2002 서울대학교 과학 영재 센터

정보분과

순환교육 교재 II.

차례

학습 목표

리컬젼 명령을 도입하여 주어진 리컬젼 그림을 리컬젼 명령으로 분석한다. 이를 바탕으로 코흐와 시어핀스키 삼각형, 그리고 고사리 등의 프랙탈 그림의 알고리즘을 이해하고, 이를 통해 무한, 극한 등의 해석학적 개념을 학습한다.

I. 자기 닮음과 프랙탈

자연에서 일어나는 현상들은 각자 독립적이고 불규칙적으로 보이지만 대부분의 현상은 기존에 존재하는 것에서 발전되고 변천되어 나타난다. 예를 들어 구름의 모양, 대기의 난류, 나뭇잎의 잎맥, 해안선 등 언뜻 생각나는 자연 현상의 갖가지 모양들은 모두 일부분의 형상이 이웃한 지점 또는 다른 지점의 형상에 영향을 준다는 것을 알 수 있게 해 준다.

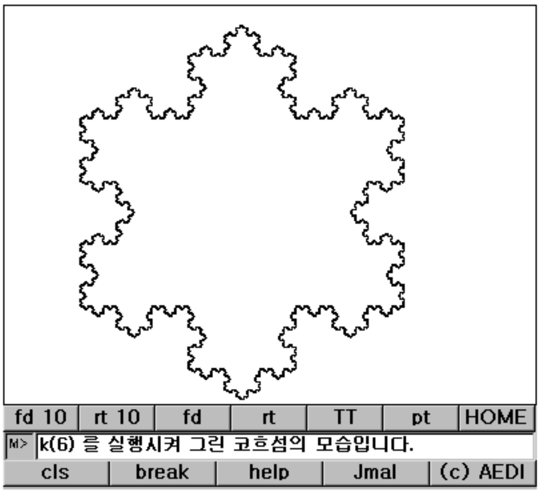
이와 같은 전체와 부분사이의 통일성을 우리는 알고리즘으로 설명할 수 있는데, 여기에서는 이미 증식되어 자리잡은 세포들이 그 반복의 재료이며 세포의 유전 정보가 생성의 알고리즘이라 할 수 있다.

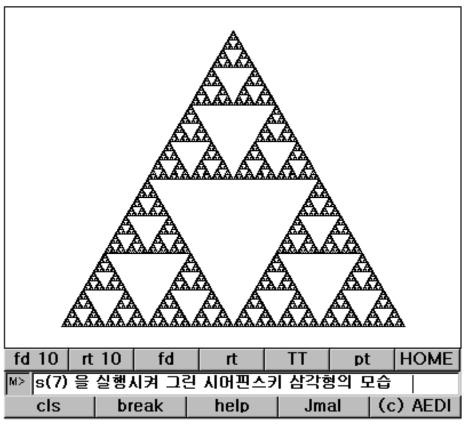
수학에서도 전체와 부분들 사이에 수학적으로 닮음의 있는 대상을 연구한다.

특별히 전체와 부분들 사이의 닮음을 자기닮음(self-similarity)이라 하며, 그림의 한 부분을 확대한 모습이 전체의 그림과 닮은 꼴인 그림을 간단히   프랙탈 (fractal) 그림이라 부른다.

Fractal 이라는 용어는 멘델브로트가 사물이 아주 잘게 "쪼개져 있는 (fractured)" 것을 묘사하는 용어로 만들어 냈는데, 그는 fractal 을 '파괴되어 있다.'는 뜻의 프랑스어에서 따왔다고 한다. 즉, 프랙탈의 사전적인 의미는 '조각'으로, 전체가 전체의 모양과 닮은 부분들로 조각나 있다는 것이다.

직관적으로 프랙탈 그림이란 전체와 부분이 서로 닮은 기하학적 그림이고, 수학적으로 말하면 프랙탈은 자기닮음성과 프랙탈 차원을 가진 그림이 된다.

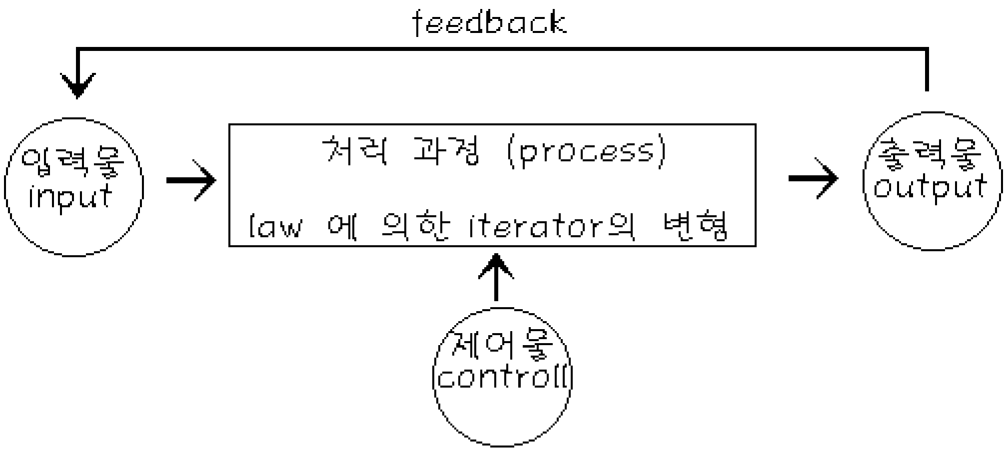




프랙탈은 카오스 (chaos) 와 친척 관계이며, 두 현상 모두 컴퓨터 실험을 통해 탐구할 수 있다.

* 에듀넷의 http://javamath.web.edunet4u.net 주소의 고등수학 학습방에 있는 인터넷과 컴퓨터를 통한 프랙탈 수학과 카오스 수학의 탐구 실험실을 통해 직접 그림을 그려보며 탐구해 볼 수 있다.

특별히 프랙탈에 대한 알고리즘을 탐구할 때에는 iterator(반복되는 것), feedback(출력물의 일부를 입력물로 되돌리는 것), law(규칙성)을 잘 파악해야 한다. 즉 iterator 를 적절한 law 에 의한 변형을 거쳐 feedback 작용을 하게 할 수 있다. 자세한 예는 다음 단원인 프랙탈과 L- system에서 자세히 다루도록 하자.



Ⅱ. 프랙탈과 L-system

자연의 모습에서 전체와 부분이 비슷하게 닮아 있는 것은 볼 수 있지만, 부분과 전체가 완전히 닮은 경우는 거의 없다. 그러나 지금 다루게 되는 Cantor set (칸토르 집합)이나 Koch 곡선 등은 전체와 부분의 완전한 닮아 있음을 알 수 있다. 따라서 이러한 모양은 일부분을 iterator 로 하는 알고리즘으로 전체를 만들어낼 수 있다.

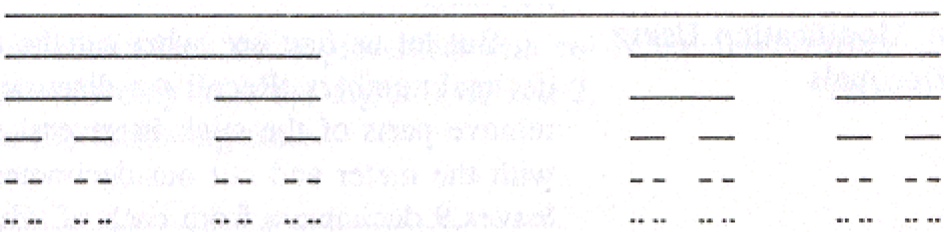
L-system은 생물학자인 린덴마이어(Lindenmayer)가 만들어낸 것으로 원래는 식물의 성장에 대한 설명을 위해 고안된 것이다. 그러나 이것은 컴퓨터 알고리즘에도 많이 사용된다. L-system 에서는 초기 단계를 axiom 으로, 규칙을 production rule 로 부른다.

실제로 L-system 에 의해 다음의 프랙탈 모양을 구성하는 방법을 생각해 보자. 다음을 기본으로 간단하게 알고리즘을 생각해 볼 수 있다.

F : 선을 그으면서 앞으로 이동하기
f : 선을 남기지 않고 이동하기
+ : 정해진 각도 만큼 왼쪽으로 돌기
- : 정해진 각도 만큼 오른쪽으로 돌기

· Cantor set

-Cantor set은 다음과 같이 생긴 모양이다. 단계별로 변하는 모양을 살펴 보면 전체 길이에서 가운데에 해당하는 1/3을 계속 제거해 나감으로써 구성할 수 있다는 것을 알수 있다.



- 전 단계에서 제거 되었던 부분은 다음단계에서도 유지(f→f)
- 전 단계에서 선분이었던 부분은 가운데의 1/3을 제거(F→FfF)

즉, F→FfF→FfFfFfF→FfFfFfFfFfFfF

그런데 이것은 그림에서 보는 바와 같이 전 단계에서 제거되었던 부분의 길이가 다음 단계에서 동일하게 유지되지 않는다. 따라서 제거되는 부분의 길이가 유지되도록 명령을 바꾸어줄 필요가 있다. 이것은 각 단계에서 F 가 길이게 구분이 없이 사용되었기 때문에 나타나는 현상이다. 따라서 F를 각 단계에서 다음과 같이 구분해 준다.

즉, F→FfF→FfFfffFfF→FfFfffFfFfffffffffFfFfffFfF
이를 L-system 으로 보면

초기 단계 : F
규칙 : F→FfF, f→fff

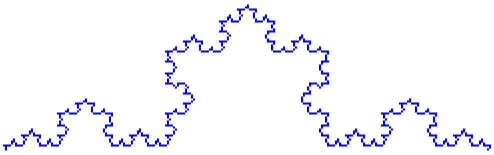
이를 거북명령에서 구현시키기 위해서 다음과 같이 수정해 쓸 수 있다.

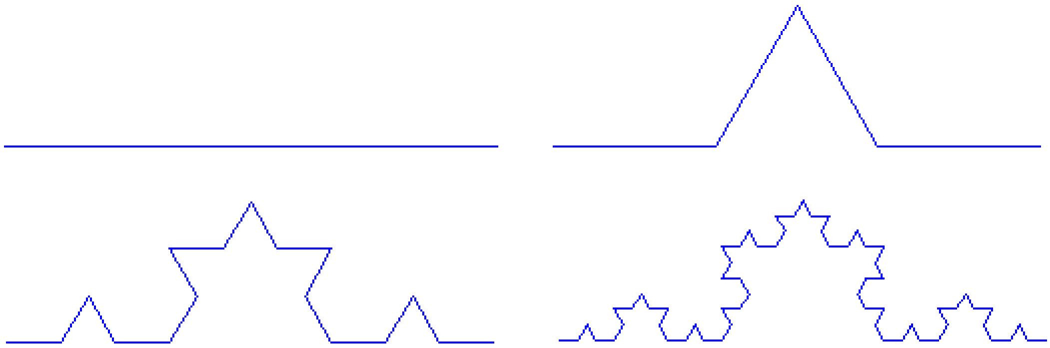
초기단계 : F(0)
규칙 : F(m)→F(m+1)f(m+1)F(m+1),f(m)→f(m)
(단, F(m+1) 의 길이는 F(m) 의 1/3, f(m) 의 길이는 F(m) 과 동일)
즉, F(0)→F(1)f(1)F(1)→F(2)f(2)F(2)f(1)F(2)f(2)F(2)→
F(3)f(3)F(3)f(2)F(3)f(3)F(3)f(1)F(3)f(3)F(3)f(2)F(3)f(3)F(3)

이를 바탕으로 거북이에게 명령을 하는 알고리즘을 생각해 보자.

· Koch 곡선

- 다음과 같이 전체 길이의 1/3 지점에 해당하는 부분에서 그 1/3 길이를 한변으로 하는 정삼각형을 구성하고 밑변에 해당하는 부분은 제거함으로써 구성되는 모양이다. 특히 2단계의 그림을 살펴보면 다음과 같다.





1단계의 그림을 1/3로 축소하여 그린다. 오른쪽으로 60도 회전한다.
1단계의 그림을 1/3로 축소하여 그린다. 왼쪽으로 120도 회전한다.
1단계의 그림을 1/3로 축소하여 그린다. 오른쪽으로 60도 회전한다.
1단계의 그림을 1/3로 축소하여 그린다.

이상에서 알 수 있듯이

n 단계의 그림은 기본적으로 n-1 단계의 그림을 1/3로 축소하여 그리기를 네 번 반복한다.
한번에 이동하는 길이는 전단계의 1/3로 줄어들고
전 단계에서 돌았던 각은 그대로 유지한다.

즉, F→F+F--F+F→F+F--F+F+F+F--F+F--F+F--F+F+F+F--F+F
여기서의 +와 -에서 정해진 각의 크기는 60도 이다.

이를 L system 으로 보면

초기 단계 : F
규칙 : F→F+F--F+F, +→+, -→-
여기서도 앞의 문제에서와 같이 길이 구분을 위해서는 다음과 같이 쓸 수 있다.

즉, F(0)→F(1)+F(1)--F(1)+F(1)→F(2)+F(2)--F(2)+F(2)+F(2)+F(2)--F(2)+F(2)--F(2)+F(2)--F(2)+F(2)+F(2)+F(2)--F(2)+F(2)

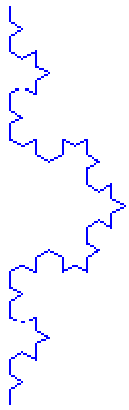
이를 L - system 으로 보면

초기 단계 : F(0)

규칙 : F(m)→F(m+1)+F(m+1)--F(m+1)+F(m+1), + → +, - → -

다음은 LOGO 로 Koch 곡선을 그리는 명령이다.

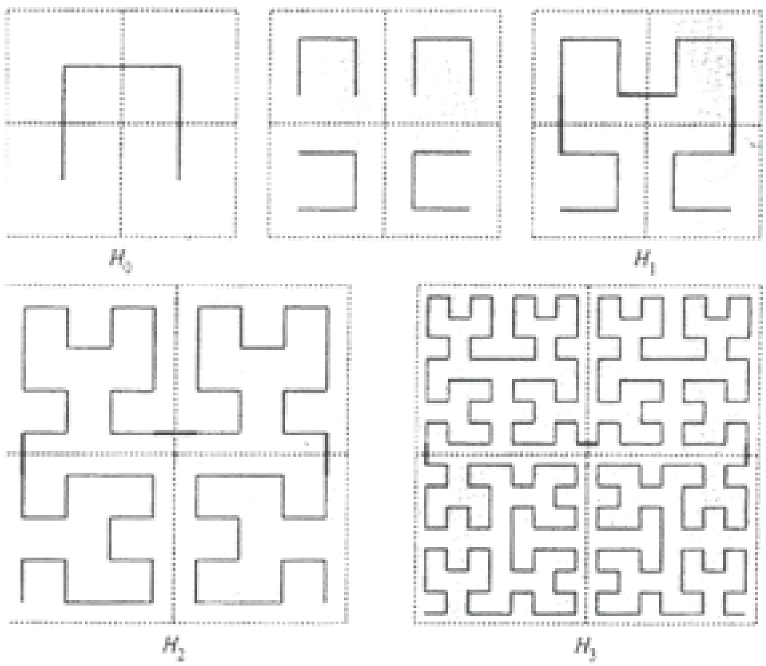
```
def L(n)

gaja_0=100;
gaja=gaja_0;

for i=1 to n;
gaja_i=gaja*1/3;
gaja=gaja_i;
next;

dolja=30;
stack=0;
tt 0,-60,90;
F(n);



def F(m)
if(m==0)  fd gaja;
else

F(m-1);
rt -60;
F(m-1);
rt 120;
F(m-1);
rt -60;
F(m-1);

 L(3)
```



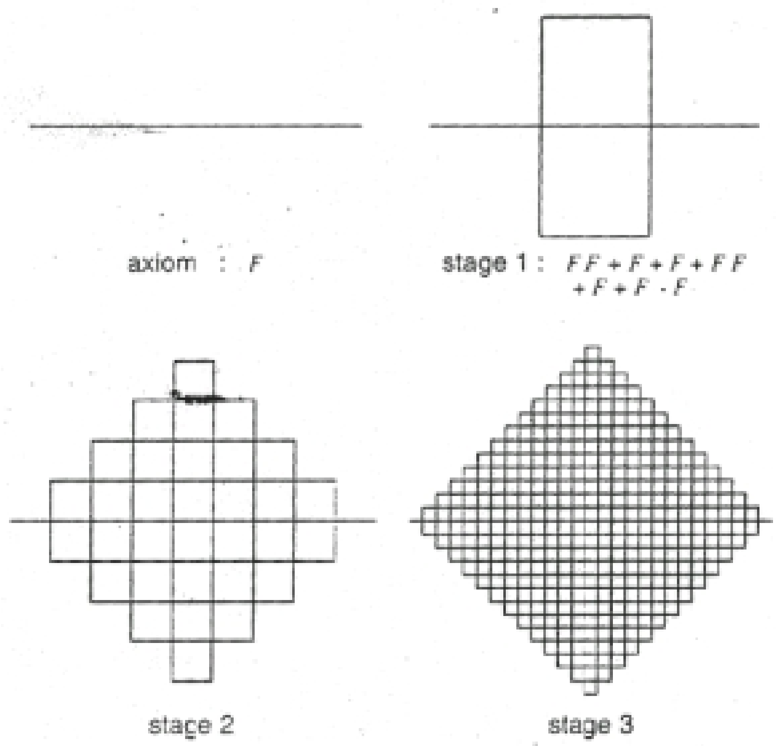
(문제)

다음 그림에서 각각 (a) 는 Peano 곡선, (b) 는 Hilbert 곡선이라 불리는 프랙탈이다. 다음의 모양을 구성하는 L-system을 생각해 보자.

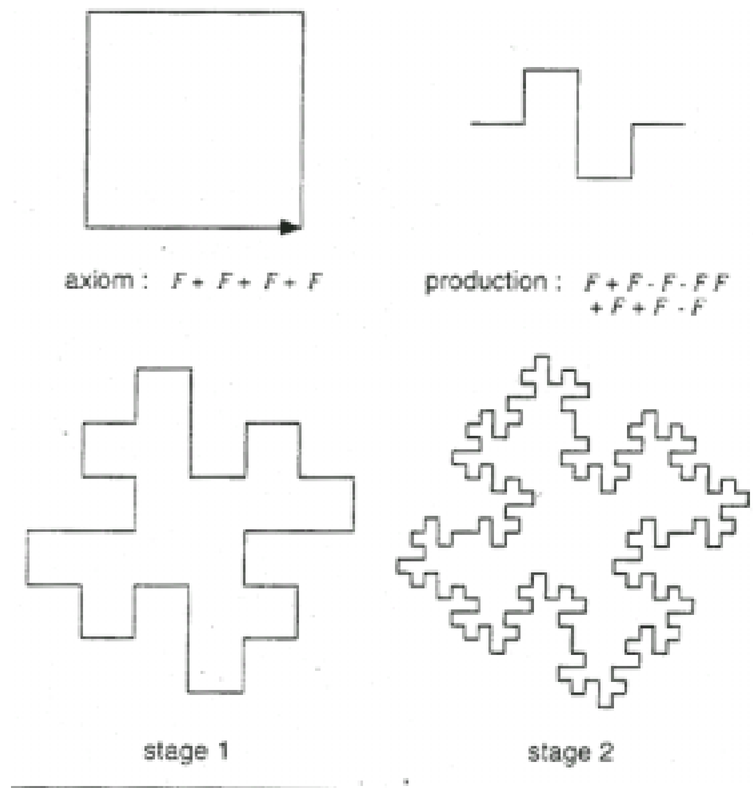
(a)



(b)


axiom : F
stage 1 : FF + F + F + FF
+ F + F - F
stage 2
stage 3

(c)

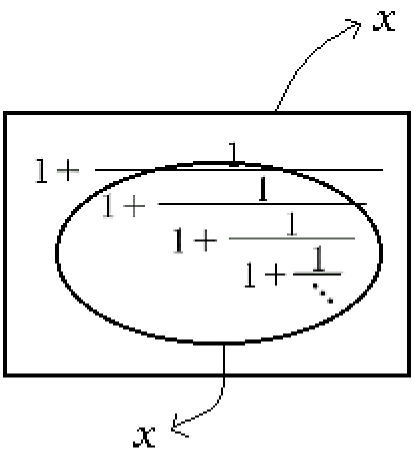

axiom : F + F + F + F
production : F + F - F - FF
+ F + F - F
stage 1
stage 2

Ⅲ. 수식에서의 프랙탈

수학에서는 이러한 자기닮음의 모습을 그림이 아닌 것에서도 볼 수 있다. 다음의 예를 생각해 보자.

$$1+\cfrac{1}{1+\cfrac{1}{1+\cfrac{1}{1+\cfrac{1}{\ddots}}}}$$ 의 값은 얼마인가?

여기서는 특정한 모양의 수식이 계속 반복되고 있다는 것을 알 수 있다. 즉, 무한히 계속되는 분수꼴의 계산값을 x 라고 하자. 그러면 다음과 같이 생각할 수 있다.



분수꼴에서 분모에 해당하는 것이 본래의 것과 완전히 닮았기 때문에 $x=1+\dfrac{1}{x}$ 과 같은 관계식을 얻을 수 있다.

또 다른 예를 생각 해 보자.

무한순환소수 0.33333… 의 값을 생각해 보자.

$$0.33333\cdots = 0.3 + 0.03333\cdots$$
$$0.03333\cdots = 0.1 \times 0.33333\cdots$$

와 같은 관계가 있으므로 0.33333…을 분수꼴로 나타낸 것을 a 라고 할 때, $a=0.3+0.1a$ 로 나타낼 나타낼 수 있다.

$1+2+2^2+2^3+2^4\cdots$ 의 값은 무엇인가?
$1+5+5^2+5^3+5^4\cdots$ 의 값은 무엇인가?
그러면 일반적으로 $1+r+r^2+r^3+r^4\cdots$ 의 값은 어떻게 될까?
이문제도 위의 문제와 유사하다. 이식을 잘 살펴보면 계속 r만큼씩 곱한 것을

합하고 있다. 따라서 다음과 같이 바꾸어 생각해 볼 수 있다.

$1+r+r^2+r^3+r^4\cdots = 1+r(1+r+r^2+r^3\cdots)$

$$\underbrace{\boxed{1+r+r^2+r^3+r^4\cdots}}_{S} = 1+r\underbrace{\boxed{(1+r+r^2+r^3\cdots)}}_{S}$$

$1+r+r^2+r^3+r^4\cdots$ 의 값을 s라고 할 때, s가 $1+r(1+r+r^2+r^3+\cdots)$ 이므로 $s=1+rs$ 라는 관계식을
얻는다. 위의 예 모두에서 전체와 부분이 서로 닮았음을 이용하여 계산값을 구할 수 있다.

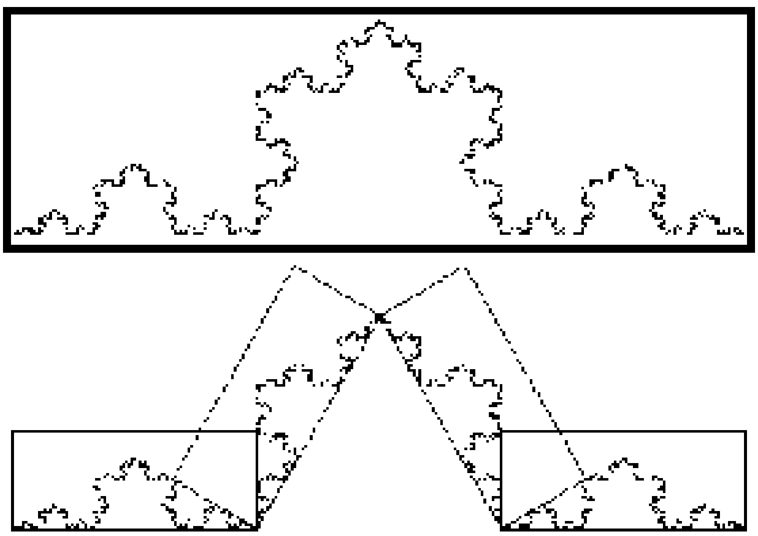
(문제) 제곱해서 a $(a\geq 0)$ 가 되는 수를 제곱근 a 또는 $\sqrt{a}$ 라고 한다. 즉, $\sqrt{a}$를 제곱하면 그 결과는 a가 된다.
프랙탈의 자기 닮음을 이용하여 $\sqrt{2\sqrt{2\sqrt{2\sqrt{2\sqrt{\cdots}}}}}$ 의 값을 구해보자.

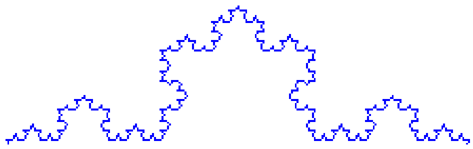
Ⅳ. 프랙탈 모양 탐구

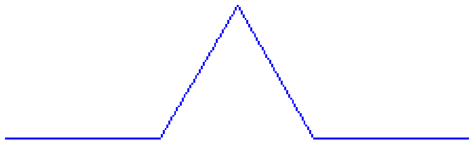
지금까지 자기 닮음의 특성을 가진 프랙탈에 대해서 기본적으로 살펴 보았다.
여기서는 구체적으로 각각의 여러 가지 프랙탈 모양과 프랙탈 모양이 가지고 있는 특성에 대해 생각해보자.

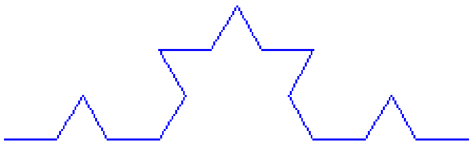
· Koch 곡선

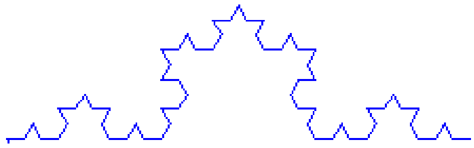
이제 다음과 같이 전체의 모습과 작은 사각형 안의 모습이 닮음비 3의 관계가 있는 코흐 (Koch) 곡선을 살펴 보자.
우선 이 곡선의 전체 길이가 어떻다고 말할 수 있을까?
전체 길이를 x 라고 할 때, 작은 사각형 안에 그려진 부분의 길이는 닮음비가 3 이므로 $\frac{x}{3}$ 가 된다. 따라서 $x=\frac{4x}{3}$ 이라는 식을 얻는다. 이 식을 통해 전체의 길이가 어떻다고 얘기할 수 있을까?
다음으로, 큰 사각형과 코흐 곡선 사이에 있는 부분의 면적을 계산하여 보자. 이 면적을 y 라고 하고 작은 사각형의 밑변의 길이를 a 라고 할 때, 작은 사각형과 코흐 곡선 사이의 면적은 $\frac{y}{9}$ 가 된다. 이제 큰 사각형 안에서 그려낼 수 있는 삼각형의 면적을 고려하면 $y=4(\frac{y}{9})+\sqrt{3}(\frac{a}{2})^2$
라는 식을 얻는다. 이 식에서 면적을 구할 수 있다. 이러한 계산을 통해 둘레의 길이는 무한이면서 내부의 면적은 유한인 예를 보게 된다.



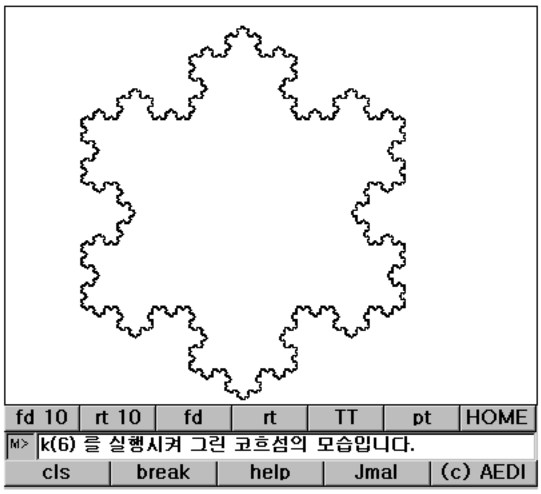






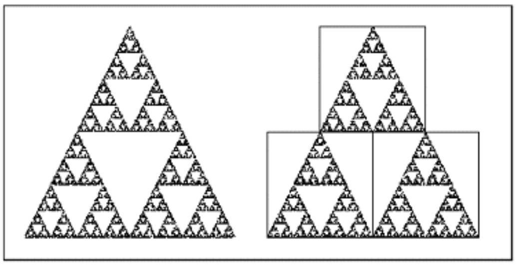


이 Koch 곡선을 이어 붙여서 다음과 같은 Koch 의 섬을 만들 수 있다. 앞의 알고리즘을 이용해 Koch 의 섬을 만드는 알고리즘을 생각해 보자.



· Sierpinski 삼각형

Sierpinski 삼각형은 정삼각형을 사등분하여 가운데 부분을 없애는 작업을 반복해서 만들어지는 모양이다.다음은 전체와 부분이 수학적으로 닮아있는 시어핀스키 삼각형의 모습을 그린 것이다. 작은 사각형 안의 모습과 전체의 모습 사이의 닮음비가 2이다.



이때, 시어핀스키 삼각형의 면적을 얼마일까? 시어핀스키 삼각형의 면적을 x 라고 할 때, 작은 사각형 안에 그려진 부분의 면적은 $\frac{x}{4}$ 가 된다. 작은 사각형 안의 그림이 세 개 모여서 전체를 이루므로 $x = \frac{3x}{4}$ 가 되는데, 이 식을 만족하는 값은 0 이다. 따라서 시어핀스키 삼각형의 면적은 0 이 된다. Koch 곡선에서 보았듯이 이 경우도 모양을 구성하는 길이는 무한대이지만 넓이는 0이 된다. 이러한 특성을 가진 모양들을 수학에서는 수학적 괴물(monster)이라고 부른다.

· Pythagoras 의 나무

다음과 같은 성질을 가지는 함수 p(x)를 생각하자.

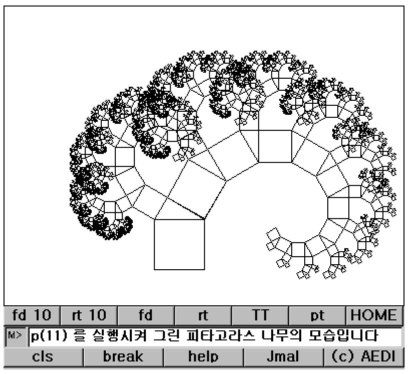
p(0) : 한 변의 길이가 30 인 정사각형에 대응된다.

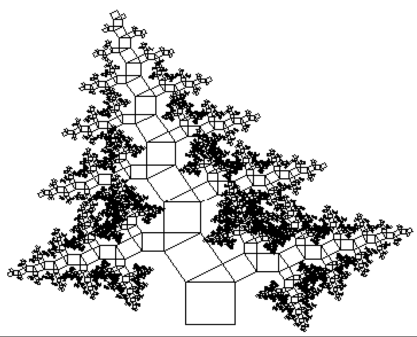
p(1) : p(0) 의 정사각형을 밑으로 하는, 피타고라스 정리에서 나오는 그림에 대응된다.

p(2) : p(1)에서 새로 그려진 정사각형들을 각각 밑으로 해서, 피타고라스 정리에 나오는 그림을 그린 것에 대응된다.

p(n) : p(n-1)에서 새로 그려진 정사각형을 각각 밑으로 해서, 새롭게 정사각형을 추가하는 것이다.

이렇게 계속 그려나가면 어떤 그림이 그려지는가? 왼쪽과 같은 모습으로 가는데, 이러한 모습의 그림을 피타고라스 나무라고 부른다. 여기에 적당한 변형을 가하면 다음 오른쪽과 같은 그림도 얻을 수 있다.

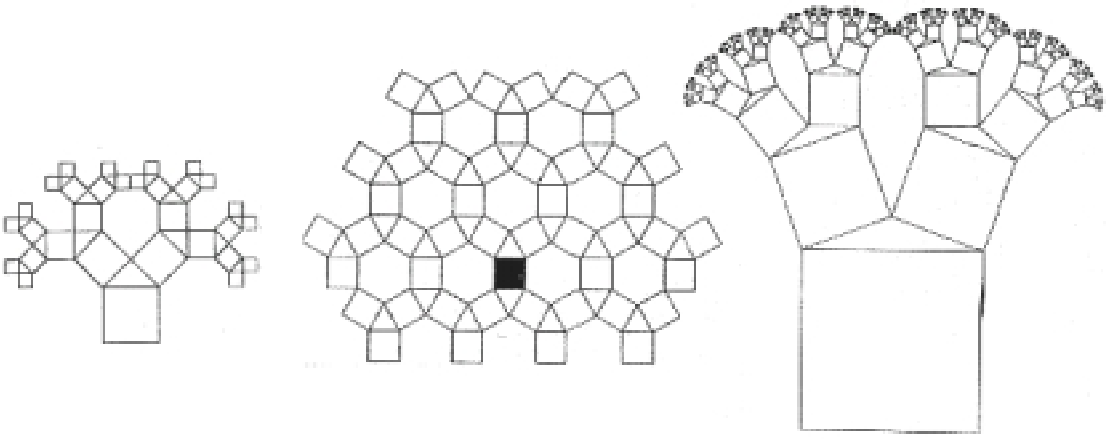




피타고라스 정리를 이용한 피타고라스 나무의 모습

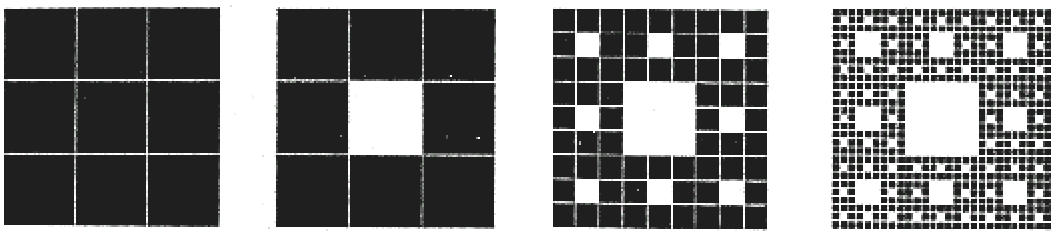
위의 그림들은 피타고라스 정리에 나오는 그림 또는 조금 변형된 그림을 반복의 재

료로 삼아 피타고라스 정리에 의한 알고리즘으로 그려진 나무 모양의 그림이다. 여기에 전체의 통일감을 잃지 않을 정도의 자유도를 줄 수 있다면 실제의 자연의 형상과 수학적으로 그려진 그림의 차이가 점점 없어질 것이다. 이와 같이 엉뚱한 재료를 단순 반복하여 얻은 그림이 자연의 복잡한 형상과 비슷하다는 사실에서 우리는 자연의 복잡성을 새롭게 보는 시각을 얻게 된다.



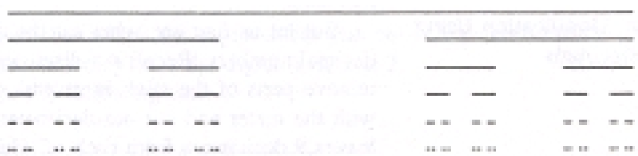
위의 피타고라스 나무를 잘 살펴보면 전체와 부분 사이에 서로 비슷한 모습을 발견할 수 있다. 뿐만 아니라 다음에 제시된 나무와 고사리 모양의 그림에서 조그만 사각형 안의 그림이 전체의 그림과 비슷함을 볼 수 있다.

(문제)
 다음의 그림은 시어핀스키 삼각형을 변형시킨 모양이다. 이 도형의 넓이는 얼마일까?



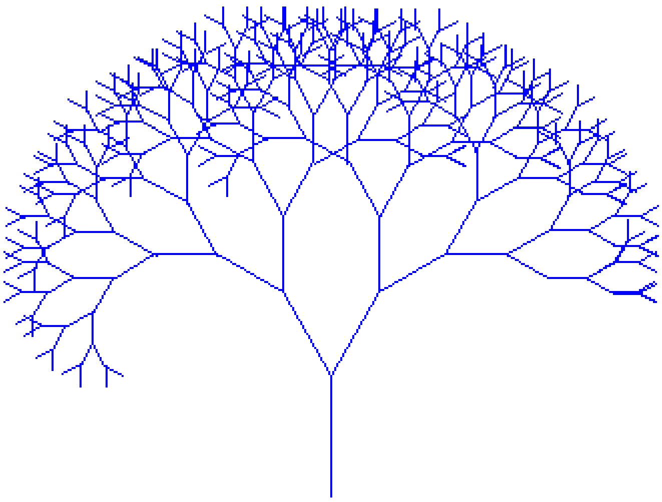
· Cantor set

다음과 같은 Cantor set을 생각해 보자. 이 단계를 계속 반복해 나갈 때 제거되는 부분의 길이는 어떻게 될까? 또 남아 있는 부분의 길이는 어떻게 될까?



· 나뭇가지

다음은 거북이가 프랙탈 나무를 그리는 모습이다.
특별히 게시판에 올려진 **L-system 나뭇가지 그리기**에서 L(8)을 입력한 경우의 그림이다. 이 경우 나뭇가지 끝의 개수와 나무에 나타나는 꺾인 점의 개수는 어떻게 될까?
n 번째 단계에서 생기는 나뭇가지 끝의 개수는 어떻게 될까? 그리고 n 번째에서의 나무에 나타나는 꺾인 점의 개수는 어떻게 될까?



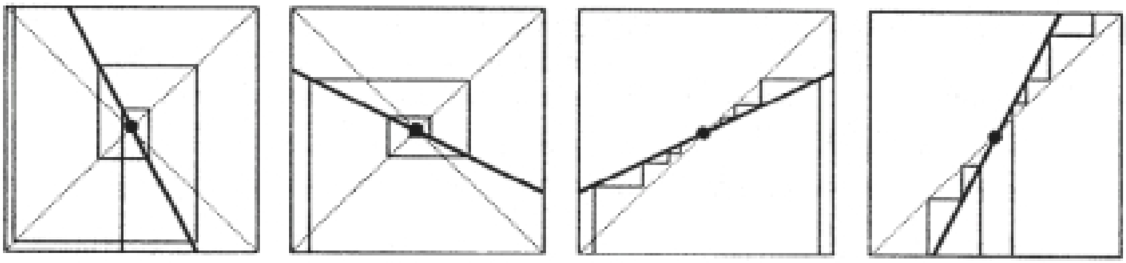
이 그림의 명령은 다음과 같다.

```
def L(n) {

gaja_0=30;
gaja=gaja_0;

for i=1 to n;
gaja_i=gaja*0.8;
gaja=gaja_i;
xx_i=10;
yy_i=10;
dd_i=10;
next;
dolja=30;
stack=0;
tt 0,-60,90;

F(n);
}
```

```
def F(m) {
if(m==0)  {fd gaja_stack;}
else {
fd gaja_stack;
stack=stack+1;print stack;
xx_stack=xc(); yy_stack=yc();dd_stack=dc();
rt dolja; F(m-1); tt xx_stack, yy_stack,dd_stack;

rt -dolja; F(m-1); tt xx_stack, yy_stack,dd_stack;
stack=stack-1;
}
}

L(3);
```

Ⅴ. 오비트 수열과 카오스 수학

다음의 그림들을 살펴보자.

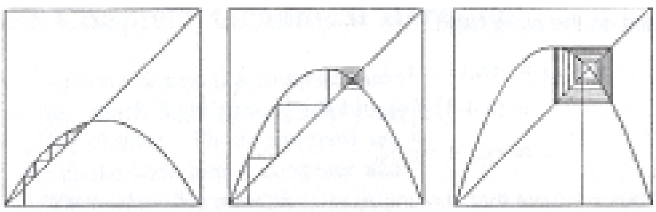


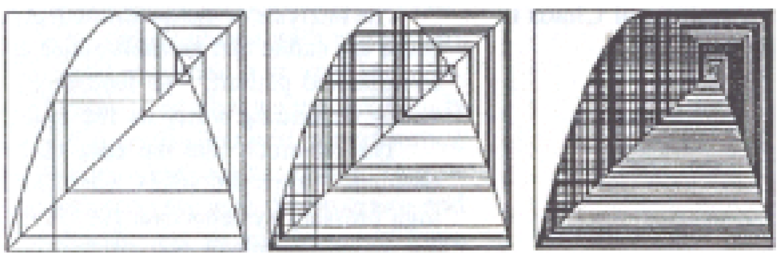
첫 번째와 세번째의 경우에는 직선의 점점 ∞,-∞로 가고 있는 것을 볼 수 있다. 반면에 두 번째와 네번째의 경우에는 점점 가운데의 점으로 접근해 오는 것을 볼 수 있다. 두가지의 차이점은 무엇일까?

위와 같이 주어진 함수 f(x)에 대해 x_0 에서의 오비트(orbit)란 다음과 같이 함수 f(x)의 합성에 의해 만들어지는 수열 x_0, x_1, x_2, ...을 말한다. 즉, 함수 f(x) 에 대해 출발점을 x_0 라고 할 때,

$$x_1=f(x_0),\quad x_2=f(x_1x1)=f(f(x_0))=f^2(x_0),\quad x_3=f(f(f(x_0)))=f^3(x_0)\ \ldots\ldots..$$

이렇게 초기값 x_0로 부터 함수의 합성에 의해 만들어지는 수열은 주어진 함수 f(x) 와 초기값 x_0 에 따라 다양한 특징을 갖는다.





위의 그림 같은 경우를 '안정적이다(stable)' 라고 말하고
아래 그림 같은 경우를 '불안정적이다.(unstable)' 라고 말한다.

이것은 위의 그림 같은 일차함수에서만 볼 수 있는 현상은 아니다.

다음의 함수에 대해 생각해 보자.

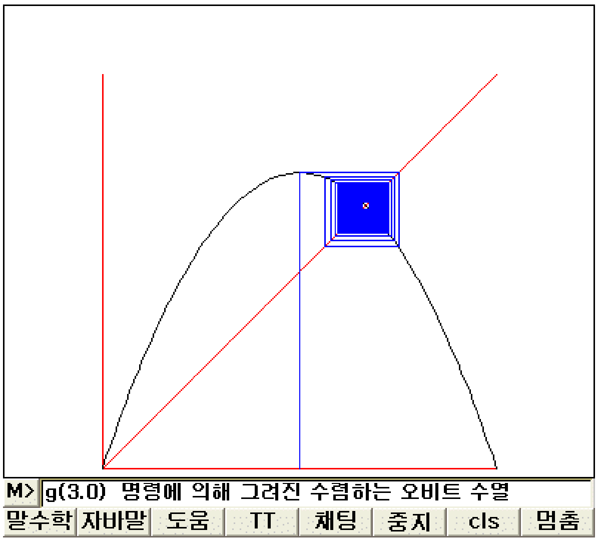
$f(x)=x(1-x)$
1. 초기값(x_0)이 0.3 일 때 어떻게 되는가?
2. 초기값(x_0)이 0.5 일 때와 0.7 일 때에는 각각 어떻게 되는가?
   0.3일때와 비교해 보자.
3. 초기값(x_0)이 -0.5 일때와 1.5 일 때에는 각각 어떻게 되는가? 이러한 현상이 나타나는 이유는 무엇인가?

$f(x)=3x(1-x)$
1. 초기값(x_0)이 0.5 일때와 0.25,0.75 일 때를 각각 비교해 보자.
   이러한 특성이 나타나는 이유를 생각해 보자.

일반적으로 $f(x)=ax(1-x)$ 일 때, 초기값(x_0)에 따른 현상의 변화에 대해서 설명할 수 있겠는가?

이번에는 초기값을 0.5로 고정하고 $f(x)=ax(1-x)$ 의 a의 값을 차례로 변화시킬 때 생기는 오비트 수열의 특징에 대해 알아보자.

$f(x)=ax(1-x)$ 의 a의 값을 차례로 변화시킬 때 생기는 오비트 수열의 특징에 대해 알아보자. 이러한 모양의 함수식은 (0,1)을 지나는 무수히 많은 이차함수들을 표현하게 된다.
 이를 위해 주어진 a의 값에 대응하여 만들어지는 오비트 수열의 모습을 컴퓨터 화면에 그리는 함수 g(a)를 생각해 보자. 다음은 a의 값이 3일 때, g(3) 에 대응하는 오비트 수열 (다시 말해서 $f(x)=3x(1-x)$ 에 대한 오비트 수열)의 모습을 컴퓨터 화면에 그린 것이다.



여기서 g(3) 에 대응하여 컴퓨터 화면에 그려지는 파란 선들은 y=x 인 초록색 직선과 관계하여 다음과 같은 방법으로 그려진다.
1. $(x_0,0)$ 즉 (0.5, 0) 을 출발점으로 하여 (x_0, x_1) 까지 파란선을 긋는다.
2. (x_0, x_1) 에서 (x_1, x_1) 으로 파란선을 긋는다.
3. 다음 (x_1, x_1) 에서 (x_1, x_2) 까지 파란선을 긋는다.

 즉, n번째의 (x_n, x_n) 에서 $(x_n, f(x_n))$ 즉 (x_n, x_{n+1}) 까지 파란선을 긋고, 다음 $(x_n,$

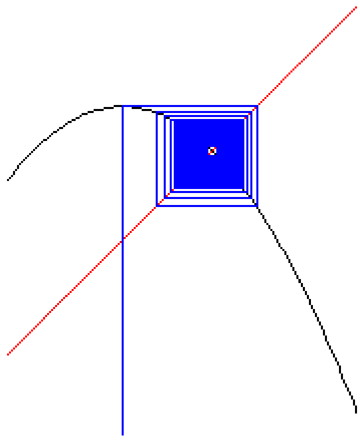
x_{n+1}) 에서 (x_{n+1}, x_{n+1}) 까지 파란선을 긋는다.

a 대신 다른 숫자를 넣으면서 관찰해 보자. 어떤 현상이 일어나는가?

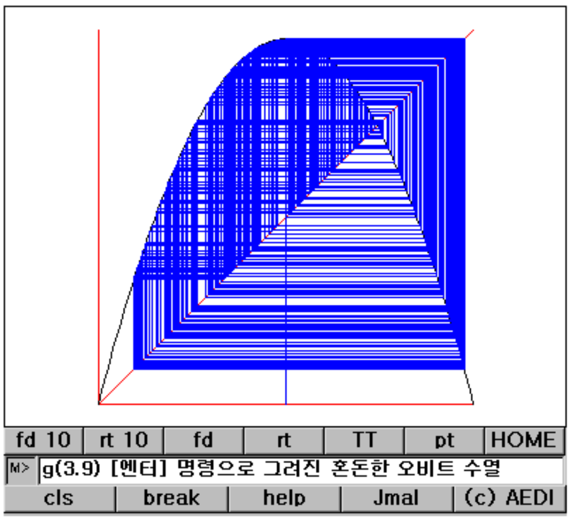
예를 들어, a가 3.1 이라면 g(3.1)이 만드는 수열은 궁극적으로 2개의 값을 반복하여 갖는 2-싸이클을 만든다. 그러다가 a가 3.45 이상으로 커지면 오비트는 궁극적으로 4개의 값을 반복하여 갖는 4-싸이클을 보여준다. a가 3.57이상으로 커지면 오비트는 규칙성 없이 이리 저리로 변화됨을 볼 수 있다 (이것이 카오스 오비트이다). 그러다가 g(3.83)에서 주기가 3인 수열이 잠시 나타나다가 a가 4에 가까이 가면서 또 다시 혼돈의 모습이 나타나게 된다. 이러한 안정과 불안정은 왜 일어나는가? 언제 안정적이고 언제가 불안정적인가?

다음의 명령은 안정적 카오스 모양을 보여준다.

```
def axx(a,s)  {
cls; if(a>-0.0001) {
if(a<4.00001) {
window 0.75 ; tt red; tt ;
x=s; y=0; tt red ;
tt -0.5,-0.5,90 ; fd 1 ;
tt -0.5,-0.5,0 ; fd 1;
tt -0.5,-0.5 ; fd 0.5,0.5 ;
tt -0.5,-0.5; tt black; i=0 ;
while(i<0.999) { fd i-0.5, a*i*(1-i) - 0.5 ; i=i+0.01  ; }
fd i-0.5,-0.5 ; tt blue ;
tt x-0.5,-0.5,90; i=0 ;
while(i<500)  {y = a*x*(1-x); fd x-0.5, y-0.5 ;
x=y; fd x-0.5,x-0.5; i=i+1 }
}
}
}

def g(a) { axx(a,0.5)  ;}
button f { '[중지] 버튼을 누르세요';0,-71;}
g(3.0);
```



다음은 a가 3.9 일 때의 오비트의 모습, 즉 f(x)=(3.9)x(1-x) 에 대한 0.5의 오비트 수열의 항이 얼마 후 혼돈한 모습으로 변하는 것을 보여주는 컴퓨터 화면의 모습이다.



위에서 a의 값이 3.57 주변인 경우에 조그마한 a값의 변화에 의해 오비트 수열이 규칙적인 모습에서 갑자기 혼돈의 모습으로 바뀌는 것을 볼 수 있다. 마치 0도씨의 물에 돌을 던지는 순간 얼음이 얼기 시작하는 것과 같이 조그마한 변화가 질적으로 큰 차이가 있는 변화를 몰고 온다. 수학자들은 이러한 현상을 나비효과라고 부르는데, 나비효과의 예로 홍콩에서 나비 한 마리가 무심코 날개를 팔랑거려 일으킨 조그마한 바람이 미국 뉴욕에 태풍을 일으킬 수도 있음을 얘기한다.

이번에는 함수를 고정시키고 초기값(x_0)을 변화시켜 보자. 초기값의 변화에 따라서 안정, 불안정에는 변화가 없는가?

(문제) 이차함수 f(x)=x^2+c 에 대하여 탐구해보자.

f(x)=x^2 일 경우에 대해 생각해 보자.
1. 초기값(x_0)이 0 일때 어떻게 되는가?
2. 초기값(x_0)이 0.3 일때와 -0.3 일 때에는 각각 어떻게 되는가?
0.3일때와 비교해 보자.
3. 초기값(x_0)이 -0.7 일때와 0.7 일 때에는 각각 어떻게 되는가? 이러한 현상이 나타나는 이유는 무엇인가?

f(x)=x^2+3
위의 경우와 동일하게 생각해 보자. 위의 경우와 비교하여 어떤 결과가 나오는가? 이러한 결과가 나오는 이유는 무엇인가?

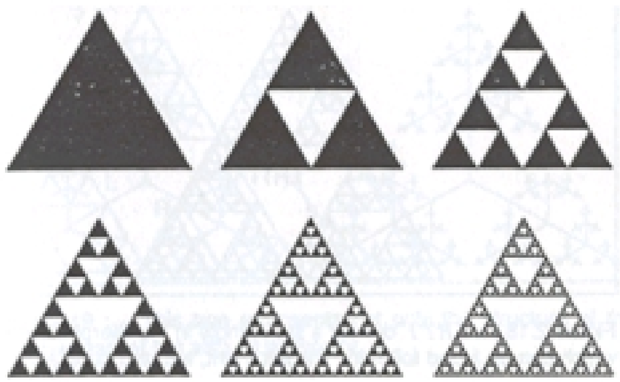
이번에는 초기값(x_0)을 0 으로 잡은 후 절편 c를 변화시킬 때 생기는 오비트 수열에 대해 탐구하여 보자.
먼저 c가 -1.1 일 때에는 어떤 현상이 나타나는가?
이제 c를 -1.76 으로 잡았을 때 나오는 오비트 수열의 특징을 살펴보자.

Ⅵ. 카오스 게임

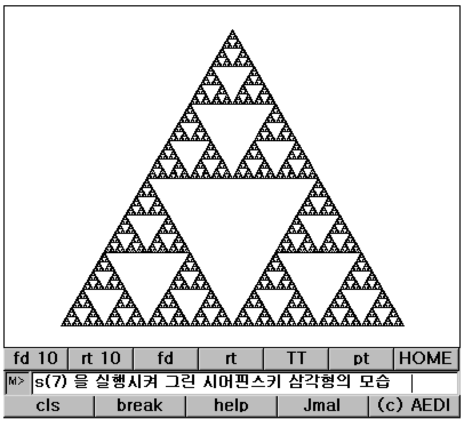
큰 책상위에 정삼각형이 그려져 있고 그 꼭지점 중 하나에 개구리 한 마리가 앉아 있다고 하자. 이제 개구리가 매 초 마다 정삼각형의 세 꼭지점 중에 하나를 마음내키는대로 선택하고, 현재의 위치와 그 꼭지점의 중간 지점으로 점프하며 계속 자리를 옮겨간다고 하자. 만일 개구리의 한 쪽발에 잉크가 발라져있어 개구리가 자리를 옮길 때마다 잉크에 의해 점이 찍힌다면 책상에는 어떤 그림이 점점 그려질까? 그려지는 그림은 불규칙할 것인가, 또 개구리마다 모두 다른 그림을 그리게 될 것인가?

답은 개구리나 모두 학생들이 그리는 그림은 소위 시어핀스키 삼각형이라고 불리우는 그림으로 점점 수렴해 간다는 것이다. 개구리가 삼각형의 꼭지점을 마음대로 택하였는데 어떻게 그런 일이 일어날 수 있을까? 세 꼭지점을 마음대로 택한다고 하지만 현재의 위치와 꼭지점의 중점에 점을 찍기에 수학적으로 점을 찍는 것을 설명할 수 있다.

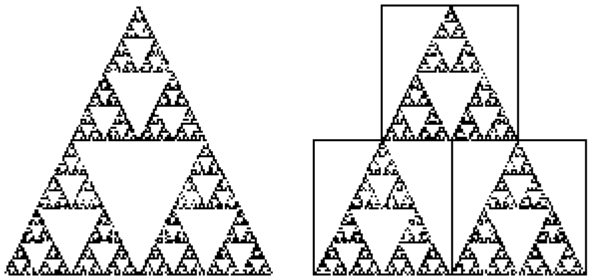
이제 다음과 같은 함수 s(x)를 통해 음이 아닌 정수 x의 값이 변함에 따라 이 함수가 대응시키는 그림이 어떻게 변화하는지 알아보자.



s(0) : 정삼각형
s(1) : 정삼각형을 사등분하여 가운데 부분을 없앤 것이다.
s(2) : s(1)에서 그린 정삼각형을 다시 사등분한 후 가운데 부분을 없앤 것
s(n) : s(n-1)에서 새로 그린 정삼각형을 4등분하여 가운데 부분을 없앤 것.
이렇게 계속 그려나가면 어떤 그림이 그려지는가? 다음과 같은 모습으로 가는데, 이러한 모습의 그림을 시어핀스키 삼각형이라고 부른다.



이 그림의 특징은 다음에서 볼 수 있듯이 왼쪽의 시어핀스키 삼각형을 2분의 1로 축소시킨 그림 3개를 적당한 위치로 이동을 시키면 다음 오른쪽 그림을 얻게 된다.



여기서 사각형안에 들어가 있는 각각의 프랙탈 조각들은 아핀변환 한 개와 대응된다. 여기서 아핀변환이란 다음의 식과 같이 평면상의 좌표 (x,y)를 (x',y')에 대응시키는 함수이다.

$$\begin{pmatrix} x' \\ y' \end{pmatrix} = \begin{pmatrix} a & b \\ c & d \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} + \begin{pmatrix} e \\ f \end{pmatrix}$$

위에 제시된 3개의 프랙탈 조각들은 본래의 시어핀스키 삼각형을 $\frac{1}{2}$ 배 한 후, 위로 옆으로 자리를 이동시켜 얻을 수 있다. 즉, 각각의 프랙탈 조각에서 다음과 같이 3개의 아핀변환을 얻을 수 있다.

W_1 : 원점을 기준으로 하여 원래의 그림을 $\frac{1}{2}$배, 평행이동하지 않는다.

$W_1(0,0)=(0,0)$

W_2 : 원점을 기준으로 하여 원래의 그림을 $\frac{1}{2}$배, 초기 단계 삼각형의 한변의 길이의 $\frac{1}{2}$만큼 x축으로 평행이동

$W_2(0,0)=(0.5,0)$

W_3 : 원점을 기준으로 하여 원래의 그림을 $\frac{1}{2}$배, 초기 단계 삼각형의 한변의 길이의 $\frac{1}{4}$만큼 x축으로 $\frac{1}{2}$만큼 y축으로 평행이동

$W_3(0,0)=(0.25,0.5)$

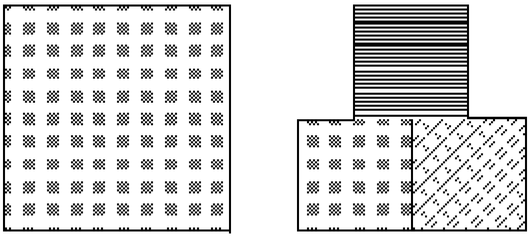
여기서, 이 세개의 아핀변환은 다음 그림 왼쪽에 있는 정사각형 A 를 그 옆에 세개의 사각형으로 이루어진 그림으로 각각 옮겨준다.
이제 위의 꽃가지 그림과 관계되는 3개의 아핀변환 W_1, W_2, W_3 을 살펴보자.

■ : 원점과 좌표 (1,0) 과 (0,1) 과 (1,1) 을 지나 만들어지는 한변의 길이가 1인 정사각형

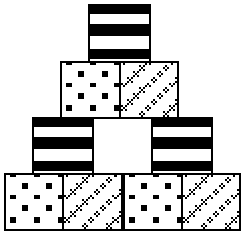
W(1) : W_1(■) 과 W_2(■) 와 W_3(■) 의 변환된 그림들을 모은 것

W(1) = W_1(■) $\bigcup$ W_2(■) $\bigcup$ W_3(■)

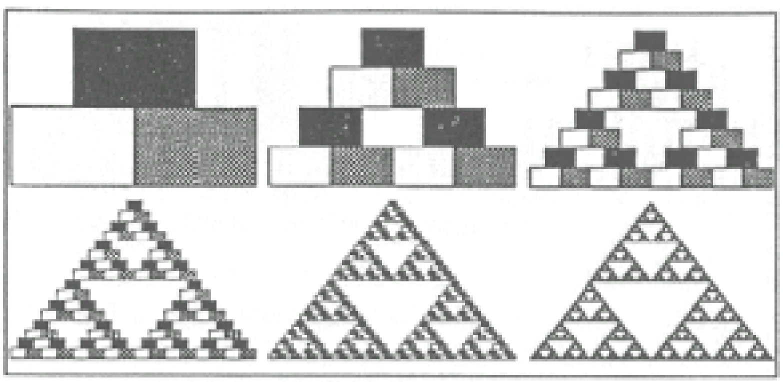
즉, 첫번째 사각형 A 에 대해 W(1) 가 W_1(A)와 W_2(A)와 W_3(A) 의 합으로 대응되는 함수라면, W(1) 는 다음의 오른쪽 그림이 된다.

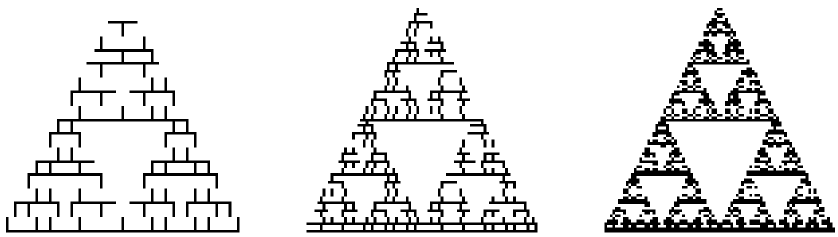


이제 W(2)를 W_1(W(1)) $\bigcup$ W_2(W(1)) $\bigcup$ W_3(W(1)) 이라고 하자. (여기서 W(1)은 좌표 평면 상의 도형이다). 그러면 다음과 같은 그림을 얻을 수 있다.



일반적으로 자연수 n에 대해, W(n) = W_1 (W(n−1)) $\bigcup$ W_2 (W(n−1)) $\bigcup$ W_3 (W(n−1))이라 하자. 이때 W(1), W(2), W(3), ... 의 모습들을 순서대로 무한히 많이 그리게 되면 그 모습이 바로 앞에서 프랙탈 조각으로 나누기 위해 먼저 소개된 시어핀스키 삼각형이 된다.

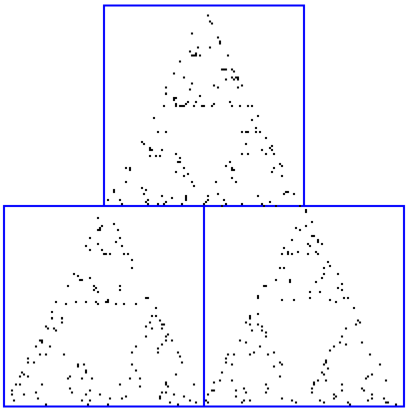




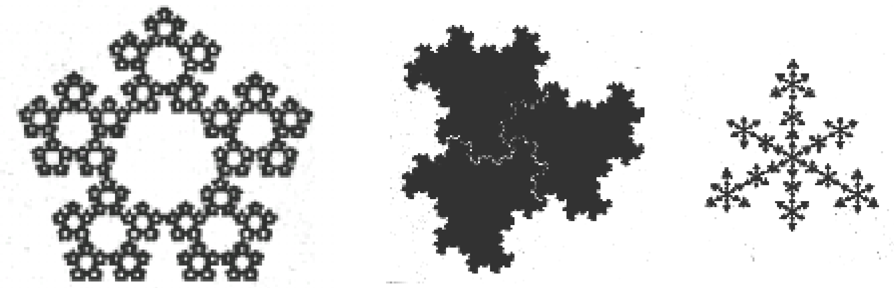
다른 수학적 언어로 표현하면, 앞에서 정의한 W(2)는 그림의 첫번째 사각형 A 에 대한 합성함수 값인 W(W(A)) 라고 할 수 있다. 따라서, $W^3(A)$ 를 W(W(W(A))) 라 할 때, n 의 값이 점점 커질 때 $W^n(A)$ 의 모습들이 시어핀스키 삼각형의 모습으로 점점 다가가게 된다. 그런데 개구리나 학생 들이 찍은 처음의 점과 n 번째의 점이 모두 $W^n(A)$ 의 안에 들어 가기에 각 그림은 시어핀스키 삼각형의 손 바닥 안을 벗어날 수 없으며, 결국 시어핀스키 삼각형 모양의 그림으로 수렴하게 된다.

다음의 명령이 카오스 게임이 진행되는 과정을 나타낸다.

```
def gift(N)  {window 1 ; tt; xx=0;x=0;y=0; n=1;
while(n<N)  {pk=rnd(3);
if(pk==1)  { aff(0.5, 0, 0, 0.5, 0, 0) ;}
else {
if(pk==2)  { aff( 0.5, 0, 0, 0.5, 0.5, 0) ;}
else  {aff( 0.5, 0, 0, 0.5, 0.25, 0.5); }
}
n=n+1;
}
}

def aff(a,b,c,d,ee,ff) { xx=x ; x=a*x+b*y+ee ;
y=c*xx+d*y+ff ;
if(n>10)  { pt(x-e,y-f);}
}

gift(250);
```

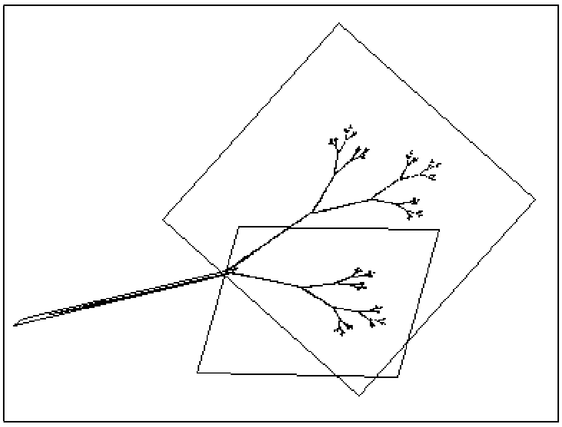


(문제) 다음의 모양에 대해 아핀 변환을 생각해보자. 몇 개의 아핀 변환으로 만들어질 수 있는가?



Ⅶ. 수학적 유전자 코드와 프랙탈 수학

우리는 앞에서 주어진 시어핀스키 삼각형을 프랙탈 조각들로 나눈 후, 각 프랙탈 조각들로 부터 아핀변환을 만들고, 그 아핀변환을 사용하여 전체의 모습을 극한과정을 통해 그려낼 수 있음을 보았다. 이제 아핀변환과 프랙탈 그림과의 관계를 살펴보기 위해 다음과 같은 꽃가지 모양의 프랙탈 그림을 도입한 후, 전체의 모습과 닮은 프랙탈 조각과 아핀변환을 통해 본래의 모습을 그려보자. 여기서 각각의 사각형 안에 있는 나무가지의 모습은 전체 나무 가지의 모습과 닮았음을 주목하자.



위에서 사용되는 아핀변환은 모두 세 개 임을 알 수 있다.아핀 변환은 다음과 같은 식에 의해 결정된다.

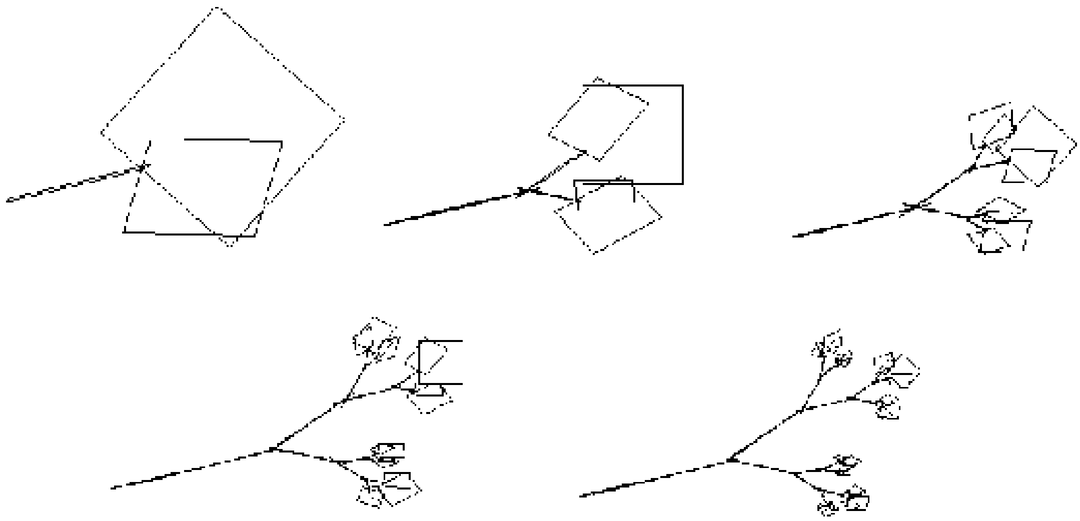
$$ax + by = e,$$
$$cx + dy = f$$

따라서 위의 그림은 세 개의 아핀변환을 이루는 6x3=18개의 숫자(계수)들과 관련이 있는데, 이 숫자들은 마치 그림의 유전자 코드 와 비슷한 것이어서 그 숫자들을 조금만 바꾸어도 돌연변이에 해당하는 약간 다른 그림을 보게된다. 실제로 컴퓨터에서 나무가지를 그리는데 사용된 18개의 숫자들을 약간씩 변형시키며 그림을 그려보면 여러 형태의 비슷한 모양의 나무가지를 보게 된다.

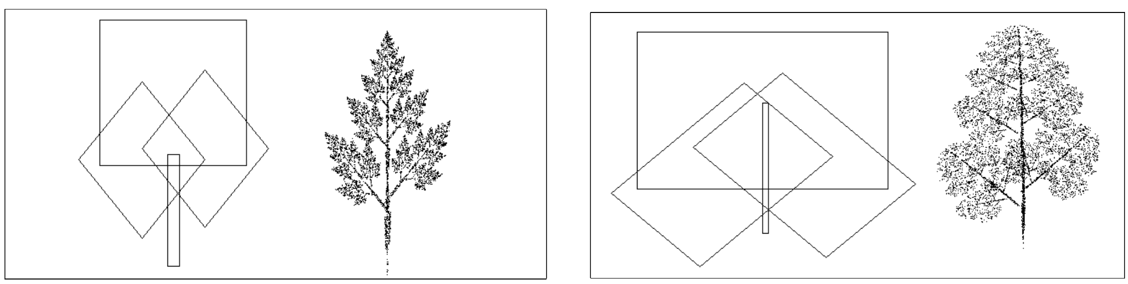
이제 위의 꽃가지 그림과 관계되는 3개의 아핀변환을 반복의 재료로 삼아 나무가지의 그림을 그려나가는 다음의 과정을 살펴보자. 다음에서 보듯이 이 과정을 컴퓨터로 계속하면 점점 원래의 나무가지의 모습이 컴퓨터 화면에 떠오르게 된다. 즉, 다음의 과정을 계속할 때 그 과정의 극한값(그림)에 해당하는 것이 존재하며 그것이 나무의 모양을 갖게 된다.

여기에서도 이와같이 주어진 그림을 그 것의 프랙탈 조각들로 나눈 후, 찾아진 프랙탈 조각들로 부터 계속 반복적으로 시행함으로써 전체의 모습을 그려낼 수 있음을 보게 된다.
이러한 관찰에서 출발하여 미국 조지아 공과대학 교수인 반슬리(Barnsley)는 주어진 그림에서 프랙탈 조각을 찾고, 또한 찾아낸 조각들을 자기닮음성을 그려줄 수

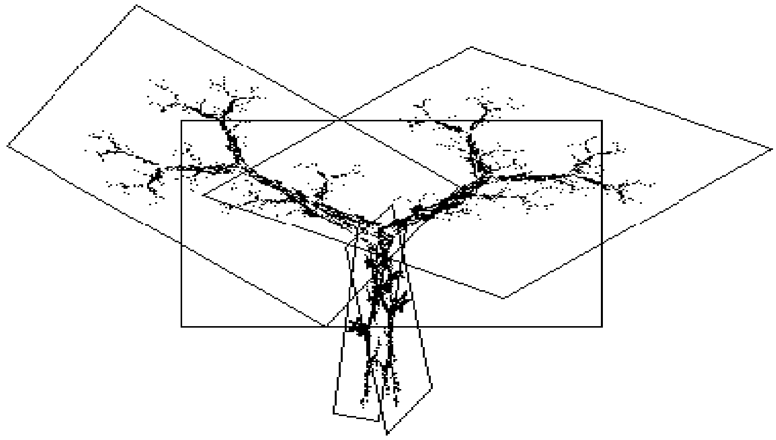
있는 아핀변환들의 모임을 통해 표현하고, 또한 컴퓨터를 통해 주어진 그림을 그려내는 기발한 방법을 발견하게 된다.
여기서 프랙탈 조각으로 부터 만들어진 아핀변환들의 모임을 IFS (Iterated Function System)라고 부르는데, IFS는 건축가가 집을 묘사하는 것처럼 산과 구름 같은 복잡한 그림을 정의하고 또한 표현할 수 있다.

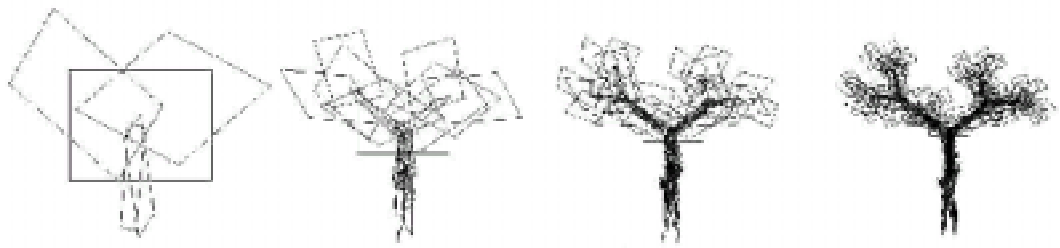


이제 단지 네개의 프랙탈 조각에서 나온 IFS를 통해 그릴 수 다음의 나무 모습들을 보자. 여기서 프랙탈 조각들을 약간씩 변형시키면 여러 모양의 나무 모양도 그려낼 수 있음에 주목하자.

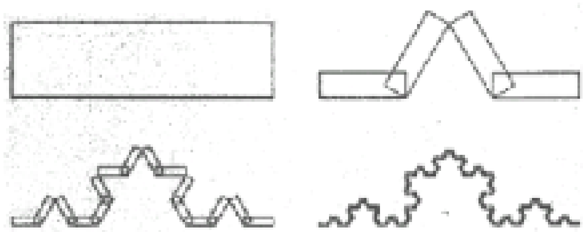


사용된 아핀변환의 개수가 많을 수록 좋은 프랙탈 그림을 그릴 수 있는데, 다음은 5개의 아핀변환을 사용하여 그린 가을철 낙엽진 나무의 모습이다. 참고로 IFS 코드를 조금만 고치면 여름철 잎이 무성한 나무 그림을 그릴 수도 있다.





■ **Koch 곡선에서의 아핀 변환**



Koch 곡선에서 사각형의 모양이 변하는 것을 잘 살펴보자.
그러면 다음과 같은 아핀 변환 네 개가 사용되고 있음을 알 수 있다.

T_1 : 크기가 전단계의 $\frac{1}{3}$로 축소, 평행 이동 하지 않음

$T_1(0,0)=(0,0)$

T_2 : 원점을 중심으로 하여 반시계방향으로 60도 회전, $\frac{1}{3}$만큼 축소, x축으로 $\frac{1}{3}$ 만큼 평행이동

$T_2(0,0)=(\frac{1}{3},0)$

T_3 : 원점을 중심으로 하여 시계방향으로 60도 회전, $\frac{1}{3}$만큼 축소, x축으로 $\frac{1}{2}$, y축으로 r만큼 평행이동

$T_3(0,0)=(\frac{1}{2},r)$

T_4 : 크기가 전단계의 $\frac{1}{3}$로 축소, x축으로 $\frac{2}{3}$ 만큼 평행이동

$T_4(0,0)=(\frac{2}{3},0)$

즉,

$$T_1\begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} \frac{1}{3} & 0 \\ 0 & \frac{1}{3} \end{pmatrix}\begin{pmatrix} x \\ y \end{pmatrix} + \begin{pmatrix} 0 \\ 0 \end{pmatrix}$$

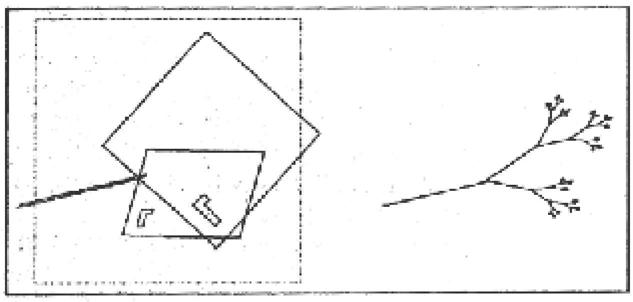
$$T_2\begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} \frac{1}{6} & -r \\ r & \frac{1}{6} \end{pmatrix}\begin{pmatrix} x \\ y \end{pmatrix} + \begin{pmatrix} \frac{1}{3} \\ 0 \end{pmatrix}$$

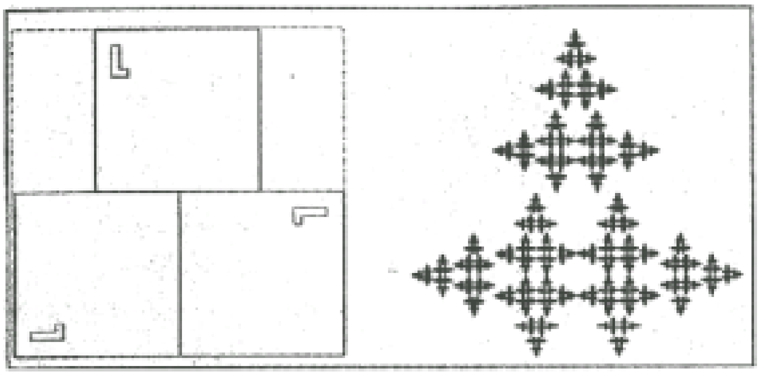
$$T_3\begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} \frac{1}{6} & r \\ -r & \frac{1}{6} \end{pmatrix}\begin{pmatrix} x \\ y \end{pmatrix} + \begin{pmatrix} \frac{1}{2} \\ 0 \end{pmatrix}$$

$$T_4\begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} \frac{1}{3} & 0 \\ 0 & \frac{1}{3} \end{pmatrix}\begin{pmatrix} x \\ y \end{pmatrix} + \begin{pmatrix} \frac{2}{3} \\ 0 \end{pmatrix}$$

인 네 개의 아핀 변환에 의해 변형됨을 알 수 있다.

(문제)
다음 그림의 아핀 변환에 대하여 설명해 보자. (수식으로 써도 좋고 어떤 변화를 했는지 말로 기술해도 좋다.)





(문제)
다음 그림에 해당하는 아핀 변환의 개수를 위에서 보인 사각형의 모양으로 표시해 보자.