Západočeská univerzita v Plzni
Fakulta aplikovaných věd
KIV/KPG

Fraktály

Pavel Zelenka
A16B0176P
zelenkap@students.zcu.cz

4. května 2018

1 Zadání

Zadáním úkolu je vytvoření nového programu nebo rozšíření programu ze cvičení, tak aby vykresloval fraktály, které nebyly na cvičení.

2 Analýza problému

Fraktál je seběpodobný útvar, to znamená, že lze pozorovat stále opakující se tvar. Mezi známe fraktály patří Sierpińského koberec, Sierpińského trojúhelník či křivka vyplňující prostor. V této práci se budu zabývat fraktály zvanými **Hexaflake**, **Sierpińského šestiúhelník**, **T-Square** a **Hilbertovou křivkou** vyplňující prostor.

2.1 Hexaflake a Sierpińského šestiúhelník

Hexaflake je iterativně konstruovaný fraktál skládající se ze šestiúhelníků. V nulté počáteční iteraci existuje jeden šestiúhelník, který se po první iteraci rozdělí na 7 menších šestiúhelníků, které jsou uvnitř plochy původního šestiúhelníku. Každou další iteraci se každý šestiúhelník znovu rozdělí na 7 menších. Celkový počet šestiúhelníků je tedy 7^{n-1}. Strana každého nového šestiúhelníků odpovídá $\frac{1}{3}$ velikosti strany předchozího šestiúhelníku.

Sierpińského šestiúhelník se implementuje obdobně s rozdílem, že se šestiúhelník rozděluje jen na 6 menších šestiúhelníků. Velikost strany zůstává stejná jako u Hexaflake, ale vypustí se prostřední šestiúhelník, tedy střed šestiúhelníku není v další iteraci vyplněn.

2.2 Hilbertova křivka

Hilbertova křivka lze implementovat iterativním způsobem, kdy se bude využívat binární reprezentace indexů bodů. Algortmus bude v cyklu provádět bitový posun a budou se zjišťovat vždy poslední 2 bity, dle kterých se bude měnit pozice.

2.3 T-square

T-square je iterativně konstruovaný fraktál skládající se ze čtverců. V nulté počáteční iteraci existuje jeden čtverec. V první iteraci vzniknou 4 nové čtverce o poloviční velikosti strany, každý z těchto čtverců bude mít střed na jedné z hran čtverce z předchozí iterace.

3 Popis řešení

Vykreslování probíhá ve třídě *Drawing*. Algortitmy fraktálů se nacházejí v balíčku *fractal*.

3.1 Hexaflake

Hexaflake je reprezentován třídou *Hexaflake.java*, v metodě *draw()* se provede výpočet středu okna a umístí se do tohoto místa počáteční bod. Velikost strany počátečního šestiúhelníku je pevně daná na 1/3 šířky nebo výšky okna, dle menšího rozměru. Dle požadovaného počtu iterací se určí počet šestiúhelníků po poslední iteraci, tento údaj slouží pro duhové barvy.

Metoda *calculateHexagon(double hw, double hh, double side)* provede výpočet hran šestiúhelníku.

Input: šířka_okna, výška_okna, strana, strany
první_bod = null;
předchozí_bod = null;
for *int i = 0; i ¡ 6; i++* **do**
    x_souřadnice = šířka_okna/2 + strana + cos($i \cdot 2 \cdot pi/6$);
    y_souřadnice = výška_okna/2 + strana + sin($i \cdot 2 \cdot pi/6$);
    if *předchozí_bod != null* **then**
        nový_bod = bod se souřadnicemi x_souřadnice a y_souřadnice;
        strany[i-1] = úsečka mezi body předchozí_bod a nový_bod;
        if *i == 5* **then**
            strany[i] = úsečka mezi body nový_bod a první_bod;
        end
    else
        první_bod = nový_bod;
    end
    předchozí_bod = nový_bod;
end

Algorithm 1: Vytvoření šestiúhelníku

Vykreslování a iterace probíhá v metodě *drawHexagon(Point center, double side, int iteraction, int max).* Metoda má jako parametry bod ve středu šestiúhelníku, velikost strany, pořadí současné iterace a celkový počet iterací.

3.2 Hilbertova křivka

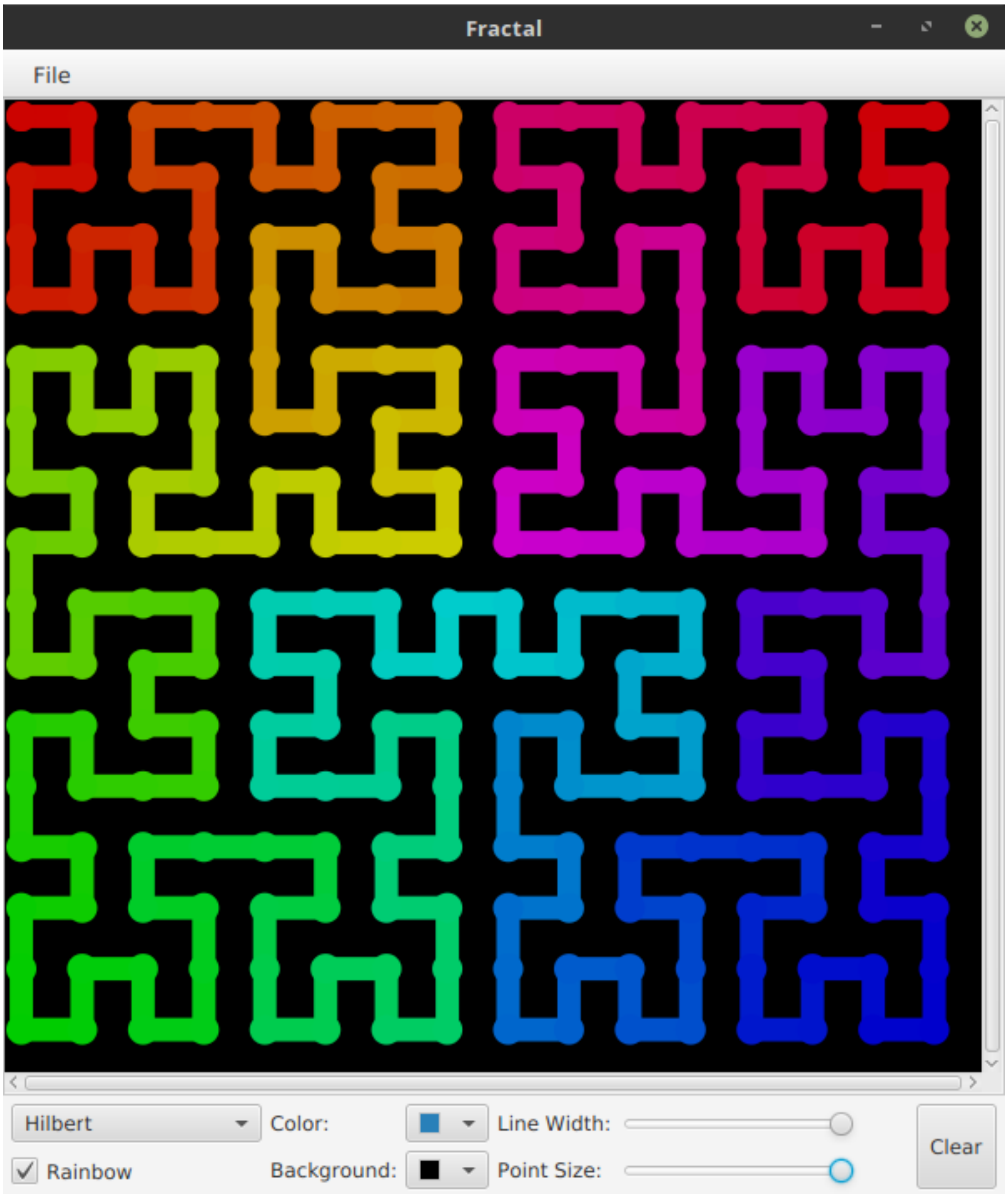
V práci je implementován algoritmus, který index bodu Hilbertovy křivky převádí do kartézské soustavy souřadnic bez použití rekurze. Algoritmus předpokládá, že bod s indexem 0 je na pozici $[0, 0]$. V první iteraci existují právě 4 body a počáteční křivka je pevně daná.

Pozice následujících bodů se dopočítávají podle indexu. Poslední 2 bity označují umístění bodu v rámci čtveřice bodů (tj. tvar křivky v první iteraci), následující dvojice bitů označuje umístění předešlé čtveřice bodů v rámci šestnáctice bodů (tj. křivka v druhé iteraci).

Obrázek 1: Indexy bodů v třetí iteraci

4 Uživatelská dokumentace

Spuštění aplikace se provede souborem `Fractal.jar`, který se nachází ve složce *App*. Po spuštění je nutné vybrat v dolním panelu křivku. Iterace se provede kliknutím na libovolné místo ve vykreslovací oblasti.



Obrázek 2: Okno aplikace

5 Závěr

U aplikace není úplně vyřešeno omezení na určitý počet iterací fraktálu a proto u vyšší iterace může dojít ke zpomalení počítače či k vyčerpání dostupné paměti a pádu aplikace. Každá třída reprezentující fraktál má nastavený strop počtu iterací, ovšem problém je závislý i na dalších faktorech a nepodařilo se mi jej spolehlivě vyřešit.

6 Reference

Hexagon – Wolfram MathWorld. [online].
Dostupné z: en.wikipedia.org/wiki/Hexaflake
Hexaflake – Wikipedia. [online].
Dostupné z: en.wikipedia.org/wiki/Hexaflake
Iterative algorithm for drawing Hilbert curve – Marcin Chwedczuk. [online].
Dostupné z: marcin-chwedczuk.github.io/iterative-algorithm-for-drawing-hilbert-curve