Wyższa Szkoła Bankowa
we Wrocławiu

Introduction
to

Algorytmy
i struktury
danych

Piotr Ciskowski
Wrocław, 2023

zadanie 1.a. NWD 1

Przetłumacz tak, aby działało:

```
// Algorytm Euklidesa wyznaczania NWD
#include <iostream.h>

int main()
{
  int a,b;
  // Poproś o dwie liczby                  (do we/wy użyj cin/cout)
  // wczytuj dwie liczby (a i b)
  // aż liczby a i b będą większe od zera
  // Dopóki a jest różne od b
  // jeśli a jest większe od b, to odejmij b od a
  // w przeciwnym razie odejmij a od b
  // Wyświetl a lub b jako NWD
}
```

Najpierw narysuj schemat blokowy

Sprawdź, czy to działa

Zaimplementuj w Pythonie

zadanie 1.a. NWD 1

rozwiązanie

Przetłumacz tak, aby działało:

```
// Algorytm Euklidesa wyznaczania NWD
#include <iostream.h>

int main()
{
  int a,b;
  do { cout << ”\nPodaj dwie liczby: ” ; // Poproś o dwie liczby
       cin  >>  a  >> b ;                 // wczytuj dwie liczby (a i b)
     } while ( a <= 0 || b <= 0 ) ;      // aż liczby a i b będą większe od zera
  while ( a != b )                        // Dopóki a jest różne od b
    if ( a > b) a -= b;                      // jeśli a jest większe od b, to odejmij b od a
    else        b -= a;                      // w przeciwnym razie odejmij a od b
  cout << ”\nNWD = ” << a ;               // Wyświetl a lub b jako NWD
}
```

Zamień część obliczającą NWD na funkcję i odpowiednio dostosuj program.

zadanie 1.a. NWD 1

rozwiązanie

Przetłumacz tak, aby działało:

```
// Algorytm Euklidesa wyznaczania NWD
#include <iostream.h>

int main()
{
  int a,b;
  do { cout << "\nPodaj dwie liczby: " ; // Poproś o dwie liczby
       cin  >>  a  >> b ;                // wczytuj dwie liczby (a i b)
     } while ( a <= 0 || b <= 0 ) ;      // aż liczby a i b będą większe od zera
  while ( a != b )                       // Dopóki a jest różne od b
  { if ( a > b) a -= b;                     // jeśli a jest większe od b, to odejmij b od a
    else        b -= a;                     // w przeciwnym razie odejmij a od b
    cout << "\na = " << a << ", b = " << b ;   // co jest grane?
  }
  cout << "\nNWD = " << a ;              // Wyświetl a lub b jako NWD
}
```

Sprawdź działanie programu dla:

- **a = 12222**
- **b = 56**

zadanie 1.b. NWD 2

Przetłumacz tak, aby działało:

```
// Algorytm wyznaczania NWD
#include <stdio.h>
#include <conio.h>
int main()
{ int a,b,r;
  // Wyczyść ekran
  // Poproś o dwie liczby              (do we/wy użyj printf/scanf)
  // wczytaj dwie liczby (a i b)
  // Dopóki reszta z dzielenia a przez b jest różna od zera
  // pod a podstaw b
  // i pod b podstaw tę resztę
  // Wyświetl b lub resztę jako NWD
}
```

Najpierw narysuj schemat blokowy

Sprawdź, czy działa

Zaimplementuj w Pythonie

zadanie 1.b. NWD 2

rozwiązanie – bez funkcji

Przetłumacz tak, aby działało:

```
// Algorytm wyznaczania NWD
#include <stdio.h>
#include <conio.h>

int main()
{ int a,b ;
  clrscr() ;                    // po nowemu: system (”CLS”);
  printf ( ”Podaj dwie liczby: ”  ) ;  scanf ( ”%d%d” , &a , &b ) ;
  int r ;
  while ( r=a%b )               // Dopóki reszta z dzielenia a przez b jest różna od zera
  { a=b ;                       // pod a podstaw b
    b=r ;                       // i pod b podstaw tę resztę
  }
  printf ( ”\nNWD= %d” , b ) ;
}
```

Na koniec pętli podejrzyj a i b

Sprawdź, jak działa program, gdy podasz:

- a = 12222
- b = 56

zadanie 2.a. Wielkanoc

Wielkanoc jest świętem ruchomym
– wypada w pierwszą niedzielę po pierwszej wiosennej pełni księżyca.

Napisz program wyznaczający datę świąt wielkanocnych w podanym roku wg algorytmu Gaussa:

- algorytm działa dla lat od 325 do 2200
- przyjmujemy oznaczenia: a – reszta z dzielenia roku przez 19
  b – reszta z dzielenia roku przez 4
  c – reszta z dzielenia roku przez 7
  d – reszta z dzielenia wyrażenia $(19a+x)$ przez 30
  e – reszta z dzielenia wyrażenia $(2b+4c+6d+y)$ przez 7
  f – $22+d+e$
- przy czym: $x = 15$, $y = 6$ jeśli rok < 1583,
  $x = 22$, $y = 2$ jeśli $1583 \leq$ rok < 1700,
  $x = 23$, $y = 3$ jeśli $1700 \leq$ rok < 1800,
  $x = 23$, $y = 4$ jeśli $1800 \leq$ rok < 1900,
  $x = 24$, $y = 5$ jeśli $1900 \leq$ rok < 2100,
  $x = 24$, $y = 6$ jeśli $2100 \leq$ rok < 2200
- Wielkanoc wypada f-tego marca,
  gdy $f > 31$, przechodzi na kwiecień

Narysuj schemat blokowy

Sprawdź, czy działa

Zaimplementuj w Pythonie

zadanie 3. Arab w Rzym

Napisz program, który przy użyciu zagnieżdżonych pętli oraz instrukcji switch wyświetli liczbę arabską w zapisie rzymskim:

- Po wczytaniu liczby arabskiej powinien on 13 razy powtórzyć pętlę,
  w której pod zmienną z podstawi sobie następujące wartości:
  1000, 900, 500, 400, 100, 90, 50, 40, 10, 9, 5, 4, 1
  (są to kolejne (od góry) liczby mające odpowiedniki w zapisie rzymskim)

- Dalej w tej samej pętli
  dopóki zamieniana liczba arabska będzie większa od danego odpowiednika,
  - program napisze rzymski symbol: M, CM, D, XD, C, XC, L, XL, X, IX, V, IV, I
  - - i odejmie go od zamienianej liczby

- **Popraw drobny błąd w powyższym algorytmie**

Narysuj schemat blokowy

Sprawdź, czy działa

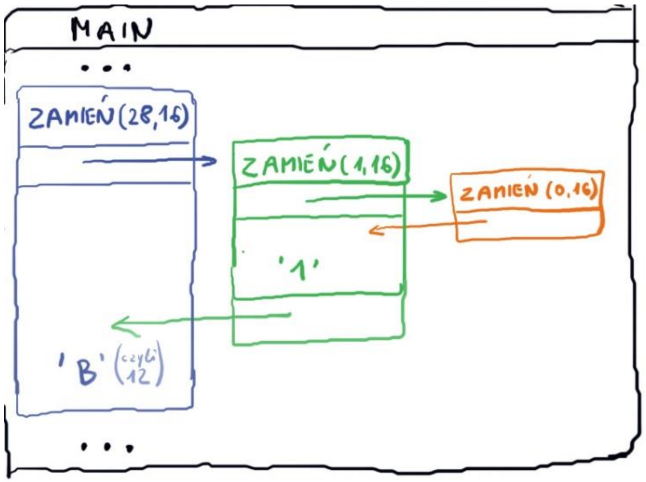
Zaimplementuj w Pythonie – w wersji z tablicą lub słownikiem

zadanie 4. Dziesiętna na inną

Napisz program wczytujący liczbę dziesiętną
oraz podstawę systemu, na który powinien zamienić tę liczbę (od 2 do 36).
W programie należy zdefiniować funkcję zamien
przyjmującą za parametry liczbę do zamiany oraz podstawę systemu.
Funkcja ta powinna nie robić nic, gdy liczba do zamiany jest równa zero,
a gdy jest różna od zera:

- wywoływać samą siebie z liczbą do zamiany podzieloną (całkowicie) przez podstawę systemu,
- po czym wyświetlić na ekranie resztę z tego dzielenia
  (w postaci cyfr 0...9 lub dużych liter: A, B, C, … - wykorzystać tablicę znaków ASCII).

przykład: liczba $28_{(10)}$ to 16+12, czyli $1B_{(16)}$



Narysuj schemat blokowy

Sprawdź, czy działa

Zaimplementuj w Pythonie

przykład 5. obliczanie wartości liczby *e*

Funkcję e^x można przedstawić w postaci szeregu: $e^x = 1 + \frac{x^1}{1!} + \frac{x^2}{2!} + \frac{x^3}{3!} + \cdots + \frac{x^i}{i!} + \cdots$

Napisz funkcję, która oblicza sumę tego szeregu dla podanej wartości *x*:

- metodą nieefektywną – obliczając każdy wyraz osobno i korzystając z zewnętrznej funkcji do obliczania silni
- lepszą metodą - obliczając każdy wyraz na podstawie poprzedniego

Niech funkcja kończy obliczenia dla wyrazu, dla którego $\left|\frac{x^i}{i!}\right| < \varepsilon$
gdzie ε jest z góry zadaną dokładnością

Oblicz $e^{-5.5}$ z dokładnością 10^{-10}

Porównaj wyniki z wynikami wbudowanej funkcji *exp*(*x*)

Oblicz 1000 razy $e^{-5.5}$ z dokładnością 10^{-10}
i porównaj czas obliczeń nieefektywnych, efektywnych oraz wbudowanej funkcji exp

Narysuj schemat blokowy

Sprawdź, czy działa

Zaimplementuj w Pythonie

zadanie 6. obliczanie wartości wielomianu - schemat Hornera

Należy obliczyć wartość wielomianu $W_n(x) = a_0 x^n + a_1 x^{n-1} + \cdots + a_{n-1} x + a_n$
dla pewnej wartości argumentu $x = z$

***W**$_n$**(*z*)** **można obliczyć wprost ze wzoru – wtedy liczba działań wynosi: … mnożeń i … dodawań**

Jeśli wielomian przedstawimy w takiej postaci:

$$W_n(x) = \left(\left(\left(\left(\left(a_0 x + a_1\right)x + a_x\right)x + a_3\right)\cdots + a_{n-1}\right)x + a_n\right)$$

to będziemy mogli wykorzystać schemat Hornera:

$$\begin{aligned} b_0 &= a_0 \\ b_1 &= b_0 z + a_1 \\ b_2 &= b_1 z + a_2 \\ &\ldots \\ b_i &= b_{i-1} z + a_i \\ &\ldots \\ W_n(z) &= b_n \end{aligned}$$

do wykonania którego potrzeba … mnożeń i … dodawań

Narysuj schemat blokowy

Sprawdź, czy działa

Zaimplementuj w Pythonie

zadanie 6. obliczanie wartości wielomianu - schemat Hornera

$$W_n(x) = a_0 x^n + a_1 x^{n-1} + \cdots + a_{n-1} x + a_n$$

$$W_n(x) = \left(\left(\left(\left(\left(a_0 x + a_1\right) x + a_x\right) x + a_3\right) \cdots + a_{n-1}\right) x + a_n\right)$$

$$b_0 = a_0$$
$$b_1 = b_0 z + a_1$$
$$b_2 = b_1 z + a_2$$
$$\dots$$
$$b_i = b_{i-1} z + a_i$$
$$\dots$$
$$W_n(z) = b_n$$

a_0	a_1	a_2	…	$a_{n\text{-}1}$	a_n	
	$+$	$+$		$+$	$+$	
	b_0z	b_1z	…	$b_{n\text{-}2}z$	$b_{n\text{-}1}z$	$\cdot\, z$
$b_0 = a_0$	b_1	b_2	…	$b_{n\text{-}1}$	$\underline{\underline{b_n}}$	$\underline{\underline{b_n = W_n(z)}}$

zadanie 7. liczby pierwsze

- Liczba pierwsza to taka liczba naturalna, która nie dzieli się przez żadną inna liczbę oprócz siebie i jedności. Liczby złożone (nie będące pierwszymi) dają się przedstawić jako iloczyn liczb pierwszych. Liczby pierwsze wykorzystuje się w kryptografii
- Jednym z prostszych i chyba najpopularniejszym algorytmem generowania liczb pierwszych jest sito Erastotenesa – eliminuje liczby podzielne przez jakąkolwiek liczbę oprócz badanej
- Algorytm zapisany kiedyś w pseudojęzyku (dla języka Pascal) zapisz jako funkcję w MATLABie
- Znajdź inne algorytmy generowania liczb pierwszych, ew. sprawdzania, czy dana liczba jest liczbą pierwszą

Narysuj schemat blokowy

Sprawdź, czy działa

Zaimplementuj w Pythonie

zadanie 7. liczby pierwsze

```
Początek
  Czytaj(M);
  Pisz(2);
  i:=3;
  dzielnik:=2;
  Dopóki i<M wykonuj
    Początek
      Powtarzaj
        Dopóki (i mod dzielnik=0) wykonuj
         Początek
           i:=i+1;
           dzielnik:=2
         Koniec;
        dzielnik:=dzielnik+1
      aż do dzielnik ≥ i div 2;
    Jeśli i<=M to Pisz(i);
    i:=i+1;
    dzielnik:=2
  Koniec
Koniec
```

- Napisz program rysujący w MATLABie zbiory Julii i Mandelbrota
- O rysowaniu fraktali można poczytać tu: J. Kudrewicz. *Fraktale i chaos*. WNT, Warszawa 1993

- Pomijając zawiłe wyprowadzenia matematyczne,
możemy przyjąć następujące metody rysowania fraktali:

  - jest przekształcenie $P(z) = z^{pot} + c$, w którym z i c to liczby zespolone
  - zaczynamy z punktu początkowego z_0 i iterujemy go tak długo,
  aż wyjdzie poza okrąg o promieniu r_{gran}

  - dla zbioru Mandelbrota zaczynamy od $z_0=0$ i obserwujemy liczbę kroków,
  po których iterowany punkt wychodzi poza okrąg graniczny o promieniu r_{gran}
  w zależności od parametru c
  – współrzędne x i y wykresu to $Re(c)$ i $Im(c)$
  - kolor punktu zależy od liczby kroków

  - dla zbioru Julii przyjmujemy z kolei z góry parametr c,
  a obserwujemy liczbę kroków, po której punkt ucieka poza okrąg graniczny
  w zależności od punktu początkowego z_0
  - współrzędne x i y wykresu to $Re(z_0)$ i $Im(z_0)$
  - kolor punktu zależy od liczby kroków

przykład 8. fraktale

- zestawy gotowych parametrów fraktali, na podstawie wspomnianej książki:

```
case  1, jakiZbior=['Mandelbrot']; pot=2; c=0; rGran=2.3;
         ReMin=-2.0; ReMax=2.0; ImMin=-2.0; ImMax=2.0; maxIter=20;
case  2, jakiZbior=['Julia'];      pot=2; c=-i; rGran=2.3;
         ReMin=-1.8; ReMax=1.8; ImMin=-1.3; ImMax=1.3; maxIter=20;
case  3, jakiZbior=['Mandelbrot']; pot=3; c=0; rGran=2;                       % fot. 15
         ReMin= 0.100; ReMax=0.108; ImMin= 0.7865; ImMax=0.7925; maxIter=100;
case  4, jakiZbior=['Julia'];      pot=3; c=0.5+i*0.4756; rGran=2;            % fot. 16
         ReMin=-1.6; ReMax=1.6; ImMin=-1.2; ImMax=1.2; maxIter=200;
case  5, jakiZbior=['Mandelbrot']; pot=3; c=0; rGran=2;                       % rys. 11.4
         ReMin= 0.1250; ReMax=0.1310; ImMin= 0.7610; ImMax=0.7670; maxIter=200;
case  6, jakiZbior=['Mandelbrot']; pot=3; c=0; rGran=2;
         ReMin= 0.1020; ReMax=0.1080; ImMin= 0.7865; ImMax=0.7925; maxIter=200;
case  7, jakiZbior=['Mandelbrot']; pot=3; c=0; rGran=2;
         ReMin= 0.4987; ReMax=0.5005; ImMin= 0.4755; ImMax=0.4773; maxIter=200;
case  8, jakiZbior=['Mandelbrot']; pot=3; c=0; rGran=2;
         ReMin= 0.4978; ReMax=0.5008; ImMin= 0.0810; ImMax=0.0840; maxIter=200;
case  9, jakiZbior=['Julia']; pot=3; c=i*1.088; rGran=2;                       % rys. 11.5
         ReMin=-1.5; ReMax=1.5; ImMin=-1.5; ImMax=1.5; maxIter=200;
case 10, jakiZbior=['Julia']; pot=3; c=i*1.089; rGran=2;
         ReMin=-1.5; ReMax=1.5; ImMin=-1.5; ImMax=1.5; maxIter=250;
case 11, jakiZbior=['Julia']; pot=3; c=0.4996+i*0.4776; rGran=2;
         ReMin=-1.5; ReMax=1.5; ImMin=-1.5; ImMax=1.5; maxIter=350;
```

przykład 8. fraktale

- zestawy gotowych parametrów fraktali, na podstawie wspomnianej książki:

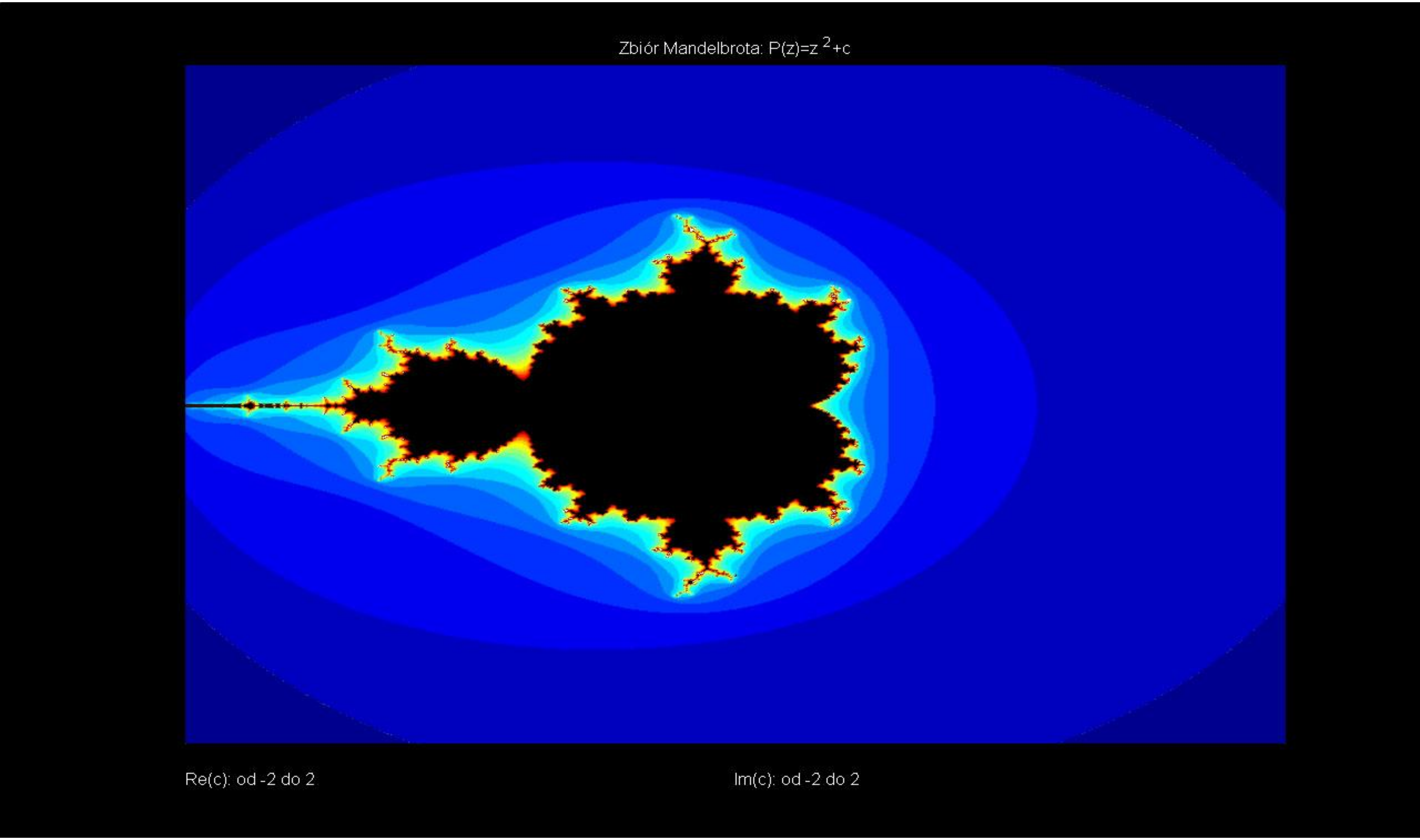
```
case 12, jakiZbior=['Julia']; pot=3; c=0.4996+i*0.4775; rGran=2;
         ReMin=-1.5; ReMax=1.5; ImMin=-1.5; ImMax=1.5; maxIter=350;
case 13, jakiZbior=['Julia']; pot=3; c=0.104945+i*0.788569; rGran=2;        % rys. 11.6
         ReMin=-1.5; ReMax=1.5; ImMin=-1.5; ImMax=1.5; maxIter= 70;
case 14, jakiZbior=['Julia']; pot=3; c=0.49876+i*0.08273; rGran=2;
         ReMin=-1.5; ReMax=1.5; ImMin=-1.5; ImMax=1.5; maxIter=300;
case 15, jakiZbior=['Julia']; pot=3; c=0.591+i*0.591; rGran=2;
         ReMin=-1.5; ReMax=1.5; ImMin=-1.5; ImMax=1.5; maxIter=100;
case 16, jakiZbior=['Julia']; pot=3; c=0.2+i*0.751; rGran=2;
         ReMin=-1.5; ReMax=1.5; ImMin=-1.5; ImMax=1.5; maxIter=150;
case 17, jakiZbior=['Julia']; pot=3; c=0.5+i*0.475; rGran=2;
         ReMin=-1.5; ReMax=1.5; ImMin=-1.5; ImMax=1.5; maxIter=180;
case 18, jakiZbior=['Julia']; pot=3; c=0.02+i*0.7698; rGran=2;
         ReMin=-1.5; ReMax=1.5; ImMin=-1.5; ImMax=1.5; maxIter=400;
case 19, jakiZbior=['Julia']; pot=2; c= 0; rGran=2;                         % rys. 10.7
         ReMin=-1.75; ReMax=1.75; ImMin=-1.75; ImMax=1.75; maxIter=100;
case 20, jakiZbior=['Julia']; pot=2; c=-0.5; rGran=2;
         ReMin=-1.75; ReMax=1.75; ImMin=-1.75; ImMax=1.75; maxIter=100;
case 21, jakiZbior=['Julia']; pot=2; c=-0.124-i*0.565; rGran=2;
         ReMin=-1.75; ReMax=1.75; ImMin=-1.75; ImMax=1.75; maxIter=100;
case 22, jakiZbior=['Julia']; pot=2; c=-1; rGran=2;
         ReMin=-1.75; ReMax=1.75; ImMin=-1.75; ImMax=1.75; maxIter=100;
```

przykład 8. fraktale

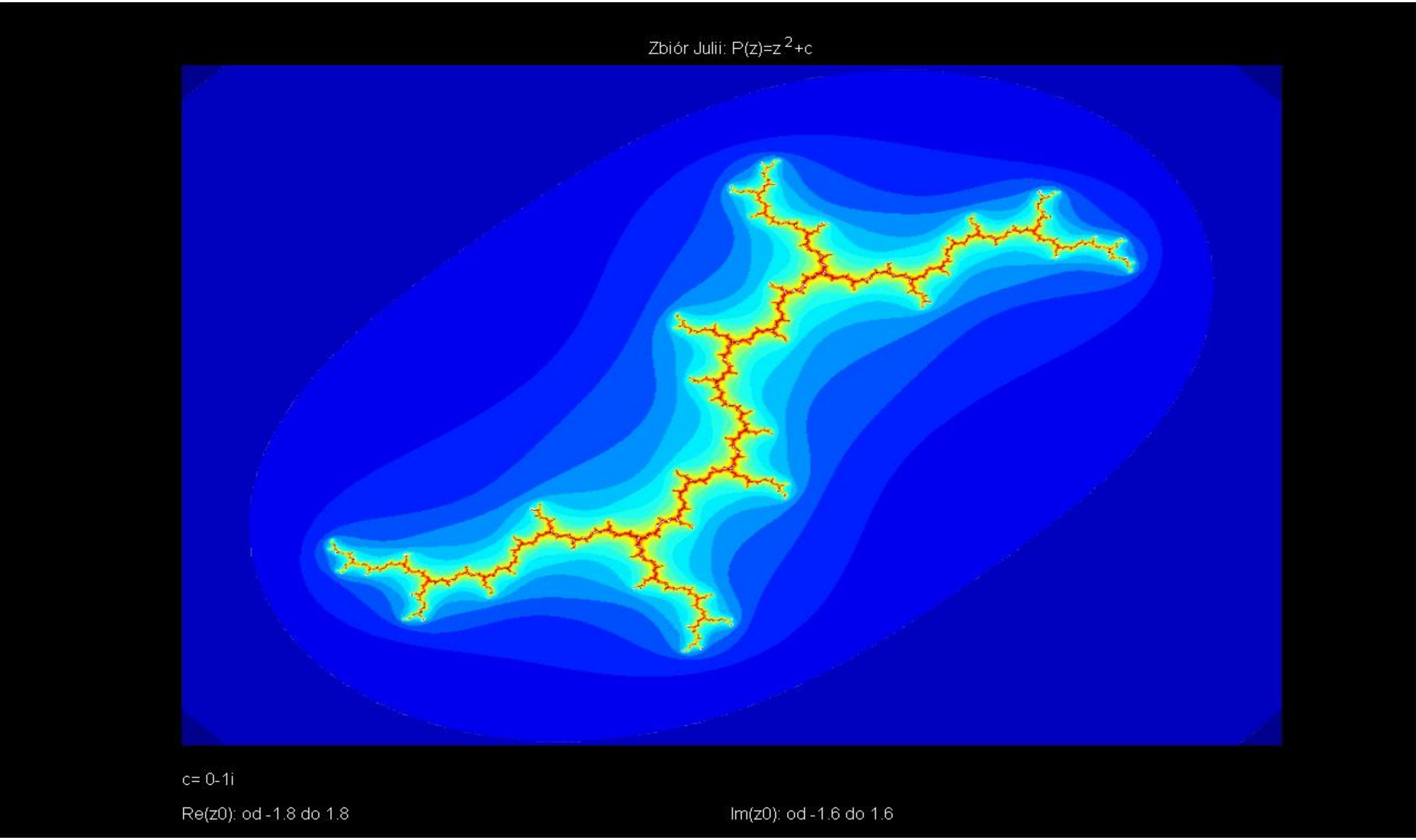
- zestawy gotowych parametrów fraktali, na podstawie wspomnianej książki:

```
case 23, jakiZbior=['Julia']; pot=2; c=-0.12256-i*0.74486; rGran=2;
         ReMin=-1.75; ReMax=1.75; ImMin=-1.75; ImMax=1.75; maxIter=100;
case 24, jakiZbior=['Julia']; pot=2; c= 0.404-i*0.38; rGran=2;
         ReMin=-1.75; ReMax=1.75; ImMin=-1.75; ImMax=1.75; maxIter=100;
case 25, jakiZbior=['Julia']; pot=2; c=-1.45; rGran=2;
         ReMin=-1.75; ReMax=1.75; ImMin=-1.75; ImMax=1.75; maxIter=100;
case 26, jakiZbior=['Julia']; pot=2; c=-i; rGran=2;
         ReMin=-1.75; ReMax=1.75; ImMin=-1.75; ImMax=1.75; maxIter=100;
case 27, jakiZbior=['Julia']; pot=2; c= 0.412+i*0.134; rGran=2;
         ReMin=-1.75; ReMax=1.75; ImMin=-1.75; ImMax=1.75; maxIter=100;
case 28, jakiZbior=['Julia']; pot=2; c= 0.35; rGran=2;
         ReMin=-1.75; ReMax=1.75; ImMin=-1.75; ImMax=1.75; maxIter=100;
case 29, jakiZbior=['Julia']; pot=2; c= i*0.95; rGran=2;
         ReMin=-1.75; ReMax=1.75; ImMin=-1.75; ImMax=1.75; maxIter=100;
case 30, jakiZbior=['Julia']; pot=2; c=-0.11-i*0.979; rGran=2;
         ReMin=-1.75; ReMax=1.75; ImMin=-1.75; ImMax=1.75; maxIter=100;
case 31, jakiZbior=['Julia']; pot=2; c=-0.84+i*0.19; rGran=2;                % rys. 10.6
         ReMin=-1.75; ReMax=1.75; ImMin=-1.75; ImMax=1.75; maxIter=100;
case 32, jakiZbior=['Julia']; pot=2; c=-0.12256117-i*0.74486177; rGran=2;    % rys. 10.5
         ReMin=-1.75; ReMax=1.75; ImMin=-1.75; ImMax=1.75; maxIter=100;      % rys. 10.4
case 33, jakiZbior=['Julia']; pot=2; c= 0.36+i*0.6; rGran=2;
         ReMin=-1.75; ReMax=1.75; ImMin=-1.75; ImMax=1.75; maxIter=100;
```

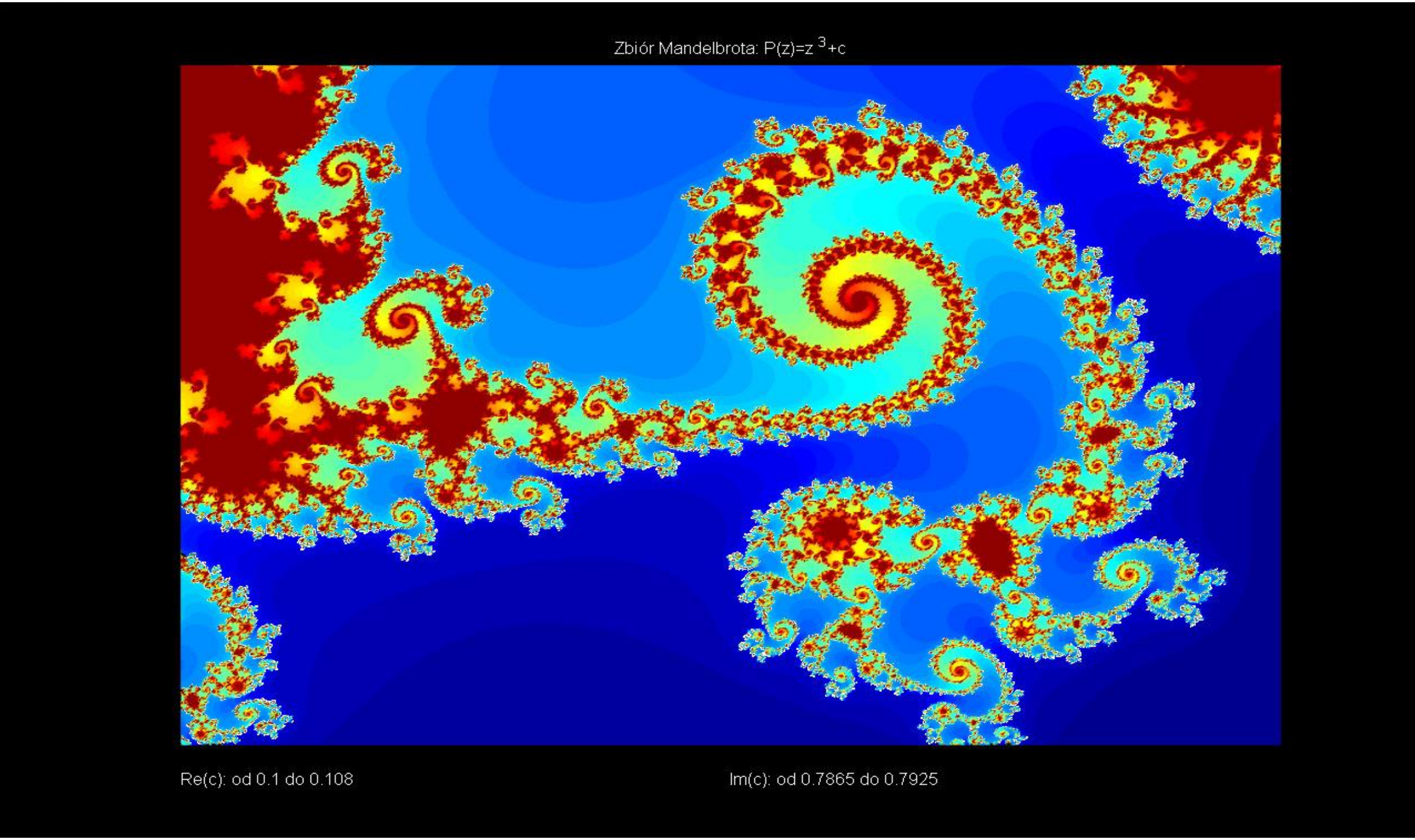
przykład 8. fraktale



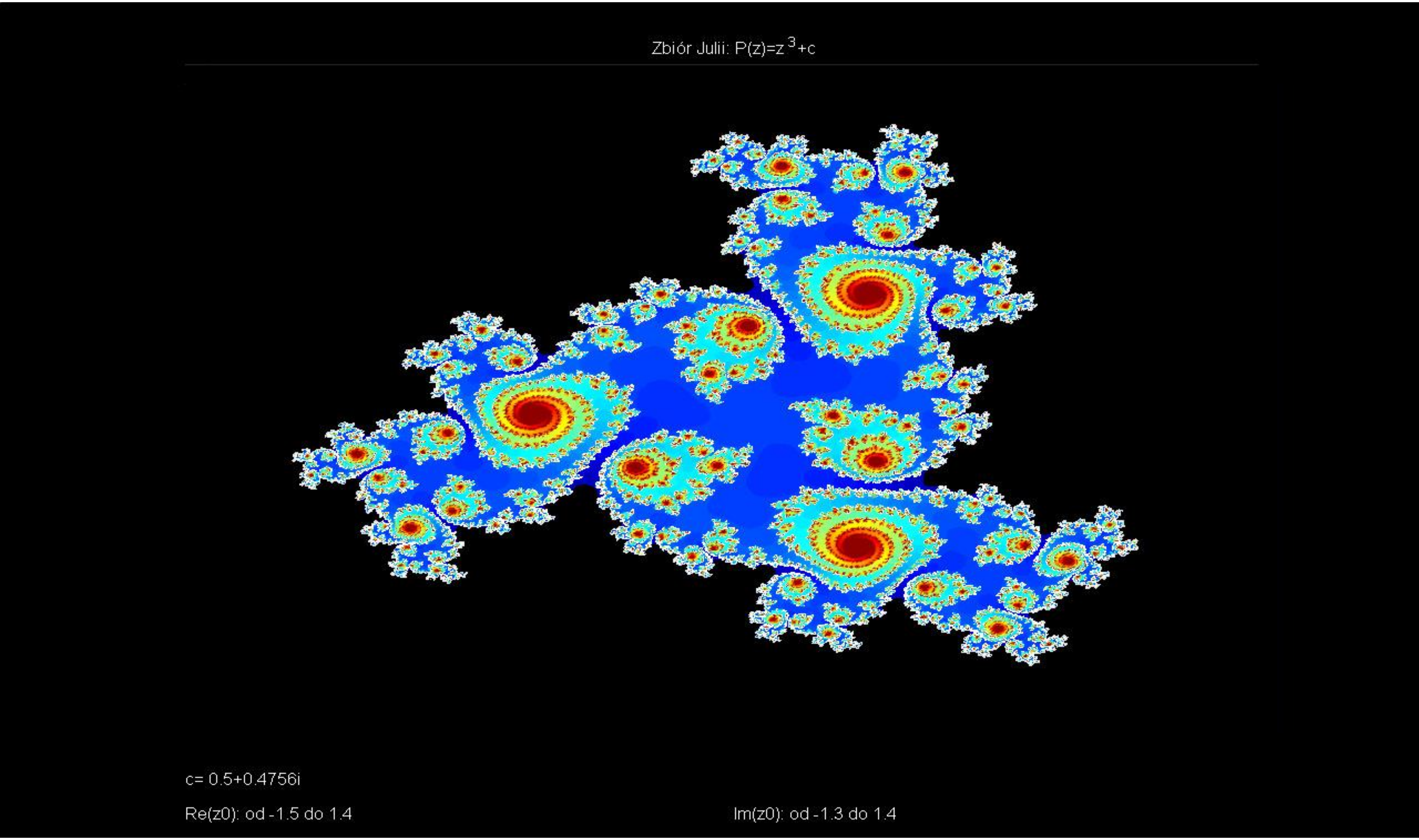
przykład 8. fraktale



przykład 8. fraktale



przykład 8. fraktale



przykład 8. fraktale