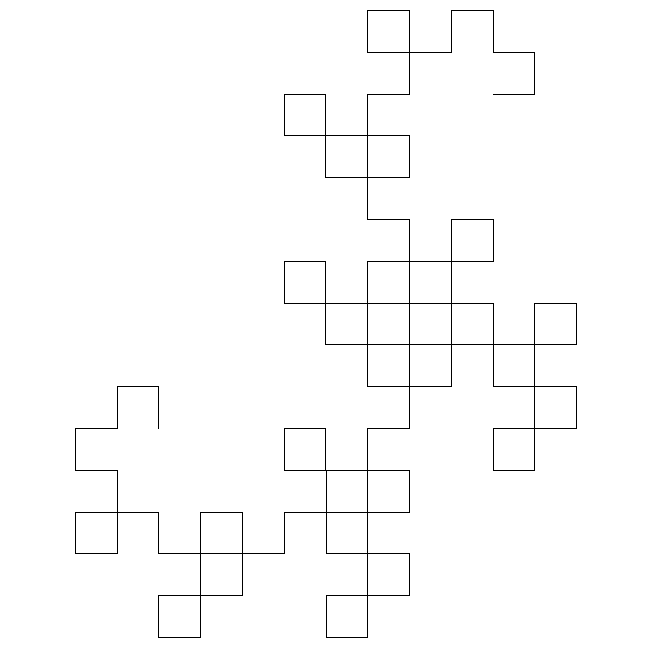
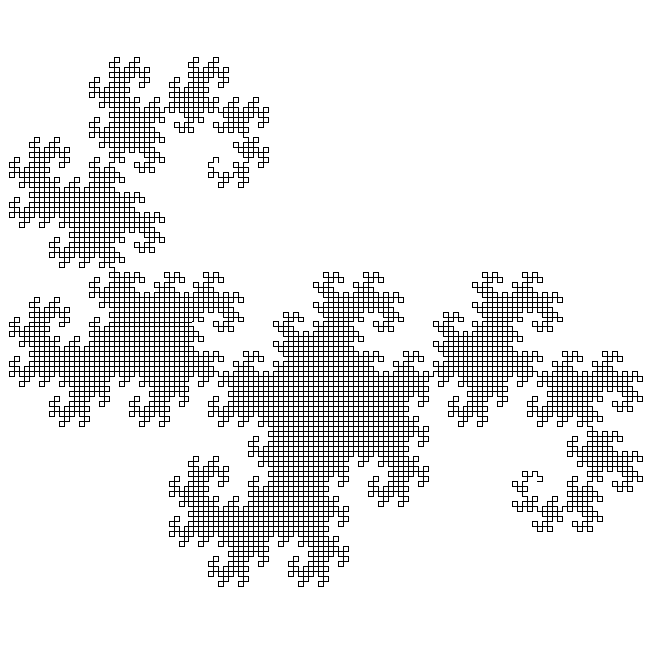
**Dragon Curve**

Written by [Paul Bourke](http://local.wasp.uwa.edu.au/%7Epbourke/fractals/)

Attributed to David and Knuth, 1970

axiom = FX  
X -> X+YF+  
Y -> -FX-Y  
angle = 90

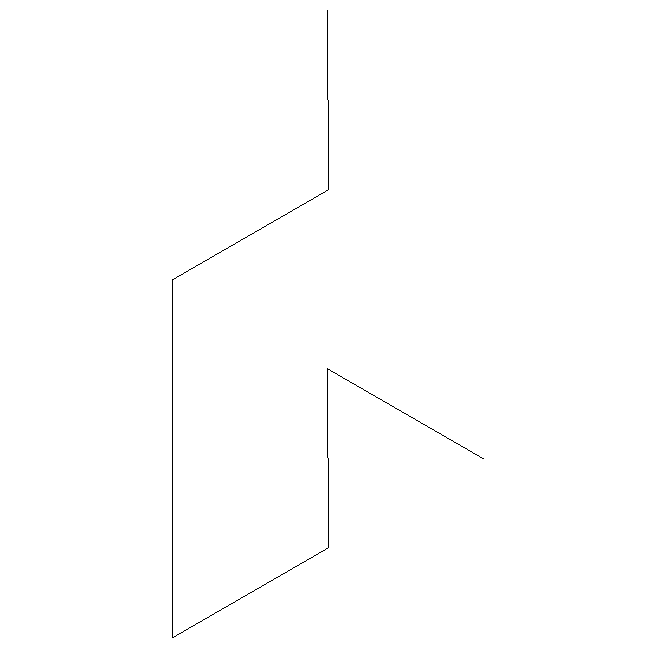
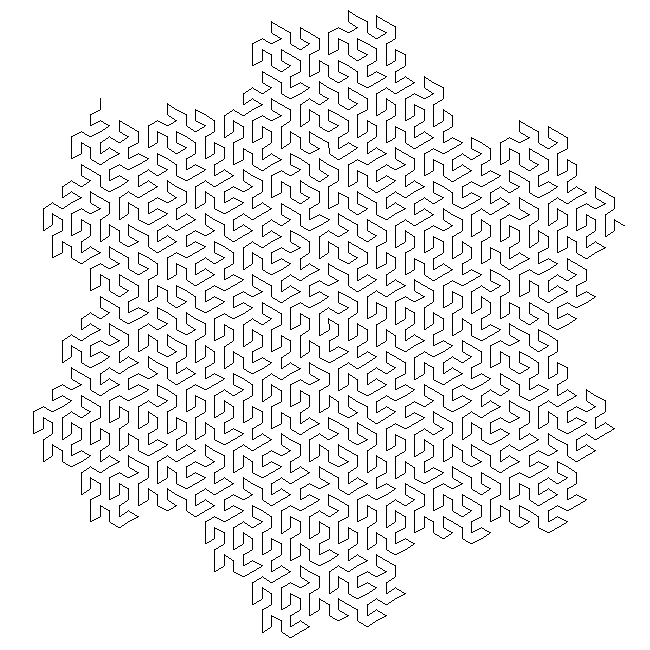
  


**Hexagonal Gosper**

Written by [Paul Bourke](http://local.wasp.uwa.edu.au/%7Epbourke/fractals/)

Attributed to Mandelbrot, 1982

axiom = XF  
X -> X+YF++YF-FX--FXFX-YF+  
Y -> -FX+YFYF++YF+FX--FX-Y  
angle = 60

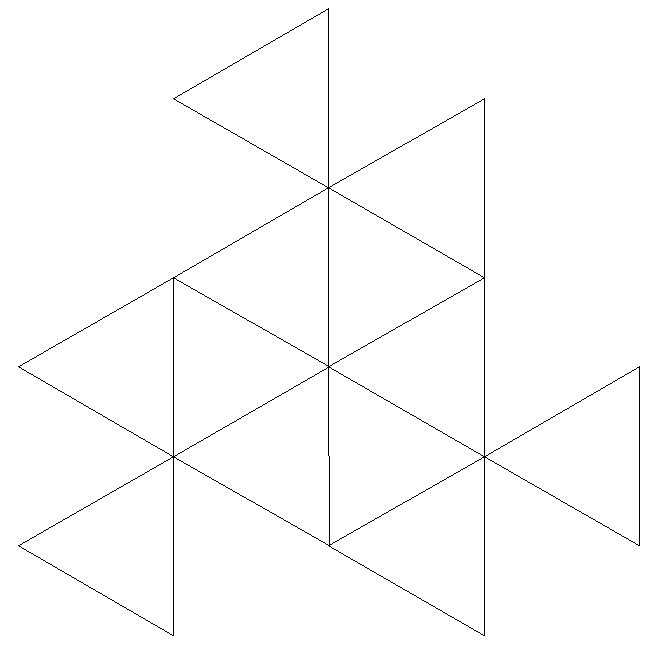
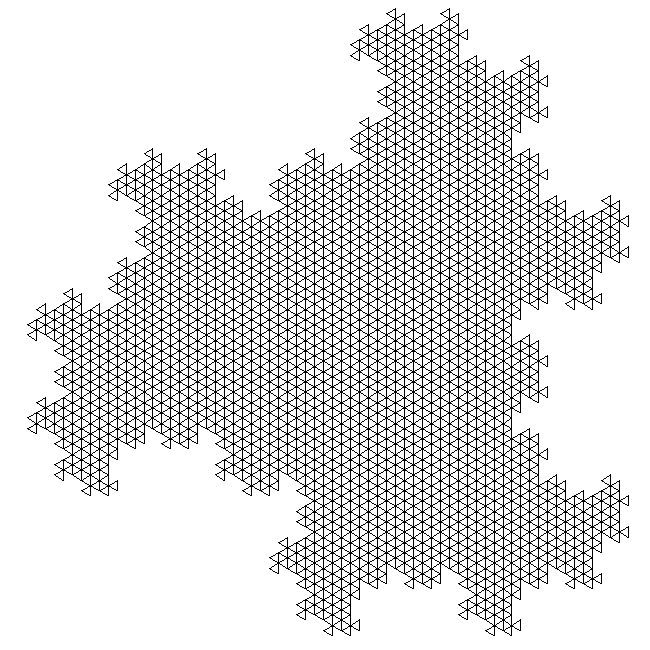
  


**Triangle**

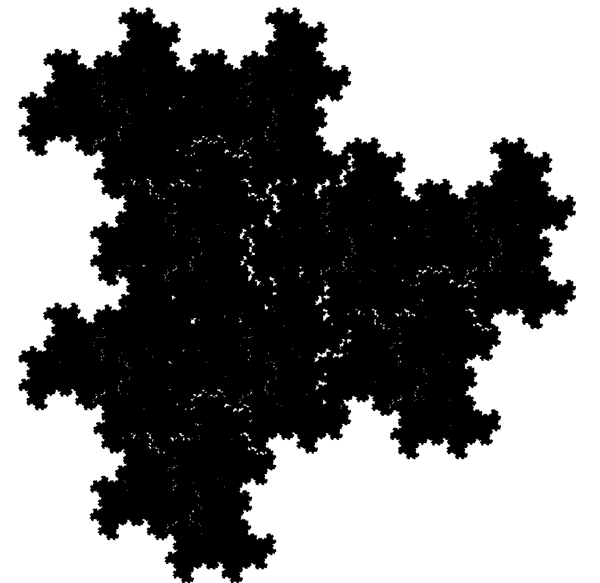
Written by [Paul Bourke](http://local.wasp.uwa.edu.au/%7Epbourke/fractals/)

**L-Systems**

axiom = F+F+F  
F -> F-F+F  
angle = 120

**As an IFS**



The IFS equations are as follows

xn+1 = a xn + b yn + e

yn+1 = c xn + d yn + f

The parameter table:

set 1 set 2 set 3

a 0.0 0.0 0.0

b 0.577 0.577 0.577

c -0.577 -0.577 -0.577

d 0.0 0.0 0.0

e 0.0951 0.4413 0.0952

f 0.5983 0.7893 0.9893

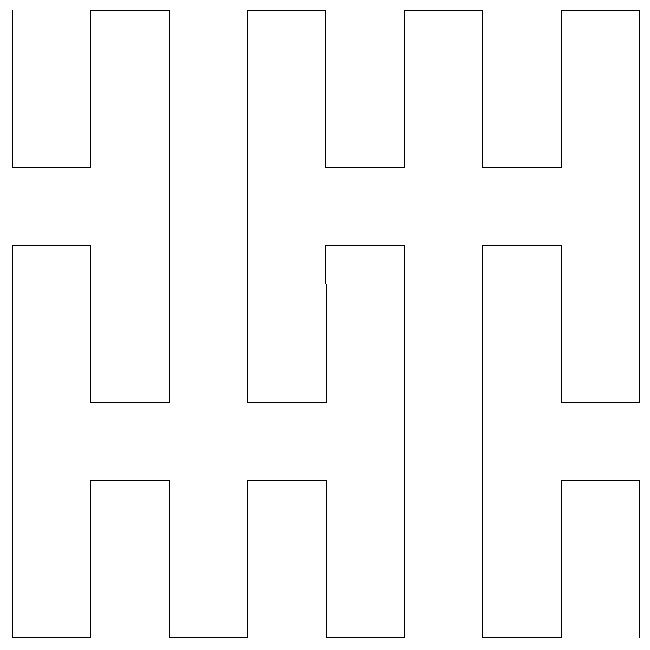
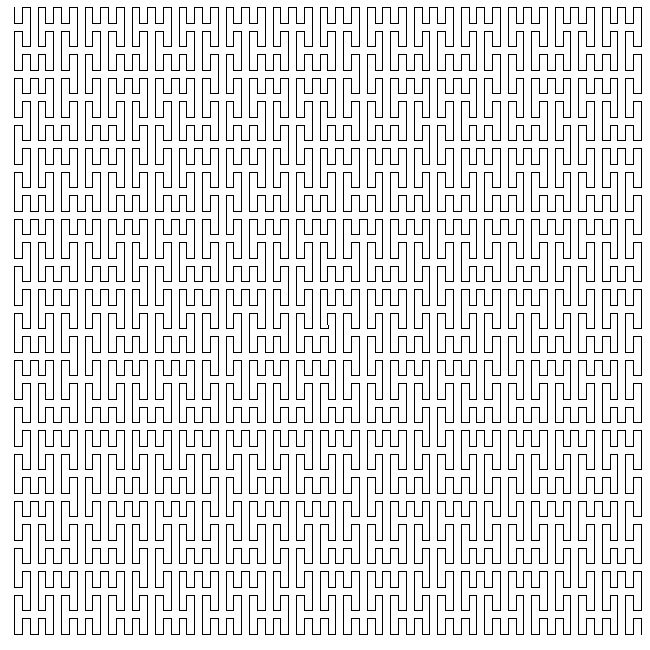
probability 1/3 1/3 1/3

**Peano Curve**

Written by [Paul Bourke](http://local.wasp.uwa.edu.au/%7Epbourke/fractals/)

Circa 1890

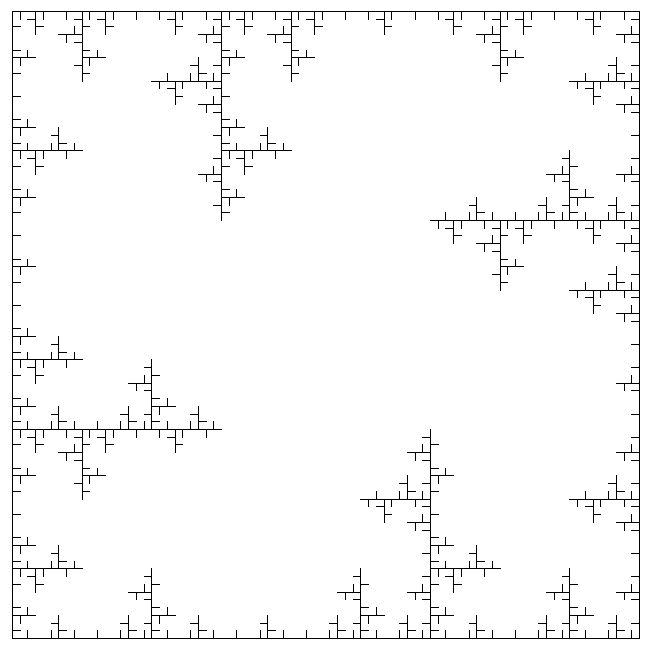
axiom = X  
X -> XFYFX+F+YFXFY-F-XFYFX  
y -> YFXFY-F-XFYFX+F+YFXFY  
angle = 90

**Crystal**

Written by [Paul Bourke](http://local.wasp.uwa.edu.au/%7Epbourke/fractals/)

axiom = F+F+F+F  
F -> FF+F++F+F  
angle = 90

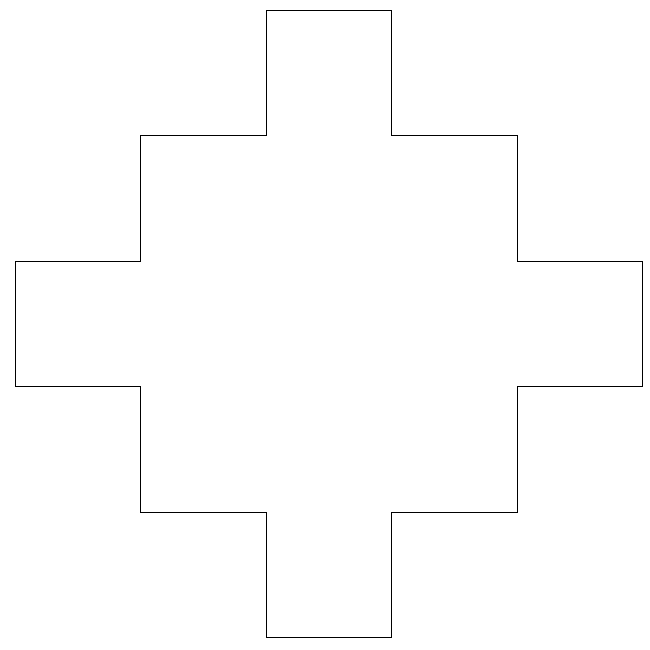
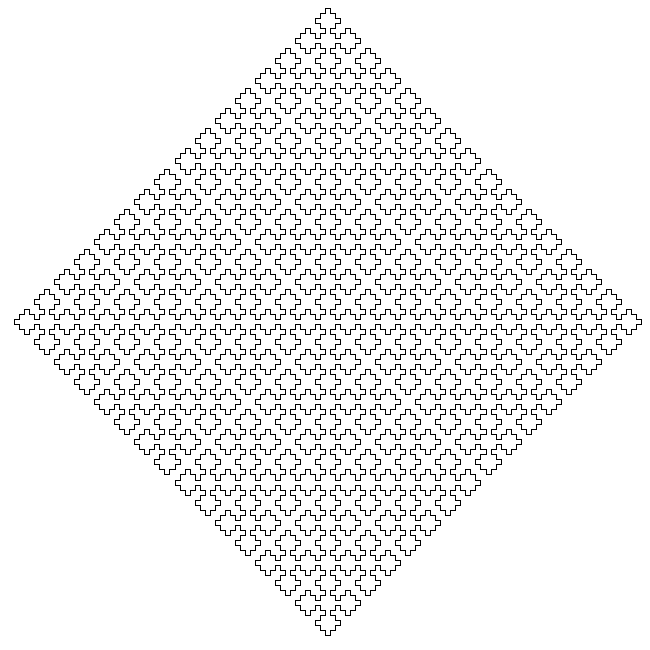
  


**Square Sierpinski**

Written by [Paul Bourke](http://local.wasp.uwa.edu.au/%7Epbourke/fractals/)

Circa 1912

axiom = F+XF+F+XF  
F -> XF-F+F-XF+F+XF-F+F-X  
angle = 90

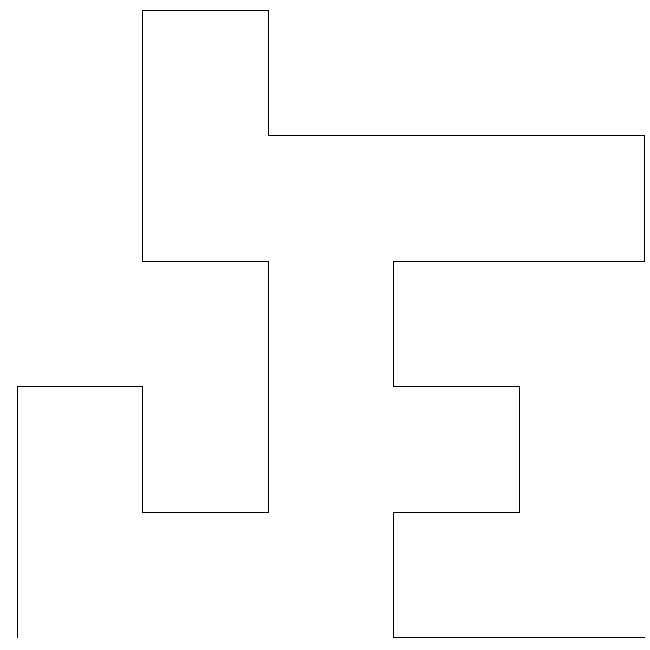
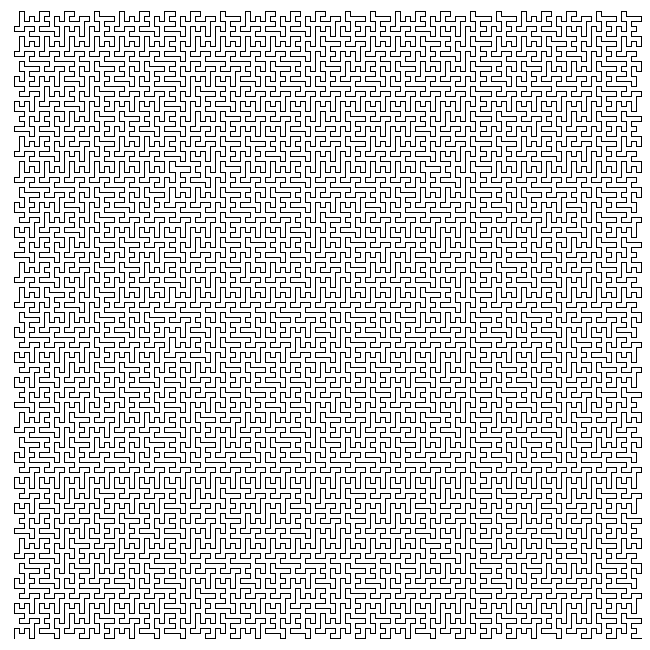
  


**Quadratic Gosper**

Written by [Paul Bourke](http://local.wasp.uwa.edu.au/%7Epbourke/fractals/)

Attributed to Dekking, 1982

axiom = -YF  
X -> XFX-YF-YF+FX+FX-YF-YFFX+YF+FXFXYF-FX+YF+FXFX+YF-FXYF-YF-FX+FX+YFYF-  
Y -> +FXFX-YF-YF+FX+FXYF+FX-YFYF-FX-YF+FXYFYF-FX-YFFX+FX+YF-YF-FX+FX+YFY  
angle = 90

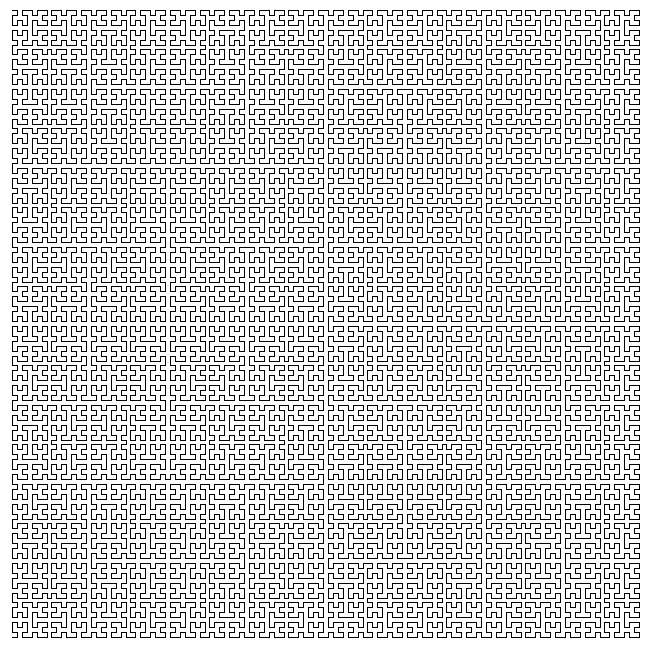
  


**Hilbert**

Written by [Paul Bourke](http://local.wasp.uwa.edu.au/%7Epbourke/fractals/)

Attributed to the German mathematician David Hilbert, circa 1891

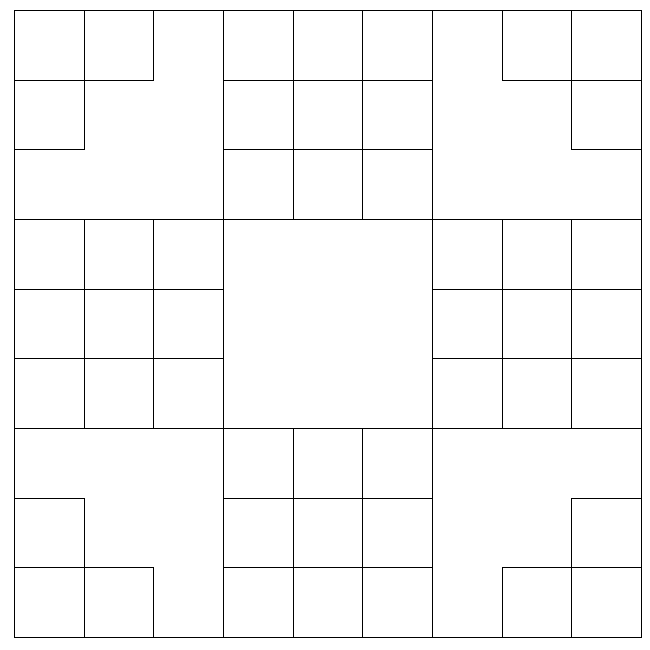
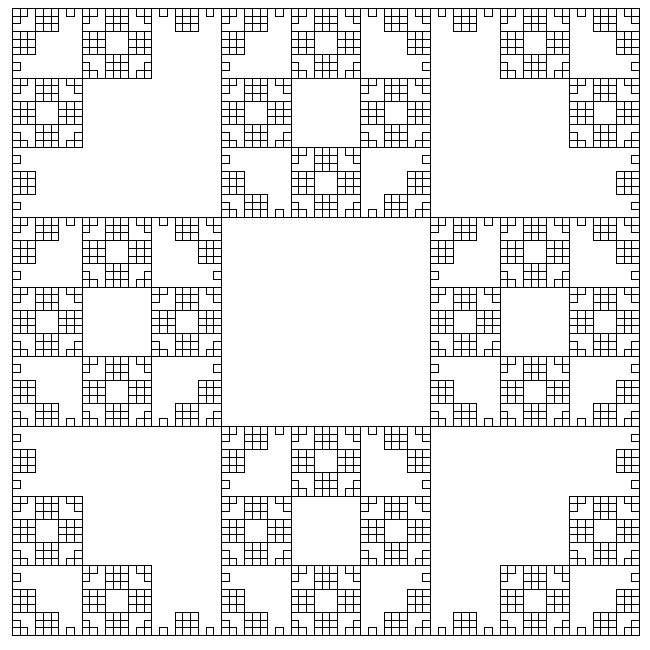
axiom = X  
X -> -YF+XFX+FY-  
Y -> +XF-YFY-FX+  
angle = 90

**Board**

Written by [Paul Bourke](http://local.wasp.uwa.edu.au/%7Epbourke/fractals/)

axiom = F+F+F+F  
F -&bt; FF+F+F+F+FF  
angle = 90

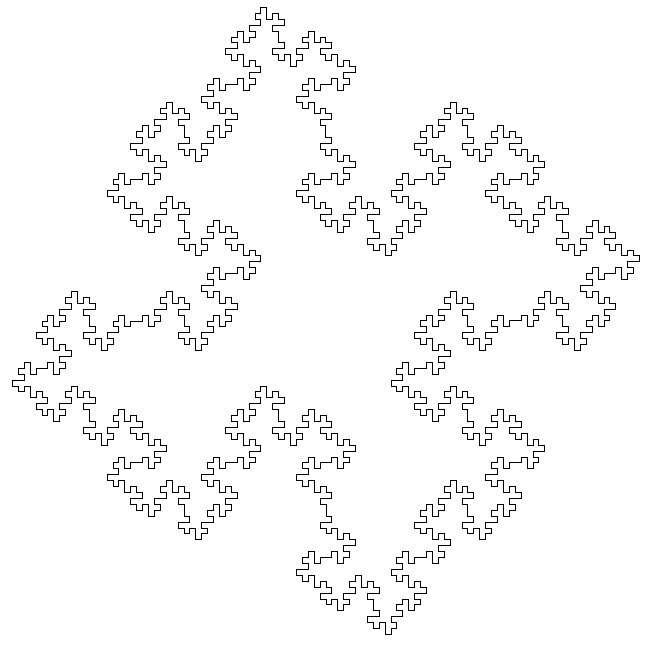
  


**Koch Curve**

Written by [Paul Bourke](http://local.wasp.uwa.edu.au/%7Epbourke/fractals/)

Fractal Dimension: 1.5

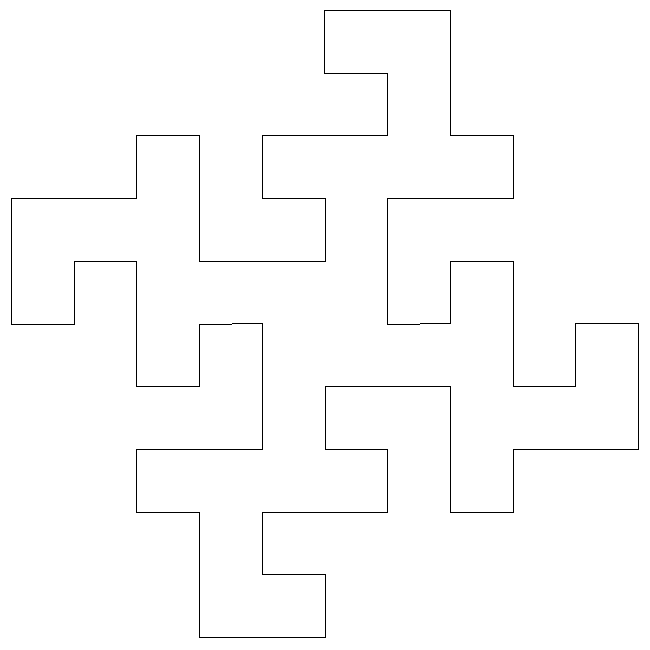
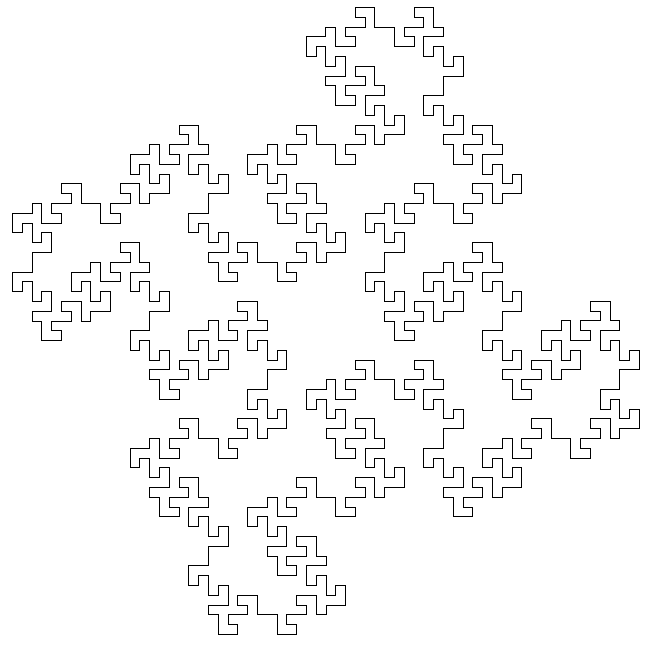
axiom = F+F+F+F  
F -> F+F-F-FF+F+F-F  
angle = 90

**Quadratic Koch Island**

Written by [Paul Bourke](http://local.wasp.uwa.edu.au/%7Epbourke/fractals/)

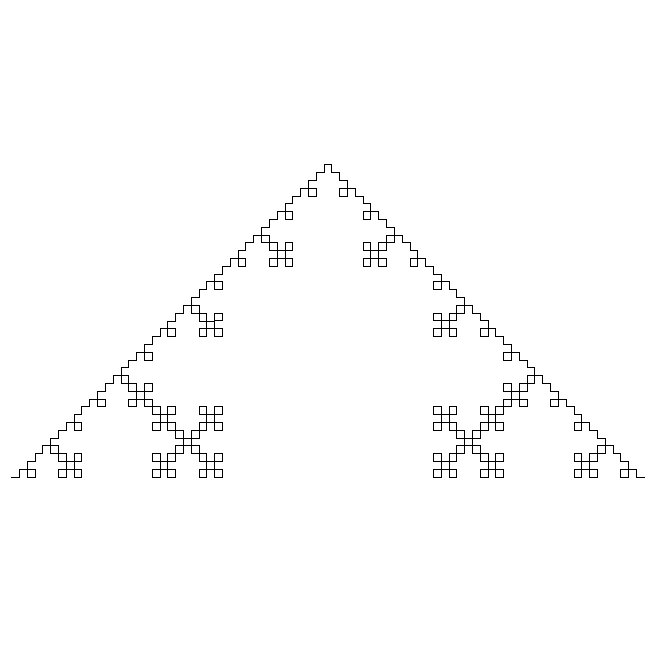
axiom = F+F+F+F  
F -> F-FF+FF+F+F-F-FF+F+F-F-FF-FF+F  
angle = 90

**Quadratic Snowflake**

Written by [Paul Bourke](http://local.wasp.uwa.edu.au/%7Epbourke/fractals/)

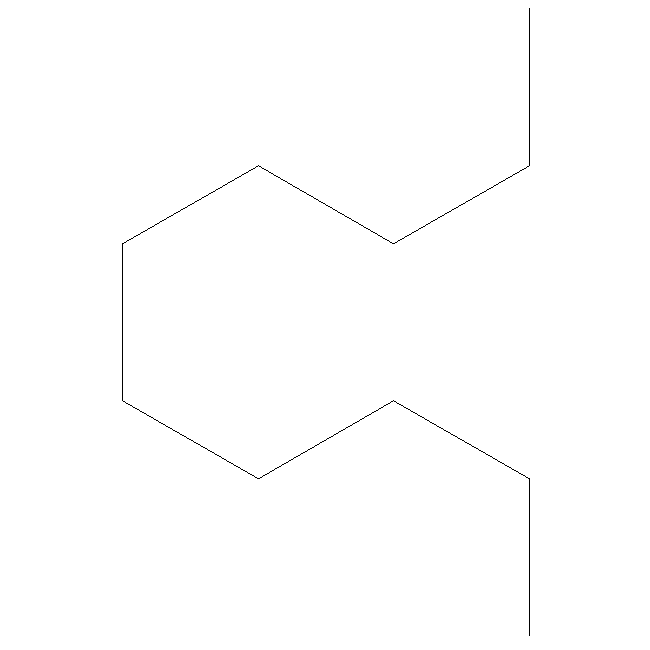
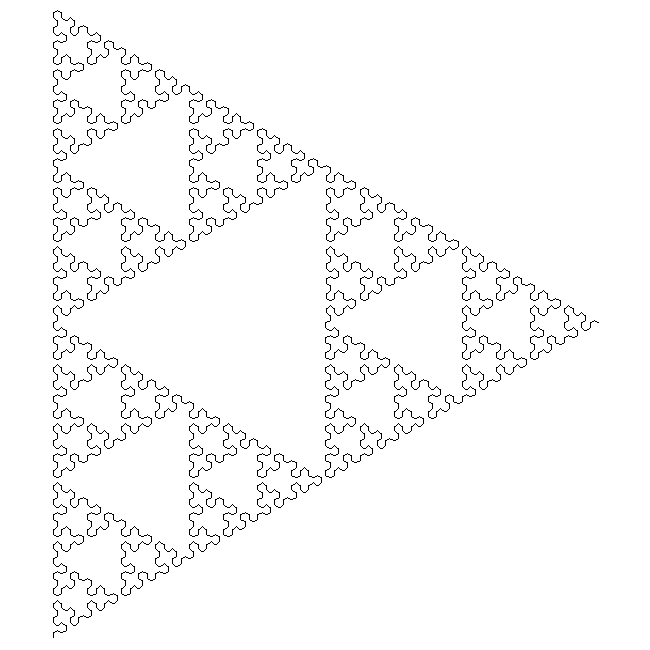
axiom = F  
F -> F-F+F+F-F  
angle = 90

**Sierpinski Arrowhead**

Written by [Paul Bourke](http://local.wasp.uwa.edu.au/%7Epbourke/fractals/)

axiom = YF  
X -> YF+XF+Y  
Y -> XF-YF-X  
angle = 60

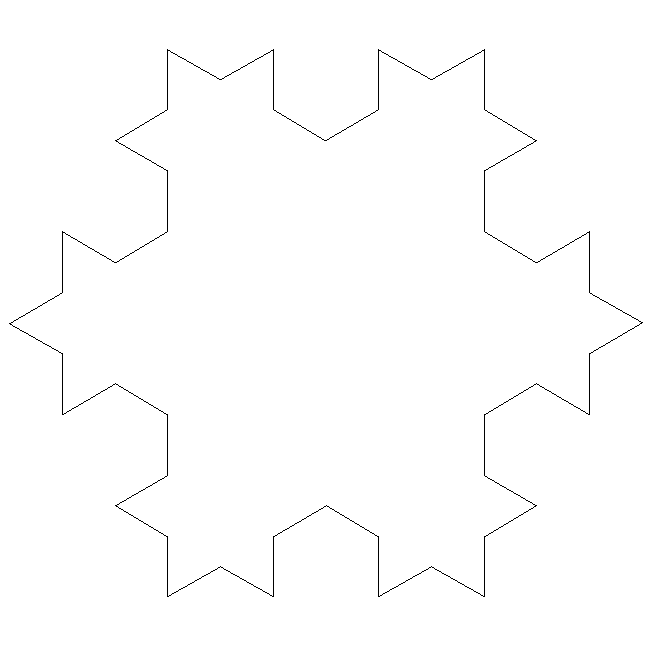
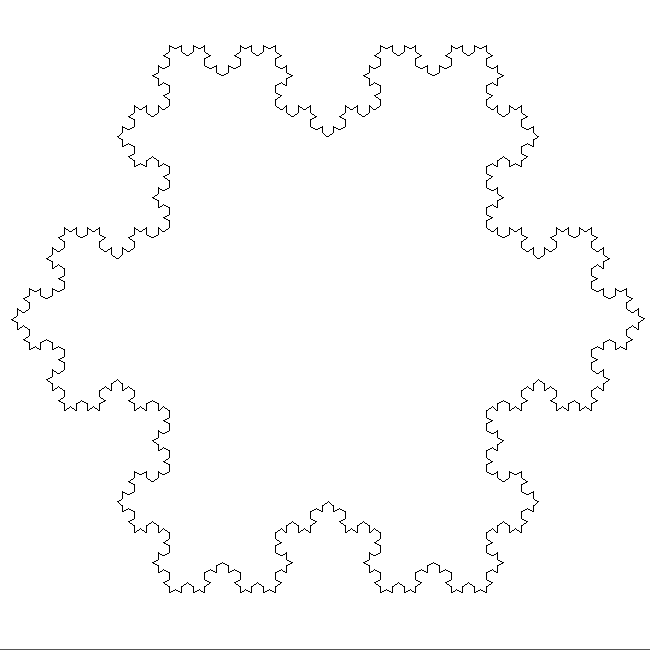
**Von Koch Snowflake**

Written by [Paul Bourke](http://local.wasp.uwa.edu.au/%7Epbourke/fractals/)

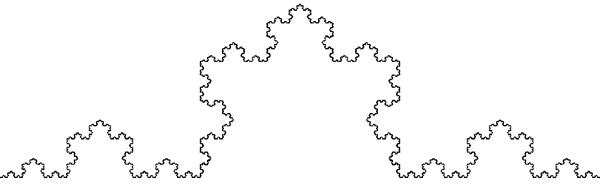
Fractal Dimension: log(4)/log(3) = 1.262

**L-ystems**

axiom = F++F++F  
F -> F-F++F-F  
angle = 60

**IFS**



The IFS equations are as follows

xn+1 = a xn + b yn + e

yn+1 = c xn + d yn + f

The parameter table:

set 1 set 2 set 3 set 4

a 0.3330 0.3330 0.1670 0.1670

b 0.0000 0.0000 -0.2890 0.2890

c 0.0000 0.0000 0.2890 -0.2890

d 0.3330 0.3330 0.1670 0.1670

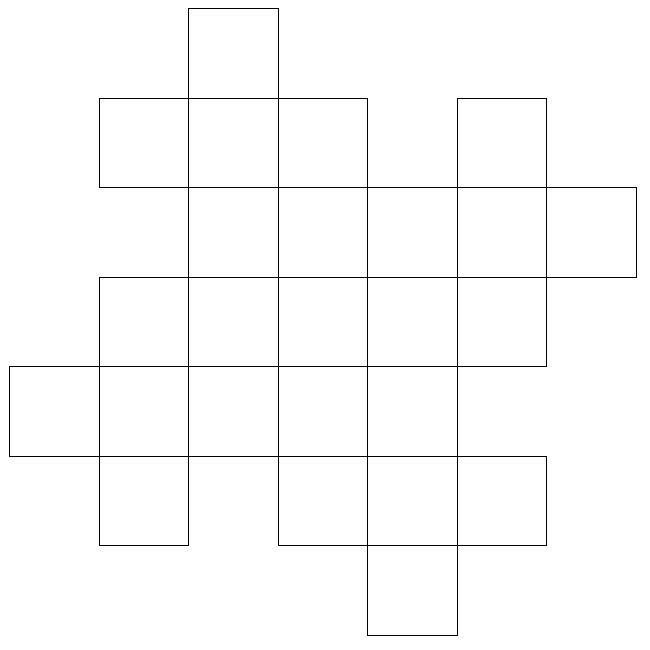
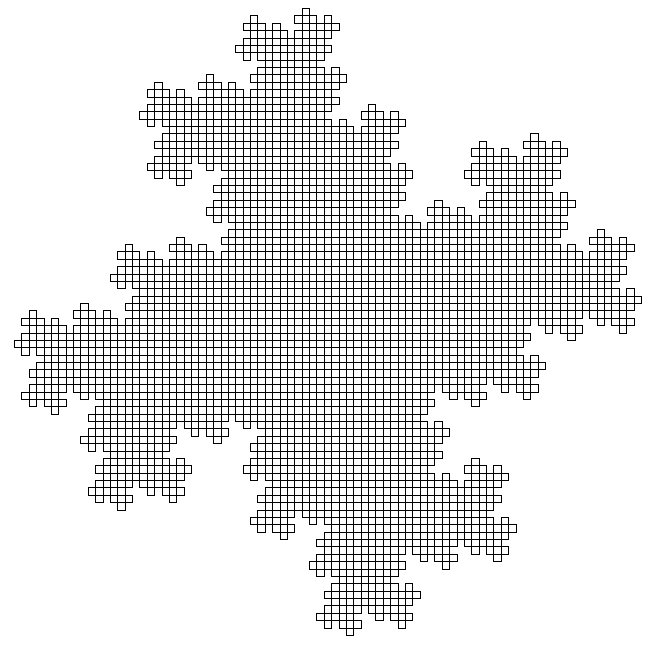
e -0.3330 0.3330 -0.0830 0.0830

f 0.0000 0.0000 0.1440 0.1440

**Cross**

Written by [Paul Bourke](http://local.wasp.uwa.edu.au/%7Epbourke/fractals/)

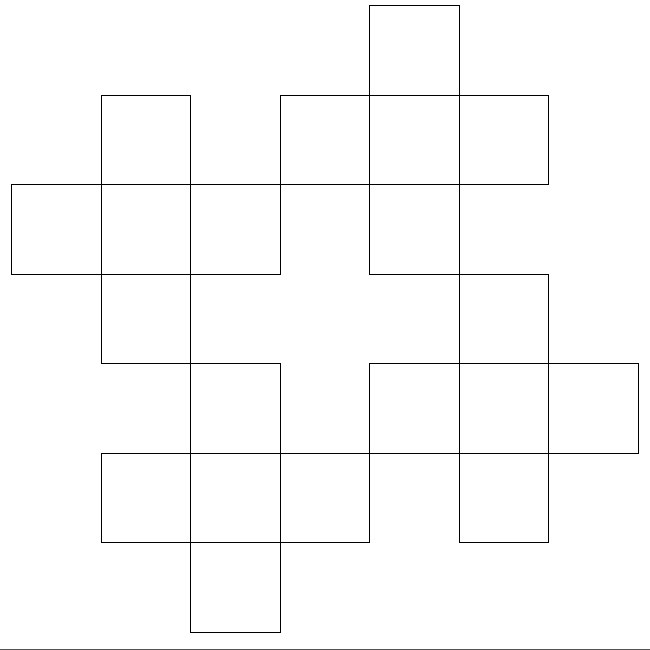
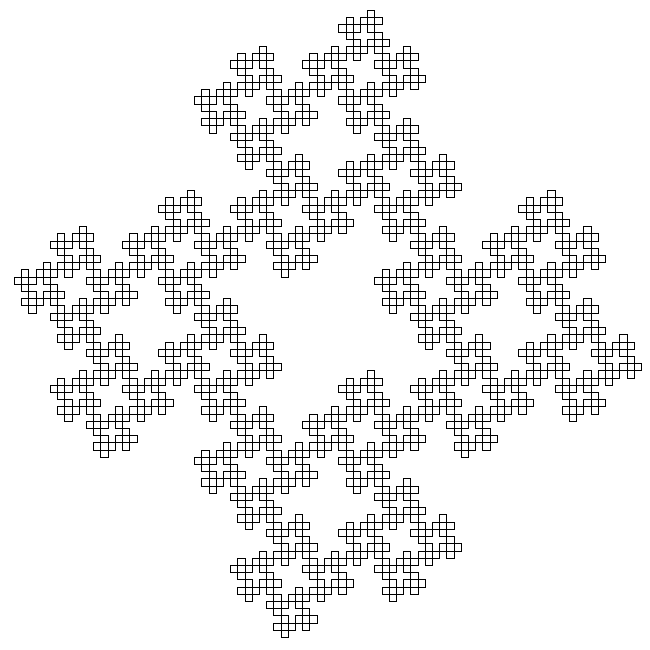
axiom = F+F+F+F  
F -> F+FF++F+F  
angle = 90

**Cross**

Written by [Paul Bourke](http://local.wasp.uwa.edu.au/%7Epbourke/fractals/)

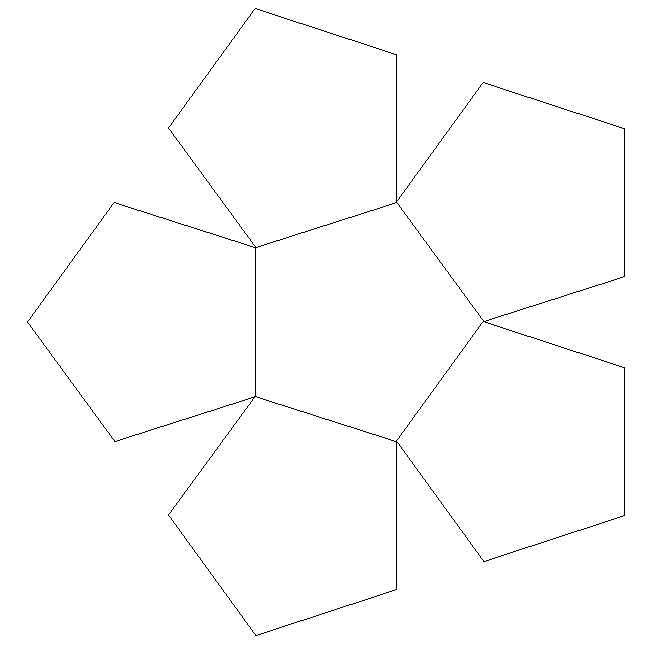
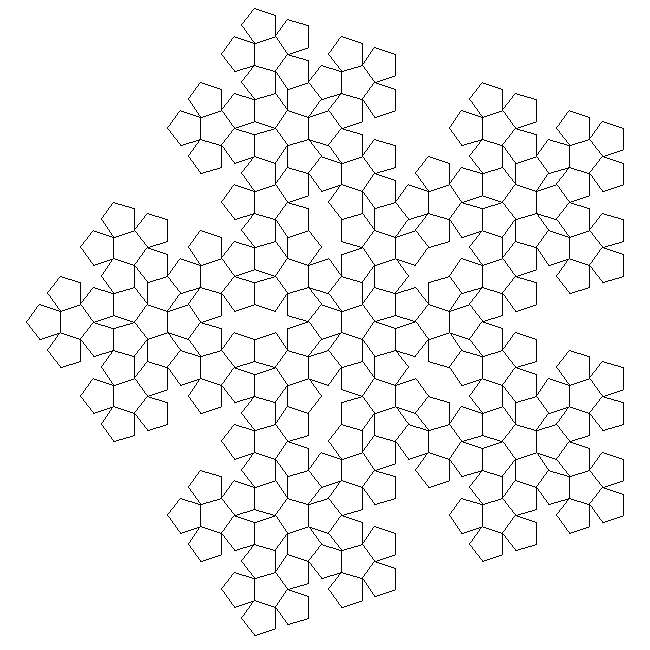
axiom = F+F+F+F  
F -> F+F-F+F+F  
angle = 90

**Pentaplexity**

Written by [Paul Bourke](http://local.wasp.uwa.edu.au/%7Epbourke/fractals/)

axiom = F++F++F++F++F  
F -> F++F++F|F-F++F  
angle = 36

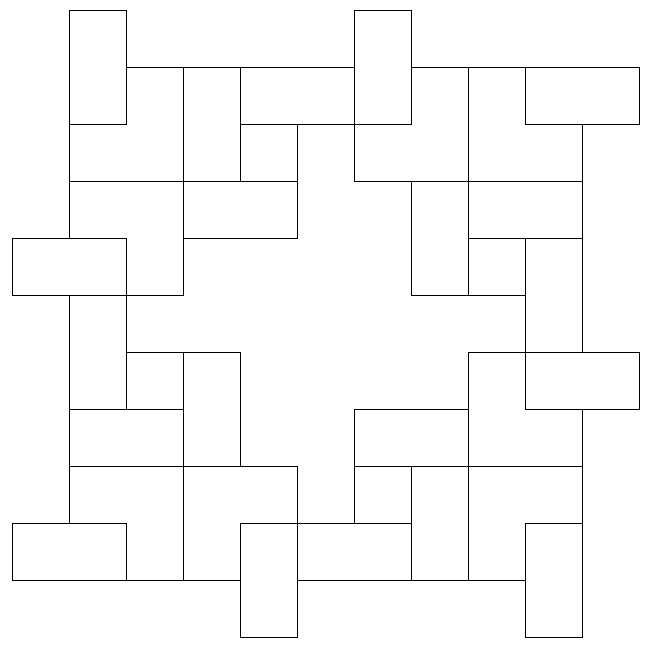
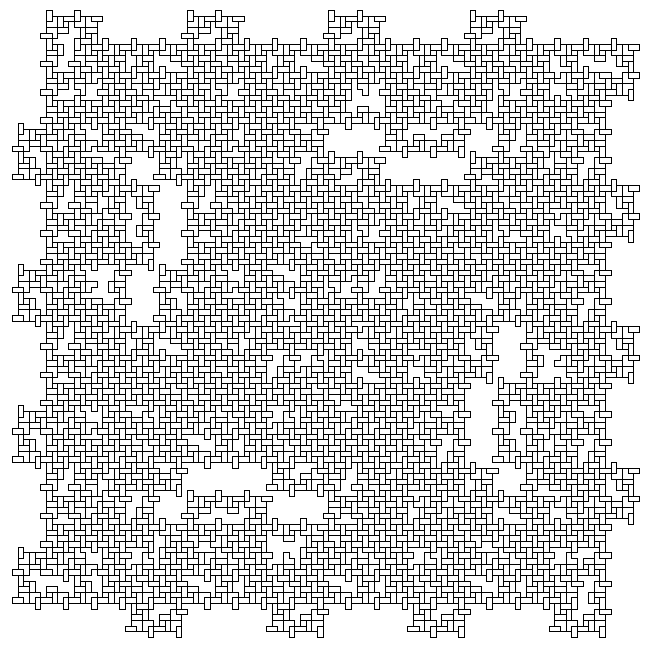
  


**Tiles**

Written by [Paul Bourke](http://local.wasp.uwa.edu.au/%7Epbourke/fractals/)

Fractal Dimension: log(7)/log(3) = 1.771

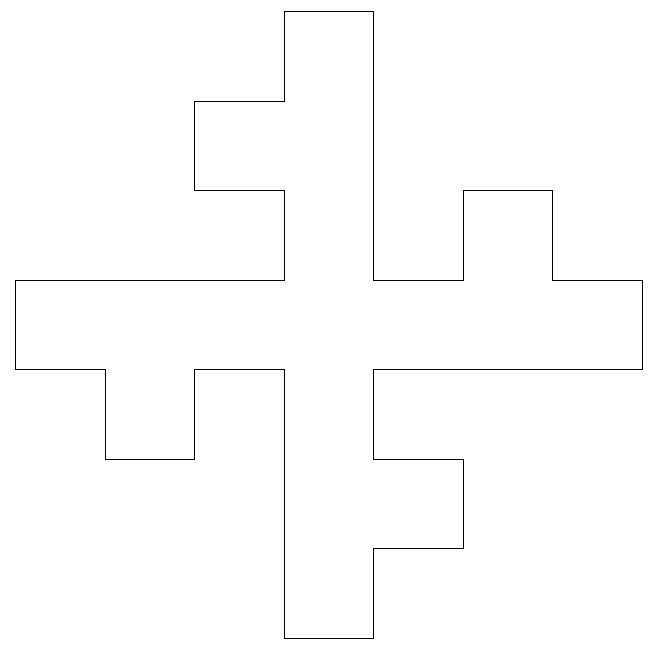
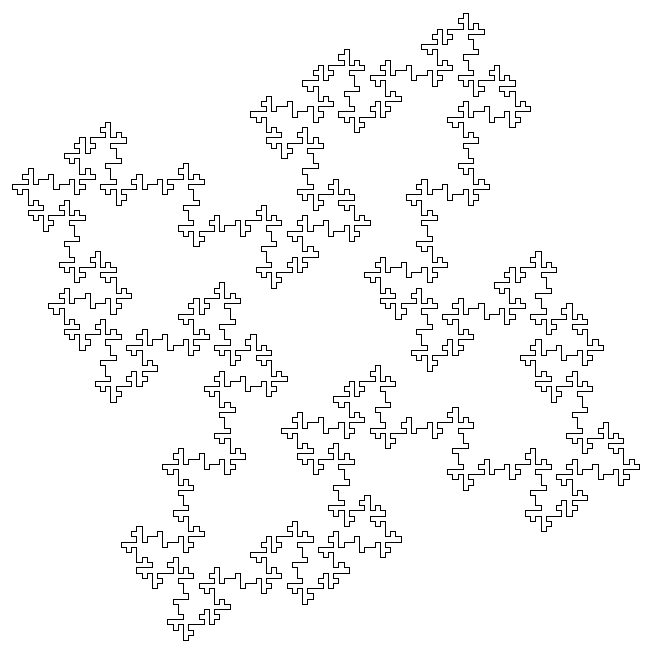
axiom = F+F+F+F  
F -> FF+F-F+F+FF  
angle = 90

**Quadratic Koch Island**

Written by [Paul Bourke](http://local.wasp.uwa.edu.au/%7Epbourke/fractals/)

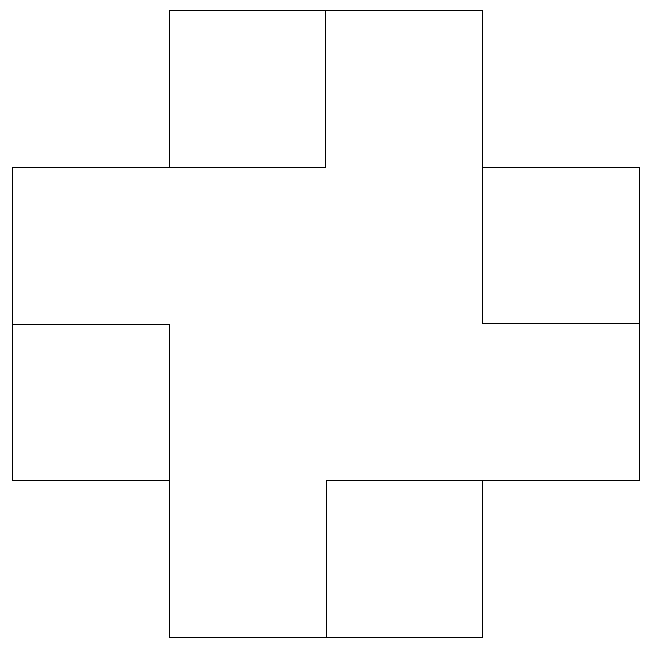
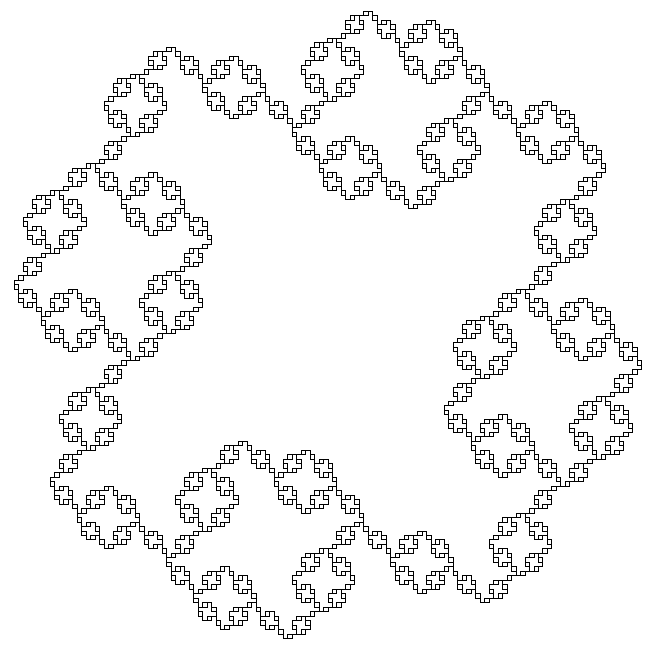
axiom = F+F+F+F  
F -> F+F-F-FFF+F+F-F  
angle = 90

**Rings**

Written by [Paul Bourke](http://local.wasp.uwa.edu.au/%7Epbourke/fractals/)

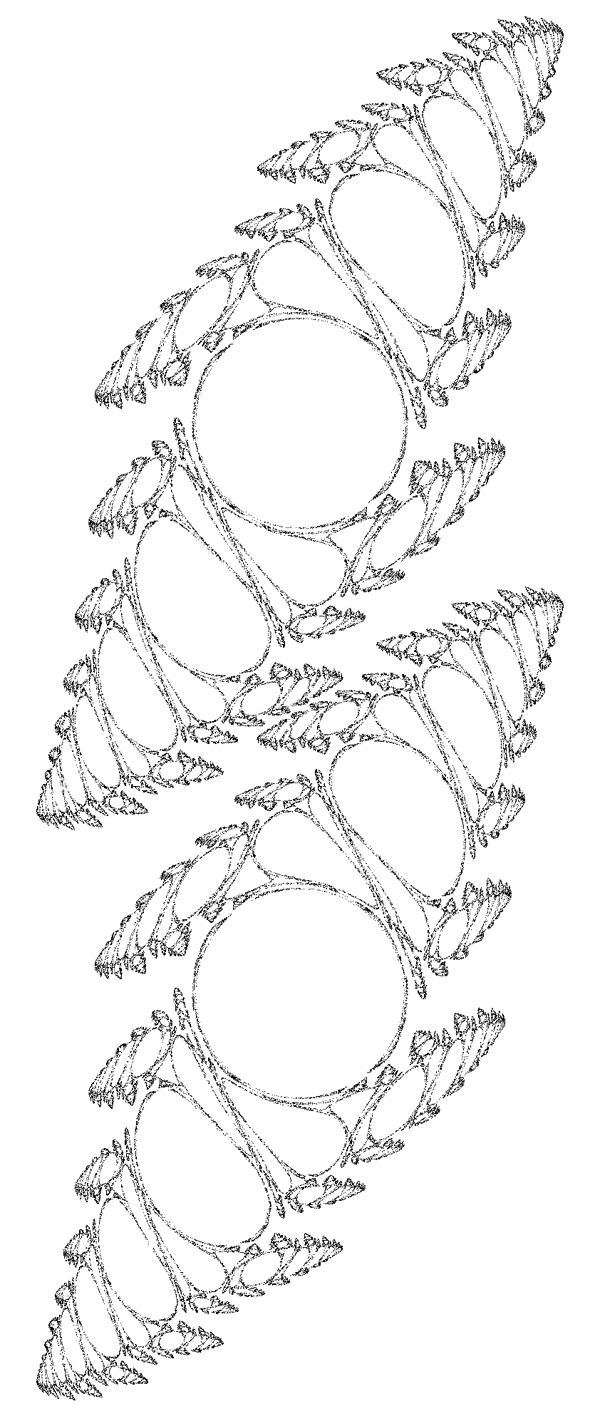
axiom = F+F+F+F  
F -> FF+F+F+F+F+F-F  
angle = 90

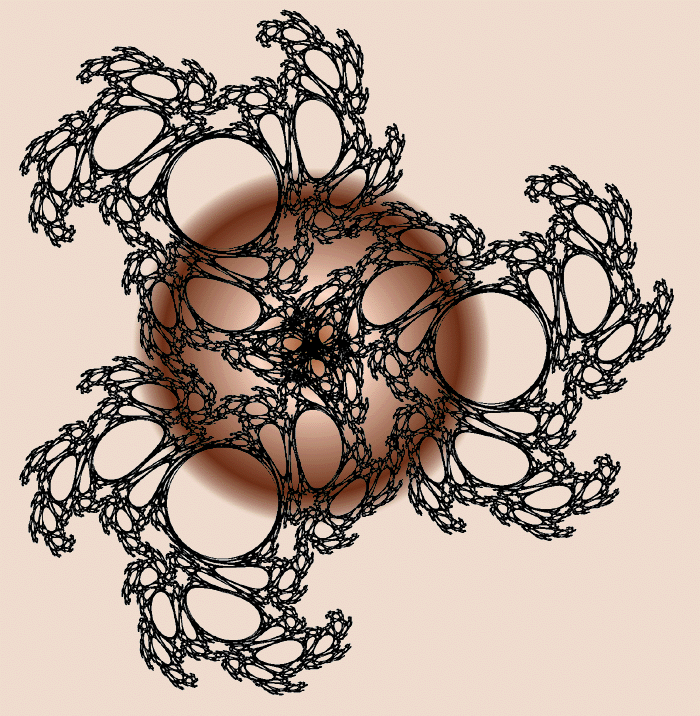
  


**Twon Dragon**

By Roger Bagula  
Compiled and graphics by [Paul Bourke](http://local.wasp.uwa.edu.au/%7Epbourke/fractals/)

[Basic source code](http://local.wasp.uwa.edu.au/%7Epbourke/fractals/twondragon/roger15.basic) -- [C Source code](http://local.wasp.uwa.edu.au/%7Epbourke/fractals/twondragon/roger15.c)





**L-System User Notes**

(Lindenmayer Systems)  
Written by [Paul Bourke](http://local.wasp.uwa.edu.au/%7Epbourke/fractals/)  
Version 2.5, July 1991

**Introduction**

This program implements some of the L-Systems discussed in "Lecture Notes in Biomathematics" by Przemyslaw Prusinkiewcz and James Hanan. A brief description of an 0L system will be presented here but for a more complete description the user should consult the literature.

The application was initially written to investigate methods of incorporating objects with a large number of drawing elements (lines, polygons) into a CAD package. L Systems is one way for example of "generating" trees at the rendering stage but not during the editing stage where the image complexity will slow down the response time.

**Simple example of a 0L system**

A string of characters (symbols) is rewritten on each iteration according to some replacement rules. Consider an initial string (axiom)

F+F+F+F

and a rewriting rule

F --> F+F-F-FF+F+F-F

After one iteration the following string would result

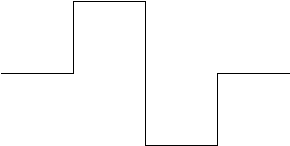
F+F-F-FF+F+F-F + F+F-F-FF+F+F-F + F+F-F-FF+F+F-F + F+F-F-FF+F+F-F

For the next iteration the same rule is applied but now to the string resulting from the last iteration

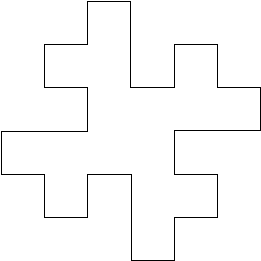
F+ F-F-FF+ F+ F-F+ F+ F-F-FF+ F+ F-F-F+ F-F-FF+ F+ F-F-F+ F-F-FF+ F+ F-FF+ F-F-FF+ F+ F-F+ F+ F-F-FF+ F+ F-F+ F+ F-F-FF+ F+ F-F-F+ F-F-FF+ F+ F-F+ F+ F-F-FF+ F+ F-F+ F+ F-F-FF+ F+ F-F-F+ F-F-FF+ F+ F-F-F+ F-F-FF+ F+ F-FF+ F-F-FF+ F+ F-F+ F+ F-F-FF+ F+ F-F+ F+ F-F-FF+ F+ F-F-F+ F-F-FF+ F+ F-F+ F+ F-F-FF+ F+ F-F+ F+ F-F-FF+ F+ F-F-F+ F-F-FF+ F+ F-F-F+ F-F-FF+ F+ F-FF+ F-F-FF+ F+ F-F+ F+ F-F-FF+ F+ F-F+ F+ F-F-FF+ F+ F-F-F+ F-F-FF+ F+ F-F+ F+ F-F-FF+ F+ F-F+ F+ F-F-FF+ F+ F-F-F+ F-F-FF+ F+ F-F-F+ F-F-FF+ F+ F-FF+ F-F-FF+ F+ F-F+ F+ F-F-FF+ F+ F-F+ F+ F-F-FF+ F+ F-F-F+ F-F-FF+ F+ F-F

Some symbols are now given a graphical meaning, for example, F means move forward drawing a line, + means turn right by some predefined angle (90 degrees in this case), - means turn left.

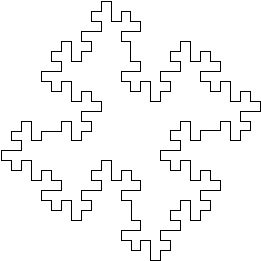
Using these symbols the initial string F+F+F+F is just a rectangle (ø = 90). The replacement rule F --> F+F-F-FF+F+F-F replaces each forward movement by the following figure



The first iteration interpreted graphically is



The next iteration interpreted graphically is:



and so on.

|  |  |
| --- | --- |
| **Example:**  Axiom X F --> FF X --> F-[[X]+X]+F[+FX]-X ø = 22.5 | http://local.wasp.uwa.edu.au/%7Epbourke/fractals/lsys/lsys04.gif |

|  |  |
| --- | --- |
| **Example:**  Axiom F+F+F+F F --> FF+F-F+F+FF ø = 90 | http://local.wasp.uwa.edu.au/%7Epbourke/fractals/lsys/lsys05.gif |

**Symbols** The following characters have a geometric interpretation.

Character Meaning

F Move forward by line length drawing a line

f Move forward by line length without drawing a line

+ Turn left by turning angle

- Turn right by turning angle

| Reverse direction (ie: turn by 180 degrees)

[ Push current drawing state onto stack

] Pop current drawing state from the stack

# Increment the line width by line width increment

! Decrement the line width by line width increment

@ Draw a dot with line width radius

{ Open a polygon

} Close a polygon and fill it with fill colour

> Multiply the line length by the line length scale factor

< Divide the line length by the line length scale factor

& Swap the meaning of + and -

( Decrement turning angle by turning angle increment

) Increment turning angle by turning angle increment

When drawing the graphical representation of the L string all other characters are ignored. The user may choose and use any other single printable characters for the replacement rules except, note: this excludes "white" characters such as spaces and tabs. For context sensitive L systems the \* character is used to represent any match.

See the SYMBOL menu item for a summary of the reserved symbols. **Menus**

|  |  |
| --- | --- |
| http://local.wasp.uwa.edu.au/%7Epbourke/fractals/lsys/lsys06.gif | As with most applications the ABOUT menu item located in the Apple menu before the desk accessories displays information such as the version number and contact address of the developer. |
| http://local.wasp.uwa.edu.au/%7Epbourke/fractals/lsys/lsys07.gif | This is the standard FILE menu although some of the items are not implemented.  NEW resets all the variables including the rules to their default values.  OPEN to load a rule description file previously saved or possibly created with a text editor.  SAVE and SAVE AS create rule description files.  PRINT and PAGESETUP control direct printing.  QUIT to exit from the program when finished. |
| http://local.wasp.uwa.edu.au/%7Epbourke/fractals/lsys/lsys08.gif | The standard EDIT menu. The first 5 items are only included for compatibility reasons and do not apply to this application. (they are used with desk accesories in some cases)  The two special copy items allow images to be transfered to other applications depending on the prefered data type. Line drawings should be copied for CAD packages and bitmaps for painting programs.  Normally images will be scaled to fit the window, it is possible to zoom in or out of areas with the last two items. |
| http://local.wasp.uwa.edu.au/%7Epbourke/fractals/lsys/lsys09.gif | This is the main menu for this application.  REDRAW updates the current display, for example, if the drawing was canceled with clover-. (period)  Menu items 2 through 4 control the iteration depth. RESET sets it to 0, the number of iterations to compute may be incremented, decremented, or set to any particular number.  FRAME, FILL and BACKGROUND are popup menus that allow the colour of the lines, polygon fills and background to be set.  PRODUCTIONS allows the current rules to be altered. AXIOM menu item sets the initial string. STATE menu item varies environment variables. SYMBOL LIST displays the meaning given to various predefined and reserved symbols.  The type of system is selectable from the last 3 items. |
| http://local.wasp.uwa.edu.au/%7Epbourke/fractals/lsys/lsys10.gif | This contains a number of predefined image specifications grouped together in categories given by the popup menu names.  Users are encouraged to submit further examples that can be added to this library. |

**Advanced features/comments**

* Always ensure that the '[' and ']' brackets match in number, ie: there should always be the same number of each type. If not a stack overflow will quickly occur, a message displayed, and the iteration depth decremented. Note: non matching brackets are not tested before the string in interpreted, a problem only occurs when a stack overflow situation is encountered.
* Polygons (using "{" and "}") cannot be nested, ie: there can only be one open at a time. If a polygon open symbol "{" is encountered while another polygon is still open it will be ignored. If a polygon is left open when the end of the string is reached it will be automatically closed.
* For a complete list of predefined symbol meanings look at the SYMBOL list under the ACTION menu.
* Best quality hardcopy line drawing images can be obtained by using MacDraw or Claris CAD. In both cases copy the image as a line drawing, paste it into MacDraw say, set the lines to black and line width 0.1mm, then print on a LaserWriter. Note: the number of lines can quickly approach the thousands and even tens of thousands, don't expect the printing to be fast.
* Time consuming drawing may be cancelled by typing clover-. (period)
* The maximum length of the strings generated is 300,000 symbols. Attempting to exceed this will give an error message and automatic decrementing of the iteration count.
* The maximum stack depth is 100. This means that a bracketed string cannot have brackets nested more than 100 deep.

ie: [ [ [ ] ] [ [ [ [ ] ] [ ] ] ] ]

1 2 3 2 1 2 3 4 5 4 3 4 3 2 1 0 is nested 5 deep

If the maximum stack depth is exceeded an appropriate error message will be displayed and the iteration depth decremented until the stack no longer overflows.

* When copying to the clipboard as lines make the drawing window as large as possible for maximum resolution.

**References**

Heinz-Otto and Deitmar Saupe  
*The Science of Fractal Images*  
Springer-Verlag

Przemyslaw Prusinkiewicz  
*Application of L-Systems to Computer Imagery*  
Lecture Notes in Computer Science #291, Pages 534-548

Przemyslaw Prusinkiewicz, James Hanan  
*Lecture Notes in Biomathematics #79*

Prusinkiewicz, P  
*Graphical Applications of L-Systems*  
Proc. of Graphics Interface 1986 - Vision Interface, 1986, Pages 247-253.

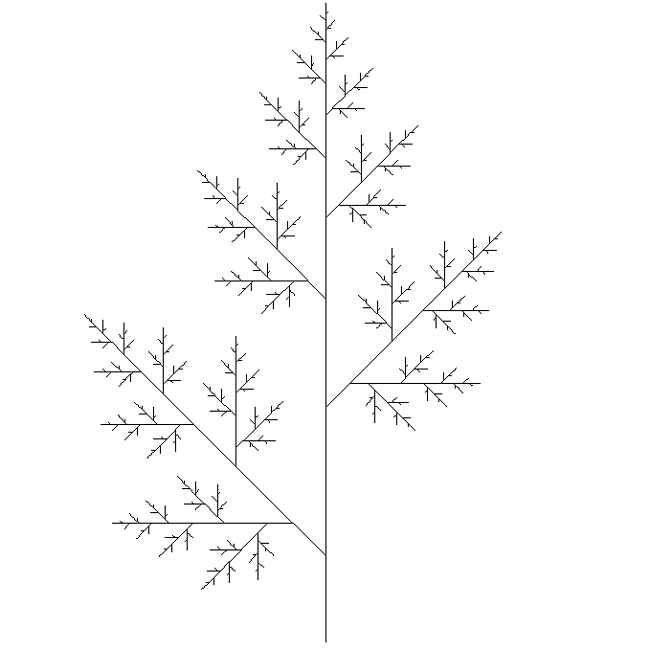
Smith, A.R.  
*Plants, Fractals, and Formal Languages*  
Computer Graphics, 18, 3, 1984, Pages 1-10

Prusinkiewicz, P and Lindenmayer A.  
*The Algorithmic Beauty of Plants*  
Springer Verleg, 1990, Pages 40-50

**L-System Leaf**

Written by [Paul Bourke](http://local.wasp.uwa.edu.au/%7Epbourke/fractals/)

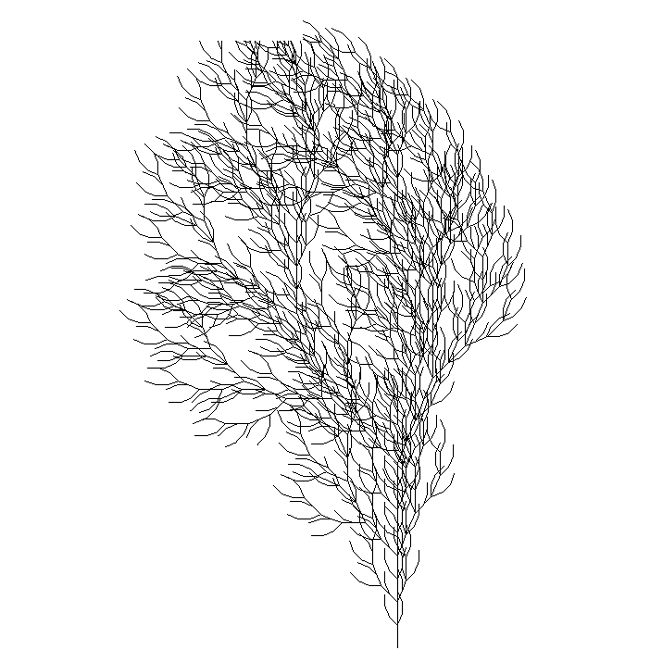
axiom = a  
F -> >F<  
a -> F[+x]Fb  
b -> F[-y]Fa  
x -> a  
y -> b  
angle = 45  
length factor = 1.36



**L-System Bush**

Written by [Paul Bourke](http://local.wasp.uwa.edu.au/%7Epbourke/fractals/)

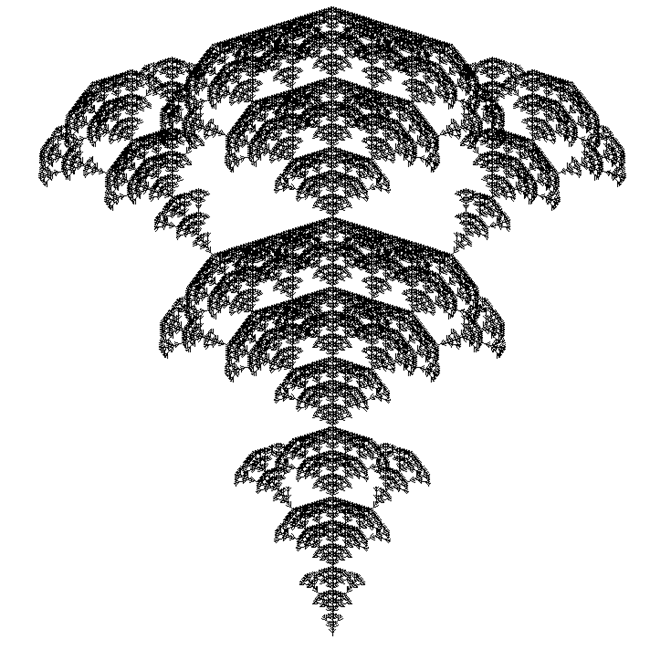
axiom = F  
F -> FF+[+F-F-F]-[-F+F+F]  
angle = 22.5



**L-System Bush**

Written by [Paul Bourke](http://local.wasp.uwa.edu.au/%7Epbourke/fractals/)

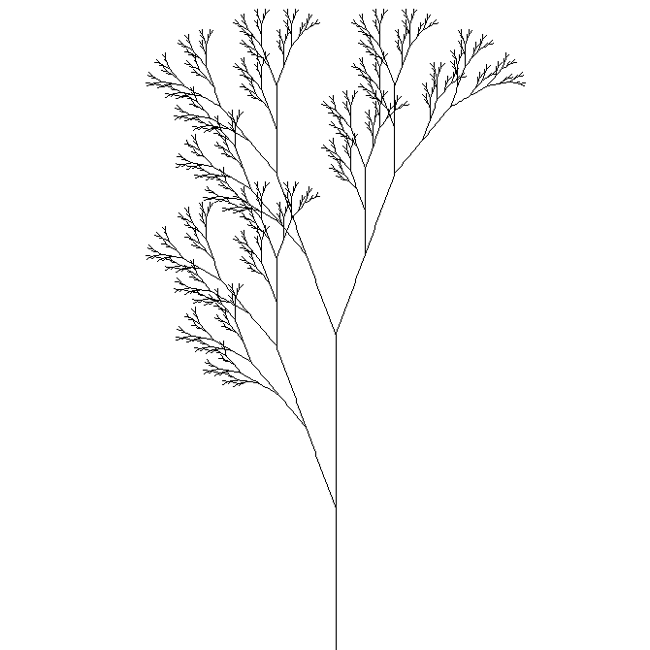
axiom = F  
F -> F[+FF][-FF]F[-F][+F]F  
angle = 35



**L-System sticks**

Written by [Paul Bourke](http://local.wasp.uwa.edu.au/%7Epbourke/fractals/)

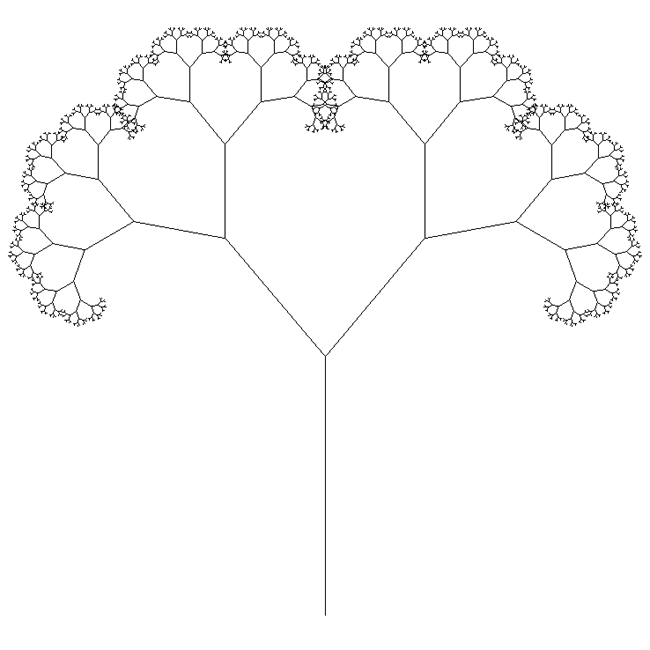
axiom = X  
F -> FF  
X -> F[+X]F[-X]+X  
angle = 20



**L-System Bush**

Written by [Paul Bourke](http://local.wasp.uwa.edu.au/%7Epbourke/fractals/)

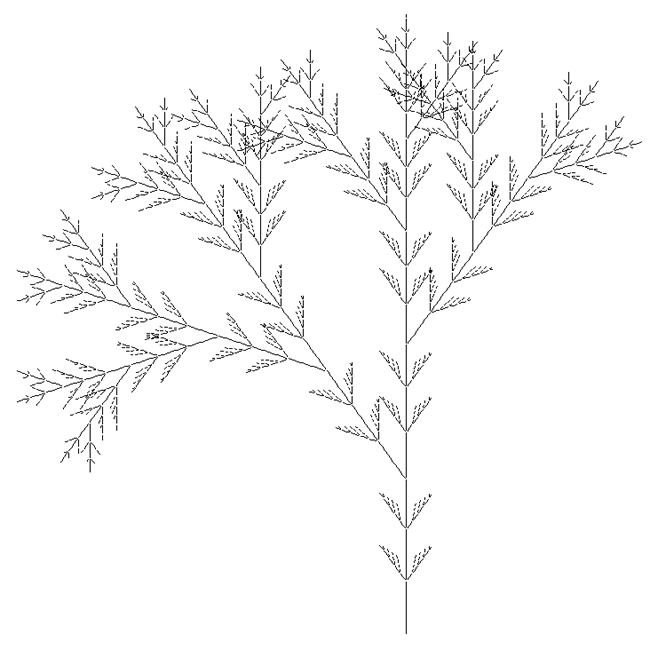
axiom = FX  
X -> >[-FX]+FX  
angle = 40



**L-System algae**

Written by [Paul Bourke](http://local.wasp.uwa.edu.au/%7Epbourke/fractals/)

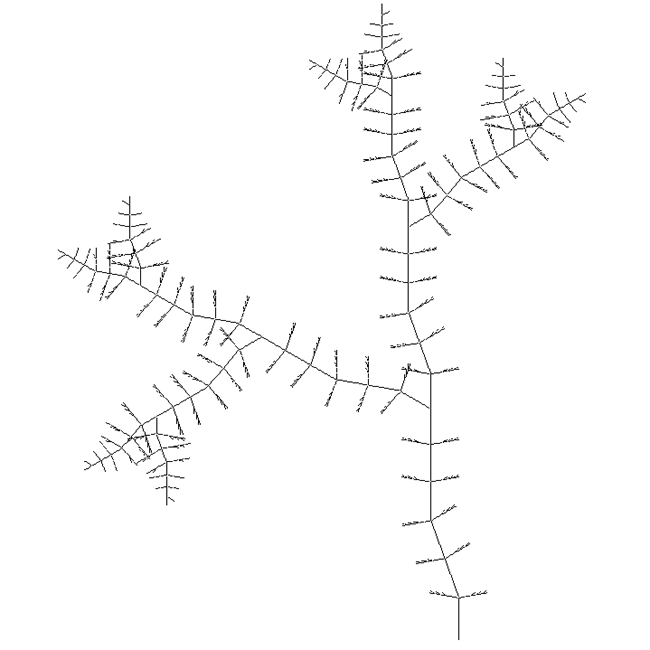
axiom = aF  
a -> FFFFFv[+++h][---q]fb  
b -> FFFFFv[+++h][---q]fc  
c -> FFFFFv[+++fa]fd  
d -> FFFFFv[+++h][---q]fe  
e -> FFFFFv[+++h][---q]fg  
g -> FFFFFv[---fa]fa  
h -> ifFF  
i -> fFFF[--m]j  
j -> fFFF[--n]k  
k -> fFFF[--o]l  
l -> fFFF[--p]  
m -> fFn  
n -> fFo  
o -> fFp  
p -> fF  
q -> rfF  
r -> fFFF[++m]s  
s -> fFFF[++n]t  
t -> fFFF[++o]u  
u -> fFFF[++p]  
v -> Fv  
angle = 12



**L-System Algae**

Written by [Paul Bourke](http://local.wasp.uwa.edu.au/%7Epbourke/fractals/)

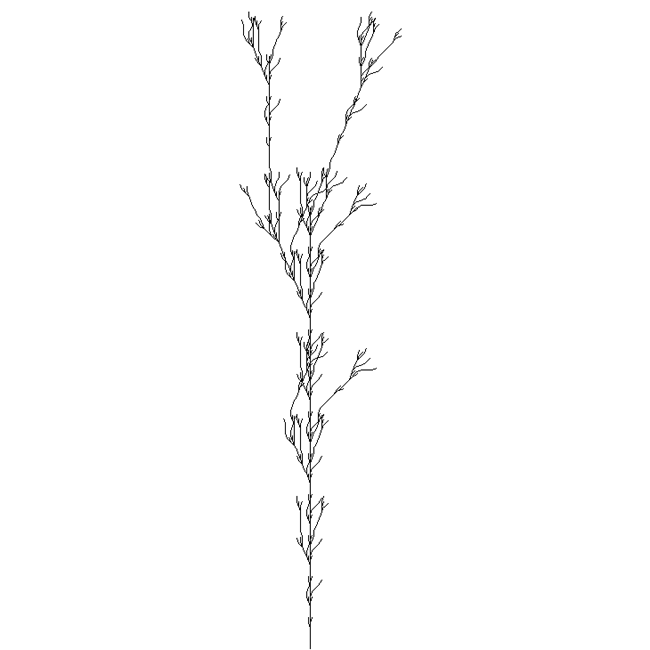
axiom = aF  
a -> FFFFFy[++++n][----t]fb  
b -> +FFFFFy[++++n][----t]fc  
c -> FFFFFy[++++n][----t]fd  
d -> -FFFFFy[++++n][----t]fe  
e -> FFFFFy[++++n][----t]fg  
g -> FFFFFy[+++fa]fh  
h -> FFFFFy[++++n][----t]fi  
i -> +FFFFFy[++++n][----t]fj  
j -> FFFFFy[++++n][----t]fk  
k -> -FFFFFy[++++n][----t]fl  
l -> FFFFFy[++++n][----t]fm  
m -> FFFFFy[---fa]fa  
n -> ofFFF  
o -> fFFFp  
p -> fFFF[-s]q  
q -> fFFF[-s]r  
r -> fFFF[-s]  
s -> fFfF  
t -> ufFFF  
u -> fFFFv  
v -> fFFF[+s]w  
w -> fFFF[+s]x  
x -> fFFF[+s]  
y -> Fy  
angle = 12



**L-System Weed**

Written by [Paul Bourke](http://local.wasp.uwa.edu.au/%7Epbourke/fractals/)

axiom = F  
F -> FF-[XY]+[XY]  
X -> +FY  
Y -> -FX  
angle = 22.5



|  |  |
| --- | --- |
| **Computer Sketching**  Written by [Paul Bourke](http://local.wasp.uwa.edu.au/%7Epbourke/fractals/) June 1990 | http://local.wasp.uwa.edu.au/%7Epbourke/fractals/sketch/sketchsmall.gif |

Computer based design software and hardware has typically been tailored towards producing sharp, precise drawings and images. Can the same technology also generate more imprecise graphical output such as a rough conceptual sketch as created by a human designer?

Two characteristics of a human sketch which do not occur in computer printed output from CAD packages are

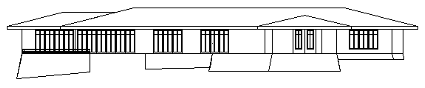
* The endpoints of lines tend to overshoot at an intersection
* Lines are not exactly straight but "wiggle"

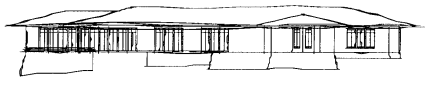
There are three ways these effects might be achieved. The first is to create or modify an output device so that it can replicate the attributes of a human drawer. There are possibilities here for robot arms which can easily be designed to hold a variety of drawing implements and programmed to draw in specified styles. A somewhat unusual attempt at computer sketching is to attach a pen to a plotter loosely with elastic and tape, the result being that the pen wobbles around during a plot giving the desired imprecise appearance.

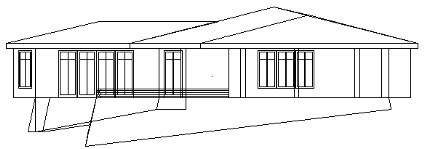
A second technique is to modify the drawing instructions on their way from the CAD software to the output device. For example the vectors could be intercepted and changed to give the appearance of a human sketch before being forwarded to the plotter.

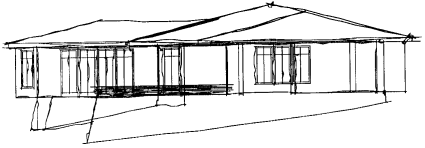
The approach used here is to modify the geometry of a 3D model itself. Both of the effects mentioned above are quite easy to simulate by computer using random and fractal methods. Non-intersecting lines are achieved by extending both ends of line segments by a random distance. The distance should be related to the line length, the application I developed allowed a range of overshoots from between zero and a user chosen maximum. The wiggly line effect was generated by a well known technique from fractal geometry called "random midpoint displacement". It takes a straight line segment and generates a fractal curve by repeatedly splitting lines and perturbing their midpoint, the degree of wiggle is easily controlled by the user.

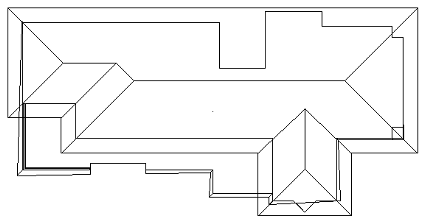
Convincing sketches can be created with these techniques. It should be noted that the result is a "3D sketch" from which hardcopy can be created from any view and projection. The following examples are LaserWriter prints showing a number of views of one of these 3D sketches with a selection of overshoot and wiggle parameters.

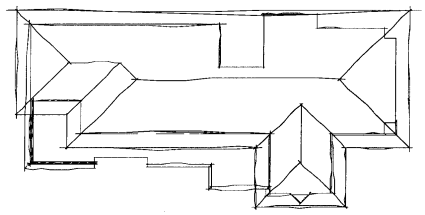
  
House, front view

  
House, front view as a sketch

  
House, side view

  
House, side view as a sketch

  
House, top view

  
House, top view as a sketch

Another noticeable effect of hand drawings is that the intensity is not constant along a line and also varies within a drawing. Typically a line is darker at the beginning and lighter near the end, lines also tend to be darker for foreground items or parts of the drawing of particular interest.. While it is not possible to maintain fine lines and introduce intensity on most printers, it is possible to simulate this effect if a plotter is being used to create the drawing. Plotters that fully support HPGL can have the pen speed, acceleration, and force set for any line segment. These pen properties will not have the desired effect with most plotters pens but will if pencils are used.