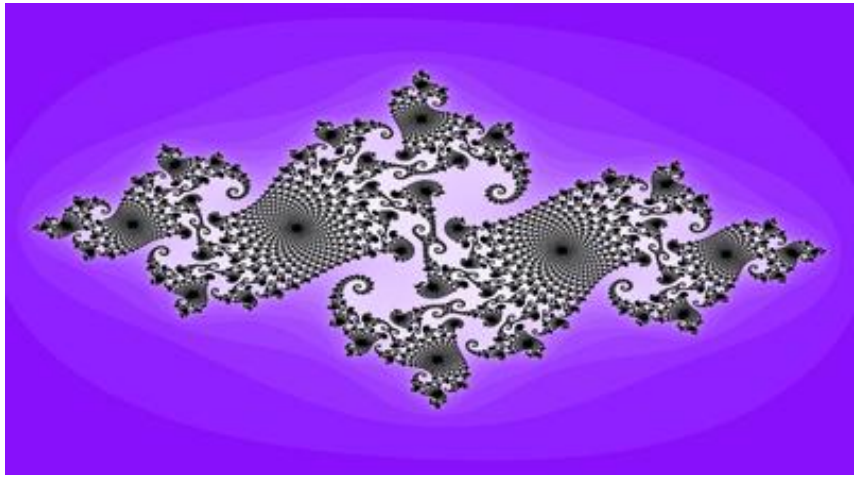
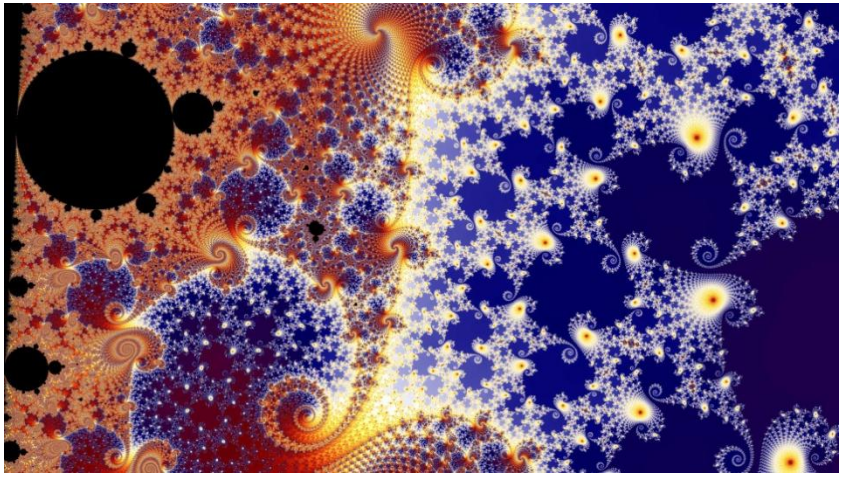
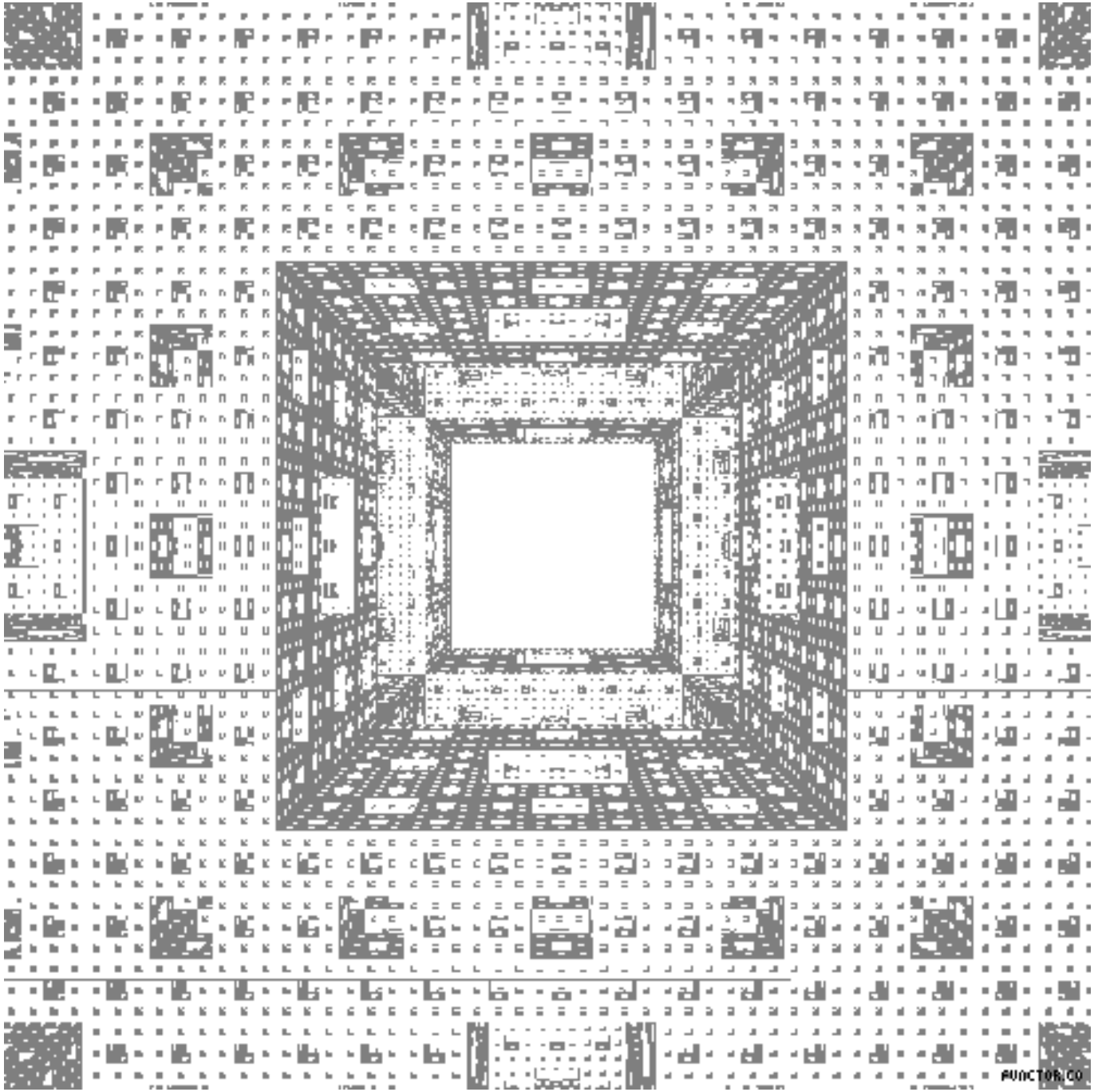
Chmury to nie kule,
góry to nie stożki,
nabrzeża to nie okręgi,
kora nie jest gładka,
a błyskawica nie jest linią prostą.
Benoit Mandelbrot, *The Fractal Geometry of Nature* (1983)

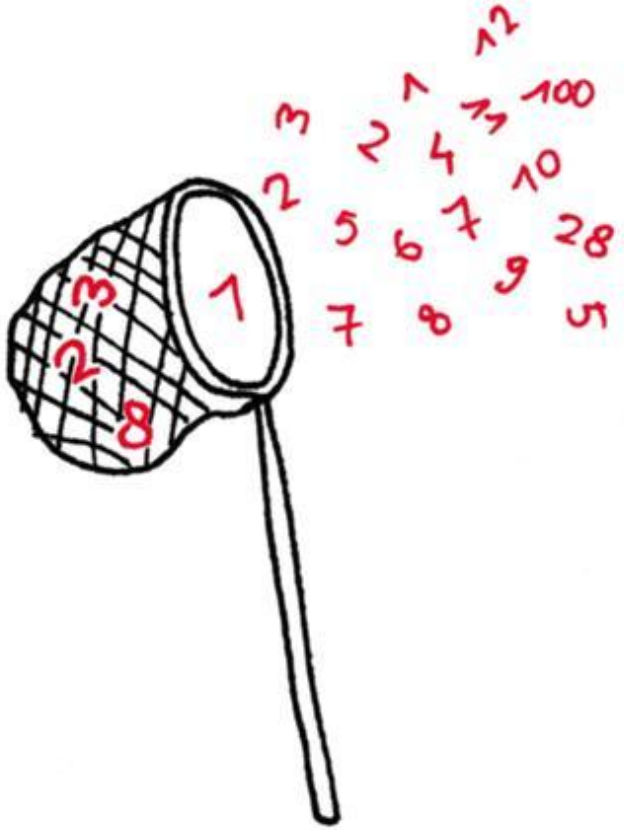
Fraktale






FUNCTOR.CO

CIĄGI A PROGRAMOWANIE



SORT-3 (n, A)

$wart \leftarrow n$
while $wart \neq 0$
  do $k \leftarrow 0$
    for $i \leftarrow 1$ **to** $wart - 1$
      do if $A[i] > A[i+1]$
        then $A[i] \leftrightarrow A[i+1]$
          $k \leftarrow i$
    $wart \leftarrow k$

Tematyka

1. Ciągi w programowaniu
2. Rekurencja vs iteracja
3. Systemy odwzorowań iteracyjnych
4. Wymiar fraktala

Ciągi w programowaniu

1. String
2. Tablica
3. Lista
4. Rekurencja
5. Iteracja, czyli … powtarzanie, a w takim razie pętle:
    - for
    - for each
    - while
    - do while
    - repeat until

Rekurencja a iteracja

0, 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, 144,...

$$F_n = \begin{cases} 0 & \text{dla } n = 0; \\ 1 & \text{dla } n = 1; \\ F_{n-1} + F_{n-2} & \text{dla } n > 1. \end{cases}$$

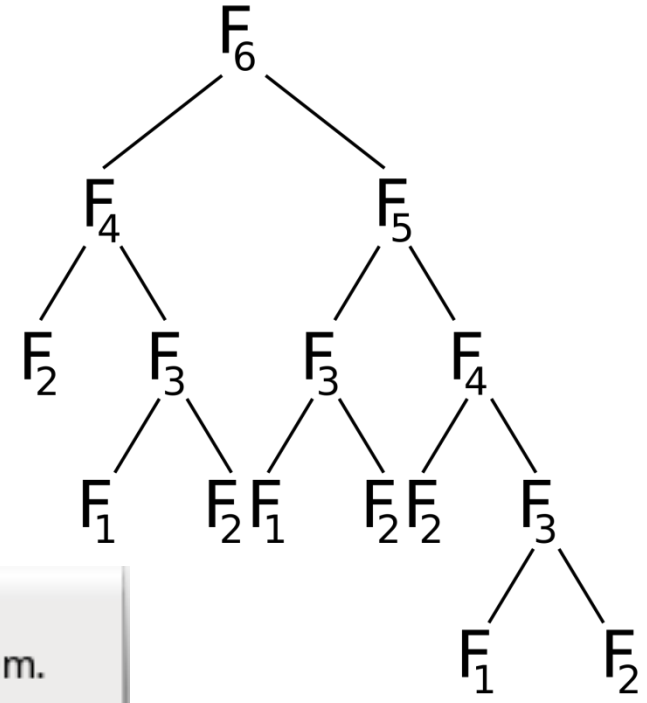
$$F_n = \frac{1}{\sqrt{5}}\left(\frac{1+\sqrt{5}}{2}\right)^n - \frac{1}{\sqrt{5}}\left(\frac{1-\sqrt{5}}{2}\right)^n$$

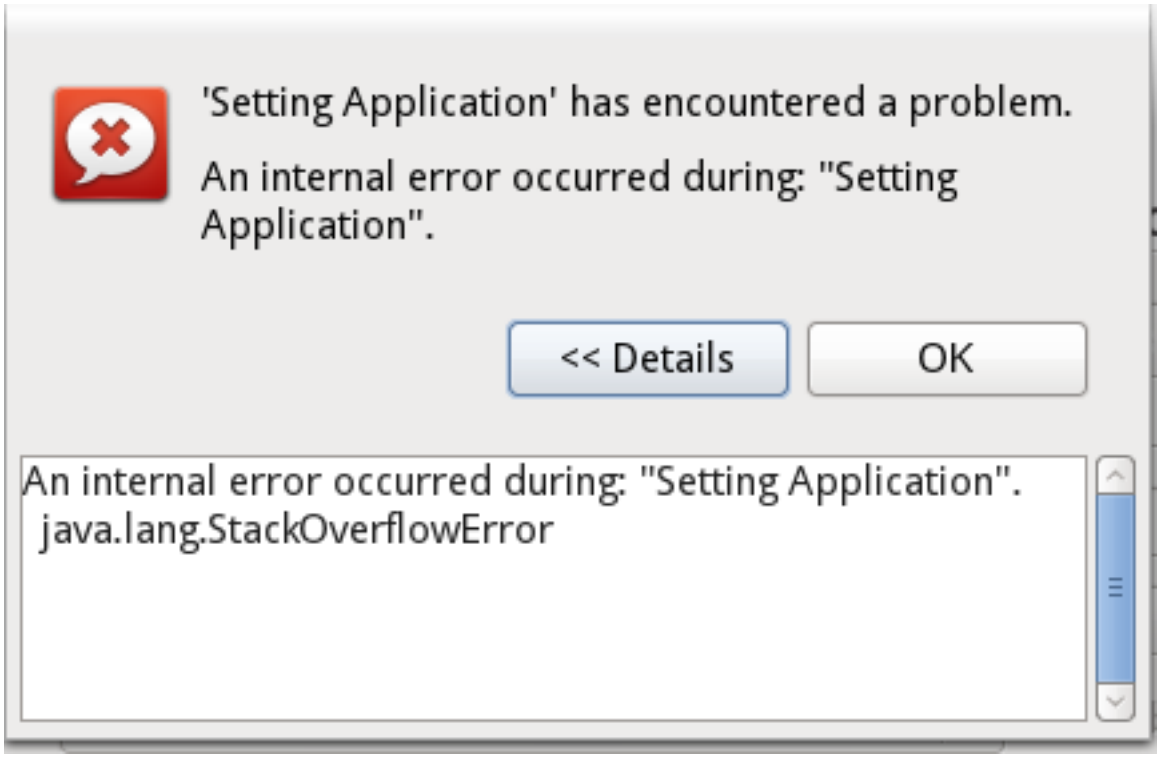
```
Fibonacci( n )
   if n=0 then return 0
   if n=1 then return 1
   f  ← Fibonacci( n-1 )+ Fibonacci( n-2 )
return f
```

```
Fibonacci( n )
  if n=0 then return 0
  if n=1 then return 1
  f' ← 0
  f  ← 1
  for i ← 2 to n
    do
      m  ← f + f'
      f' ← f
      f  ← m
    end
  return f
```

Paproć Barnsleya

Główny problem rekurencji





Czas wyliczenia:

F_{40} – parę sekund

F_{50} – ponad 7 minut

F_{90} – **człowiek tak długo nie żyje!!!**

Układy odwzorowań iterowanych

IFS – ang. iterated function system

Niech *f* będzie przekształceniem, które:
przesuwa, obraca, skaluje, pochyla i/lub odbija.
Wtedy:

$$P(x,y) \mapsto P'(x,y) = f(P(x,y))$$

$$x = ax + by + c$$

$$y = dx + ey + g$$

$$P_n(x,y) \xrightarrow{f} P_{n+1}(x,y)$$

$$n \in \{1,\ldots,k\}$$

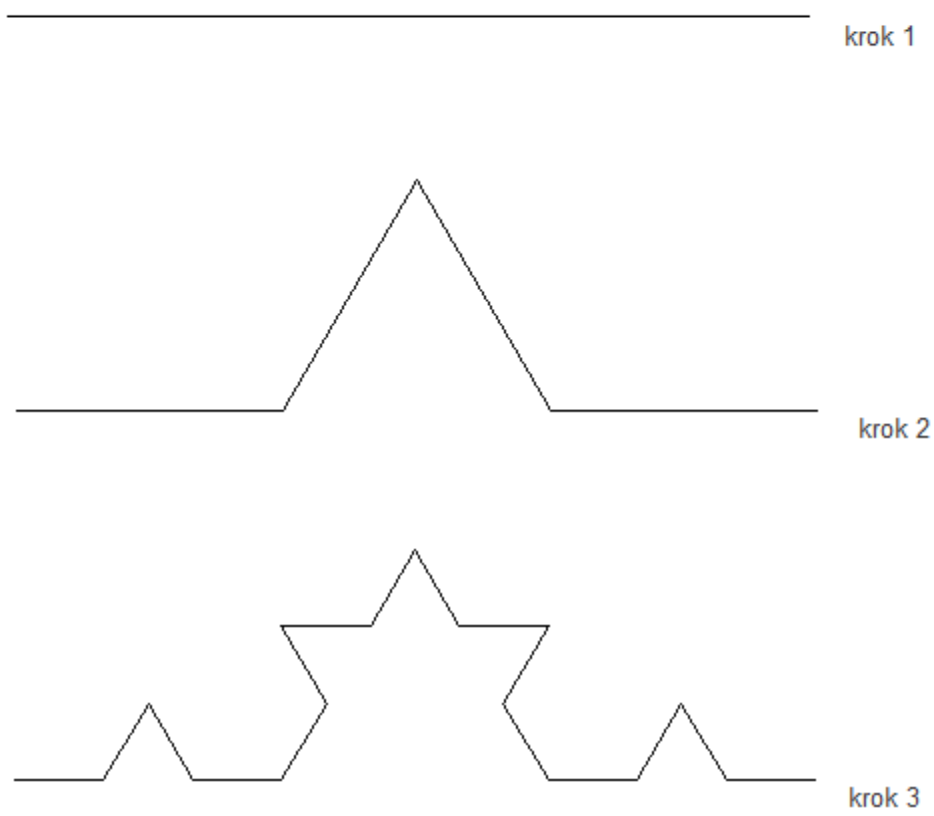
Kopiarka wielokrotnie redukująca

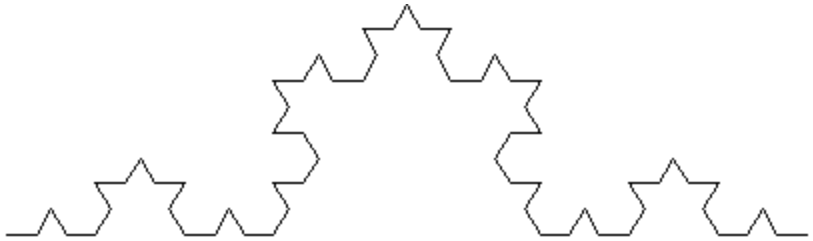
Wyposażona jest w kilka niezależnych systemów soczewek, z których każdy pomniejsza obraz początkowy i umieszcza go gdzieś na wyjściu. Istotne są następujące parametry kopiarki:

- liczba systemów soczewek,
- współczynnik pomniejszenia, osobny dla każdego systemu soczewek,
- ustawienie systemu soczewek przy tworzeniu obrazu na wyjściu.

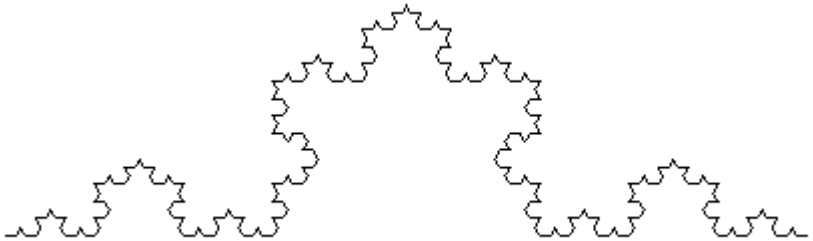
Podstawową zasadą jest sprzężenie zwrotne; obraz po przetworzeniu przez kopiarkę jest przetwarzany ponownie jako obraz wejściowy.

Krzywa Kocha

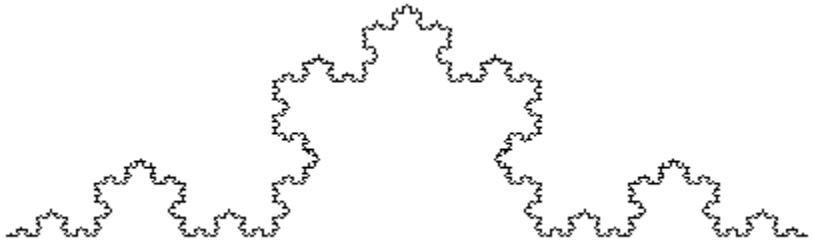




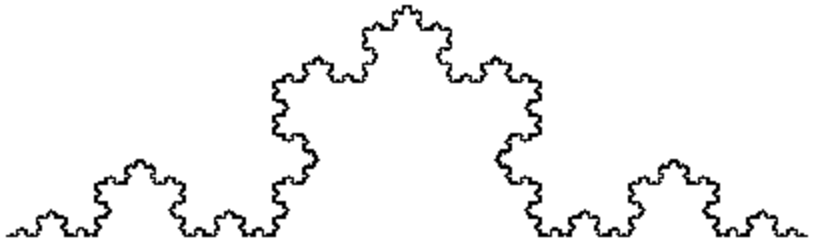
krok 4



krok 5



krok 6



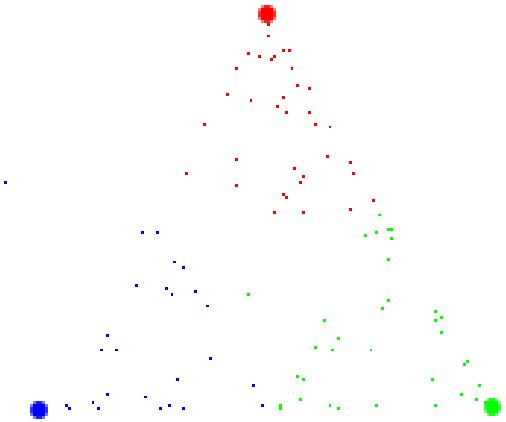
krok 7

rand 55

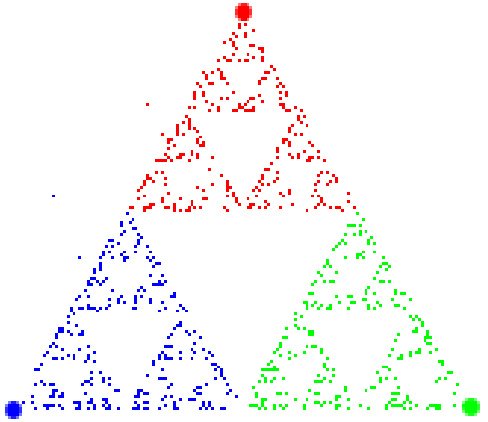
i 7444

IFSP – probabilistyczne (losowe) układy odwzorowań iterowanych

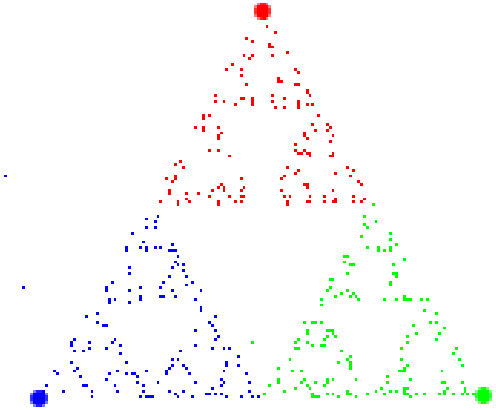
A B A A C B C B A C C A B B A C A
B C B B C A A C A B A C A B B C B
A C B B A C A B C B C B C A C B C
A C B C A C B C B A C C A B A C A



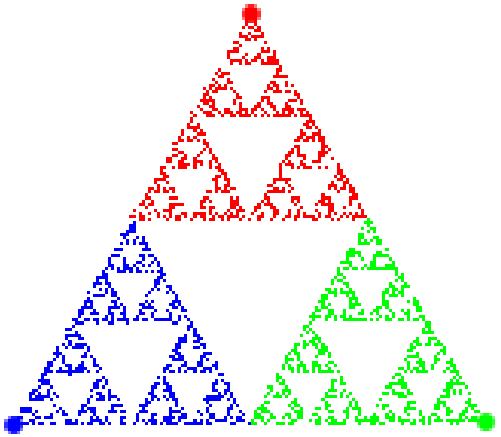
100



1000



500

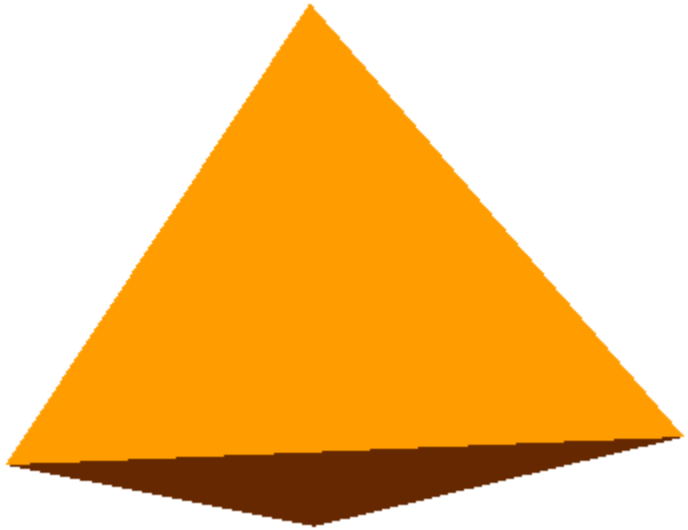


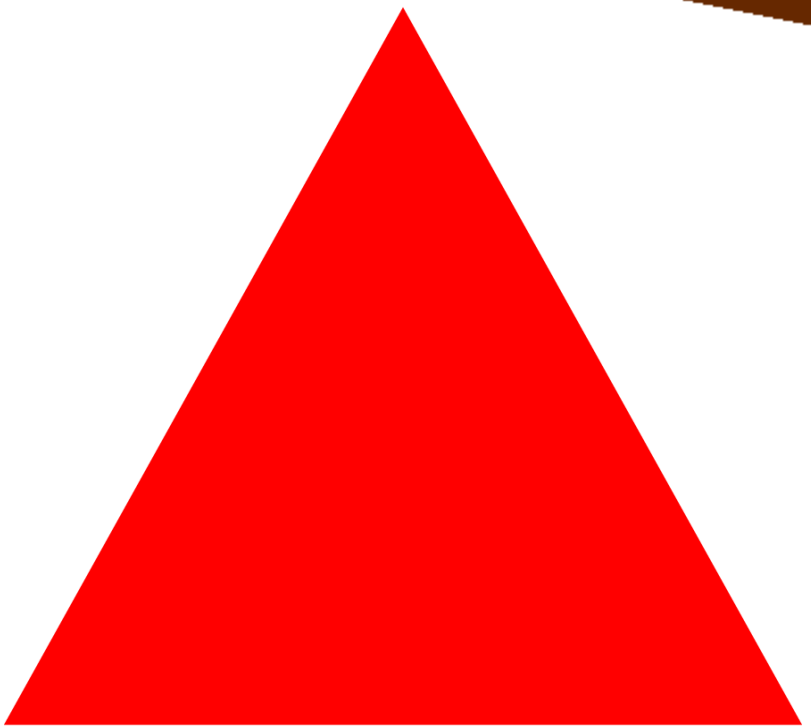
3000

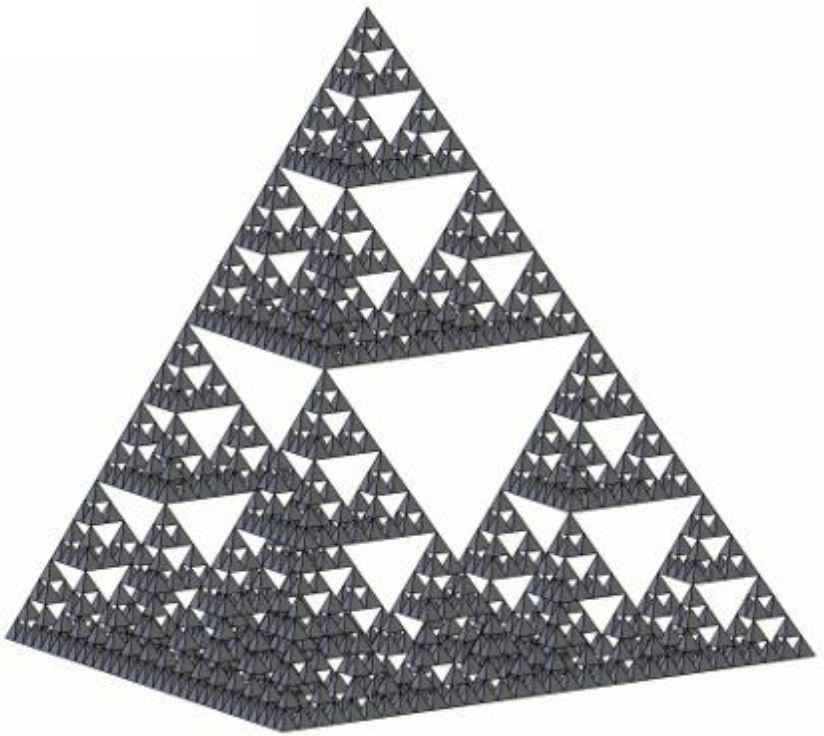
Wymiar samopodobieństwa

D – wymiar samopodobieństwa
a – liczba elementów
s – współczynnik redukcji

$$D = \frac{\log a}{\log \frac{1}{s}}$$







Wymiar pudełkowy

- Dzielimy płaszczyznę za pomocą regularnej siatki o grubości s_n
- Zliczamy w ilu kwadracikach (sześcianikach) siatki znajdują się elementy naszego zbioru, otrzymujemy pewną liczbę zależną od s, nazwijmy ją $N(s_n)$
- Zmniejszamy grubość siatki $s_n \to s_{n+1}$
- Przybliżenie wymiaru pudełkowego uzyskujemy za pomocą wzoru:

$$D = \frac{\left(\log(N(s_{n+1})) - \log(N(s_n))\right)}{\left(\log\left(\frac{1}{s_{n+1}}\right) - \log\left(\frac{1}{s_n}\right)\right)}$$

Układy odwzorowań iterowanych

IFS – ang. iterated function system

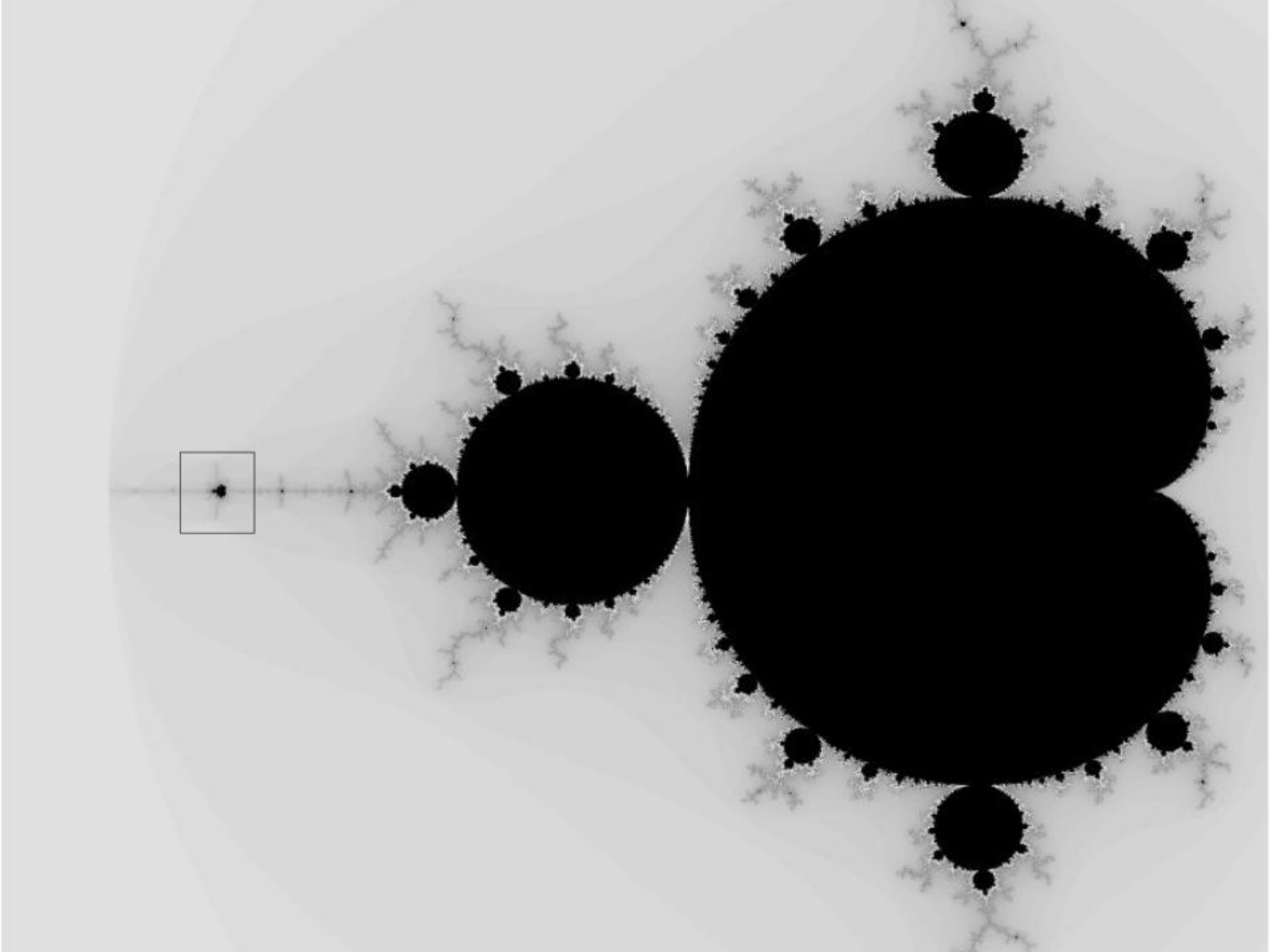
Niech *f* będzie przekształceniem, które:
przesuwa, obraca, **zwęża**, pochyla i/lub odbija.
Wtedy:

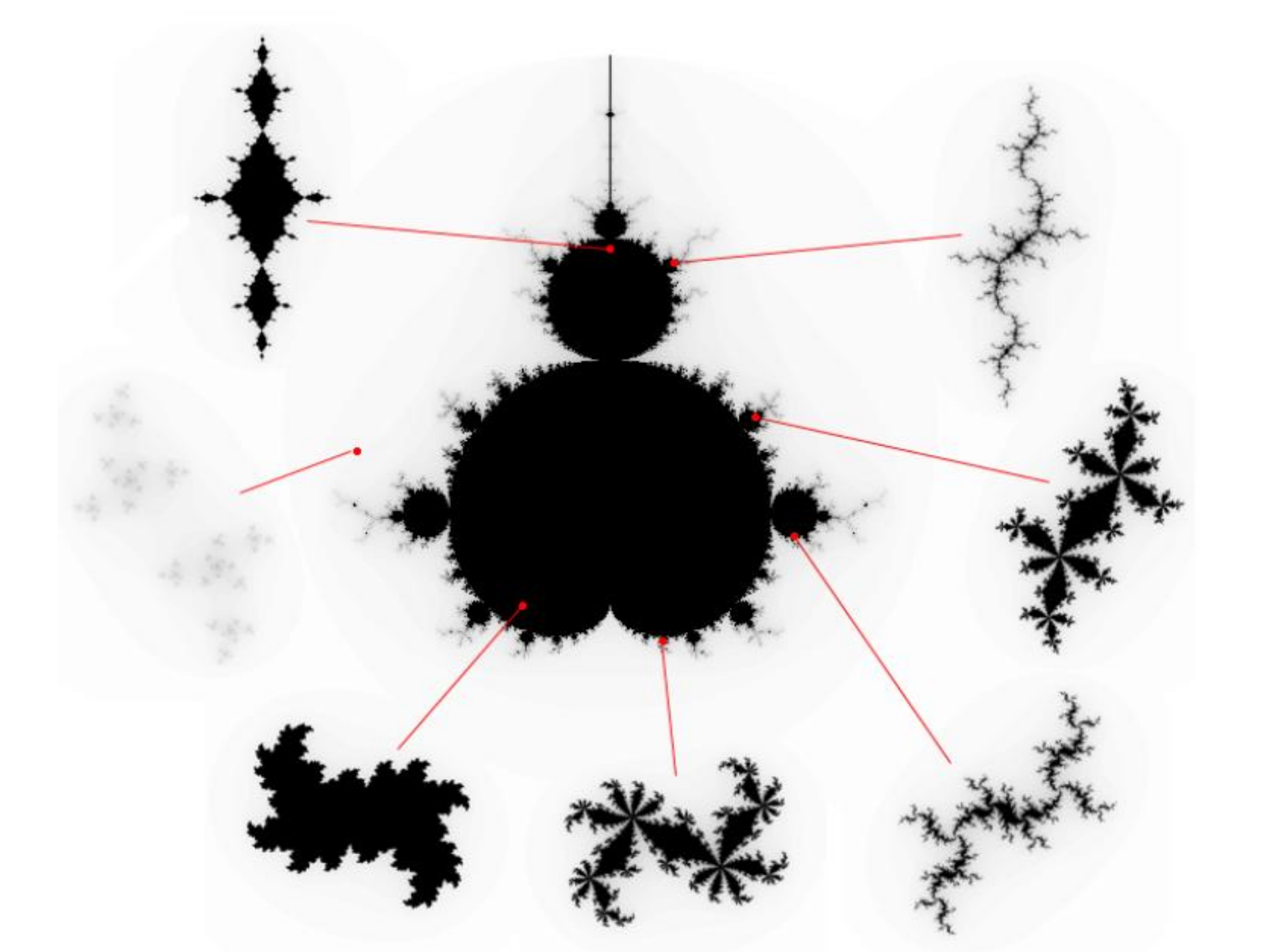
$$x_0 \overset{f_{i_1}}{\longmapsto} f_{i_1}(x_0) = x_1 \overset{f_{i_2}}{\longmapsto} f_{i_2}(x_1) = x_2 \longmapsto \dots$$

$$i_1, i_2, \dots \in \{1, \dots, k\}$$

$$x_n = f_{i_n}(f_{i_{n-1}}(\dots(f_{i_2}(f_{i_1}(x_0)))\dots)) \underset{n\to\infty}{\longrightarrow} A_*$$

A_* – nazywamy **dziwnym atraktorem**





THE END

DZIĘKUJĘ