

FRACTAIS E FUNÇÕES RECURSIVAS

TAPETE DE SIERPINSKI EM JAVA

Cleverton Hoffmann, Kael Lucas Pinheiro Cordeiro, Juliano Tonizeti Brignoli

Instituto Federal Catarinense Campus Rio do Sul

clevertonhoffmann@gmail.com.br, kaellucas@hotmail.com.br, juliano.brignoli@ifc.edu.br

Abstract.

The work deals with the use of recursive functions for the construction of fractals, seeking to use concepts learned from recursive functions during the discipline of Discrete Mathematics of the Higher Course of Computer Science. The fractal implemented Sierpinski Carpet, uses the Java programming language and its drawing libraries, applying the use and creation of the required recursive function.

Key-words: *Recursive Functions; Fractals; Computer Science.*

Resumo.

O trabalho aborda sobre o uso de funções recursivas para construção de fractais, buscando usar conceitos aprendidos de funções recursivas durante a disciplina de Matemática Discreta do Curso Superior de Ciências da Computação. O fractal implementado Tapete de Sierpinski, utiliza da linguagem de programação Java e suas bibliotecas de desenho, aplicando o uso e a criação da função recursiva necessária.

Palavras-chave: *Funções Recursivas; Fractais; Ciências da Computação.*

1. Introdução

Uma função recursiva visa a execução dela mesma várias vezes, conforme o passo de recursividade. Segundo Bittencourt (2013, p.5) em programação a função recursivas é “Um método/função que invoca o próprio método/função”. Fractais são um exemplo da aplicação de funções recursivas. São figuras geométricas produzidas por meio de equações matemáticas as quais são interpretadas pelo computador e desenhadas em formas e cores. Os fractais recursivos possuem a auto similaridade, em outras palavras um cópia do desenho anterior mas em tamanho reduzido. (PINTO, 2015).

Existem diversos tipos de fractais existentes, tanto na natureza, como criadas computacionalmente. No trabalho será explorado o estudo específico do fractal do fractal recursivo que chama-se Tapete de Sierpinski. A implementação do mesmo deu-se por meio do uso da linguagem de programação JAVA, com a IDE de programação NetBeans 8.2. Na qual foram aplicados conceitos de função recursiva, usando a biblioteca gráfica para desenho do Java Graphics2D e a visualização por meio da biblioteca JFrame e JPanel. Para a construção da interface gráfica utilizou-se de artigos e exemplos de códigos já criados. Contudo a implementação da função recursiva

deu-se por meio da exploração do funcionamento da biblioteca gráfica do Java buscando a criação das imagens encontradas de fractais na internet.

O trabalho desenvolveu-se ao longo do primeiro semestre letivo de 2019, na disciplina de Matemática Discreta, visando a aplicabilidade do uso de funções recursivas, seu uso e aplicabilidades dos algoritmos recursivos exemplificando com os fractais.

2. Análise e Discussão

Atualmente existem diversos tipos de fractais, principalmente na natureza. Alguns exemplos da natureza são os caracóis, as folhas de samambaia, as árvores, a cadeia de reprodução vegetal, entre diversos outros. (Mendonça, 2016). Para a implementação do fractal, inicialmente foram feitos alguns testes, que geraram diversas imagens, podemos assim dizer imagens simples que mostram como o uso de funções recursivas pode ser variado, e que a construção de fractais se diversifica dependendo da função utilizada. Na Figura 1 encontra-se algumas figuras geradas com os testes.

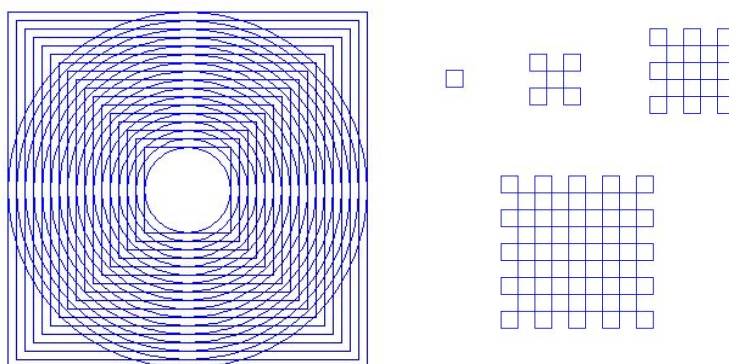


Figura 1 - Testes de implementação de fractais com função recursiva.

Fonte: Elaboração dos autores.

Além dos testes buscava se implementar outros fractais, porém sem sucesso optou-se pela implementação do fractal recursivo de Tapete de Sierpinski, por sugestão do professor. O qual foi implementado com um pouco de dificuldade, pois para a criação do fractal baseou-se na figura exposta na internet e a necessidade de compreensão do funcionamento da biblioteca gráfica do Java.

Inicialmente criou-se uma interface gráfica usando o JFrame, definindo alguns atributos como o altura da tela e largura conforme linha 2 e 3 da Tabela 1, como também a quantidade máxima de recursão que o fractal deve ter, bem como a cor inicial. Também a partir da linha 15 temos a implementação do botão de decrease, que é responsável por diminuir a variável nível possibilitando assim a interação com o fractal por níveis.

Tabela 2 - Parte do código principal do painel de visualização usando o JPanel.

Fonte: Elaboração dos autores.

```
1 public class Fractal extends JFrame{
2     private final int WIDTH = 1000; // define a largura de GUI
3     private final int HEIGHT = 600; // define a altura de GUI
4     private final int MIN_LEVEL = 0, MAX_LEVEL = 6;
5     private Color color = Color.BLUE;
6
7     private JButton increaseLevelJButton, decreaseLevelJButton;
8     private JLabel levelJLabel;
9     private FractalJPanel drawSpace;
10    private JPanel mainJPanel, controlJPanel;
11
12    // configura a GUI
13    public Fractal(){
14        super( "Fractal" );
15
16    ....
17    decreaseLevelJButton = new JButton( "Decrease Level" );
18    controlJPanel.add( decreaseLevelJButton );
19    decreaseLevelJButton.addActionListener(
20        new ActionListener(){ // classe interna anônima
21            // processa o evento decreaseLevelJButton
22            public void actionPerformed((ActionEvent event) ){
23                int level = drawSpace.getLevel();
24                level--; // diminui o nível por u
25                // modifica o nível se possível
26                if (( level >= MIN_LEVEL ) && ( level <= MAX_LEVEL )){
27                    levelJLabel.setText( "Level: " + level );
28                    drawSpace.setLevel( level );
29                    repaint();
30                } // fim do if
31            } // fim do método actionPerformed
32        } // fim da classe interna anônima
33    ); // fim de addActionListener
```

Quanto a classe responsável pela função recursiva, ela também contém atributos que definem o tamanho do JPanel. A cor e o nível de recursão são passados pelo construtor da classe. A função de recursão utiliza da função de desenho gráfico `g.fillRect(xA, yA, xB, yB)` que desenha um retângulo ou quadrado de acordo com os pontos passados como parâmetros, assim as coordenadas `xA` e `yB` definem o ponto em que a figura criada vai se localizar na tela, e o `xB` a largura do quadrado e `yB` a altura do mesmo. Assim inicialmente no nível 0 foi criado uma figura de um quadrado e na chamada da função recursiva o desenho do mesmo no nível 1 conforme Tabela 3 linha 4, em seguida no nível 1, o quadrado vazado linha 6 á 9, e em seguida as cópias necessárias, ou passos de recursão para a geração completa do fractal Tapete de

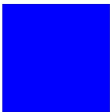
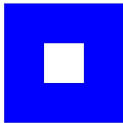
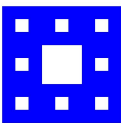
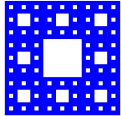
Sierpinski. A primeira chamada da função recursiva serve para o desenho da figura anterior, e assim os demais passos para o desenho de cada quadrante contando que o Tapete de Sierpinski tem oito quadrantes descontando o quadrante central vazado conforme os passos de recursão da linha 13 a 20.

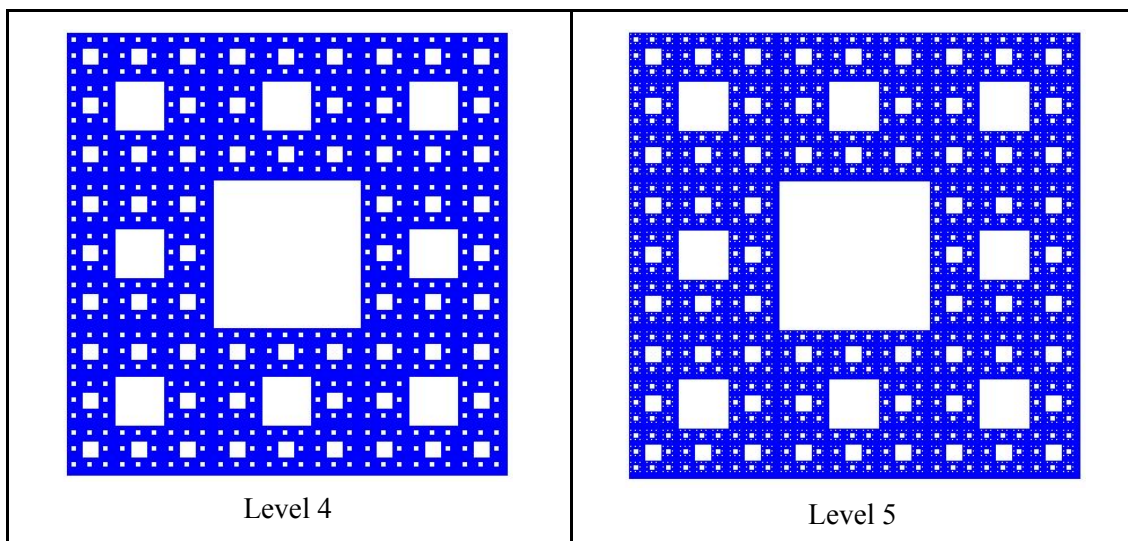
Tabela 3 - Função recursiva para desenho do fractal.
Fonte: Elaboração dos autores.

1	public void drawFractal(int level, int xA, int yA, int xB, int yB, Graphics2D g) {
2	// caso básico: desenha o quadrado inicial com um quadrado no meio
3	if (level == 0) {
4	g.fillRect(xA, yA, xB, yB);
5	} else if (level == 1){
6	g.setColor(Color.BLUE);
7	g.fillRect(xA, yA, xB, yB);
8	g.setColor(Color.WHITE);
9	g.fillRect(xA + (xB) / 3, yA + (yB) / 3, xB / 3, yB / 3);
10	} else{
11	//Passo de recursão
12	drawFractal(level - 1, xA, yA, xB, yB, g);
13	/*P1*/ drawFractal(level - 1, xA, yA, xB / 3, yB / 3, g);
14	/*P2*/ drawFractal(level - 1, xA + ((xB) / 3), yA, xB / 3, yB / 3, g);
15	/*P3*/ drawFractal(level - 1, xA + ((xB) / 3) * 2, yA, xB / 3, yB / 3, g);
16	/*P4*/ drawFractal(level - 1, xA, yA + ((yB) / 3), xB / 3, yB / 3, g);
17	/*P5*/ drawFractal(level - 1, xA + ((xB) / 3) * 2, yA + ((yB) / 3), xB / 3, yB / 3, g);
18	/*P6*/ drawFractal(level - 1, xA, yA + ((yB) / 3) * 2, xB / 3, yB / 3, g);
19	/*P7*/ drawFractal(level - 1, xA + ((xB) / 3), yA + ((yB) / 3) * 2, xB / 3, yB / 3, g);
20	/*P8*/ drawFractal(level - 1, xA + ((xB) / 3) * 2, yA + ((yB) / 3) * 2, xB / 3, yB / 3, g);
21	} // fim do else
22	} // fim do método drawFractal

Podemos ver a construção gráfica do fractal passo a passo na Tabela 4 de Figuras, o tamanho dos últimos níveis foi aumentado a escala para melhor visualização, porém o tamanho das figuras originais é a mesma.

Tabela 4 - Fractal Tapete de Sierpinski. Fonte: Elaboração dos autores.

 <p>Level 0</p>	 <p>Level 1</p>	 <p>Level 2</p>	 <p>Level 3</p>
--	--	--	--



3. Conclusão

O algoritmo implementado possibilitou o aprimoramento de conhecimentos sobre as bibliotecas gráficas e o funcionamento de funções recursivas, bem como o funcionamento dos fractais recursivos.

4. Referências

BITTENCOURT, João Ricardo. **Recursividade**. 2013. Disponível em: <<http://professor.unisinos.br/lttonietto/tsi/edc/recursividade.pdf>>. Acesso em: 01 jul. 2019.

PINTO, Fábio Lima. **O uso de recorrências e do raciocínio recursivo no ensino médio**. 2015. Disponível em: <<https://repositorio.ufba.br/ri/bitstream/ri/22971/1/disserta%C3%A7%C3%A3o%20f%C3%A1bio%20-%20digital.pdf>>. Acesso em: 01 jul. 2019.

MENDONÇA, Fernando Antônio Cavalcante. **Aplicações da Geometria Fractal: uma proposta didática para o Ensino Médio**. 2016. DISSERTAÇÃO DE MESTRADO. Disponível em: <http://www.repositorio.ufal.br/bitstream/riufal/2412/1/Aplica%C3%A7%C3%B5es%20da%20geometria%20fractal_%20uma%20proposta%20did%C3%A1tica%20para%20o%20ensino%20m%C3%A9dio.pdf>. Acesso em: 01 jul. 2019.