Notas de la clase de 1 junio, 2016

- **La alfombra de Sierpinski.** (Ver problema 3 de la tarea 15). Aquí está un código corto que resuelve este problema:

```
function alfombra(n,m)
% display the n-th Sirpienski carpet with side length 3^m, with n<m
A=ones(3^m);
I=0:3^m-1;
for j=1:n
    erase=mod(floor(I/3^(m-j)),3)==1;
    A(erase,erase)=0;
end
imshow(~A);
end
```

- **Fractales como puntos fijos de contracciones.** Denotamos por $\square$ el cuadrado unitario en el plano euclideano, $\square = \{(x, y) \mid 0 \leq x \leq 1, 0 \leq y \leq 1\}$. Una función $T : \square \to \square$. es una *contracciónn* si existe un número $0 \leq c < 1$ tal que para cualquier dos puntos $P_1, P_2 \in \square$,

$$dist(T(P_1), T(P_2)) \leq c\, dist(P1, P2),$$

donde *dist* es la distancia euclideana usual:

$$dist((x_1, y_1), (x_2, y_2)) = \sqrt{(x_1 - x_2)^2 - (y_1 - y_2)^2}.$$

Un *punto fijo* de T es un punto $P \in \square$ tal que $T(P) = P$.

Theorema A. *Toda contracción T de $\square$ tiene un punto fijo único P. Para cualquier punto $P_0 \in \square$, la sucesión de los puntos $P_0, T(P_0), T(T(P_0)), \ldots$ converge a P.*

Ahora consideramos n contracciones $T_1, T_2, \ldots, T_n$ de $\square$, y definimos una transformación F que asocia con cada subconjunto $K \subset \square$ el conjunto

$$F(K) = T_1(K) \cup T_2(K) \cup \ldots \cup T_n(K_n).$$

Esto es, $F(K)$ es la unión de las imagenes de K bajo $T_1, \ldots, T_n$.

Theorema B. *Si $T_1, \ldots, T_n$ son contracciones de $\square$ y F es la transformación asociada del espcaio de los subconjuntos de $\square$ entonces existe un sobconjunto cerrado* único $K_\infty \subset \square$ *tal que para* cualquier *subconjunto cerrado (no vacio) $K \subset \square$, la sucesión de los subconjuntos $K, F(K), F(F(K)), F(F(F(K))), \ldots$ converge al conjunto K_∞.*

Nota: "convergencia"de subconjuntos significa que $\lim_{i\to\infty} dist(K_i, K_\infty) = 0$, donde *dist* es la "distancia de Hausdorf"(ver el artículo de Wikipedia). La idea de la demostración es demostrar que F es una contracción, pero no de $\square$ sino del espacio de los subconjuntos cerrados de $\square$. Referencia: "Fractals everywhere", de Michael Bransley (theorema B es el teorema 7.1 de la p. 89).

El triángulo de Sierpinski. Usando el Teorema B podemos construir muchos fractales, escogiendo contracciones adecuadas. Para construir el llamado triangulo de Sirpienski escojemos 3 contracciones:

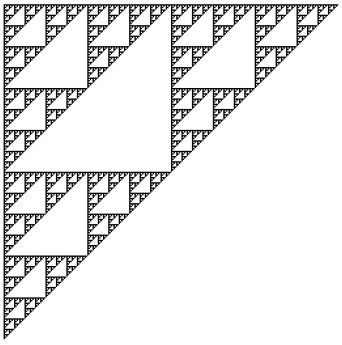
$$T_1(x, y) = (x, y)/2,$$
$$T_2(x, y) = (x + 1, y)/2,$$
$$T_3(x, y) = (x, y + 1)/2.$$

Para implementar esto en Matlab, notamos primero que cada una de estas contracciones son la composición de una re-escalación $(x, y) \mapsto s(x, y)$ y traslación $(x, y) \mapsto (x, y) + (x_0, y_0)$. Escribimos primero una función `transf` que transforma un conjunto $F_{in} \subset \square$, a un conjunto $F_{out} = sF_{in}+(x_0, y_0)$. Ambas F_{in}, F_{out} están representadas por matrices cuadrada de 0's y 1's.

```
function Fout=transf(Fin,s,x0,y0)
% transform Fin by T(x,y)=s(x,y)+(x0,y0)
len=length(Fin);
Fout=zeros(len);
len1=floor(s*len);
x1=floor(len*x0);
y1=floor(len*y0);
D=1:len1;
D1=floor(D/s);
Fout(D+x1,D+y1)=Fin(D1, D1);
end
```

Ahora podemos producir el triángulo de Sierpinski

```
clc;close all;
F=ones(2^10);
T1=zeros(res);T2=T1;T3=T1;
for i=1:9
    pause(.5);
    imshow(~F);
    T1=transf(F,.5,0,0);
    T2=transf(F,.5,.5,0);
    T3=transf(F,.5,0,.5);
    F=T1|T2|T3;
end
```

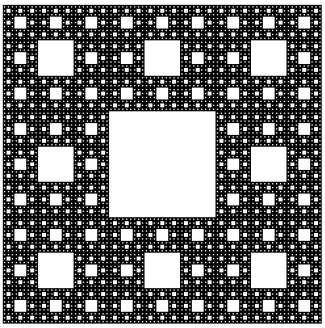


- **La alfombra de Sierpinski** (2nda vuelta). Esta se produce por el proceso descrito en Teorema B, usando 8 contracciones

$$\begin{array}{ll} T_1(x,y) &= (x,y)/3, \qquad T_5(x,y) = (x,y+2)/3, \\ T_2(x,y) &= (x+1,y)/3, \quad T_6(x,y) = (x+1,y+2)/3, \\ T_3(x,y) &= (x+2,y)/3, \quad T_7(x,y) = (x+2,y+1)/3, \\ T_4(x,y) &= (x,y+1)/3, \quad T_8(x,y) = (x+2,y+2)/3. \end{array}$$

El código que lo produce es

```
clc;close all;
s=1/3;
K=ones(3^6);
for i=1:6
    pause(.5);
    imshow(~K);
    K1=transf(K,s,0,0);
    K2=transf(K,s,s,0);
    K3=transf(K,s,2*s,0);
    K4=transf(K,s,0,s);
    K5=transf(K,s,0,2*s);
    K6=transf(K,s,s,2*s);
    K7=transf(K,s,2*s,s);
    K8=transf(K,s,2*s,2*s);
    K=K1|K2|K3|K4|K5|K6|K7|K8;
end
```



- **Agregando rotaciones.** Otro tipo de contracción se obtiene al componer una contraccion de $\square$ con una rotación. Por ejemplo, una rotación por 90^0

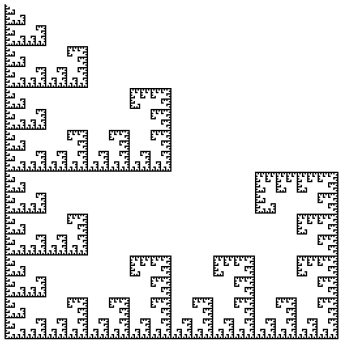
$$Rot_{90^0}(x, y) = (-y, x).$$

Fortunadamente, existe en Matlab una funcíon que hace exactamente esto: `rot90(A,k)` rota una matriz cuadrada por `k*90` grados. Así que modificamos `tranf` a `tranf1` para incluir rotación

```
function Fout=transf1(Fin, s, x0,y0,rot)
Fin=rot90(Fin,rot);
Fout=transf(Fin, s, x0,y0);
end
```

Ahora podemos generar otros fractales, como el "helecho"

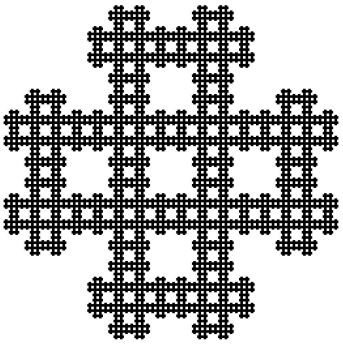
```
clc;close all;
F=ones(2^10); s=.5;
T1=zeros(res);T2=T1;T3=T1;
for i=1:9
 pause(.5);
 imshow(~F);
 T1=transf(F,s,0,0);
 T2=transf(F,s,s,0);
 T3=transf1(F,s,s,s,1);
 F=T1|T2|T3;
end
```



Tarea núm. 16

1. Intenta entender cómo funciona el código de la funcion `alfombra(m,n)` al prinicipio de estas notas. Explicalo a un amigo.
2. Escriba un código que genera el siguiente fractal: empezamos con un cuadrado, lo subdividimos en 16 cuadraditos, y le quitamos las 4 esquinas. Luego, a cada uno de los 12 cuadraditos restantes hacemos lo mismo. Etc.

   Hay que hacer esto de dos maneras:
   a) usando contracciones;
   b) usando un código similar al que se muestra al principio de estas notas (mucho más corto). .



3. Modificar la función `tranf`, para que haga lo que uno espera de ella en la notación matemática usual; esto es, que `tranf(Fin,s,0,0)` sea la imagen `Fin` s veces más pequeña, ubicada en la esquina *inferior izquierda*, y que `tranf(Fin,s,x0,y0)` sea esta última figura transladada x_0 unidades a la *derecha* y y_0 unidades hacia *arrriba*.
4. Escriber una función `hilbert(n,m)` que produce la n-esima curva de Hilbert, en un cuadrado de lado 4^m pixels. Al inicio de esta página se explica con detalle como se construyan estas curvas. Aquí estan las primeras 6 curvas.