Rysowanie zbiorów Mandelbrota i Julii

Trochę teorii

Dysponując implementacją liczb zespolonych, możemy zrobić z nich bardzo efektowny użytek — narysujemy fraktal, a konkretnie słynny zbiór Mandelbrota. Jest to zbiór takich liczb zespolonych c, dla których każdy wyraz ciągu zespolonego $\{z_n\}$ określonego w następujący sposób

$z_0 = 0$

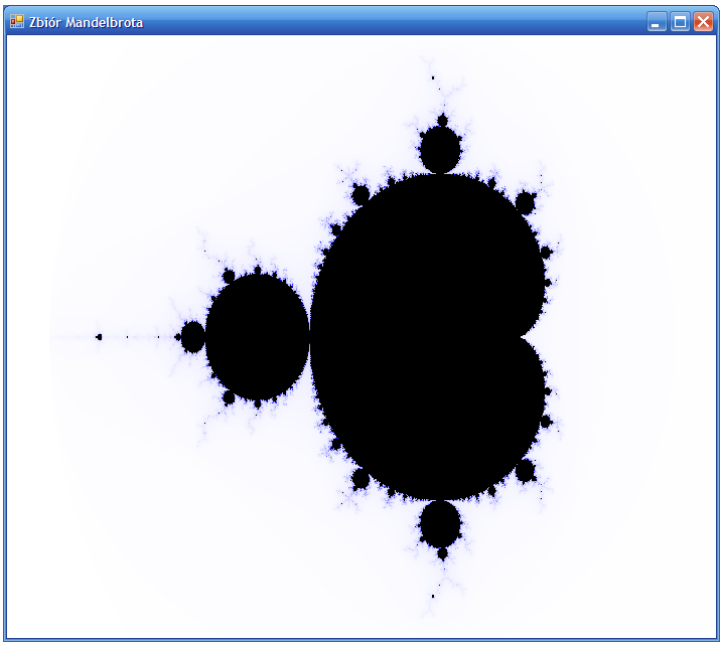
$z_1 = z_0^2 + c$

...

$z_n = z_{n-1}^2 + c$

ma promień mniejszy od 2 (jak pamiętamy z pierwszej części rozdziału, promień liczby zespolonej to pierwiastek kwadratowy sumy kwadratów części rzeczywistej i urojonej). Innymi słowy, dla liczb c należących do tego zbioru spełniony jest warunek $|z_n| < 2$ przy dowolnie dużej wartości indeksu n.

Liczby rzeczywiste mogą być umieszczone na jednej osi. Liczby zespolone wymagają płaszczyzny. Oś pionowa reprezentuje wówczas część urojoną, a pozioma — rzeczywistą. W ten sposób każdy punkt na płaszczyźnie odpowiada jednej liczbie zespolonej o wyznaczonych przez współrzędne punktu części rzeczywistej i urojonej. Gdy chcemy graficznie zaprezentować zbiór Mandelbrota, możemy zaznaczyć czarnym kolorem punkty odpowiadające tym liczbom zespolonym c, które należą do zbioru (rysunek 1). Pozostałe zostawiamy białe. Jeżeli chcemy ów zbiór narysować na powierzchni formy, ilość liczb zespolonych, które musimy przetestować, równa jest ilości pikseli, jakie zawiera obszar użytkownika tej formy.



Rysunek 1. Cały zbiór Mandelbrota

Zauważmy jednak, że zgodnie z definicją, żeby sprawdzić, czy liczba zespolona należy do zbioru Mandelbrota, trzeba wykonać nieskończoną ilość iteracji. W każdej obliczamy następny wyraz ciągu z_n i sprawdzamy, czy i on nie ma promienia większego od 2. Tego nie potrafią nawet najszybsze komputery. Dlatego, tworząc rysunek, ograniczymy liczbę iteracji do — powiedzmy — 255 i odwrócimy pytanie: będziemy szukać tych liczb c, o których wiemy, że nie należą do zbioru. Liczby c, dla których $|z_{255}| < 2$, zaznaczamy na rysunku kolorem czarnym jako rokujące nadzieję, że są elementem zbioru Mandelbrota (tego — oczywiście — nie możemy być do końca pewni bez kontynuowania testu), a pozostałe zaznaczamy na biało. Aby dodatkowo uatrakcyjnić rysunek, można zamiast nudnej bieli pokolorować te piksele, które nie należą do zbioru, uzależniając ich barwę od ilości iteracji, po których promień $|z_n|$ stał się większy od 2.

Nasz algorytm (rysunek 2) składający się z dwóch pętli (zewnętrzna po wszystkich wartościach zmiennej `c` i wewnętrzna z indeksem `indeks`) można zatem zapisać w następującej sekwencji instrukcji (por. przedstawiony na rysunku 2 schemat blokowy tego algorytmu):

(1) Wybieramy wartość zmiennej `c` odpowiadającą kolejnemu pikselowi rysunku (część rzeczywista ustalana na podstawie współrzędnej poziomej, urojona — pionowej).

Inicjujemy zmienną `z` wartością 0.

Inicjujemy zmienną przechowującą liczbę iteracji `indeks` wartością 0.

(2) Sprawdzamy, czy ilość iteracji przekroczyła 255.

Jeżeli tak, rysujemy czarny piksel i przechodzimy do punktu (1), wybierając nową wartość zmiennej `c`.

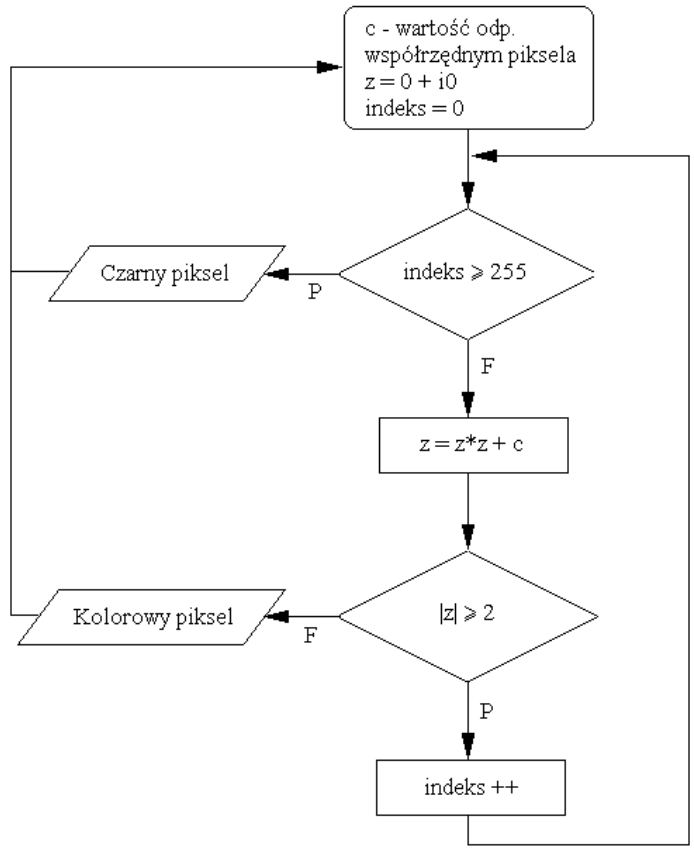
(3) Obliczamy nową wartość zmiennej `z` (kolejny wyraz ciągu z_n), przypisując jej wartość `z*z+c`.

(4) Sprawdzamy, czy `Abs(z)>=2`.

Jeżeli warunek jest prawdziwy, kolorujemy piksel w zależności od ilości iteracji (wartość zmiennej `indeks`) i przechodzimy do punktu (1), wybierając kolejną wartość zmiennej `c`.

Jeżeli warunek nie jest spełniony, zwiększamy wartość `indeks` o jeden i przechodzimy do punktu (2).

(5) Algorytm kończy się, gdy przetestowane zostaną wszystkie liczby `c` odpowiadające powierzchni rysunku.



Rysunek 2. Schemat blokowy przedstawiający algorytm testowania, czy liczba c odpowiadająca punktowi na rysunku należy do zbioru Mandelbrota