a + v; e = b - v;
Graphics[{Polygon[{a, d, c}], Polygon[{c, e, b}]}]
```



```
koch =.
koch[Polygon[{a_, b_, c_}]] :=
    {v = (b - a)/3.; d = a + v; e = b - v; Polygon[{a, c, d}], Polygon[{b, c, e}]}
koch[x_List] := koch /@ Flatten[x]

n =.
Manipulate[
    Graphics[
        Nest[koch, Polygon[{{0, 0}, {1, 0}, {1 / 2, Sqrt[3] / 6}}], n]],
    {{n, 2, "numcurvas"}, 0, 10, 1},
    SaveDefinitions → True]
```



```
Clear[n, p1, p2, p3, p4, p5, p6]
p1 = {0, 0}; p2 = {1, 0}; p3 = {1 / 2, Sqrt[3] / 6};
p4 = {1 / 2, -Sqrt[3] / 2}; p5 = {0, -1 / Sqrt[3.]};
p6 = {1, -1 / Sqrt[3.]};
Manipulate[
    Graphics[

  Nest[koch, List[Polygon[{p1, p2, p3}], Polygon[{p1, p4, p5}], Polygon[{p2, p4, p6}]], n]],
    {{n, 2, "numcurvas"}, 0, 10, 1},
    SaveDefinitions → True]
```

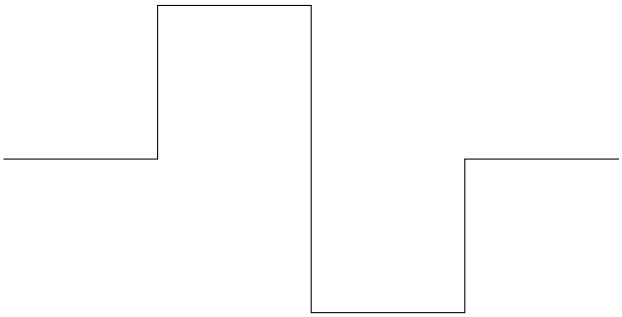


Solución actividad 7:

```
Clear[a, b]
a = {-1, 0}; b = {1, 0};
Graphics[Line[{a, b}]]
```

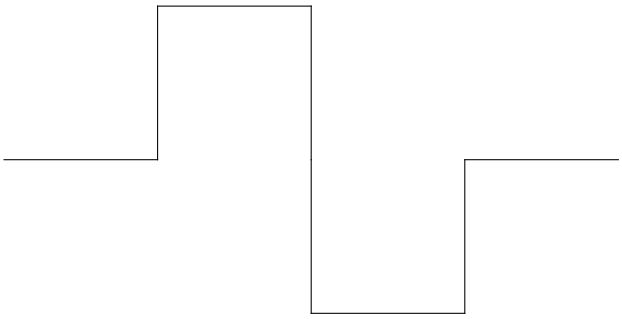


```
Clear[c, d, e, f, g, h, i, v]
v = (b - a)/4.; c = a + v; d = c + RotationTransform[90. °][v]; e = d + v; f = c + v;
g = f + RotationTransform[-90. °][v]; h = g + v; i = b - v;
Graphics[Line[{a, c, d, e, f, g, h, i, b}]]
```

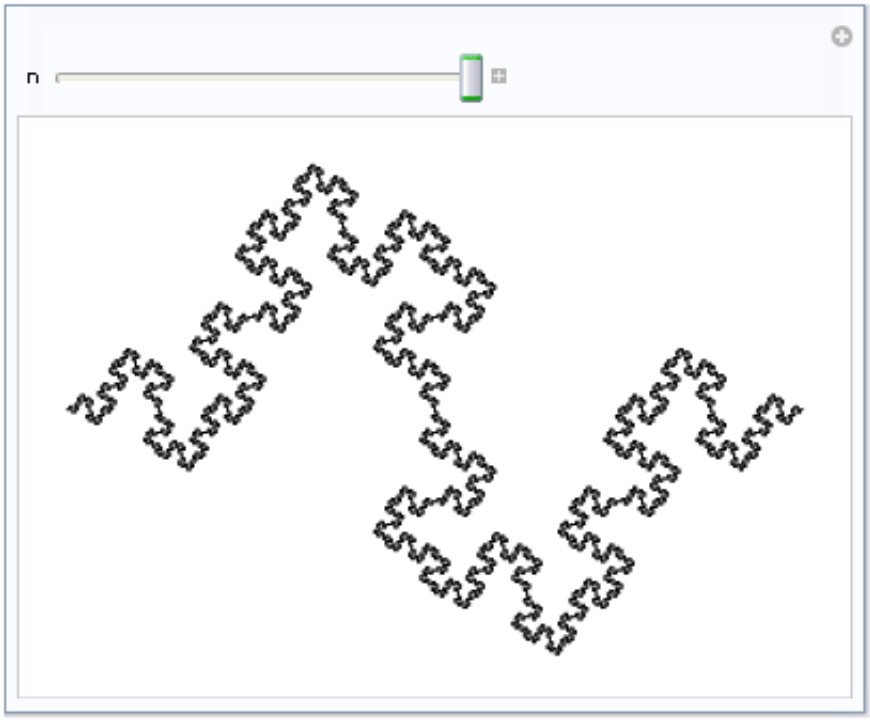


Una forma alternativa de definir los puntos c,d,e,f, g, h e i es:

```
Clear[c, d, e, f, g, h, i]
c = (3 a + b)/4; d = (a + b)/2; e = (a + 3 b)/4; f = RotationTransform[90 °, c][d];
g = RotationTransform[-90 °, d][c];
h = RotationTransform[90 °, d][c]; i = RotationTransform[90 °, e][d];
Graphics[{Line[{a, c}], Line[{c, f}], Line[{f, g}], Line[{g, d}], Line[{d, h}], Line[{h, i}],
         Line[{i, e}], Line[{e, b}]}]
```

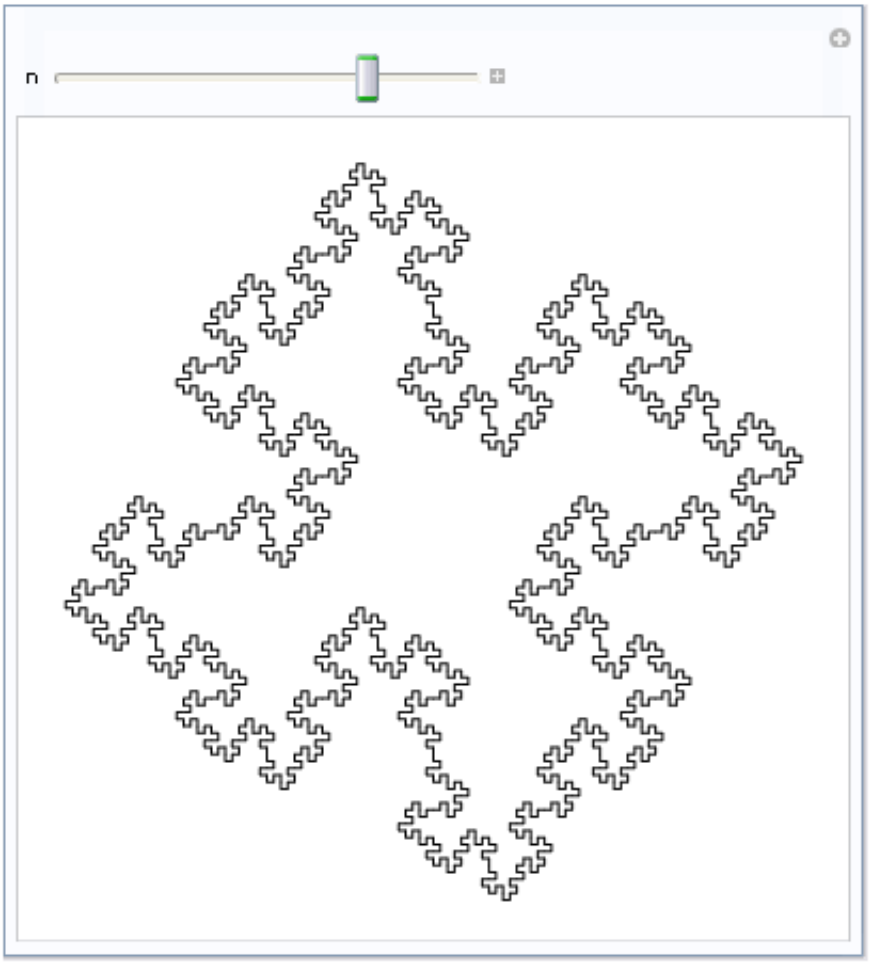


```
costado =.
costado[Line[{a_, b_}]] :=
    {v = (b - a)/4.; c = a + v; d = c + RotationTransform[π/2.][v];
  e = d + v; f = c + v; g = f + RotationTransform[-π/2.][v]; h = g + v; i = b - v;
    Line[{a, c}], Line[{c, d}], Line[{d, e}], Line[{e, f}],
  Line[{f, g}], Line[{g, h}], Line[{h, i}], Line[{i, b}]}
costado[x_List] := costado /@ Flatten[x]

n =.
Manipulate[
    Graphics[
        Nest[costado, Line[{{0, 0}, {1, 0}}], n]],
    {n, 0, 4, 1}, SaveDefinitions → True]
```



Para dibujara la curva generada con la función *costado* sobre un cuadrado se utiliza un código similar al empleado para dibujar el copo de Nieve de Koch.

```
Clear[p1, p2, p3, p4]
p1 = {0, 0}; p2 = {1, 0}; p3 = RotationTransform[90 °][p2]; p4 = p2 + RotationTransform[90 °][p2];

n =.
Manipulate[
    Graphics[
        Nest[costado, {Line[{p1, p2}], Line[{p1, p3}], Line[{p2, p4}], Line[{p3, p4}]}, n]],
    {n, 0, 4, 1}, SaveDefinitions → True]
```



Solución actividad 8:

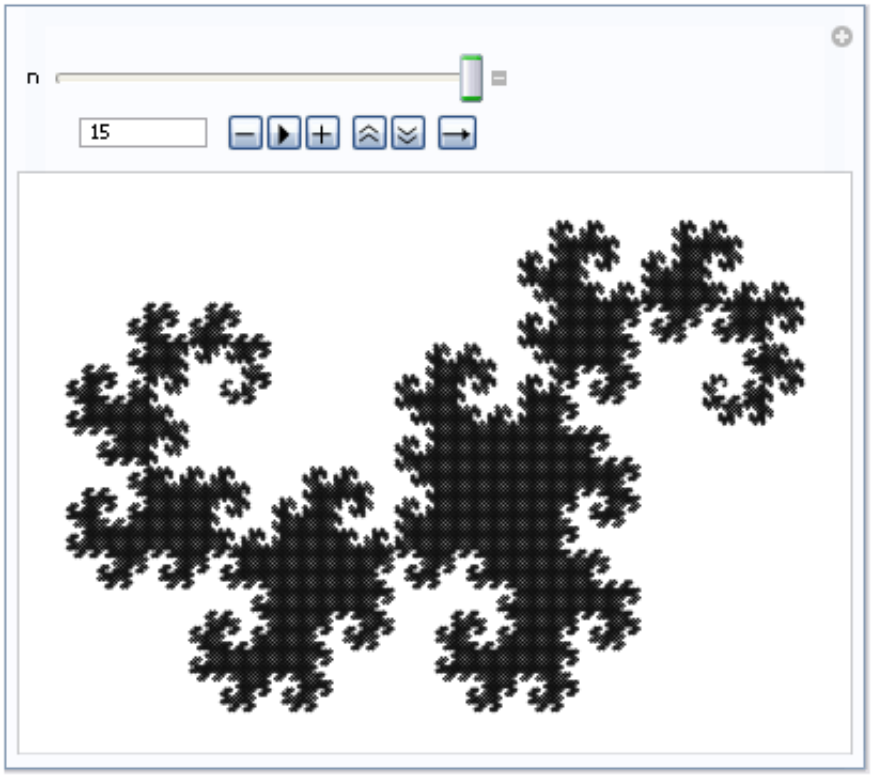
```
Clear[a, b, c, v]
```

```
a = {0, 0}; b = {1, 0}; v = (b - a)/Sqrt[2]; c = a + RotationTransform[-45 °][v];
```

```
List[Graphics[Line[{a, b}]], Graphics[{Line[{a, c}], Line[{b, c}]}]]
```

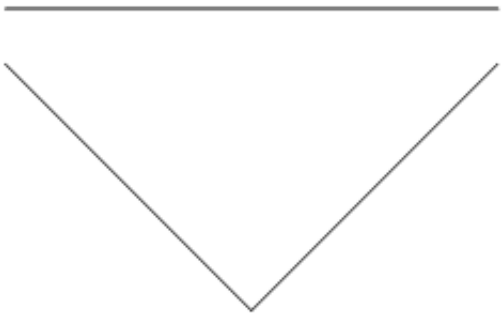
Un código para graficar el dragón es:

```
dragon = .
dragon[Line[{a_, b_}]] :=
        {c = a + RotationTransform[-π/4.][(b - a)/√2.]; Line[{a, c}], Line[{b, c}]}
dragon[x_List] := Map[dragon, Flatten[x]]

n = .
Manipulate[
     Graphics[
          Nest[dragon, Line[{{0, 0}, {1, 0}}], n]],
     {n, 0, 15, 1}, SaveDefinitions → True]
```



Solución 2.

```
Clear[a, b, c, v]
a = {-1, 0}; b = {1, 0}; v = (b - a)/Sqrt[2]; c = a + RotationTransform[-45 °][v];

Graphics[Line[{a, b}]]
Graphics[{Line[{a, c}], Line[{c, b}]}]
```

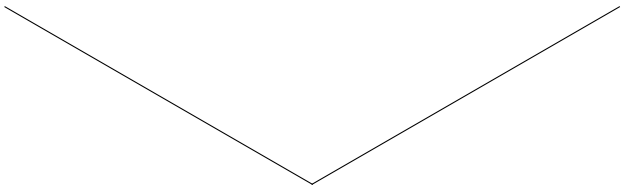


```
dragon =.
dragon[Line[{a_, b_}], signo_] :=
        {c = a + RotationTransform[signo Pi/4.][(b - a)/Sqrt[2.]]; Line[{a, c}], Line[{c, b}]}
dragon[x_List] := Thread[dragon[Flatten[x], Table[(-1)^k, {k, Length[Flatten[x]]}]]]

n =.
Manipulate[
     Graphics[
          Nest[dragon, List[Line[{{0, 0}, {1, 0}}]], n]],
     {n, 0, 15, 1}, SaveDefinitions → True]
```

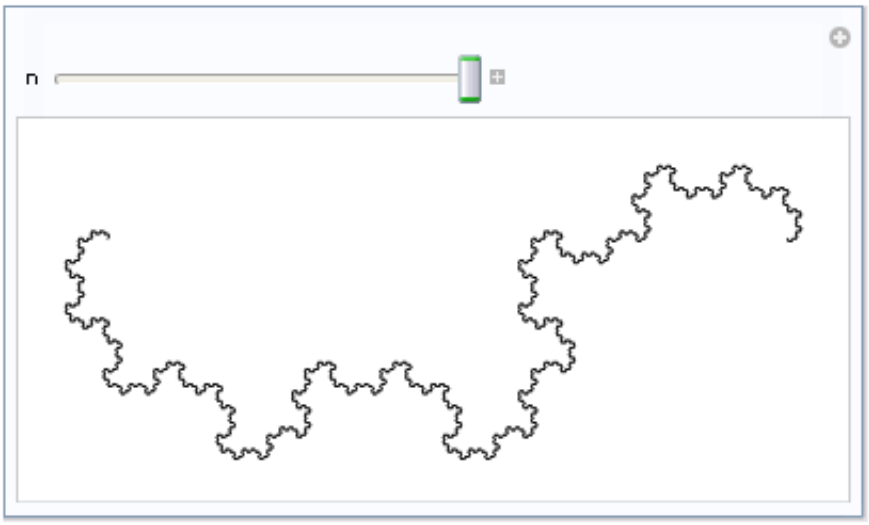
Solución actividad 9:

Primera solución:

```
Clear[a, b, c, v]
a = {0, 0}; b = {1, 0}; v = (b - a)/Sqrt[3]; c = a + RotationTransform[-30 °][v];

Graphics[{Line[{a, c}], Line[{c, b}]}]
```

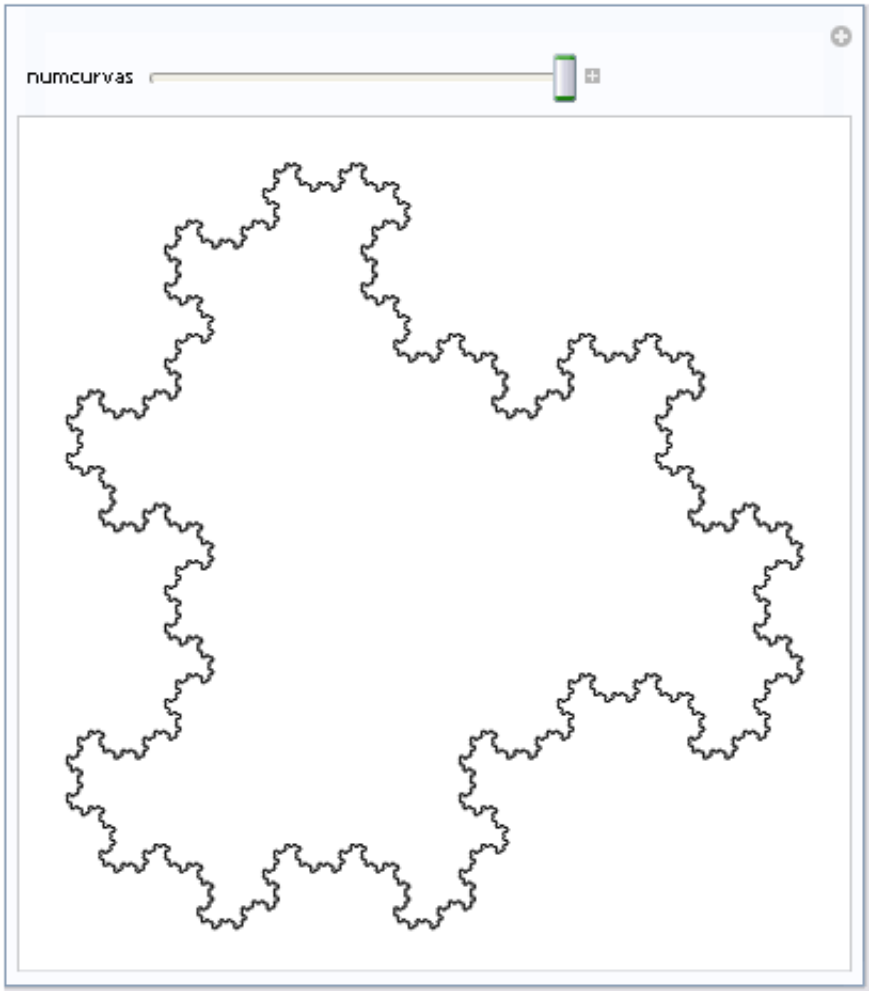


```
dragon2 =.
dragon2[Line[{a_, b_}]] :=
        {c = a + RotationTransform[-30. °][(b - a)/Sqrt[3.]]; Line[{a, c}], Line[{b, c}]}
dragon2[x_List] := Map[dragon2, Flatten[x]]

n =.
Manipulate[
     Graphics[
          Nest[dragon2, Line[{{0, 0}, {1, 0}}], n]],
     {n, 0, 10, 1}, SaveDefinitions → True]
```



Con el código siguiente se dibuja la curva dragón sobre un triángulo:

```
Clear[n, p1, p2, p3]

p1 = {0, 0}; p2 = {1, 0}; p3 = {1/2, Sqrt[3]/2};

Manipulate[
    Graphics[
        Nest[dragon2, {Line[{p1, p2}], Line[{p2, p3}], Line[{p3, p1}]}, n]],
    {{n, 0, "numcurvas"}, 0, 10, 1},
    SaveDefinitions → True]
```



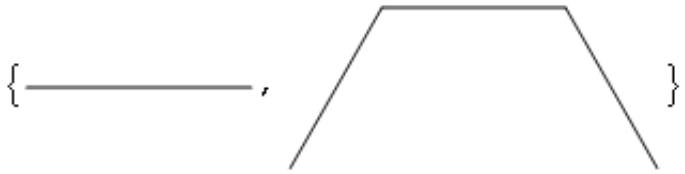
Solución actividad 10:

Primera solución:

```
Clear[a, b, c, d, v]
```

```
a = {0, 0}; b = {1, 0}; v = (b - a)/2; c = a + RotationTransform[60 °][v]; d = c + v;
```

```
List[Graphics[Line[{a, b}]], Graphics[{Line[{c, a}], Line[{c, d}], Line[{b, d}]}]]
```

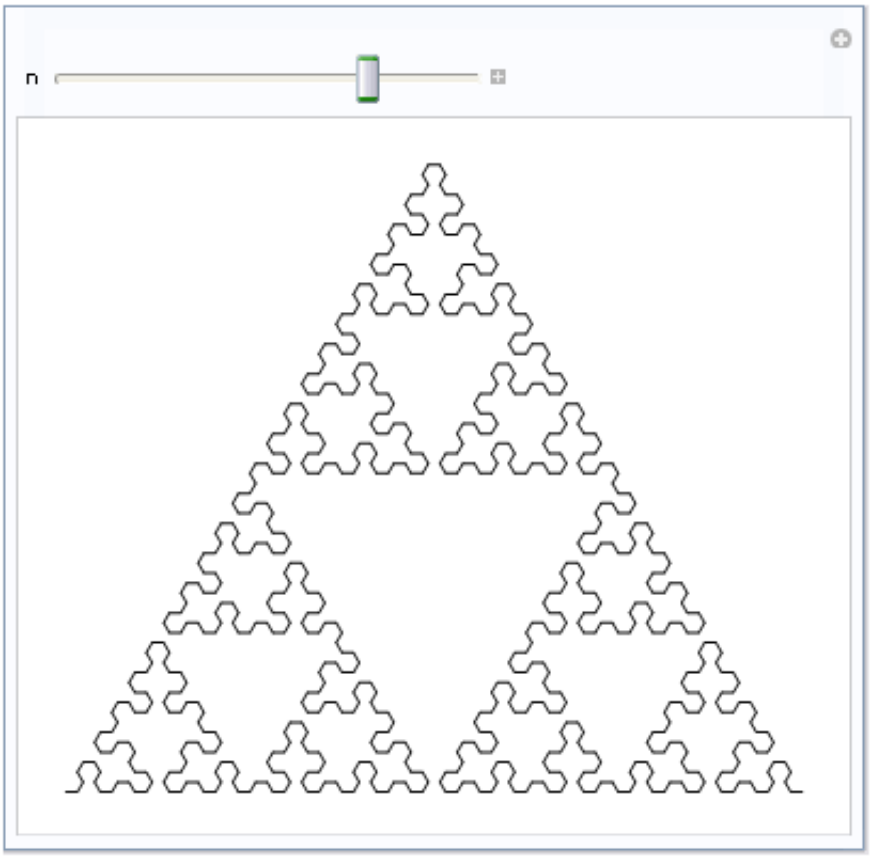


```
dragonsierpinski = .
dragonsierpinski[Line[{a_, b_}]] :=
```

```
{v = (b - a)/2.; c = a + RotationTransform[π/3.][v]; d = c + v; Line[{c, a}], Line[{c, d}], Line[{b, d}]}
```

```
dragonsierpinski[x_List] := Map[dragonsierpinski, Flatten[x]]
```

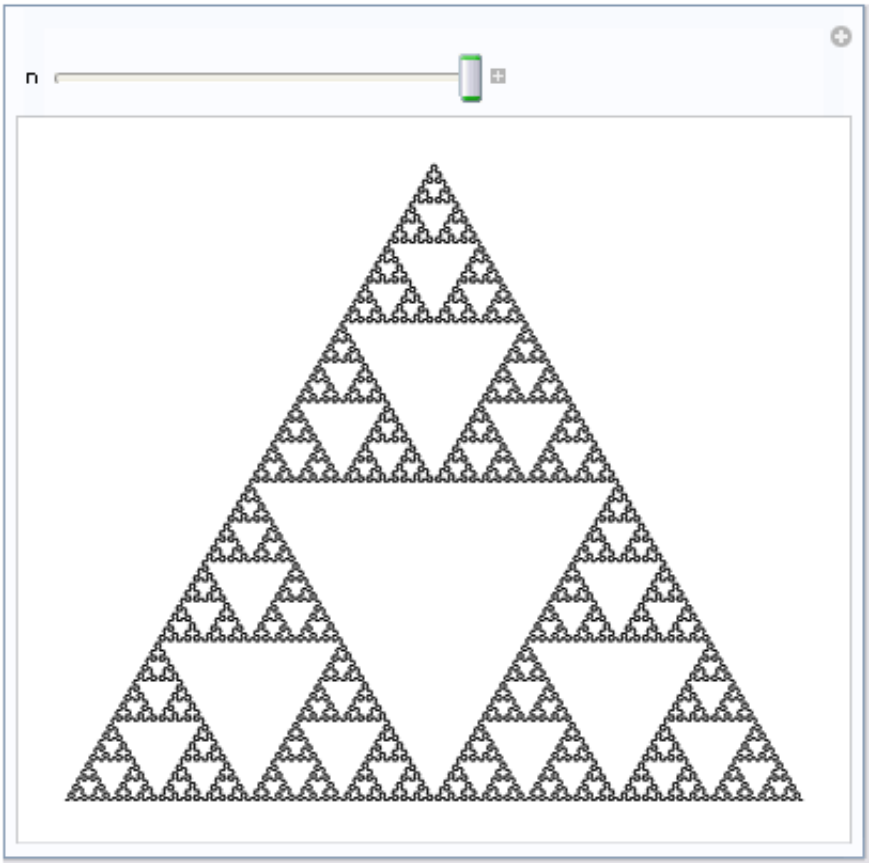
```
n =.
Manipulate[
     Graphics[
          Nest[dragonsierpinski, Line[{{0, 0}, {1, 0}}], n]],
     {n, 0, 8, 1}, SaveDefinitions → True]
```



Segunda solución:

```
Clear[a, b, c, d, v]
a = {0, 0}; b = {1, 0}; v = (b - a)/2; c = b + RotationTransform[120 °][v]; d = c - v;
Graphics[Line[{a, b}]]
Graphics[{Line[{b, c}], Line[{c, d}], Line[{d, a}]}]
```

```
dragonsierpinski =.
dragonsierpinski[Line[{a_, b_}], signo_] :=
        {v = (b - a)/2.; c = b + RotationTransform[signo 2 π/3.][v];
  d = c - v; Line[{b, c}], Line[{c, d}], Line[{d, a}]}
dragonsierpinski[x_List] := Thread[
  dragonsierpinski[Flatten[x], Table[(-1)^(k+1), {k, 1, Length[Flatten[x]]}]]]

n =.
Manipulate[
     Graphics[
          Nest[dragonsierpinski, List[Line[{{0, 0}, {1, 0}}]], n]],
     {n, 0, 8, 1}, SaveDefinitions → True]
```



Solución actividad 11:

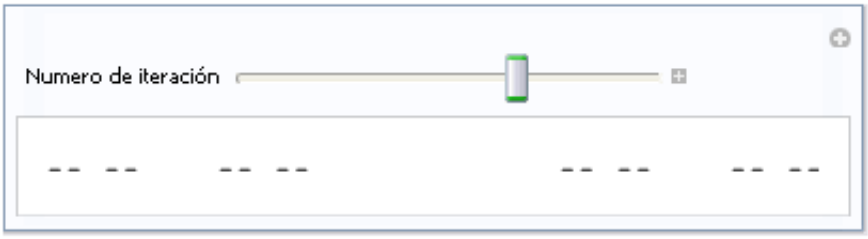
```
Graphics[{Green, Line[{{0, 0}, {1, 0}}]}]
```



```
Graphics[{Red, {Line[{{0, 0}, {1 / 3, 0}}], Line[{{2 / 3, 0}, {1, 0}}]}}]
```



```
cantor =.
cantor[Line[{a_, b_}]] :=
    {v = (b - a)/3.; Line[{a, a + v}], Line[{b - v, b}]}
cantor[x_List] := cantor /@ Flatten[x]

i =.
Manipulate[
     Graphics[
              Nest[cantor, Line[{{0, 0}, {1, 0}}], i], PlotRange → {{0, 1}, {-.025, .025}}],
    {{i, 3, "Numero de iteración"}, 0, 6, 1}, SaveDefinitions → True]
```



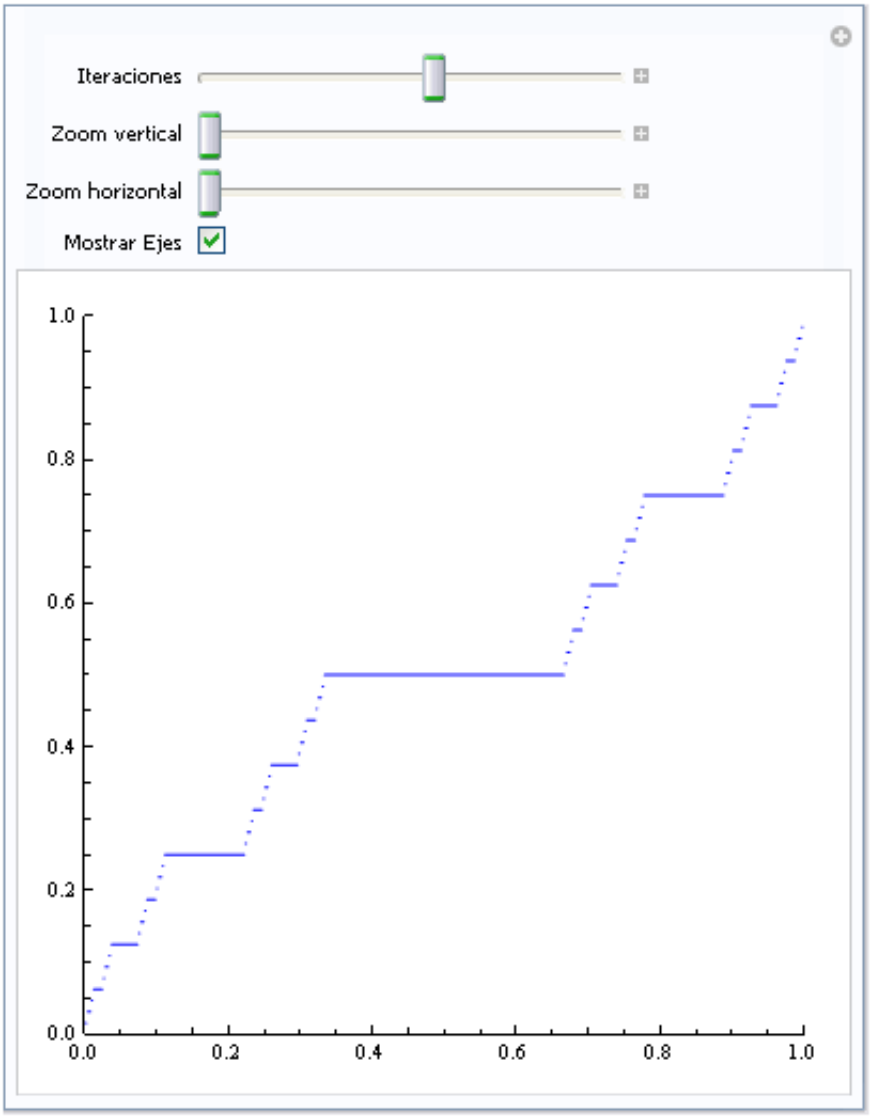
Solución actividad 12:

En el siguiente código se define la función de Cantor.

```
fcantor =.
fcantor[valoresiniciales[a_, fa_, b_, fb_]] :=
     {c = (b - a)/3.; fc = (fa + fb)/2.; valoresiniciales[a, fa, a + c, fc],
      Line[{{a + c, fc}, {b - c, fc}}], valoresiniciales[b - c, fc, b, fb]}
fcantor[Line[{{a_, fa_}, {b_, fb_}}]] := Line[{{a, fa}, {b, fb}}]
fcantor[x_List] := fcantor /@ Flatten[x]
```

Este código permite graficar la función de Cantor.

```
n =.
Manipulate[
     Graphics[{Blue,
        (Flatten[Nest[fcantor, valoresiniciales[0, 0, 1, 1], n]])[[Table[2 k, {k, 1, 2^n - 1}]]]},
       Axes → ejes, PlotRange → {{x, 1 / y}, {x, 1 / y}}],
     {{n, 2, "Iteraciones"}, 1, 10, 1}, {{y, 1, "Zoom vertical"}, 1, 10, 1},
     {{x, 0, "Zoom horizontal"}, 0, 1}, {{ejes, False, "Mostrar Ejes"}, {True, False}},
    SaveDefinitions → True]
```


Iteraciones
Zoom vertical
Zoom horizontal
Mostrar Ejes
1.0
0.8
0.6
0.4
0.2
0.0
0.0
0.2
0.4
0.6
0.8
1.0

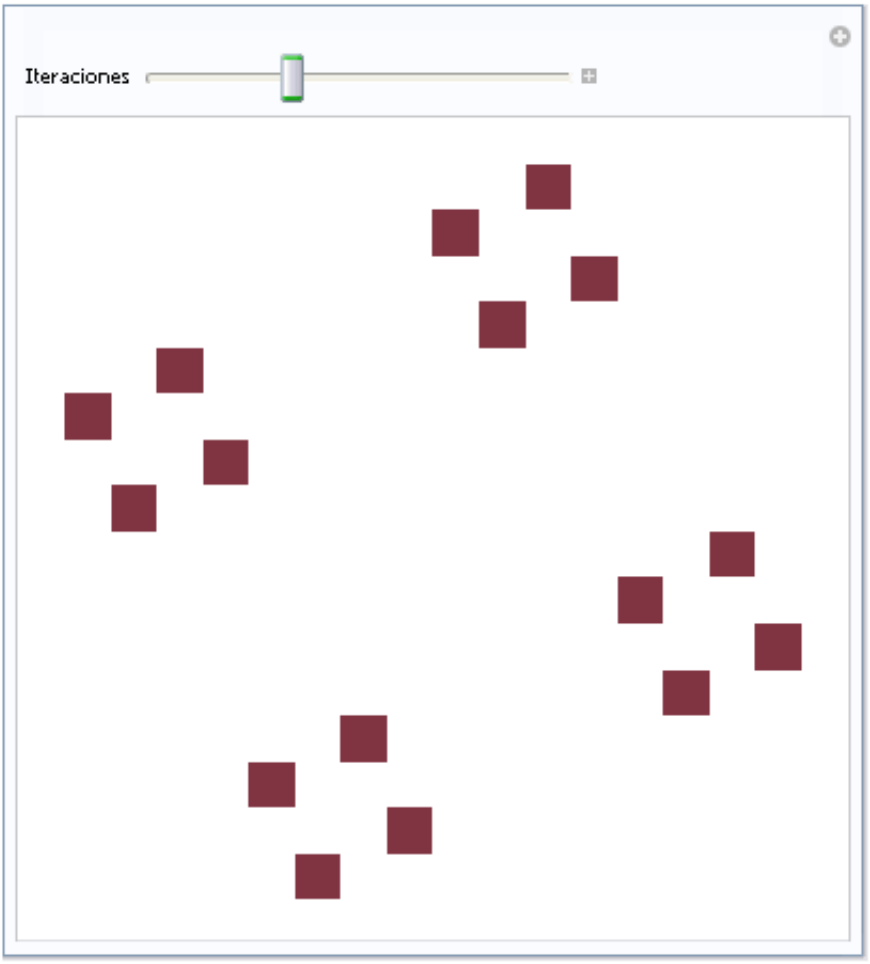
Solución actividad 13:

Primera solución:

```
(*Una variación del conjunto de cantor, dado los vértices diagonalmente opuestos*)

Clear[a, b]
a = {0, 0}; b = {1, 1}; Graphics[{Orange, Rectangle[a, b]}]
```

```
Clear[v, vx, vy, p1, p2, p3, p4, p5, p6, p7, p8]
v = (b - a)/4; vx = Projection[v, {1, 0}]; vy = Projection[2 v, {0, 1}]; p1 = a + vx;
p2 = p1 + v; p3 = p2 + vx; p4 = p3 + v; p5 = a + vy; p6 = p5 + v; p7 = p6 + vx; p8 = p7 + v;
Graphics[{RGBColor[0, 1, 0, .5], Rectangle[p1, p2], Rectangle[p3, p4], Rectangle[p5, p6],
          Rectangle[p7, p8]}]
```

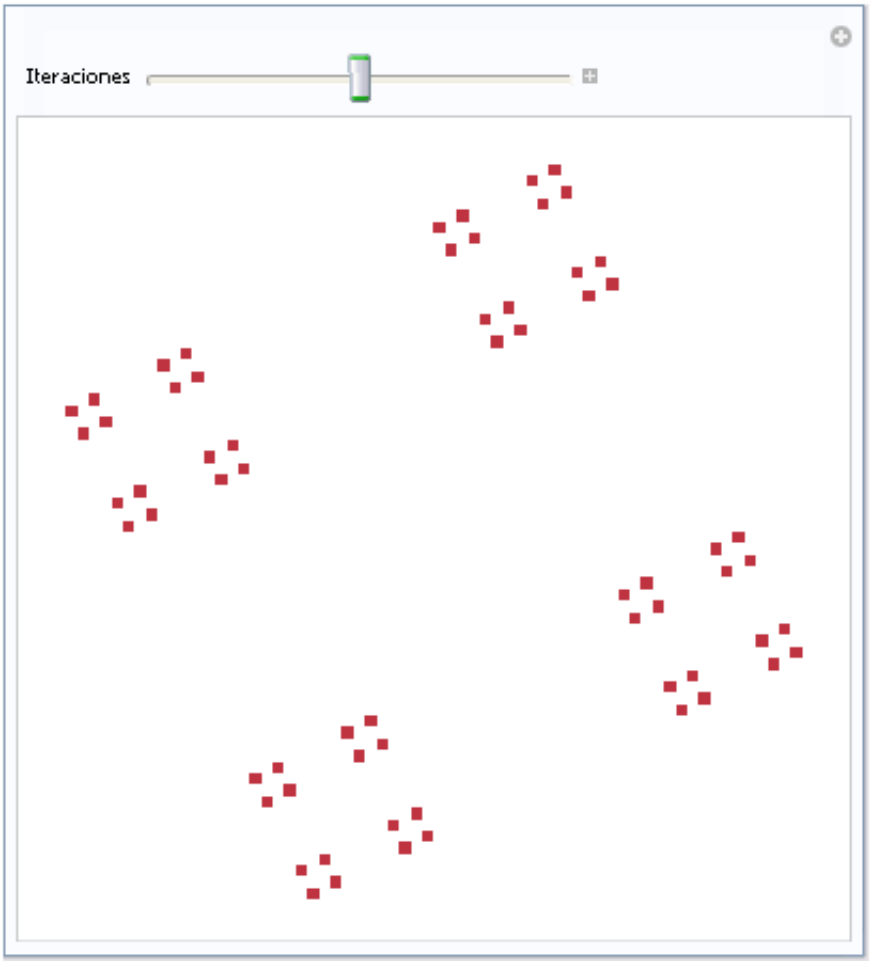
```
cantor2D =.
cantor2D[Rectangle[a_, b_]] :=
    {v = (b - a)/4.; vx = Projection[v, UnitVector[1], Dot]; vy = Projection[2 v, UnitVector[2], Dot];
  p1 = a + vx; p2 = p1 + v; p3 = p2 + vx; p4 = p3 + v; p5 = a + vy;
    p6 = p5 + v; p7 = p6 + vx; p8 = p7 + v; Rectangle[p1, p2],
  Rectangle[p3, p4], Rectangle[p5, p6], Rectangle[p7, p8]}
cantor2D[x_List] := cantor2D /@ Flatten[x]

n =.
Manipulate[
    Graphics[
        {RGBColor[1 / 2, 1 / 5, 1 / 4], Nest[cantor2D, Rectangle[{0, 0}, {1, 1}], n]}],
    {{n, 0, "Iteraciones"}, 0, 6, 1}, SaveDefinitions → True]
```



Segunda solución:

```
Clear[v, vx, vy, p1, p2, p3, p4, p5, p6, p7, p8]
v = (b - a)/4; vx = {v[[1]], 0}; vy = {0, 2 v[[2]]}; p1 = a + vx; p2 = p1 + v;
p3 = p2 + vx; p4 = p3 + v; p5 = a + vy; p6 = p5 + v; p7 = p6 + vx; p8 = p7 + v;
Graphics[{RGBColor[0, 1, 0, .5], Rectangle[p1, p2], Rectangle[p3, p4], Rectangle[p5, p6],
         Rectangle[p7, p8]}]
```

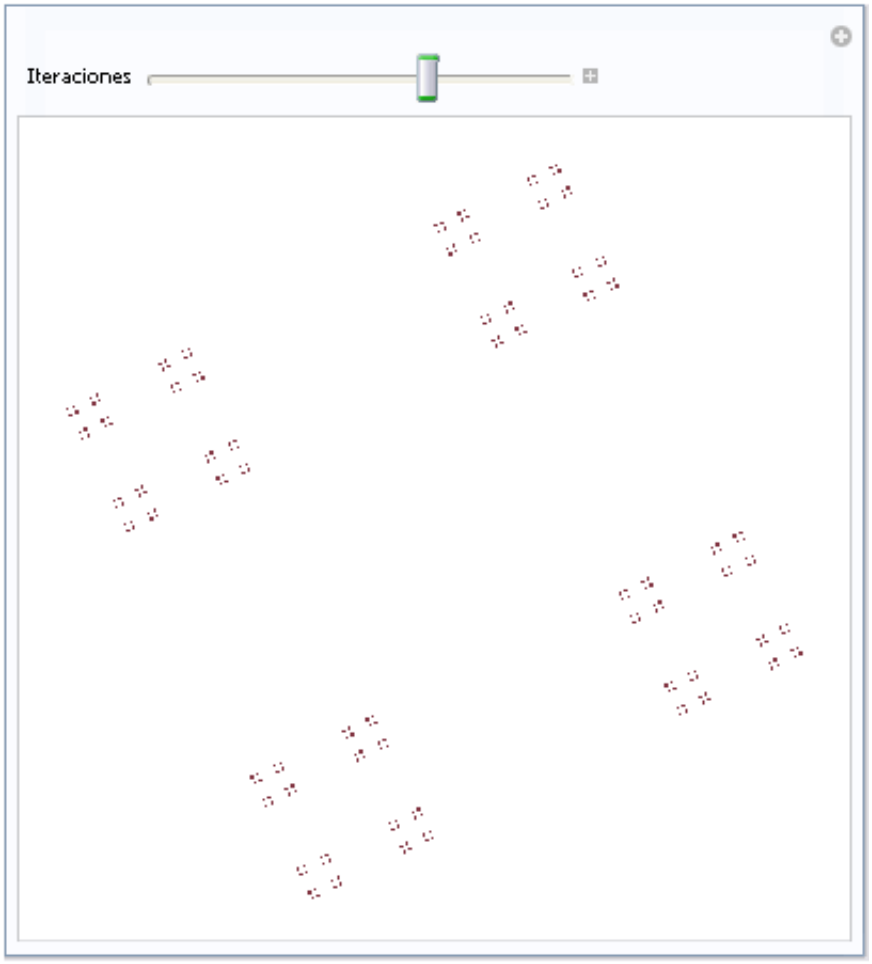
```
cantor2D = .
cantor2D[Rectangle[a_, b_]] :=
    {v = (b - a)/4.; vx = {v[[1]], 0}; vy = {0, 2 v[[2]]}; p1 = a + vx; p2 = p1 + v;
  p3 = p2 + vx; p4 = p3 + v; p5 = a + vy; p6 = p5 + v; p7 = p6 + vx; p8 = p7 + v;
    Rectangle[p1, p2], Rectangle[p3, p4], Rectangle[p5, p6], Rectangle[p7, p8]}
cantor2D[x_List] := cantor2D /@ Flatten[x]

n = .
Manipulate[
    Graphics[
        {RGBColor[3 / 4, 1 / 5, 1 / 4], Nest[cantor2D, Rectangle[{0, 0}, {1, 1}], n]}],
    {{n, 0, "Iteraciones"}, 0, 6, 1}, SaveDefinitions → True]
```



Tercera solución:

```
Clear[l, p1, p2, p3, p4, p5, p6, p7, p8]
l = Norm[b - a] / (4 √2); p1 = {a[[1]] + l, a[[2]]};
p2 = {a[[1]] + 2 l, a[[2]] + l}; p3 = {b[[1]] - l, a[[2]] + l};
p4 = {b[[1]], a[[2]] + 2 l}; p5 = {a[[1]], a[[2]] + 2 l}; p6 = {a[[1]] + l, b[[2]] - l};
p7 = {a[[1]] + 2 l, b[[2]] - l}; p8 = {b[[1]] - l, b[[2]]};
Graphics[{RGBColor[0, 1, 0, .5], Rectangle[p1, p2], Rectangle[p3, p4], Rectangle[p5, p6],
          Rectangle[p7, p8]}]
```

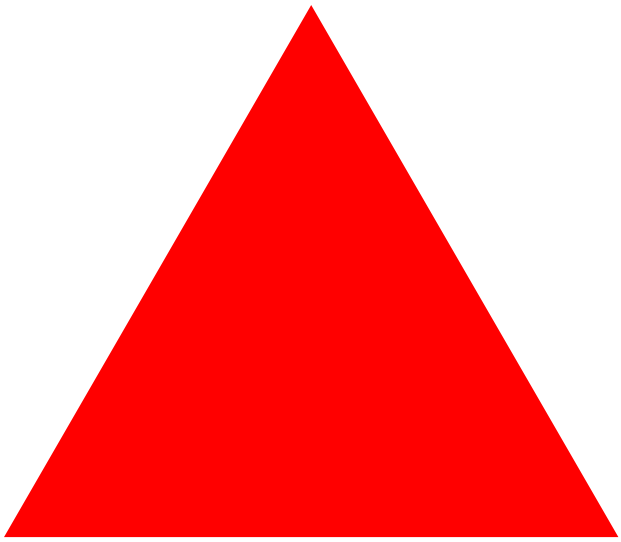
```
cantor2D = .
cantor2D[Rectangle[a_, b_]] :=
    {l = Norm[b - a] / (4 Sqrt[2.]); p1 = {a[[1]] + l, a[[2]]}; p2 = {a[[1]] + 2 l, a[[2]] + l};
    p3 = {b[[1]] - l, a[[2]] + l}; p4 = {b[[1]], a[[2]] + 2 l}; p5 = {a[[1]], a[[2]] + 2 l};
    p6 = {a[[1]] + l, b[[2]] - l}; p7 = {a[[1]] + 2 l, b[[2]] - l}; p8 = {b[[1]] - l, b[[2]]};
    Rectangle[p1, p2], Rectangle[p3, p4], Rectangle[p5, p6], Rectangle[p7, p8]}
cantor2D[x_List] := cantor2D /@ Flatten[x]

n = .
Manipulate[
    Graphics[
        {RGBColor[1 / 2, 1 / 5, 1 / 4], Nest[cantor2D, Rectangle[{0, 0}, {1, 1}], n]}],
    {{n, 0, "Iteraciones"}, 0, 6, 1}, SaveDefinitions → True]
```


Iteraciones

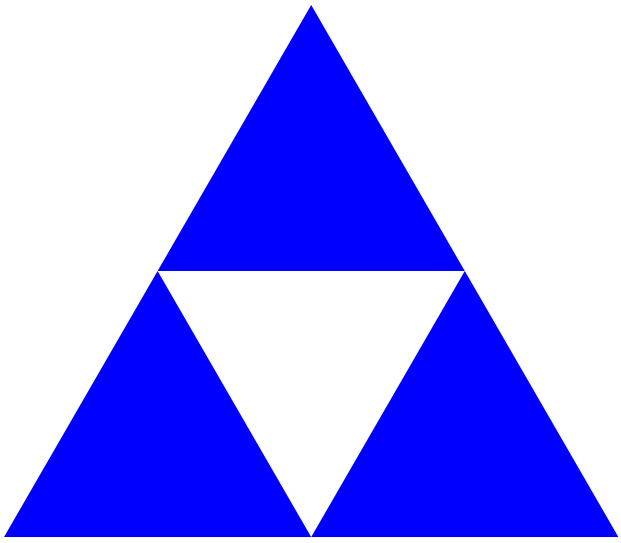
Solución actividad 14:

Primera solución:

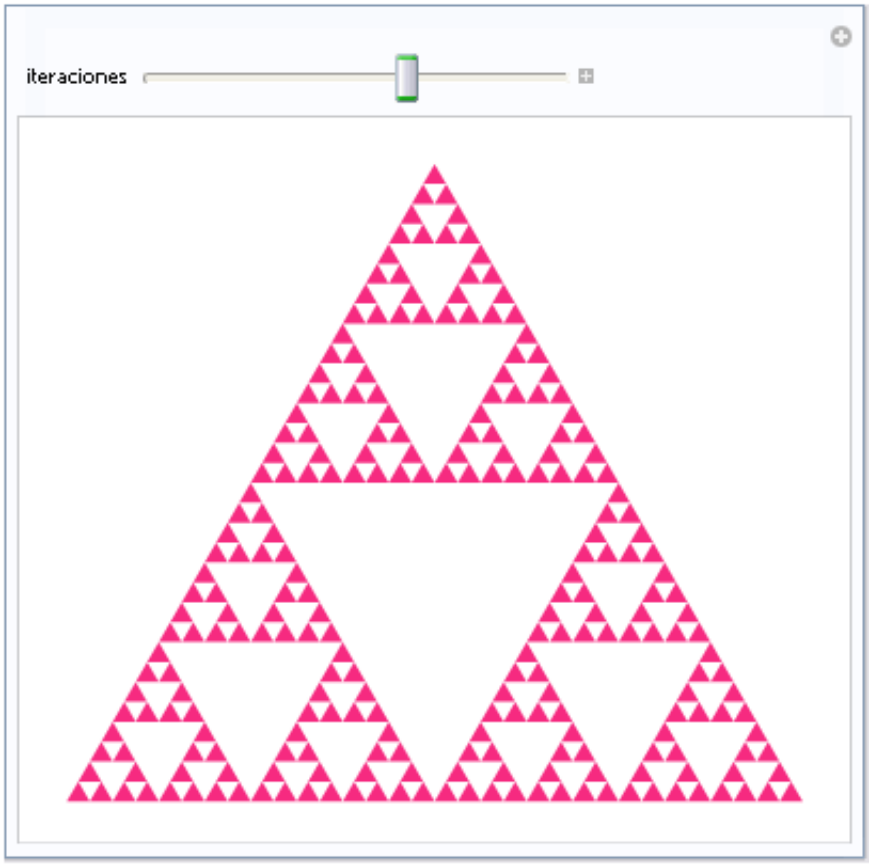
```
Clear[a, b, c]
a = {0, 0}; b = {1, 0}; c = RotationTransform[60 °, a][b];
Graphics[{Red, Polygon[{a, b, c}]}]
```



```
Clear[v, ab, bc, ac]
v = (b - a)/2; ab = a + v; ac = a + RotationTransform[60 °][v]; bc = ac + v;
Graphics[{Blue, {Polygon[{a, ab, ac}], Polygon[{b, ab, bc}], Polygon[{c, ac, bc}]}}]
```



```
sierpinski =.
sierpinski[Polygon[{a_, b_, c_}]] :=
 (*Otra ordenación de los vértices es:{a,ab,ac}, {ab,b,bc},{ac,bc,c}*)
    {v = (b - a)/2.; ab = a + v; ac = a + RotationTransform[π/3.][v]; bc = ac + v;
  Polygon[{a, ab, ac}], Polygon[{b, bc, ab}], Polygon[{c, ac, bc}]}
sierpinski[x_List] := sierpinski /@ Flatten[x]

n =.
Manipulate[
    Graphics[
            {RGBColor[n^2/(1 + n^2), 1/(1 + n), n/(5 + n)],
            Nest[sierpinski, Polygon[{{0, 0}, {1, 0}, {1/2, Sqrt[3]/2}}], n]}],
    {{n, 4, "iteraciones"}, 0, 8, 1},
    SaveDefinitions → True]
```

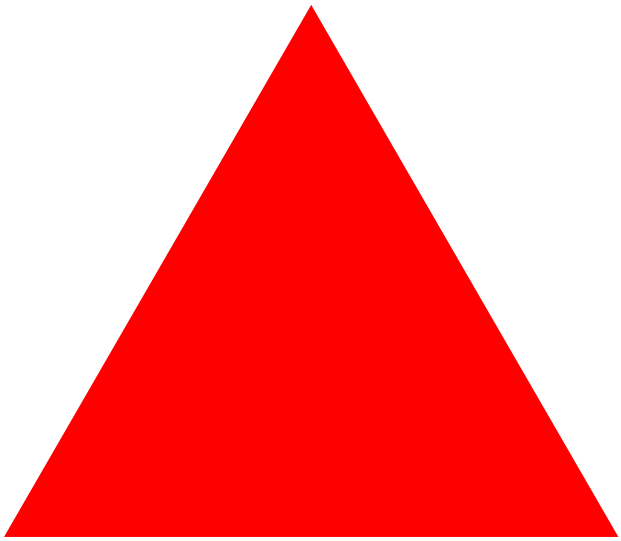


Segunda solución:

```
Clear[a, b, c]
```

$$a = \{0, 0\}; b = \{1, 0\}; c = \left\{\frac{1}{2}, \frac{\sqrt{3}}{2}\right\};$$

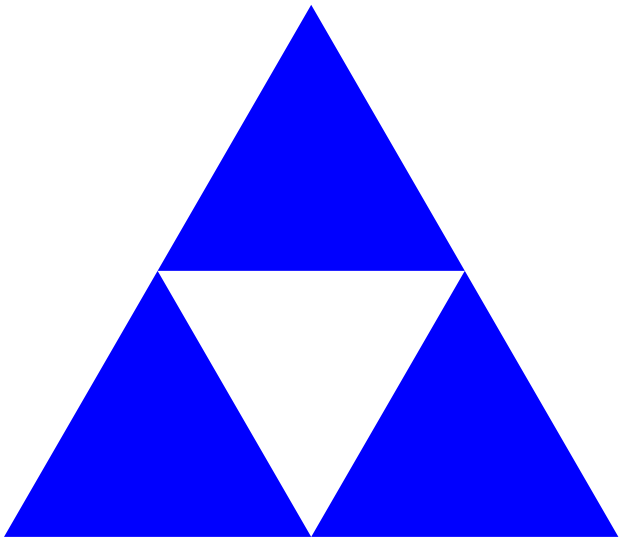
```
Graphics[{Red, Polygon[{a, b, c}]}]
```



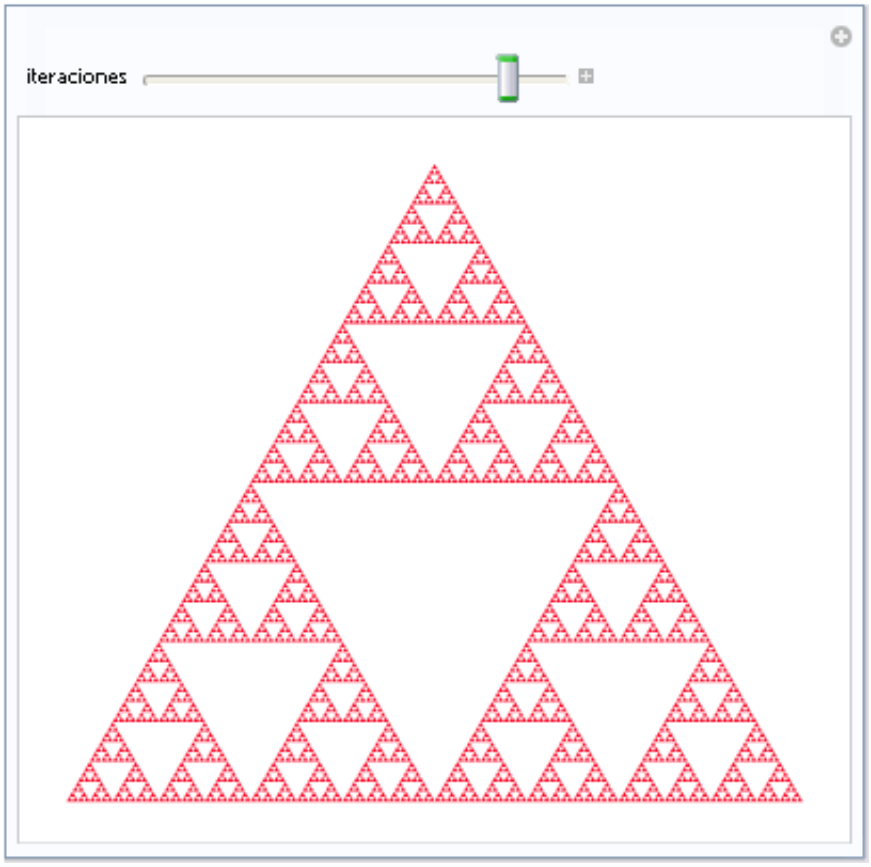
```
Clear[ab, bc, ac]
```

$$ab = \frac{a + b}{2}; bc = \frac{b + c}{2}; ac = \frac{a + c}{2};$$

```
Graphics[{Blue, {Polygon[{a, ab, ac}], Polygon[{b, ab, bc}], Polygon[{c, ac, bc}]}}]
```



```
sierpinski =.
sierpinski[Polygon[{a_, b_, c_}]] :=
    {ab = (a + b)/2.; bc = (b + c)/2.; ac = (a + c)/2.;
  Polygon[{a, ab, ac}], Polygon[{ab, b, bc}], Polygon[{ac, bc, c}]}
sierpinski[x_List] := sierpinski /@ Flatten[x]

n =.
Manipulate[
    Graphics[
            {RGBColor[n^2/(1 + n^2), 0, 1/(1 + n)],
            Nest[sierpinski, Polygon[{{0, 0}, {1, 0}, {1/2, Sqrt[3]/2}}], n]}],
    {{n, 4, "iteraciones"}, 0, 8, 1},
    SaveDefinitions → True]
```

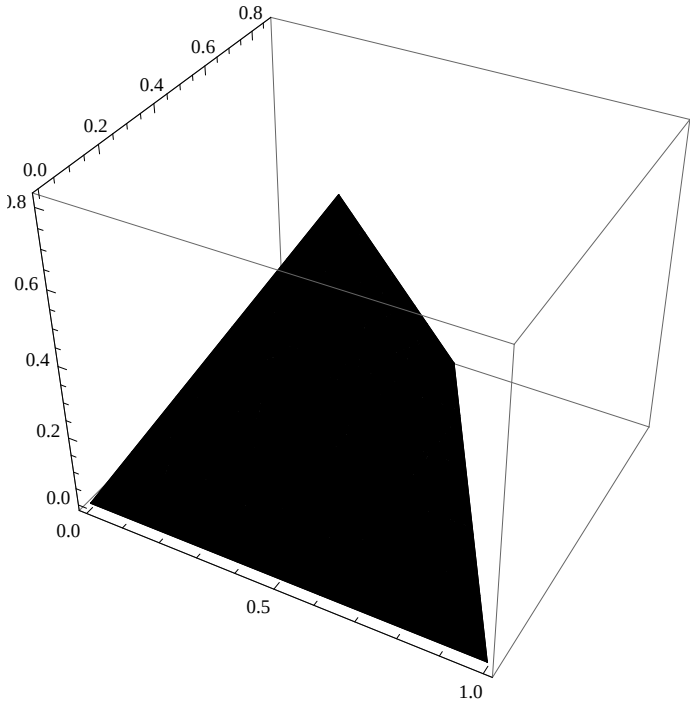


Solución actividad 16:

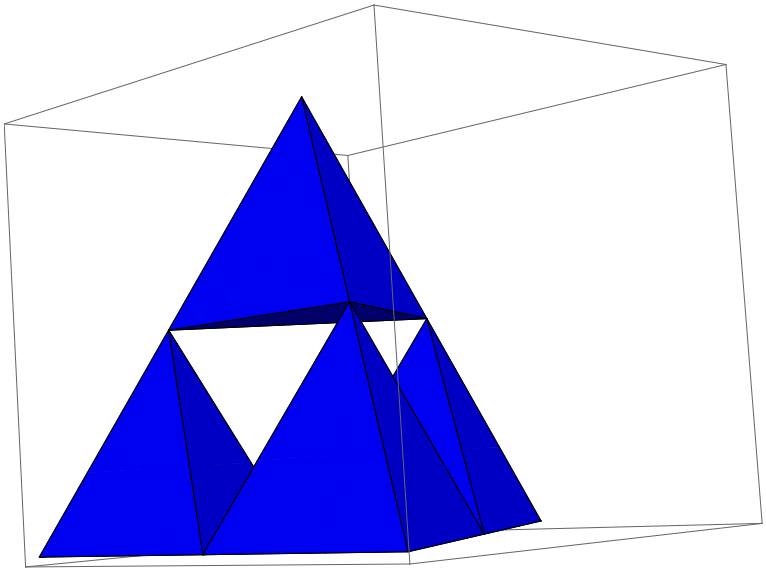
```
Clear[v, j]
```

$$\mathbf{v} = \left\{\{0, 0, 0\}, \{1, 0, 0\}, \left\{\frac{1}{2}, \frac{\sqrt{3}}{2}, 0\right\}, \left\{\frac{1}{2}, \frac{\sqrt{3}}{6}, \frac{\sqrt{6}}{3}\right\}\right\};$$

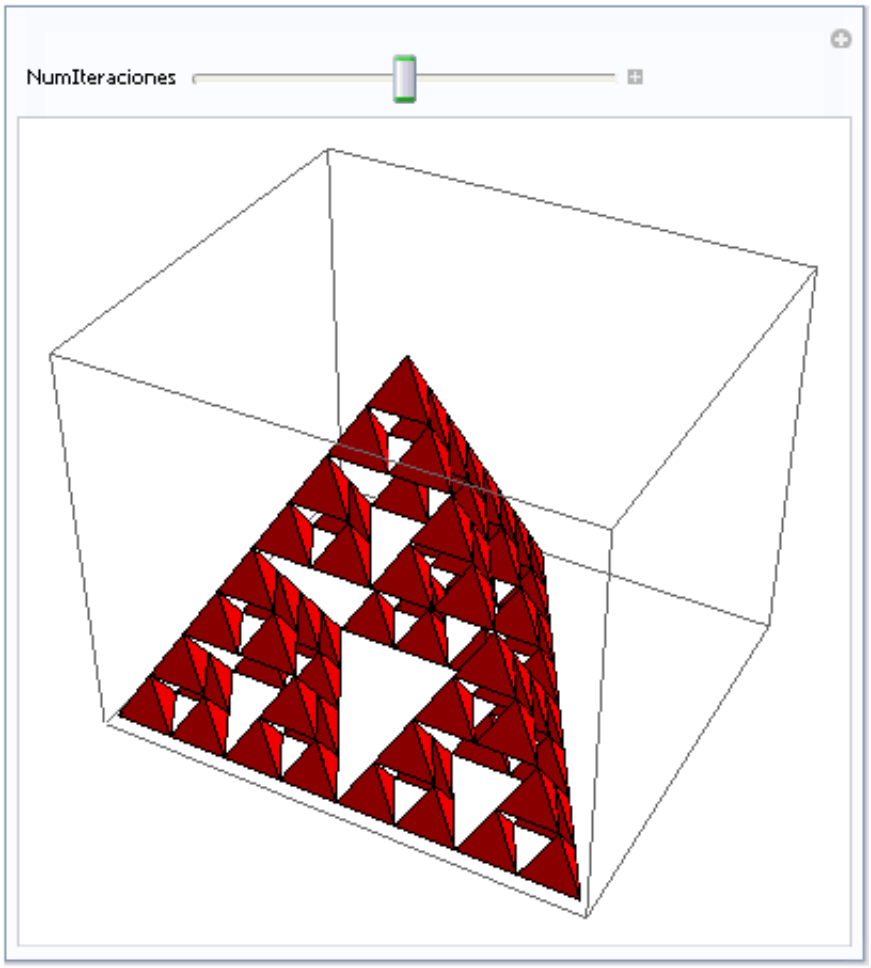
```
j = {{1, 2, 3}, {1, 2, 4}, {1, 3, 4}, {2, 3, 4}};
Graphics3D[{Black, GraphicsComplex[v, Polygon[j]]}, Axes → True]
```



```
Clear[v1, v2, v3, v4]
v1 = Table[(v[[1]] + v[[n]])/2., {n, 1, 4}]; v2 = Table[(v[[2]] + v[[n]])/2., {n, 1, 4}];
v3 = Table[(v[[3]] + v[[n]])/2., {n, 1, 4}]; v4 = Table[(v[[4]] + v[[n]])/2., {n, 1, 4}];
Graphics3D[{Blue, {GraphicsComplex[v1, Polygon[j]], GraphicsComplex[v2, Polygon[j]],
    GraphicsComplex[v3, Polygon[j]], GraphicsComplex[v4, Polygon[j]]}}]
```



```
tetraedro =.
tetraedro[GraphicsComplex[v_, k_]] :=
    {v1 = Table[(v[[1]] + v[[n]])/2., {n, 1, 4}]; v2 = Table[(v[[2]] + v[[n]])/2., {n, 1, 4}];
    v3 = Table[(v[[3]] + v[[n]])/2., {n, 1, 4}]; v4 = Table[(v[[4]] + v[[n]])/2., {n, 1, 4}];
    GraphicsComplex[v1, k], GraphicsComplex[v2, k], GraphicsComplex[v3, k],
    GraphicsComplex[v4, k]}
tetraedro[x_List] := tetraedro /@ Flatten[x]

Clear[v0, i, n]

v0 = {{0, 0, 0}, {1, 0, 0}, {1/2, Sqrt[3]/2, 0}, {1/2, Sqrt[3]/6, Sqrt[6]/3}};
i = {{1, 2, 3}, {1, 2, 4}, {1, 3, 4}, {2, 3, 4}};
Manipulate[
    Graphics3D[
        {RGBColor[1, 0, 0], Nest[tetraedro, GraphicsComplex[v0, Polygon[i]], n]}],
    {{n, 0, "NumIteraciones"}, 0, 6, 1}, SaveDefinitions → True]
```

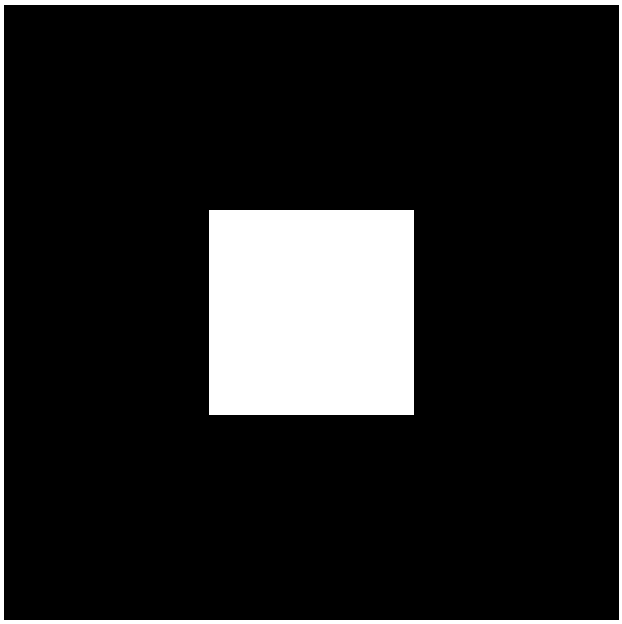


Solución actividad 17:

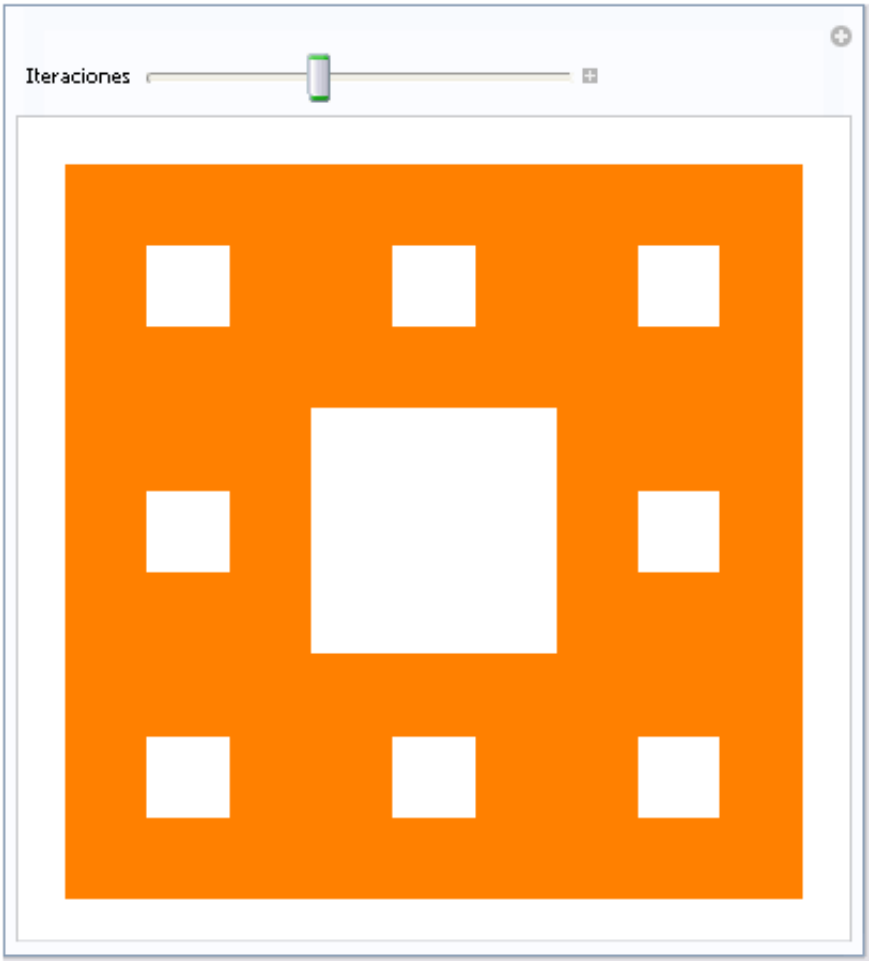
Primera solución: Aquí se presentan códigos que utilizan *Rectangle* en vez de *Poligon* y se emplean rotaciones:

```
Clear[a, b]
a = {0, 0}; b = {1, 1}; Graphics[Rectangle[a, b]]
```

```
Clear[p1, p2, p3, p4, p5, p6, v]
v = (b - a)/3.; p1 = a + v; p2 = p1 + RotationTransform[-90 °][v]; p3 = p2 + v; p4 = b - v;
p5 = p1 + RotationTransform[90 °][v]; p6 = p5 + v;
Graphics[{Rectangle[a, p1], Rectangle[p1, p2], Rectangle[p2, p3], Rectangle[p3, p4],
         Rectangle[p4, b], Rectangle[p1, p5], Rectangle[p5, p6], Rectangle[p4, p6]}]
```



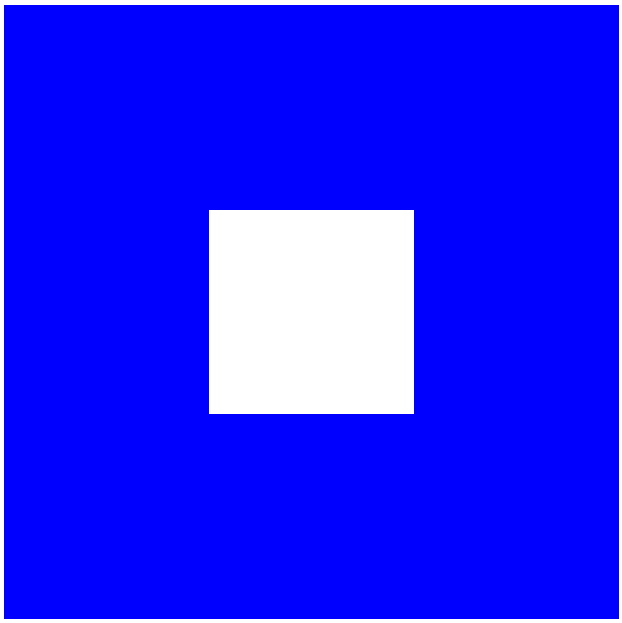
```
alfombra =.
alfombra[Rectangle[a_, b_]] :=
    {v = (b - a)/3.; p1 = a + v; p2 = p1 + RotationTransform[-π/2.][v];
  p3 = p2 + v; p4 = b - v; p5 = p1 + RotationTransform[π/2.][v]; p6 = p5 + v;
    Rectangle[a, p1], Rectangle[p1, p2], Rectangle[p2, p3], Rectangle[p3, p4],
    Rectangle[p4, b], Rectangle[p1, p5], Rectangle[p5, p6], Rectangle[p4, p6]}
alfombra[x_List] := alfombra /@ Flatten[x]

n =.
Manipulate[
    Graphics[{Orange, Nest[alfombra, Rectangle[{0, 0}, {1, 1}], n]}],
    {{n, 1, "Iteraciones"}, 0, 5, 1}, SaveDefinitions → True]
```



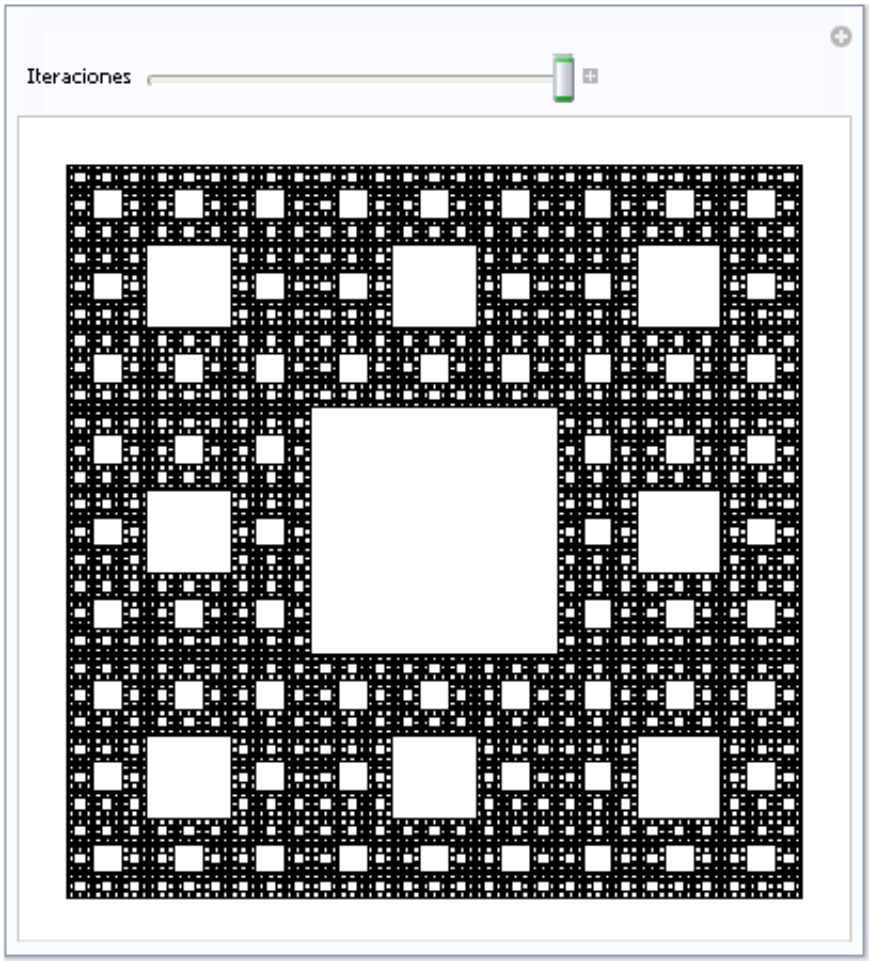
```
Clear[a, b]
a = {0, 0}; b = {1, 1}; Graphics[{Yellow, Rectangle[a, b]}]
```



```
Clear[p1, p2, p3, p4, p5, p6]
p1 = (2 a + b)/3.; p2 = (2 b + a)/3.; p6 = {a[[1]], p2[[2]]}; p3 = {p2[[1]], a[[2]]}; p4 = {b[[1]], p1[[2]]};
p5 = {p1[[1]], b[[2]]};
Graphics[{Blue, Rectangle[a, p1], Rectangle[p1, p3], Rectangle[p3, p4],
          Rectangle[p2, p4], Rectangle[p2, b], Rectangle[p2, p5], Rectangle[p5, p6],
          Rectangle[p1, p6]}]
```

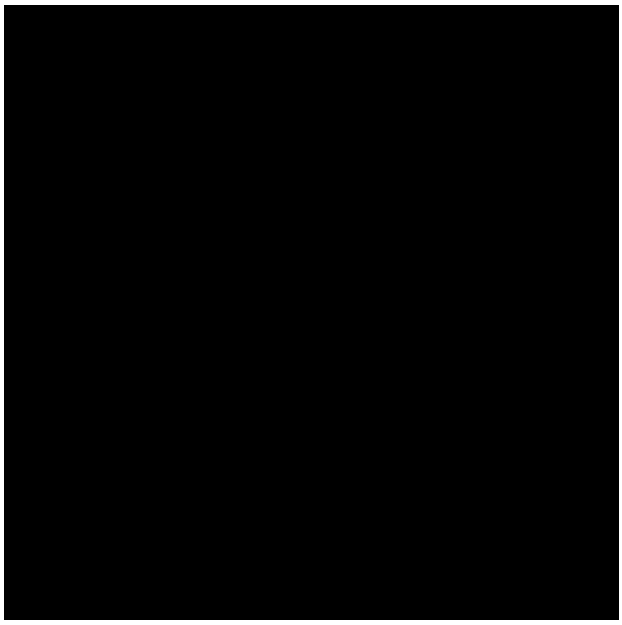


```
alfombra =.
alfombra[Rectangle[a_, b_]] :=
    {p1 = (2 a + b)/3.; p2 = (2 b + a)/3.; p3 = {p2[[1]], a[[2]]}; p4 = {b[[1]], p1[[2]]};
    p5 = {p1[[1]], b[[2]]}; p6 = {a[[1]], p2[[2]]};
  Rectangle[a, p1], Rectangle[p1, p3], Rectangle[p3, p4],
    Rectangle[p2, p4], Rectangle[p2, b], Rectangle[p2, p5],
  Rectangle[p5, p6], Rectangle[p1, p6]}
alfombra[x_List] := alfombra /@ Flatten[x]

n =.
Manipulate[
    Graphics[Nest[alfombra, Rectangle[{0, 0}, {1, 1}], n]],
    {{n, 0, "Iteraciones"}, 0, 5, 1}, SaveDefinitions → True]
```

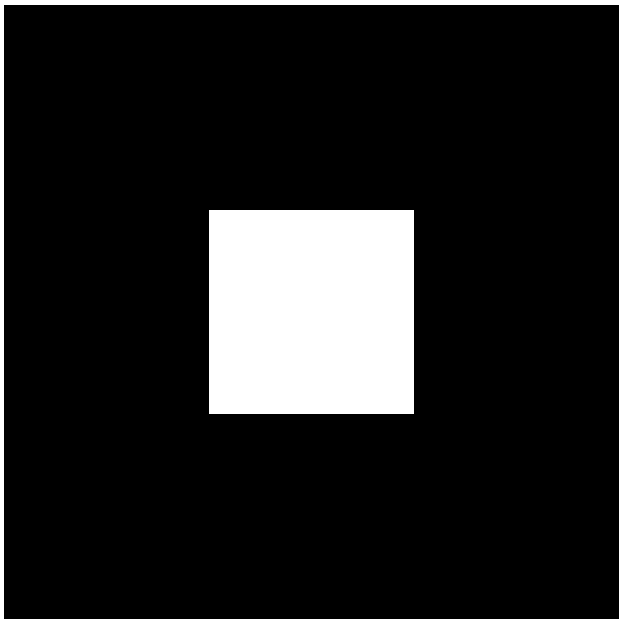


Tercera solución: En esta solución se determinan primero los puntos sobre la diagonal principal y luego se rotan uno alrededor del otro para obtener los puntos restantes.

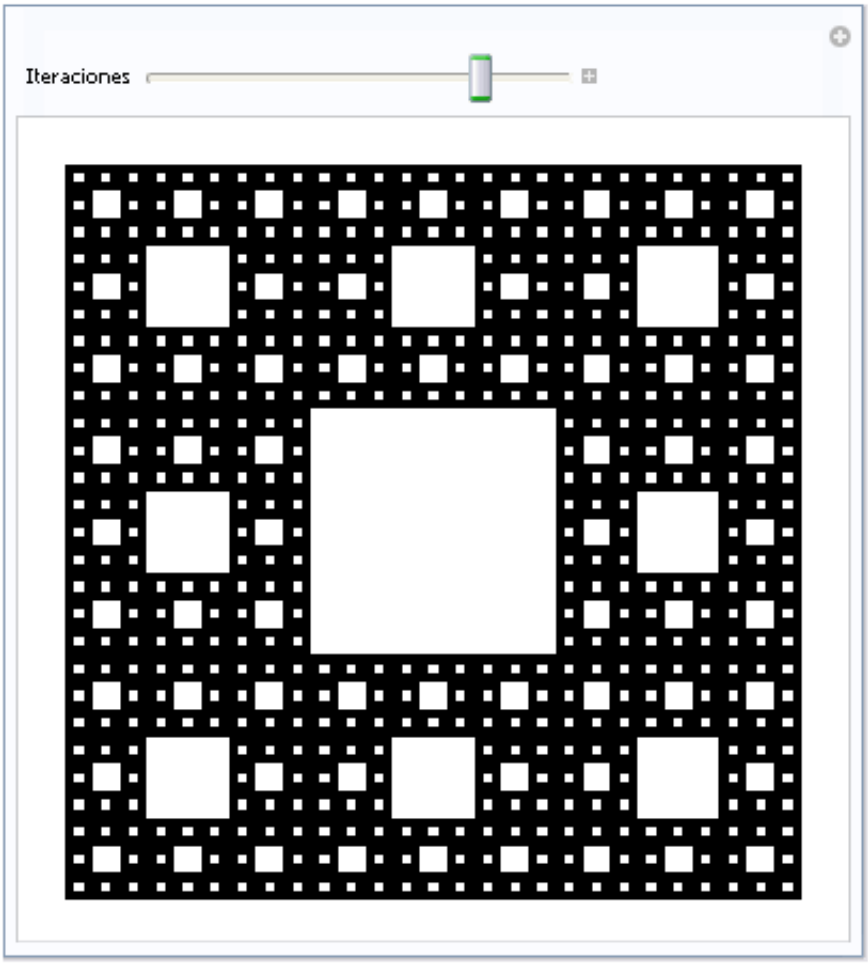
```
a = {0, 0}; b = {1, 1}; Graphics[Rectangle[a, b]]
```



```
c = a + (b - a)/3.; d = (a - b)/3. + b; e = RotationTransform[90 °, c][d]; f = RotationTransform[-90 °, c][d];
g = RotationTransform[90 °, d][c]; h = RotationTransform[-90 °, d][c];

Graphics[{Rectangle[a, c], Rectangle[c, e], Rectangle[f, c], Rectangle[f, g], Rectangle[g, d],
          Rectangle[d, b], Rectangle[d, h], Rectangle[e, h]}]
```



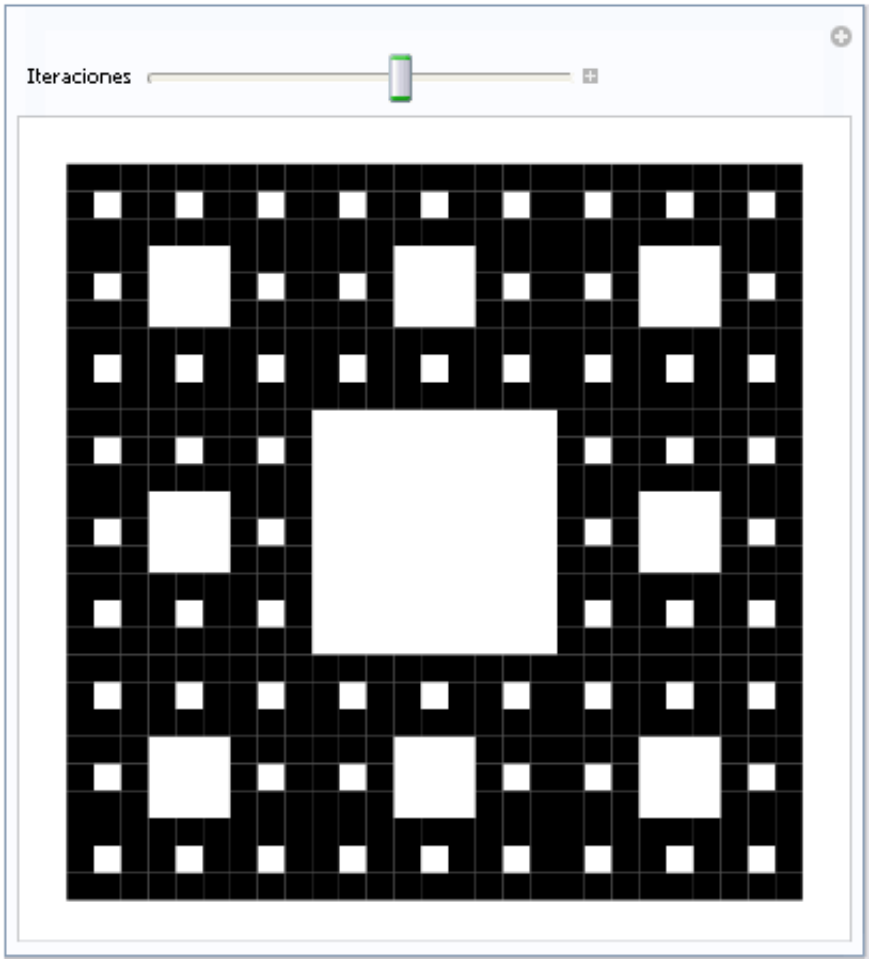
```
alfombra =.
alfombra[Rectangle[a_, b_]] :=
    {c = a + (b - a)/3.; d = (a - b)/3. + b;
  e = RotationTransform[π/2., c][d]; f = RotationTransform[-π/2., c][d];
      g = RotationTransform[π/2., d][c]; h = RotationTransform[-π/2., d][c];
    Rectangle[a, c], Rectangle[c, e], Rectangle[f, c], Rectangle[f, g], Rectangle[g, d],
    Rectangle[d, b], Rectangle[d, h], Rectangle[e, h]}
alfombra[x_List] := alfombra /@ Flatten[x]

n =.
Manipulate[
    Graphics[
        Nest[alfombra, Rectangle[{0, 0}, {1, 1}], n]],
    {{n, 0, "Iteraciones"}, 0, 5, 1}, SaveDefinitions → True]
```



Cuarta solución:

```
Clear[a, b, c, d]
a = {0, 0}; b = {1, 0}; c = {1, 1}; d = {0, 1};
Graphics[{Brown, Polygon[{a, b, c, d}]}]
```

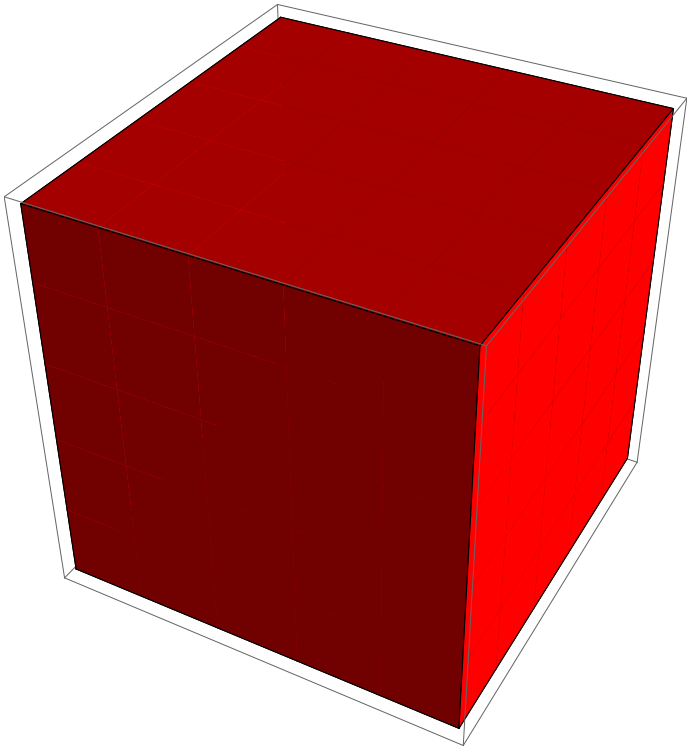
```
alfombra =.
alfombra[Polygon[{a_, b_, c_, d_}]] :=
    {e = (2 a + b)/3.; f = (a + 2 b)/3.; g = (2 b + c)/3.; h = (2 c + b)/3.;
  i = (2 c + d)/3.; j = (2 d + c)/3.; k = (2 d + a)/3.; l = (2 a + d)/3.; m = (2 a + c)/3.; n = (2 c + a)/3.;
    o = (2 b + d)/3.; p = (2 d + b)/3.; Polygon[{a, e, m, l}], Polygon[{e, f, o, m}], Polygon[{f, b, g, o}],
    Polygon[{o, g, h, n}], Polygon[{n, h, c, i}], Polygon[{p, n, i, j}], Polygon[{k, p, j, d}],
    Polygon[{l, m, p, k}]}
alfombra[x_List] := alfombra /@ Flatten[x]

s =.
Manipulate[
    Graphics[Nest[alfombra, Polygon[{{0, 0}, {1, 0}, {1, 1}, {0, 1}}], s]],
    {{s, 0, "Iteraciones"}, 0, 5, 1}, SaveDefinitions → True]
```


Iteraciones

Solución actividad 18:

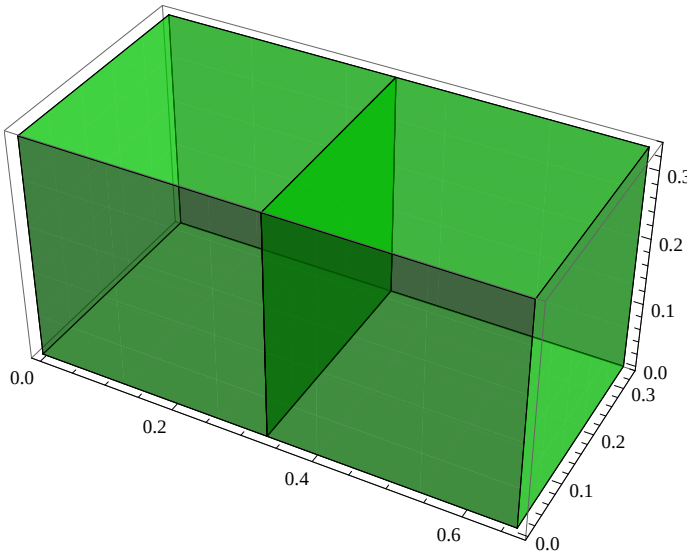
Primera solución:

```
Clear[a, b]; a = {0, 0, 0}; b = {1, 1, 1}; Graphics3D[{Red, Cuboid[a, b]}]
```



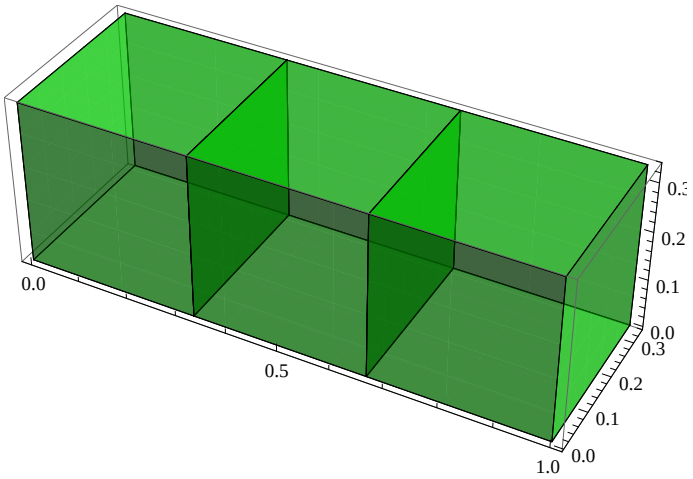
Ahora se dibuja por partes el resultado para corroborar que no se tienen errores en el diseño.

```
Clear[v, vx, p1, p2]; v = (b - a)/3.; vx = RotationTransform[180 °, UnitVector[3, 1]][v]; p1 = a + v;
p2 = p1 + vx;
Graphics3D[{RGBColor[0, 1, 0, .5], Cuboid[a, p1], Cuboid[p1, p2]}, Axes → True]
```



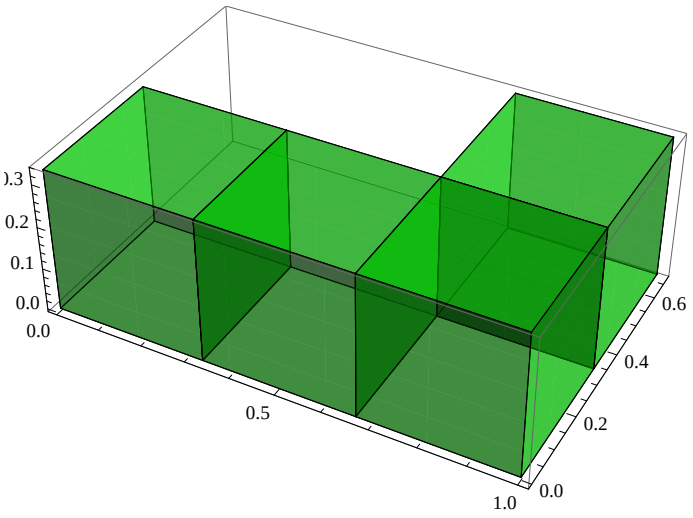
```
p3 = .; p3 = p2 + v;
```

```
Graphics3D[{RGBColor[0, 1, 0, .5], Cuboid[a, p1], Cuboid[p1, p2], Cuboid[p2, p3]}, Axes → True]
```



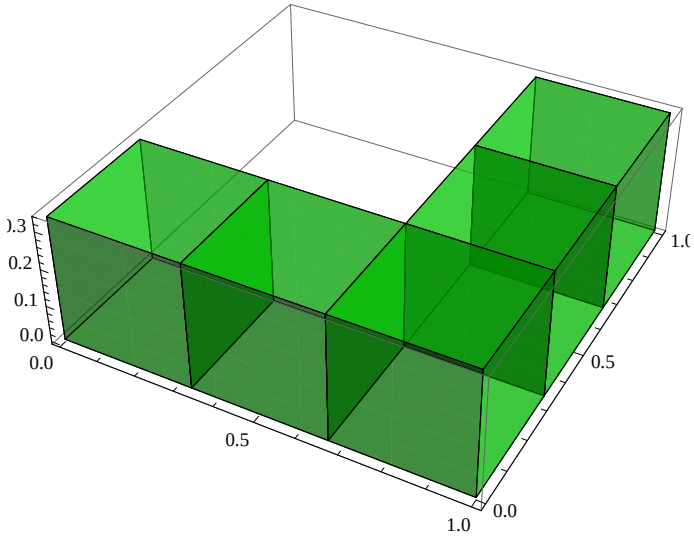

```
Clear[vy, p4]; vy = RotationTransform[180 °, UnitVector[3, 2]][v]; p4 = p3 + vy;
```

```
Graphics3D[{RGBColor[0, 1, 0, .5], Cuboid[a, p1], Cuboid[p1, p2], Cuboid[p2, p3],
            Cuboid[p3, p4]}, Axes → True]
```



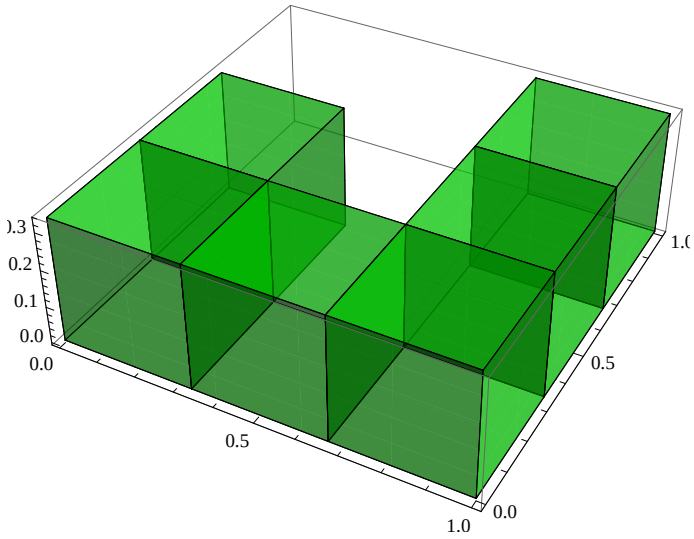
```
p5 =.; p5 = p4 + v;
```

```
Graphics3D[{RGBColor[0, 1, 0, .5], Cuboid[a, p1], Cuboid[p1, p2], Cuboid[p2, p3],
            Cuboid[p3, p4], Cuboid[p4, p5]}, Axes → True]
```



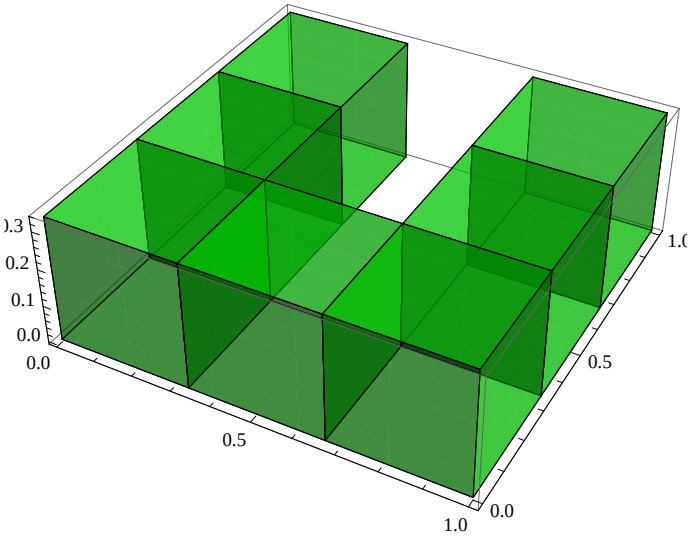

```
p6 =.; p6 = p1 + vy;
```

```
Graphics3D[{RGBColor[0, 1, 0, .5], Cuboid[a, p1], Cuboid[p1, p2], Cuboid[p2, p3],
            Cuboid[p3, p4], Cuboid[p4, p5], Cuboid[p1, p6]}, Axes → True]
```



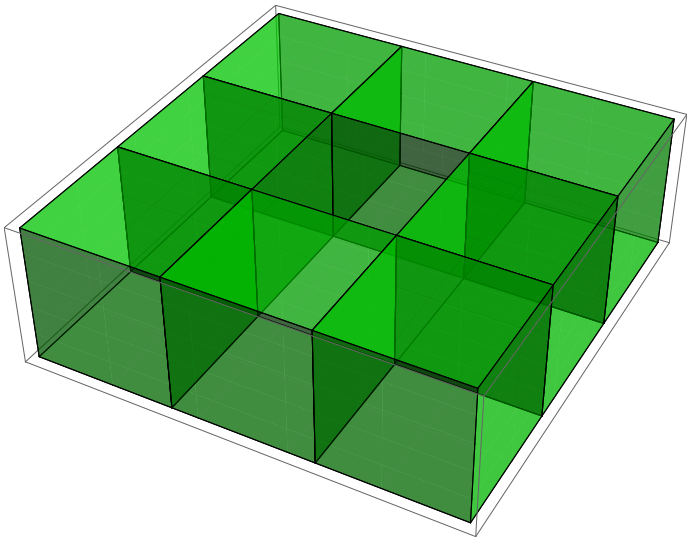
```
p7 = .; p7 = p6 + v;
```

```
Graphics3D[{RGBColor[0, 1, 0, .5], Cuboid[a, p1], Cuboid[p1, p2], Cuboid[p2, p3],
            Cuboid[p3, p4], Cuboid[p4, p5], Cuboid[p1, p6], Cuboid[p6, p7]}, Axes → True]
```

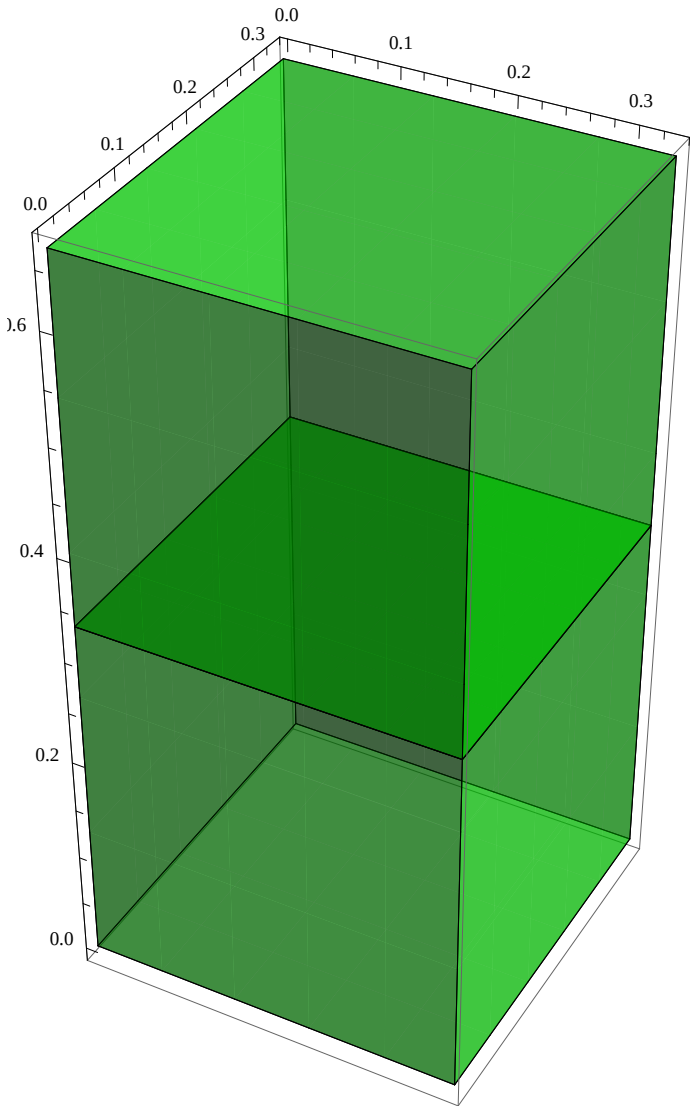



```
Clear[vz, p8]; vz = RotationTransform[180 °, UnitVector[3, 3]][v]; p8 = p1 + vz;
```

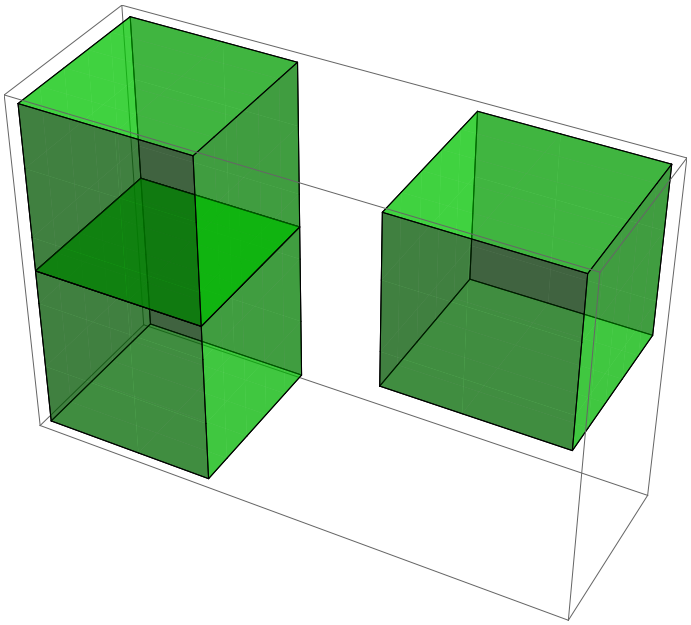
```
Graphics3D[{RGBColor[0, 1, 0, .5], Cuboid[a, p1], Cuboid[p1, p2], Cuboid[p2, p3],
            Cuboid[p3, p4], Cuboid[p4, p5], Cuboid[p1, p6], Cuboid[p6, p7], Cuboid[p4, p7]}]
```



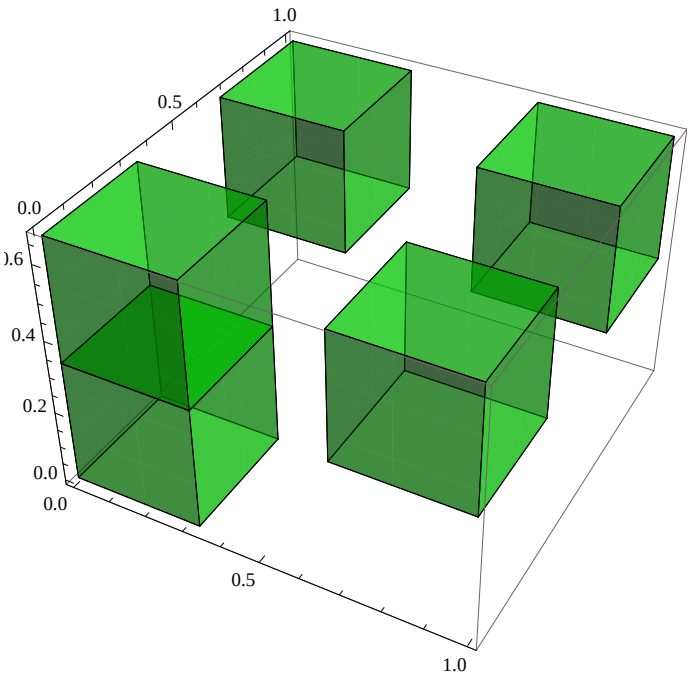
```
p9 = .; p9 = p3 + vz;

Graphics3D[{RGBColor[0, 1, 0, .5], Cuboid[a, p1], Cuboid[p1, p8]}, Axes → True]
```



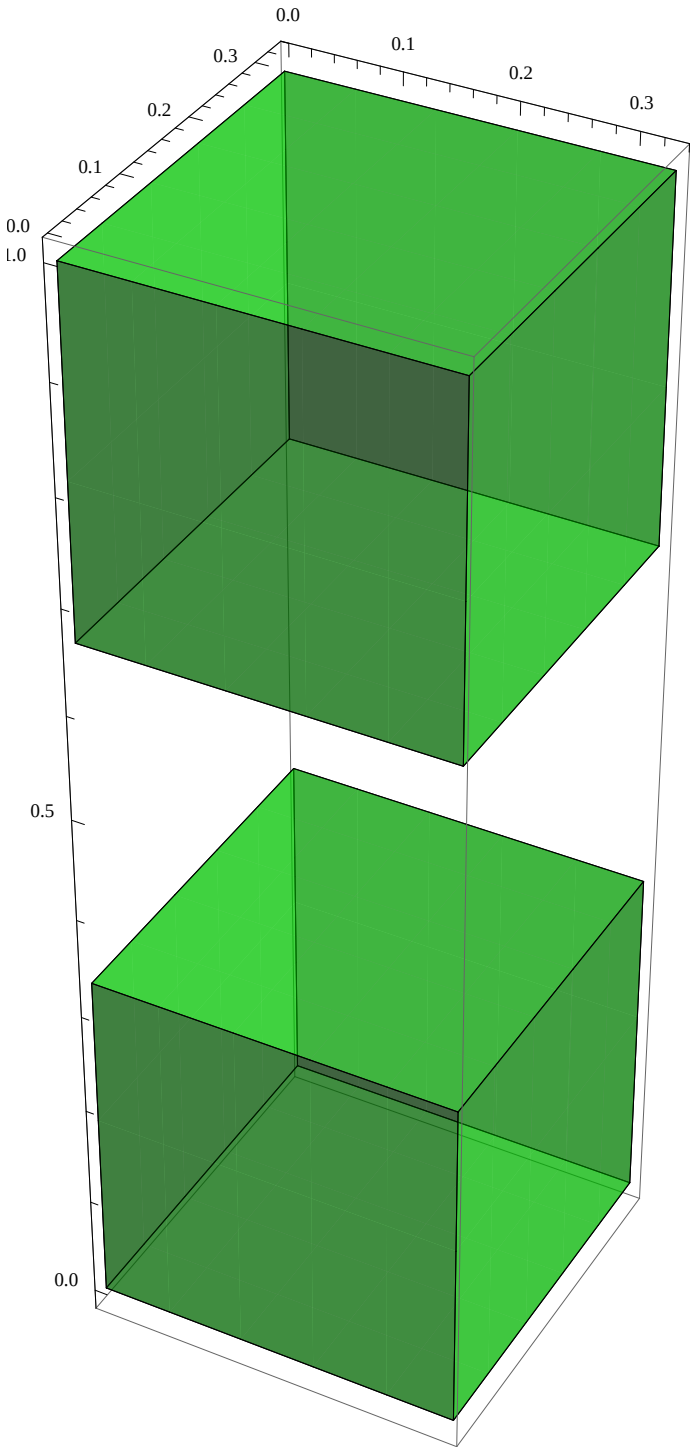
```
p10 = .; p10 = p5 + vz;

Graphics3D[{RGBColor[0, 1, 0, .5], Cuboid[a, p1], Cuboid[p1, p8], Cuboid[p3, p9]}]
```



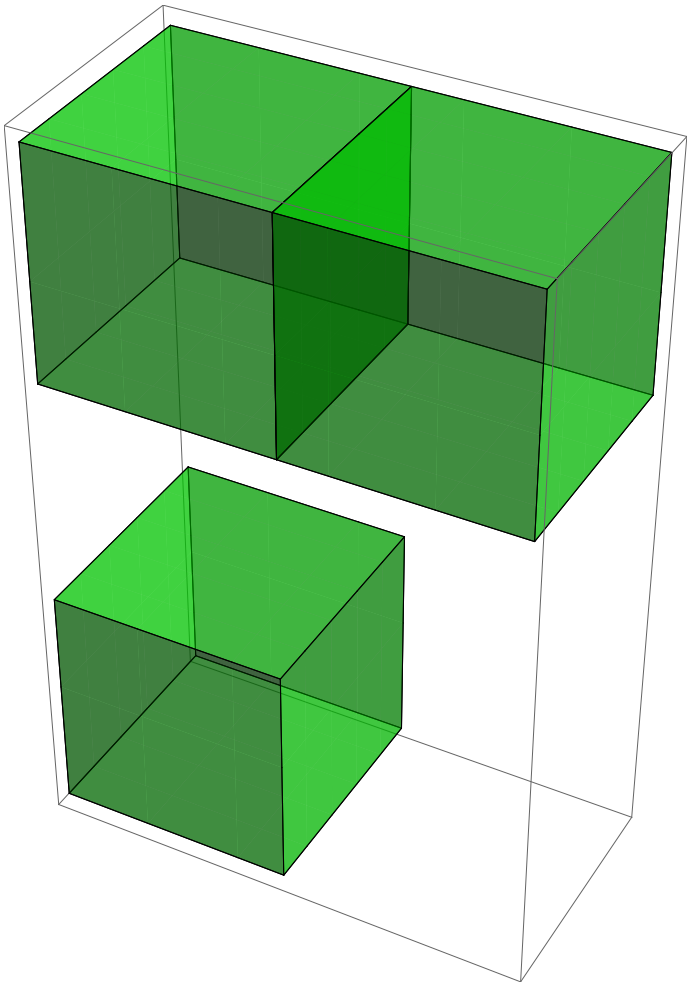
```
p11 = .; p11 = p7 + vz;

Graphics3D[{RGBColor[0, 1, 0, .5], Cuboid[a, p1], Cuboid[p1, p8], Cuboid[p3, p9],
            Cuboid[p5, p10], Cuboid[p7, p11]}, Axes → True]
```



```
p12 = .; p12 = p8 + v;

Graphics3D[{RGBColor[0, 1, 0, .5], Cuboid[a, p1], Cuboid[p8, p12]}, Axes → True]
```

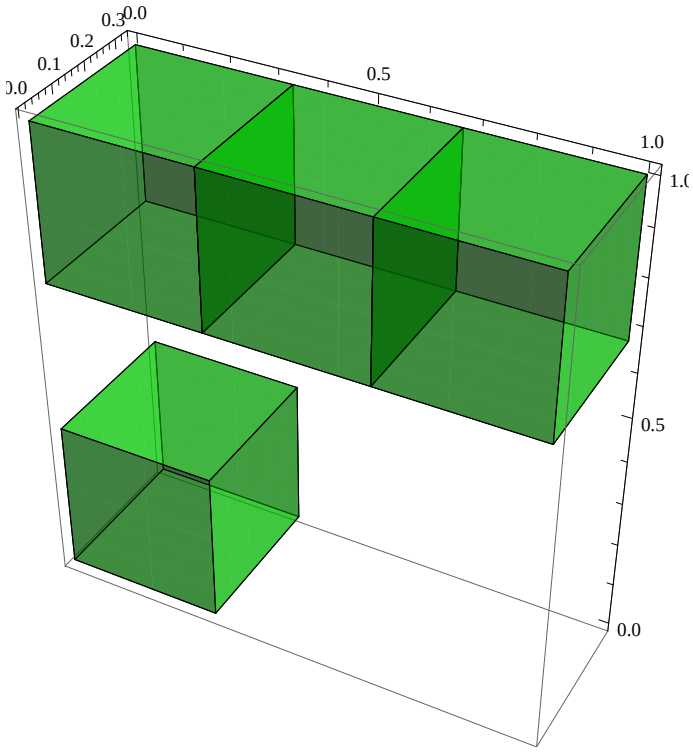


```
p13 = .; p13 = p9 + v;

Graphics3D[{RGBColor[0, 1, 0, .5], Cuboid[a, p1], Cuboid[p8, p12], Cuboid[p9, p12]}]
```

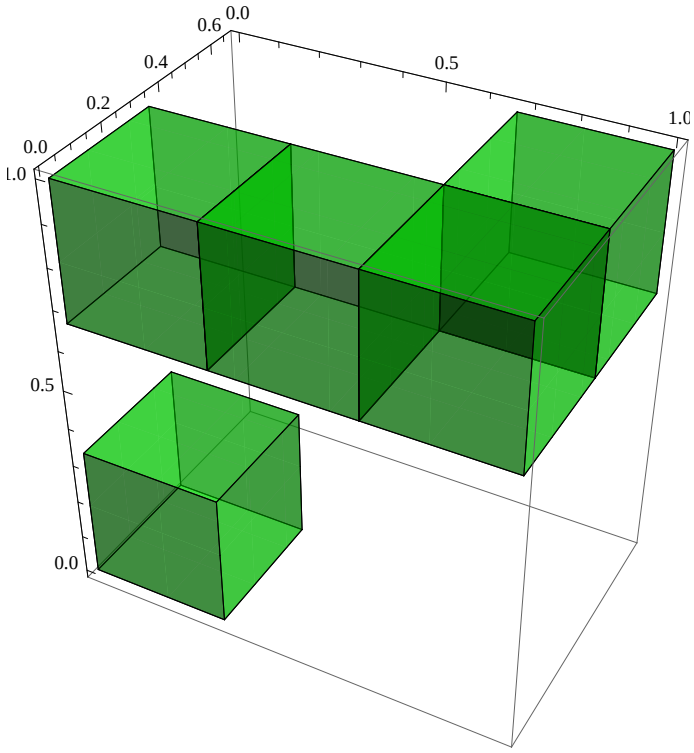


```
p14 = .; p14 = p11 + v;
```

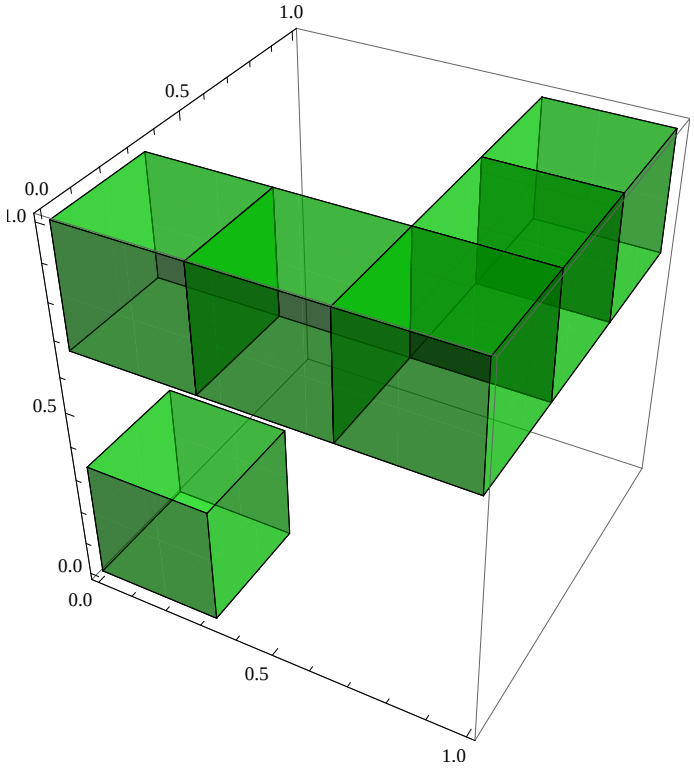
```
Graphics3D[{RGBColor[0, 1, 0, .5], Cuboid[a, p1], Cuboid[p8, p12], Cuboid[p9, p12],
            Cuboid[p9, p13]}, Axes → True]
```



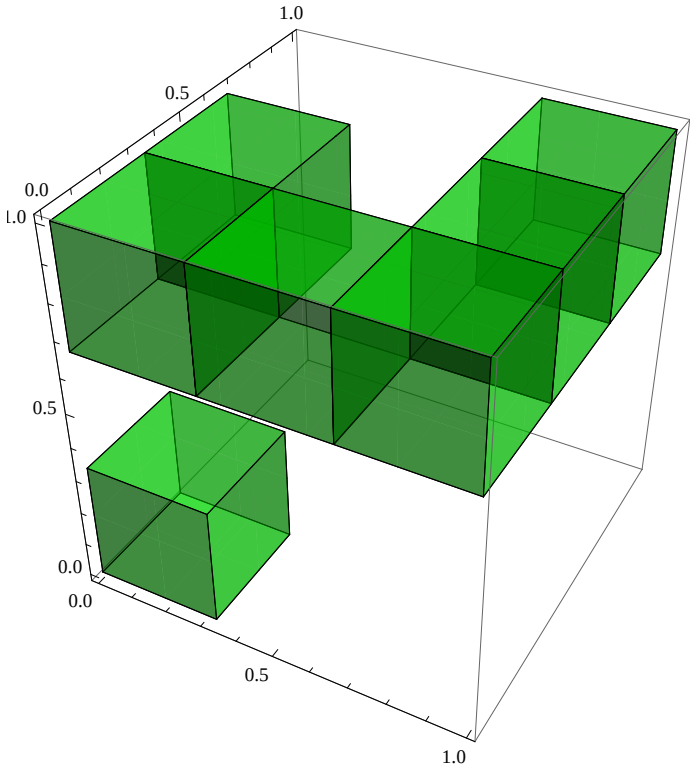
```
Graphics3D[{RGBColor[0, 1, 0, .5], Cuboid[a, p1], Cuboid[p8, p12], Cuboid[p9, p12],
    Cuboid[p9, p13], Cuboid[p10, p13]}, Axes → True]
```



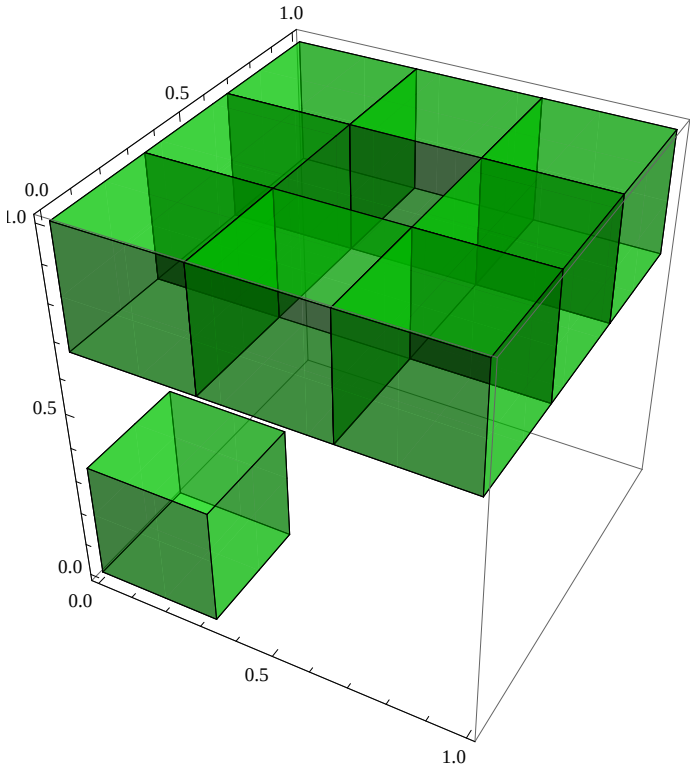

```
Graphics3D[{RGBColor[0, 1, 0, .5], Cuboid[a, p1], Cuboid[p8, p12], Cuboid[p9, p12],
    Cuboid[p9, p13], Cuboid[p10, p13], Cuboid[p10, b]}, Axes → True]
```



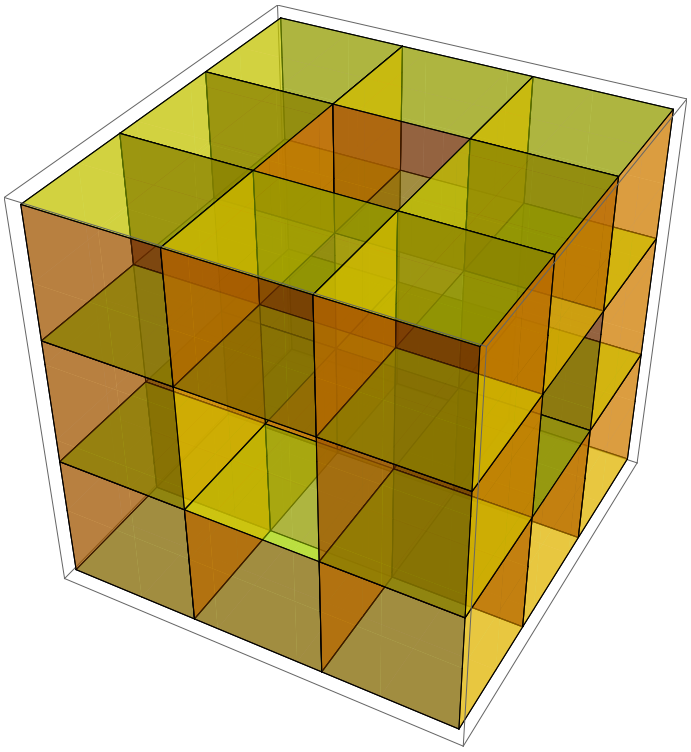
```
Graphics3D[{RGBColor[0, 1, 0, .5], Cuboid[a, p1], Cuboid[p8, p12], Cuboid[p9, p12],
            Cuboid[p9, p13], Cuboid[p10, p13], Cuboid[p10, b], Cuboid[p11, p12]}, Axes → True]
```



```
Graphics3D[{RGBColor[0, 1, 0, .5], Cuboid[a, p1], Cuboid[p8, p12], Cuboid[p9, p12],
            Cuboid[p9, p13], Cuboid[p10, p13], Cuboid[p10, b], Cuboid[p11, p12],
            Cuboid[p11, p14], Cuboid[p10, p14]}, Axes → True]
```

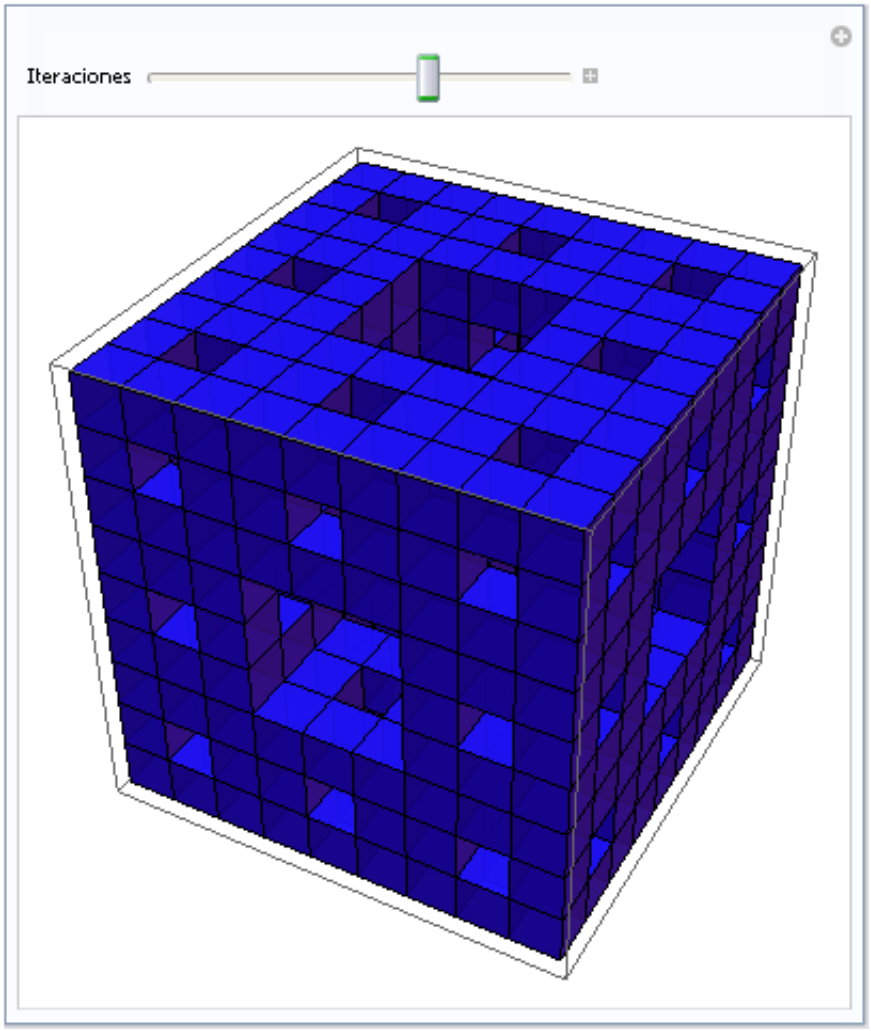


```
Graphics3D[{RGBColor[1, 1, 0, .5], Cuboid[a, p1], Cuboid[p1, p2], Cuboid[p2, p3],
            Cuboid[p3, p4], Cuboid[p4, p5], Cuboid[p1, p6], Cuboid[p6, p7], Cuboid[p4, p7],
            Cuboid[p1, p8], Cuboid[p3, p9], Cuboid[p5, p10], Cuboid[p7, p11], Cuboid[p8, p12],
            Cuboid[p9, p12], Cuboid[p9, p13],
  Cuboid[p10, p13], Cuboid[p10, b], Cuboid[p11, p12],
            Cuboid[p11, p14], Cuboid[p10, p14]}]
```



El código siguiente permite graficar la Esponja de Menger.

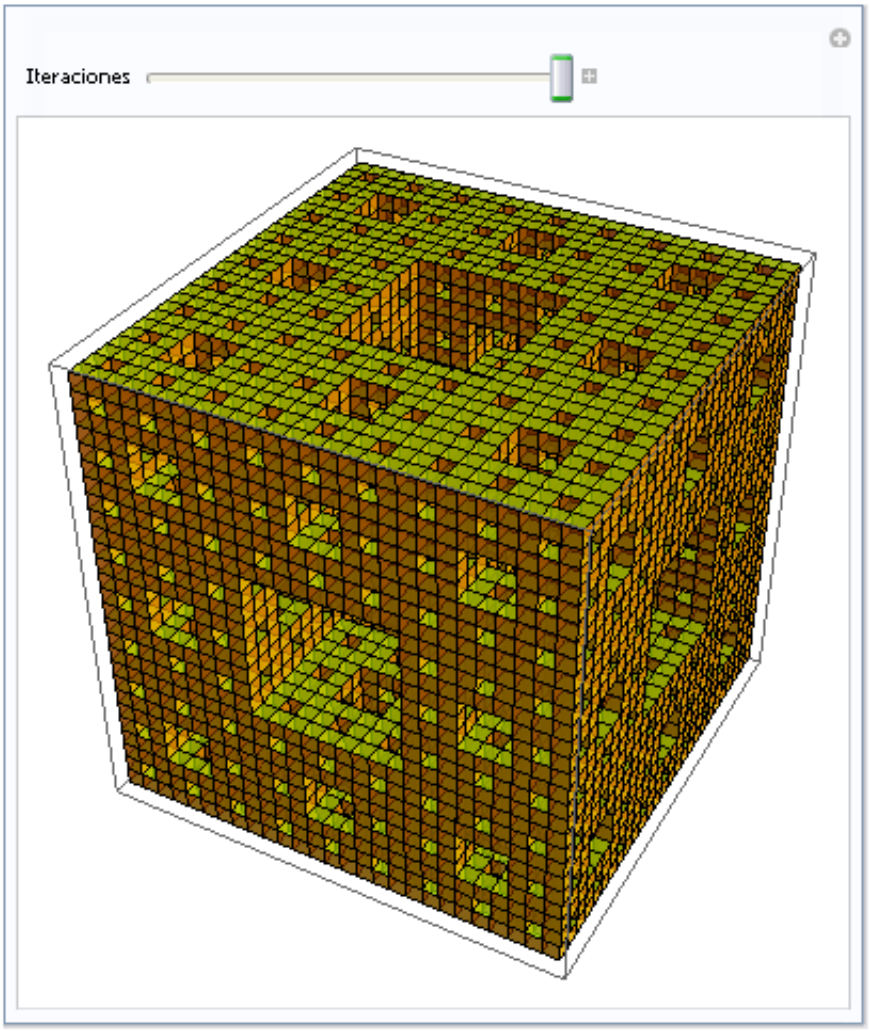
```
menger =.
menger[Cuboid[a_, b_]] :=
    {v = (b - a)/3.; vx = RotationTransform[180. °, UnitVector[3, 1]][v];
  vy = RotationTransform[180. °, UnitVector[3, 2]][v];
    vz = RotationTransform[180. °, UnitVector[3, 3]][v]; p1 = a + v;
  p2 = p1 + vx; p3 = p2 + v; p4 = p3 + vy; p5 = p4 + v; p6 = p1 + vy; p7 = p6 + v;
    p8 = p1 + vz; p9 = p3 + vz; p10 = p5 + vz; p11 = p7 + vz; p12 = p8 + v; p13 = p9 + v; p14 = p11 + v;
    Cuboid[a, p1], Cuboid[p1, p2], Cuboid[p2, p3], Cuboid[p3, p4],
  Cuboid[p4, p5], Cuboid[p1, p6], Cuboid[p6, p7], Cuboid[p4, p7],
    Cuboid[p1, p8], Cuboid[p3, p9], Cuboid[p5, p10], Cuboid[p7, p11],
  Cuboid[p8, p12], Cuboid[p9, p12], Cuboid[p9, p13],
    Cuboid[p10, p13], Cuboid[p10, b], Cuboid[p11, p12], Cuboid[p11, p14], Cuboid[p10, p14]}
menger[x_List] := menger /@ Flatten[x]

n =.
Manipulate[
    Graphics3D[
        {RGBColor[.2, .1, 1, .9], Nest[menger, Cuboid[{0, 0, 0}, {1, 1, 1}], n]}],
    {{n, 0, "Iteraciones"}, 0, 3, 1}, SaveDefinitions → True]
```



Segunda solución:

```
menger2 = .
menger2[Cuboid[a_, b_]] :=
    {p1 = (2 a + b)/3.; p2 = (2 b + a)/3.; p3 = {p2[[1]], a[[2]], a[[3]]};
  p4 = {b[[1]], p1[[2]], p1[[3]]}; p5 = {p2[[1]], p2[[2]], a[[3]]};
    p6 = {b[[1]], b[[2]], p1[[3]]}; p7 = {p1[[1]], b[[2]], p1[[3]]};
  p8 = {a[[1]], p2[[2]], a[[3]]}; p9 = {a[[1]], a[[2]], p2[[3]]};
    p10 = {p2[[1]], a[[2]], p2[[3]]}; p11 = {a[[1]], p2[[2]], p2[[3]]};
  p12 = {p1[[1]], b[[2]], b[[3]]}; p13 = {b[[1]], p1[[2]], b[[3]]};
    p14 = {p1[[1]], p1[[2]], b[[3]]};
    Cuboid[a, p1], Cuboid[p1, p3], Cuboid[p3, p4], Cuboid[p4, p5],
  Cuboid[p5, p6], Cuboid[p5, p7], Cuboid[p7, p8], Cuboid[p8, p1],
    Cuboid[p1, p9], Cuboid[p4, p10], Cuboid[p2, p6], Cuboid[p7, p11],
  Cuboid[p2, b], Cuboid[p2, p12], Cuboid[p2, p13], Cuboid[p11, p12],
    Cuboid[p10, p13], Cuboid[p10, p14], Cuboid[p11, p14], Cuboid[p14, p9]}
menger2[x_List] := menger2 /@ Flatten[x]
```

```
n =.
Manipulate[
    Graphics3D[
        {RGBColor[1, 1, 0, .7], Nest[menger2, Cuboid[{0, 0, 0}, {1, 1, 1}], n]}],
    {{n, 0, "Iteraciones"}, 0, 3, 1}, SaveDefinitions → True]
```



3.5. Soluciones a actividades de la sección 3.2

Solución actividad 19:

Para la curva de Koch, se tiene que la razón de contracción es $\frac{1}{3}$ y el número de pedazos se cuadriplica, así:

$$\text{dimensión de la curva de Koch} = \frac{\log(4^n)}{-\log\left(\frac{1}{3^n}\right)} = \frac{\log(4)}{\log(3)}$$

Que aproximadamentees

```
Log[4]
------ // N
Log[3]
```

```
1.26186
```

por lo que la denominada curva de Koch, no es realmente una curva, aunque tampoco es una superficie plana, ya que su dimensión está entre uno y dos. Aunque se asemeja más a una curva por tener dimensión más cercana de uno que de dos.

Para la curva generada por la función *costado*, se tiene que la razón de contracción es $\frac{1}{4}$ y el número de pedazos se octuplica, así:

$$\text{dimensión de la curva costado} = \frac{\log(8^n)}{-\log\left(\frac{1}{4^n}\right)} = \frac{3}{2} = 1.5$$

por lo que la denominada curva costado, no es realmente una curva, ya que su dimensión es 1.5, pero tampoco es una superficie.

En cuanto al dragón, la razón de contracción es $\frac{1}{\sqrt{2}}$, mientras que las piezas se duplican. Por lo tanto,

$$\text{dimensión del Dragón} = \frac{\log(2)}{-\log\left(\frac{1}{\sqrt{2}}\right)} = 2$$

por lo que intuitivamente el Dragón "la curva límite" ocupa una superficie.

Advertencia: El Dragón no es un fractal pues su dimensión es entera, aunque su borde si lo es.

Al modificar la figura base que engendra la "curva dragón" se tiene, que la razón de contracción es $\frac{1}{\sqrt{3}}$, mientras que el número de pedazos se duplica, por lo que

$$\text{dimensión de la curva Dragón - modificada} = \frac{\log(2)}{-\log\left(\frac{1}{\sqrt{3}}\right)} = \frac{2\log(2)}{\log(3)}$$

Que aproximadamentees

```
2 Log[2.]
---------
Log[3]
```

```
1.26186
```

por lo que la "curva límite" si es un fractal, contrario a lo ocurrido con la "curva límite" generada con la función dragón.

En el ternario de Cantor se tiene, que la razón de contracción es $\frac{1}{3}$, mientras que el número de pedazos se duplica, por lo que

$$\text{dimensión del ternario de Cantor} = \frac{\log(2^n)}{-\log\left(\frac{1}{3^n}\right)} = \frac{\log(2)}{\log(3)}$$

Que aproximadamentees

$$\frac{\text{Log[2.]}}{\text{Log[3]}}$$

0.63093

por lo que el ternario de Cantor no tiene dimensión uno, así que no es un segmento de recta, empero, no es un punto pues no tiene dimensión cero, aunque su medida sea nula.

Al generar el polvo de Cantor se tiene, que la razón de contracción es $\frac{1}{4}$, mientras que el número de cuadrados se cuadruplica, por lo que su dimensión es:

$$\text{dimensión del polvo de Cantor} = \frac{\log\left(4^n\right)}{-\log\left(\frac{1}{4^n}\right)} = \frac{\log(4)}{\log(4)} = 1$$

Así, intuitivamente el polvo de Cantor es una curva y no es un fractal pues su dimensión es entera.

En el cálculo de la dimensión del triángulo de Sierpinski, se tiene que la razón de homotecia es $\frac{1}{2}$, mientras que el número de triángulos se triplica, así en la n-ésima iteración, el factor de contracción será de $\left(\frac{1}{2}\right)^n$, mientras que el número de pedazos se incrementará en 3^n. Así utilizando la fórmula se tiene:

$$\text{dimensión Támiz Sierpinski} = \frac{\log\left(3^n\right)}{-\log\left(\frac{1}{2^n}\right)} = \frac{\log(3)}{\log(2)}$$

Que aproximadamentees

$$N\left[\frac{\text{Log[3]}}{\text{Log[2]}}\right]$$

1.58496

Se puede decir pues que (intuitivamente) el Támiz de Sierpinski ocupa más espacio que una curva, pero no llega a ser una superficie.

En el cálculo de la dimensión de Hausdorff del Tetraedro de Sierpinski, se tiene que la razón de homotecia es $\frac{1}{2}$, mientras que el número de tetraedros en cada iteración se cuadruplica, así en la n-ésima iteración, el factor de contracción será de $\left(\frac{1}{2}\right)^n$, mientras que el número de pedazos se incrementará en 4^n. Así utilizando la fórmula se tiene:

$$\text{dimensión Tetraedro de Sierpinski} = \frac{\log\left(4^n\right)}{-\log\left(\frac{1}{2^n}\right)} = \frac{2\log(2)}{\log(2)} = 2$$

por lo que intuitivamente el Tetraedro de Sierpinski "el sólido límite" es una superficie.

Advertencia: El Tetraedro de Sierpinski no es un fractal pues su dimensión es entera.

En el cálculo de la dimensión de la Alfombra de Sierpinski, se tiene que la razón de homotecia es $\frac{1}{3}$, mientras que el número de cuadrados en cada iteración se multiplica por ocho, así en la n-ésima iteración, el factor de contracción será de $\left(\frac{1}{3}\right)^n$, mientras que el número de pedazos se incrementará en 8^n. Así utilizando la fórmula se tiene:

$$\text{dimensión Alfombra de Sierpinski} = \frac{\log\left(8^n\right)}{-\log\left(\frac{1}{3^n}\right)} = \frac{3\log(2)}{\log(3)}$$

Que aproximadamentees

$$N\left[\frac{\text{3 Log[2]}}{\text{Log[3]}}\right]$$

1.89279

Se puede decir pues que (intuitivamente) la Alfombra de Sierpinski no es una línea y es casi una superficie.

En el cálculo de la dimensión de la Esponja de Menger, se tiene que la razón de homotecia es $\frac{1}{3}$, mientras que el número de cubos en cada iteración se multiplica por veinte, así en la n-ésima iteración, el factor de contracción será de $\left(\frac{1}{3}\right)^n$, mientras que el número de pedazos se incrementará en 20^n. Así utilizando la fórmula se tiene:

$$\text{dimensión Esponja de Menger} = \frac{\log\left(20^n\right)}{-\log\left(\frac{1}{3^n}\right)} = \frac{\log(20)}{\log(3)}$$

Que aproximadamentees

```
N[Log[20]/Log[3]]
```

```
2.72683
```

Se puede decir pues que (intuitivamente)la Esponja de Menger no es una superficie y es casi un sólido.

3.6. Soluciones a actividades de la sección 3.3

Solución actividad 1:

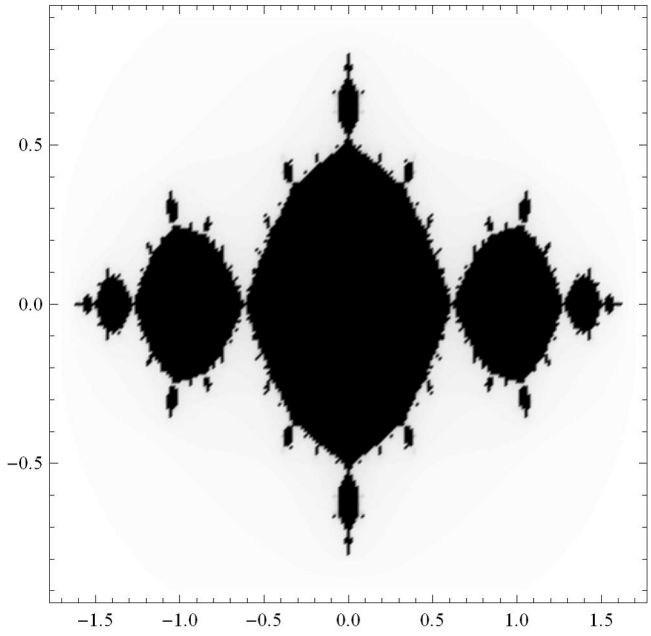
Como resultado de la actividad anterior se tiene que:

$c = -1$.

```
rango = {{-1.7, 1.7}, {-.9, .9}}; puntos = 200;
Timing[tabla = julia[-1., 100, rango, {puntos, puntos}];]
```

```
{0.969, Null}
```

```
ListDensityPlot[tabla, ColorFunction → (GrayLevel[1 - #] &),
     Frame → True, Mesh → False, DataRange → rango]
```

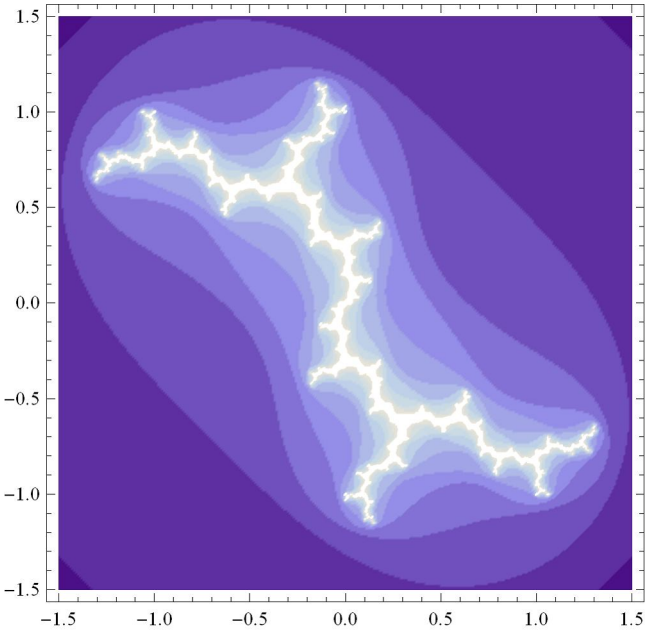


Haciendo un "zoom", es decir, dibujando sobre la región $\{(x, y) \mid x \in [-0.53, -0.41],\ y \in [-0.19, 0.31]\}$ y manteniendo los otros valores constantes, se obtiene figuras similares a la original lo que muestra la naturaleza fractal de este conjunto.

```
rango = {{-1.5, 1.5}, {-1.5, 1.5}}; puntos = 300;
Timing[tabla = julia[i // N, 100, rango, {puntos, puntos}];]
```

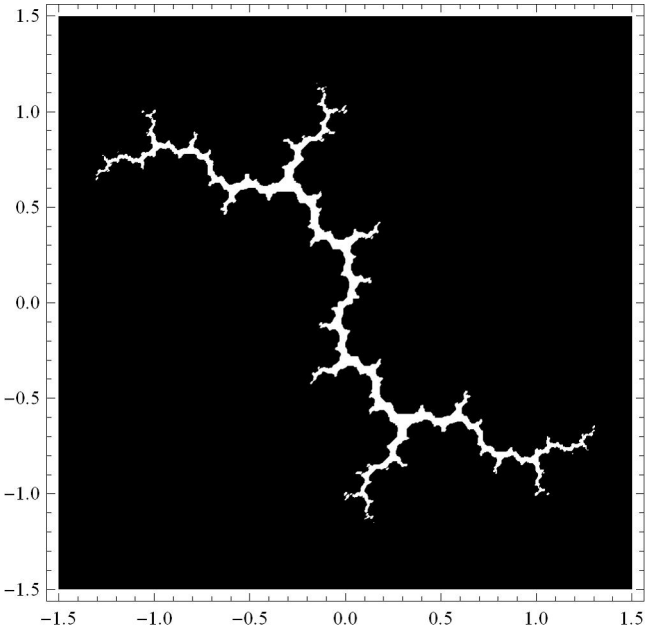
```
{0.672, Null}
```

```
ListDensityPlot[tabla, Frame → True, Mesh → False, DataRange → rango]
```



Si se desea resaltar el contorno se puede agregar el atributo **ColorFunction**.

```
ListDensityPlot[tabla, ColorFunction → (GrayLevel[-Round[#]] &),
    Frame → True, Mesh → False, DataRange → rango]
```



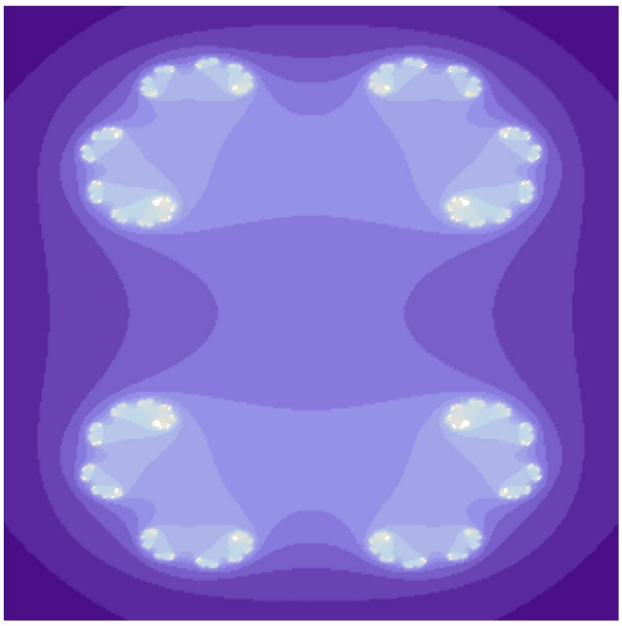
$c = 1/2$.

A este tipo de tipo de fractal se le denomina polvo fractal.

```
rango = {{-1., 1.}, {-1.4, 1.4}}; puntos = 300;
Timing[tabla = julia[.5, 150, rango, {puntos, puntos}];]
```

```
{0.735, Null}
```

```
ListDensityPlot[tabla, Frame → False, Mesh → False, DataRange → rango]
```

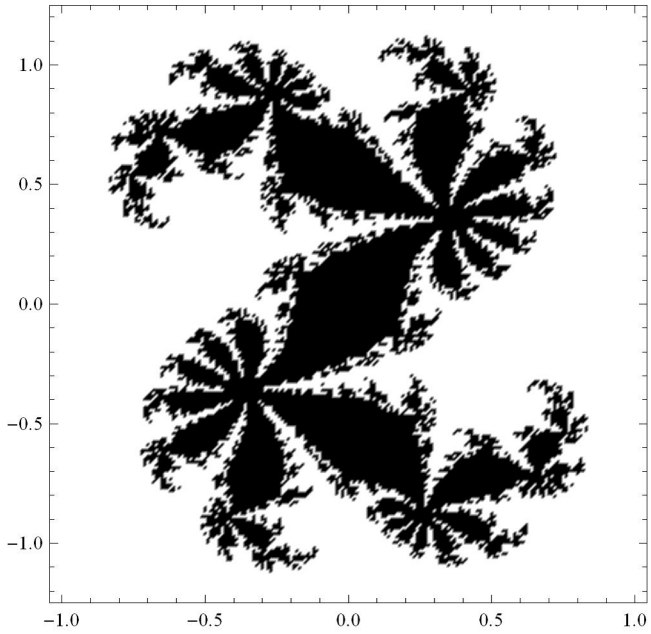


$c = 0.360284 + 0.100376\,i$.

```
c = 0.360284 + 0.100376 i; rango = {{-1., 1.}, {-1.2, 1.2}}; puntos = 300;
Timing[tabla = julia[c, 200, rango, {puntos, puntos}];]
```

```
{2.422, Null}
```

```
ListDensityPlot[tabla, ColorFunction → (GrayLevel[1 - Round[#]] &),
     Frame → True, Mesh → False, DataRange → rango]
```



Haciendo un "zoom", es decir, dibujando sobre la región $\{(x, y) \mid x \in [0.55, 0.74],\ y \in [0.53, 0.82]\}$ y manteniendo los otros valores constantes, se obtiene figuras que presentan la misma complejidad de la anterior, lo que muestra la naturaleza fractal de este conjunto.

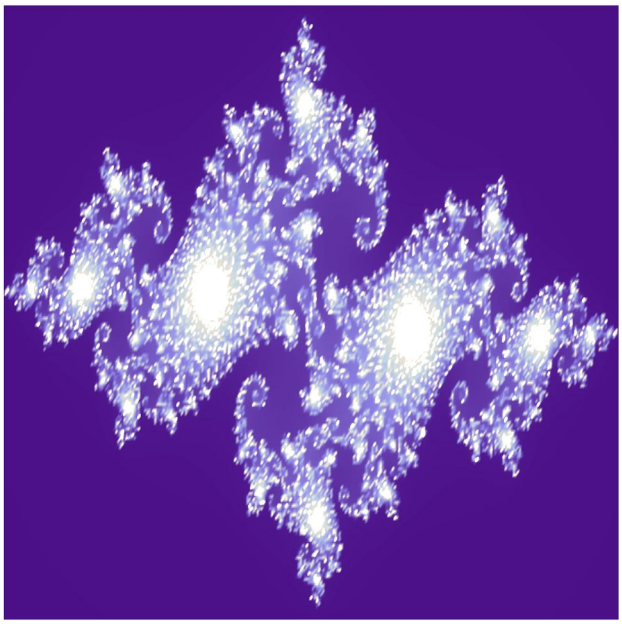
Solución actividad 2:

$c = -0.74543 + 0.11301\,i$.

```
c = -0.74543 + 0.11301 i; rango = {{-1.5, 1.5}, {-0.9, 0.9}}; divisiones = {300, 200};
Timing[tabla = julia[c, 3000, rango, divisiones];]
```

```
{2.937, Null}
```

```
ListDensityPlot[tabla, Frame → False, Mesh → False, DataRange → rango]
```



Obsérvese que en ningún caso se necesitó realizar 3000 iteraciones, como se comprueba a continuación:

```
t = Flatten[tabla];
{Min[t], Max[t]}
```

```
{1, 2034}
```

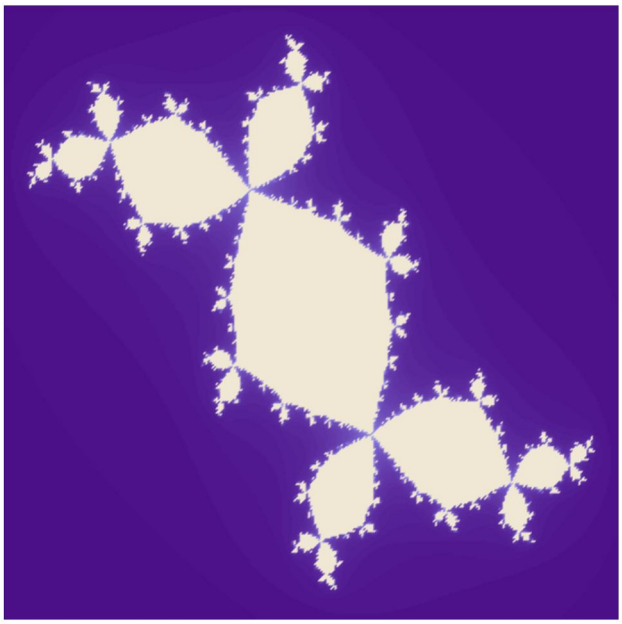
Esto indica que los puntos fijos y/o ciclos son repulsivos.

$c = -0.122 + 0.745\,i$.

```
c = -0.122 + 0.745 i; rango = {{-1.4, 1.4}, {-1.2, 1.2}}; divisiones = {300, 300};
Timing[tabla = julia[c, 100, rango, divisiones];]
```

```
{1.984, Null}
```

```
ListDensityPlot[tabla, Frame → False, Mesh → False, DataRange → rango]
```

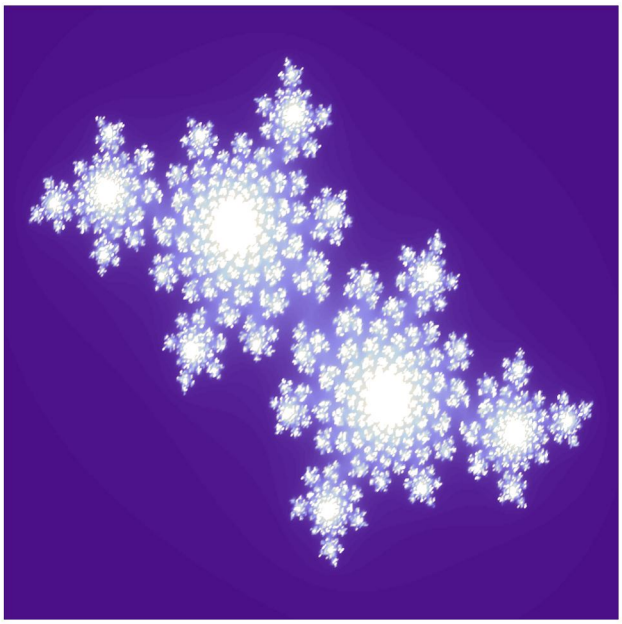


$c = -0.4 - 0.6\,i$.

```
c = -0.4 + 0.6 i; rango = {{-1.5, 1.5}, {-1.2, 1.2}}; divisiones = {350, 300};
Timing[tabla = julia[c, 300, rango, divisiones];]
```

```
{2.11, Null}
```

```
ListDensityPlot[tabla, Frame → False, Mesh → False, DataRange → rango]
```

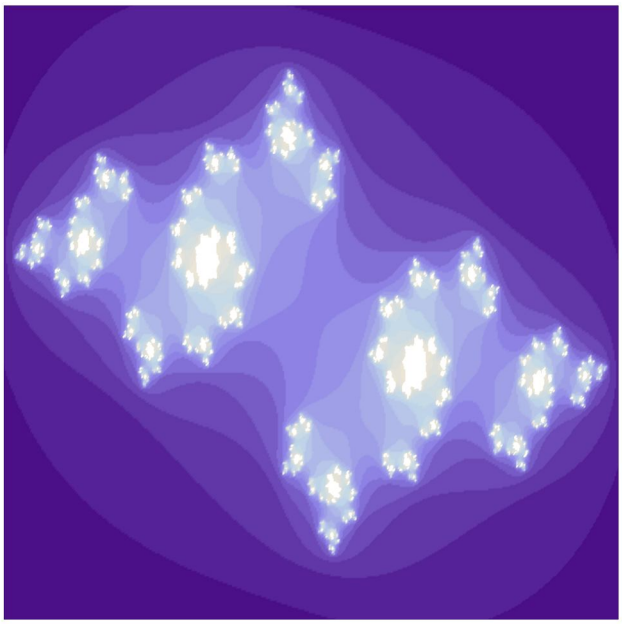


$c = -0.8 + 0.4\,i$.

```
c = -0.8 + 0.4 i; rango = {{-1.6, 1.6}, {-1.1, 1.1}}; divisiones = {400, 350};
Timing[tabla = julia[c, 300, rango, divisiones];]
```

```
{1.265, Null}
```

```
ListDensityPlot[tabla, Frame → False, Mesh → False, DataRange → rango]
```



$c = 1.5$.

```
c = -1.5; rango = {{-1.9, 1.9}, {-.7, .7}}; divisiones = {500, 250};
Timing[tabla = julia[c, 100, rango, divisiones];]
```

```
{1.016, Null}
```

```
ListDensityPlot[tabla, Frame → False, Mesh → False, DataRange → rango]
```

Solución actividad 3:

```
(* 100 iteraciones máximas por default *)
Clear[escape, mandelbrot]
escape = Compile[{{c, _Complex}, {itermax, _Integer}},
           Module[{c1 = c, iter = 1},
                While[(Abs[c1] < 2) && (iter++ ≤ itermax), c1 = c1^2 + c]; iter]];
mandelbrot[{{xmin_, xmax_}, {ymin_, ymax_}}, npuntos_, itermax_: 100] :=
     Table[escape[x + y I, itermax],
           {y, ymin, ymax, N[(ymax - ymin) / npuntos]},
           {x, xmin, xmax, N[(xmax - xmin) / npuntos]}]
```

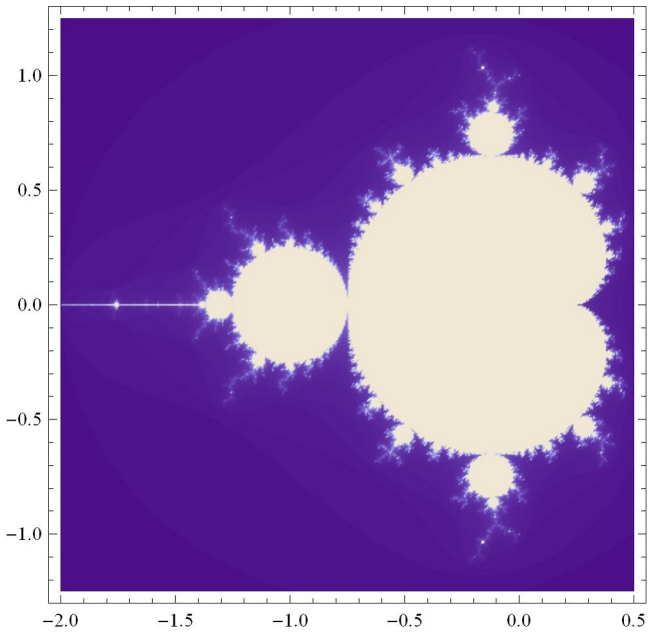
Solución actividad 4:

Como resultado de la actividad anterior se tiene:

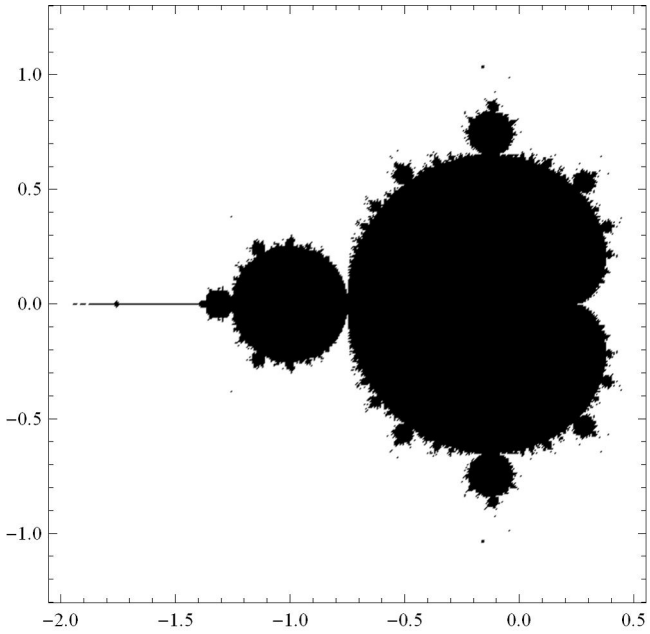
```
man =.; rango = {{-2., .5}, {-1.25, 1.25}};
Timing[man = mandelbrot[rango, 400, 125];]
```

```
{4.219, Null}
```

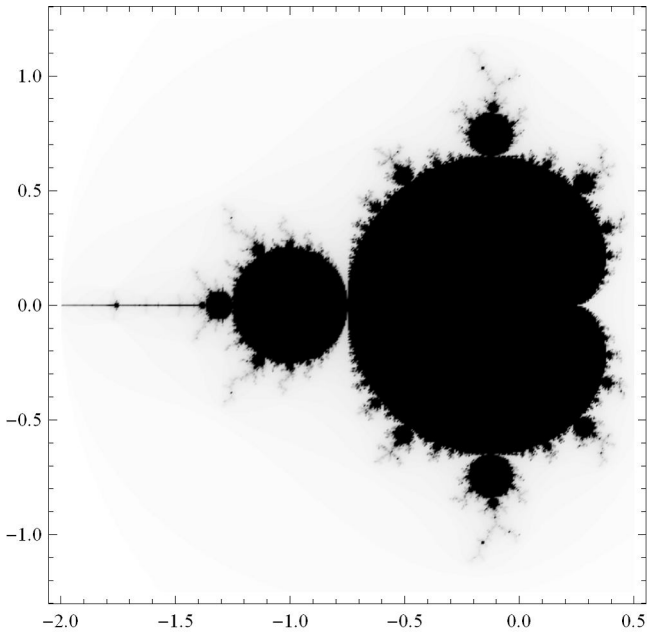
```
ListDensityPlot[man, Frame → True, Mesh → False, DataRange → rango]
```



```
ListDensityPlot[man, Frame → True, ColorFunction → (GrayLevel[1 - Round[#]] &),
                Mesh → False, DataRange → rango]
```




```
ListDensityPlot[man, Frame → True, ColorFunction → (GrayLevel[1 - #] &),
                Mesh → False, DataRange → rango]
```

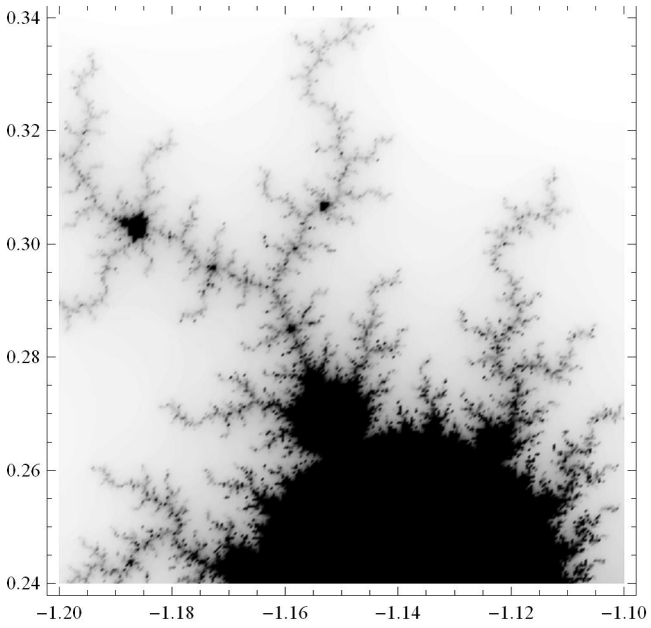


La solución del literal b) de la tercera actividad es:

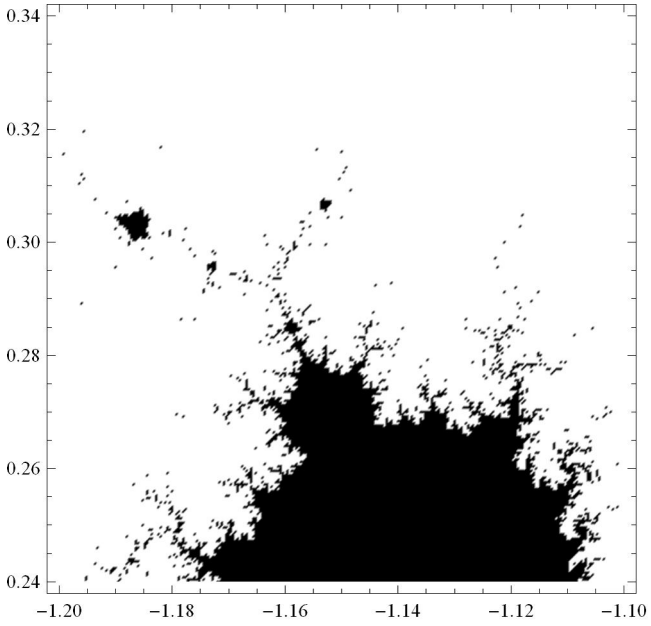
```
rango = {{-1.2, -1.1}, {.24, .34}};
Timing[man = mandelbrot[rango, 250, 125];]
```

```
{1.891, Null}
```

```
ListDensityPlot[man, Frame → True, ColorFunction → (GrayLevel[1 - #] &),
                Mesh → False, DataRange → rango]
```




```
ListDensityPlot[man, Frame → True, ColorFunction → (GrayLevel[1 - Round[#]] &),
                Mesh → False, DataRange → rango]
```

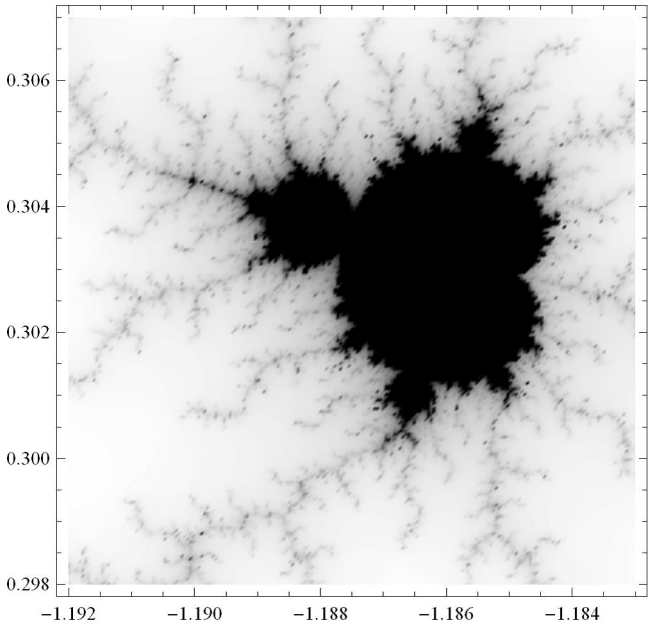


La solución del literal c) de la tercera actividad es:

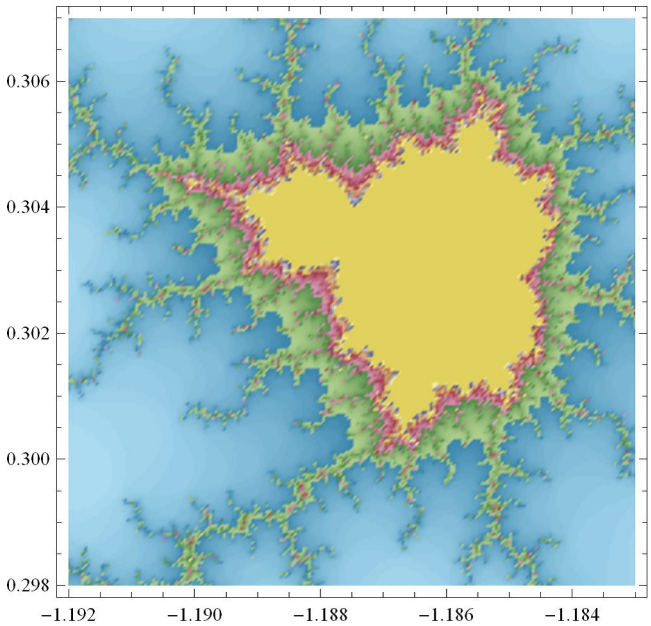
```
rango = {{-1.192, -1.183}, {.298, .307}};
Timing[man = mandelbrot[rango, 200, 175];]
```

```
{1.859, Null}
```

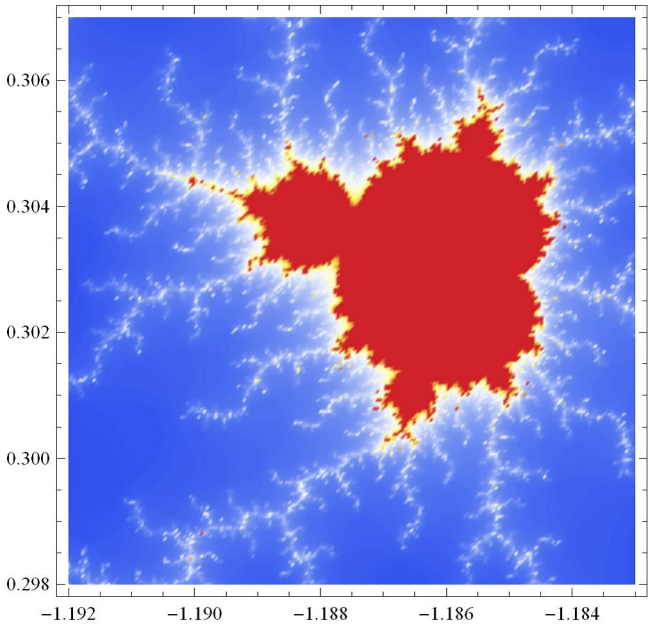
```
ListDensityPlot[man, Frame → True, ColorFunction → (GrayLevel[1 - #] &),
                Mesh → False, DataRange → rango]
```




```
ListDensityPlot[man, Frame → True, ColorFunction → "DarkBands",
                Mesh → False, DataRange → rango]
```



```
ListDensityPlot[man, Frame → True, ColorFunction → "TemperatureMap",
                Mesh → False, DataRange → rango]
```



Conclusiones

El objetivo de esta investigación es de doble naturaleza: por una parte pretende proporcionar al lector un acceso elemental a la teoría de sistemas dinámicos en general, y a los sistemas con dinámica caótica en particular. Por otra parte, lo hace con la intención explícita de aportar material al debate sobre la reforma del currículo de matemática en la universidad, ya que busca sistemáticamente la exposición a través de la experimentación y mediante el software Matemática y del análisis de ejemplos muy sencillos, en muchos casos surgidos de la investigación aplicada, próximos a la intuición de los alumnos y aptos para el trabajo en el aula. Es por esta razón que asistimos hoy en día a una revisión crítica de esta influencia, que se traduce en un intento de renovación del currículo de matemáticas. Inevitablemente si tal renovación se quiere llevar a cabo, debe apoyarse en los mismos elementos que han servido para superar el estancamiento estructuralista; debe apoyarse en las aportaciones que desde el campo de la matemática aplicada, han irrumpido con fuerza en el panorama matemático de los últimos cincuenta años. Una de las áreas protagonistas de esta revolución es la que se centra en el estudio del movimiento: la teoría de los sistemas dinámicos. Los progresos realizados en esta área, cuyos orígenes se remontan a la teoría de las ecuaciones diferenciales, iniciada por Newton y Leibnitz, permiten hoy en día comenzar a entender la conducta de sistemas dinámicos con conducta caótica, es decir, que prosiguen perpetuamente en un movimiento sin pauta aparente.

Otra característica más a destacar, es el hecho de construir una matemática actual próxima a problemas relevantes de la investigación reciente. Mediante el uso del ordenador, máquina de iterar por excelencia, es posible simular y experimentar los problemas planteados en términos de sistemas dinámicos, ganando así en viveza y plasticidad, y haciendo posible el acercamiento a la complejidad generada por problemas simples, a plantear otros nuevos y a hacer con ello una matemática interactiva muy lejana a la visión de ciencia estancada que frecuentemente se tiene de la matemática. Empero, los tópicos abordados en esta investigación al alumnado a investigar un poco más en temas que por cuestiones diversas no están presentes en el currículo de la carrera y que resultan muy interesantes.

Estas razones hacen de los sistemas dinámicos y de la teoría del caos una mina a explotar también en el campo educativo. ¿Seremos capaces de aprovechar estos factores para llevar a las aulas los nuevos tesoros aportados por las matemáticas de la actualidad?

En mi opinión una de las grandes deficiencias actuales en todo el sistema educativo es la poca familiaridad que tenemos con la computadora y su uso como herramienta para hacer cálculos, gráficos y resolver ecuaciones en el estudio de matemáticas. Es por ello que mi trabajo tendrá como método principal el experimental.

La geometría fractal y la teoría del caos han obligado a muchos científicos a observar con otros ojos la complejidad del mundo. Los fractales es toda una teoría aun en pañales que no se debe admirar quizá solo por la belleza superficial, sus forma, su dimensión, el color que se le pueda dar, etc. sino también es de destacar la belleza matemática que estos encierran; es fácil observar que ideas tan simples como las de iterar funciones que todos lo podemos por medio de un software tan eficiente como lo es el Mathematica nos construye desde una ramificación sencilla de un árbol hasta curvas extremadamente complicadas de predecir su comportamiento.

Los fractales demuestran que aún sabemos muy poco, quizá creamos vivir en una época de esplendor científico y es así, pero todavía nos falta saber muchísimas cosas. El caos está presente en la naturaleza, es más, está intrínsicamente relacionado con nuestra realidad. No es, pues, asombroso que los fractales sean un ejemplo de cómo se puede “controlar” lo incontrolable.

Otro aspecto que cabe destacar son las numerosas imágenes de belleza muy asombrosa. El conjunto de Mandelbrot es enorme y bello, también el conjunto de Julia es muy curioso y con infinitas posibilidades.

Estudios Futuros

Al nivel que está este trabajo debo de reconocer que no se me han presentado demasiadas dificultades a pesar de que he profundizado en aspectos como el Sistema de Funciones Iteradas o la demostración de muchos teoremas con los que me he enfrentado durante la investigación. Todo esto se debe a los conocimientos matemáticos necesarios para abordar todas estas problemáticas, lo que motiva al continuar investigando mucho más, entre algunos tópico se pueden mencionar los siguientes:

1. Estudiar los conjuntos de Mandelbrot para la función $f(z) = z^n + c,\ z \in \mathbb{C},\ c \in \mathbb{C}$ y $n \in \mathbb{Z},\ n > 2.$

2. Construir los conjuntos de Julia para las funciones, $E_\lambda(z) = \lambda e^z,\ S_\lambda(z) = \lambda sen(z).$

3. Investigar sobre la teoría de bifurcación, considerando familias uniparamétricas de funciones de variable real que son suaves respecto del parámetro.

4. Reflexionar sobre los conceptos, dimensión topológica y dimensión fractal.

Bibliografía

[1] AGUILERA, Néstor E. (1995). Un paseo por el Jardín de los Fractales. Red Olímpica.

[2] DEVANEY, Robert L. (1989). An Introduction to Chaotic Dynamical Systems Second Edition. Addison-Wesley Publishing Company, Inc.

[3] LÓPEZ JIMÉNEZ, Víctor (2000). ECUACIONES DIFERENCIALES: Cómo aprenderlas, cómo enseñarlas. Universidad de Murcia.

[4] MANDELBROT, Benoit B. (2003). Ed. Tusquets Editores, S.A. Versión en Español.

[5] PEREZ ORTÍZ, Juan A. (2000). Codificación Fractal de Imágenes. Creative Commons. Stanford CA, USA.