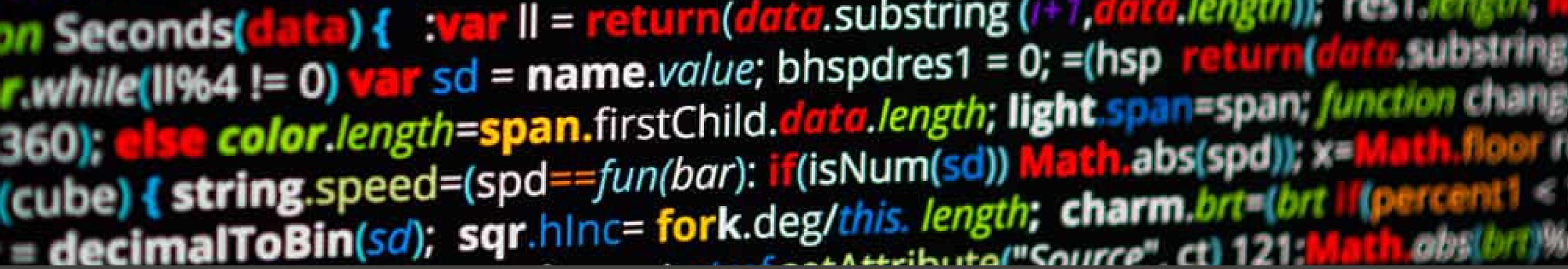

PROCESSAMENTO DE IMAGEM

JOGO DA VIDA

Vitória Maria Bezerra
Engenharia biomédica

Problema??

Jogo da vida é um autómato celular desenvolvido pelo matemático britânico John Horton Conway em 1970. É o exemplo **mais bem conhecido** de autômato celular.

- Imagem bidimensional;
- Jogo sem jogador (podendo ter entrada ou não);
- Pixel = Célula;
- Estados: Viva, Morta ou sem vida;
- Conectividade 8;
- As regras aplicadas a cada nova "geração"

Regras adaptadas

- Qualquer célula viva com menos de dois vizinhos vivos morre de solidão;
- Qualquer célula viva com mais de três vizinhos vivos morre de superpopulação;
- Qualquer célula morta com três ou mais de cinco vizinhos vivos se torna uma célula viva;
- Qualquer célula viva com dois ou três vizinhos vivos continua no mesmo estado para a próxima geração.
- Cada célula tem sete número de vidas. Isto é, cada célula viva poderá morrer até 7 vezes. Quando ele está em situação de morte, ela morrerá e quando encontrar as condições de nascimento, nascerá novamente enquanto tiver vidas suficientes.

Algoritmo

- Programa em Java com o auxilio do ImageJ

- Pedir entrada do cpf ao usuário,qtd de vida e interações.
- Calcular a cor da célula viva, e também a cor complementar para celula sem vida.
- Criar uma imagem nova com a qtdade de gerações.
- Criar o padrão aleatório com a porcentagem dada a partir dos dígitos do cpf.
- Pintar a imagem (aleatoriamente) com a porcentagem estabelecida.
- Varrer a imagem completa.
- Fazer a análise para cada caso .
- Plotar o gráfico

Código

```
// CRIANDO UMA IMAGEM COM O PERCENTUAL DESEJADO
size = (w * h); // A dimensão da imagem é altura vezes largura
image = NewImage.createRGBImage("Jogo da vida", w, h, slices, NewImage.FILL_WHITE); // O argumento é (title, int
                                                                                     // width, int height, int
                                                                                     // slices, int options)
ims = image.getImageStack(); // Passa a imagem para o stack para poder editar cada imagem, ja que imagem é um
                             // objeto que não pode ser editado
image.show(); // Mostra imagem na tela com a quantidade de frames

// PREENCHENDO ALEATORIAMENTE O MAPA INICIAL
Random ran = new Random(); // Criar um objeto aleatório
int i = 0; // inicia o contador com 0
int perc1 = (perc * size) / 100; // Para obter a porcentagem, iremos multiplicar a porcentagem(/100)
                                 // multiplicado pela qtd de pixels
imp1 = ims.getProcessor(1); // A variavel imp irá acessar o meu processor apenas para imagem 1.

do {
    xran = ran.nextInt(w); // retorna um valor inteiro de 0 até w para x
    yran = ran.nextInt(h); // retorna um valor inteiro de 0 até h para y
    aux = imp1.getPixel(xran, yran); // usamos uma variável auxiliar para pegar o pixel que já foi escolhido
                                     // aleatoriamente
    if (aux == dead) { // Para certificar que o mesmo pixel não seja pintado mais de uma vez, vamos
                       // restriguir a pintura para apenas celulas mortas, ou seja, as brancas
        imp1.set(xran, yran, alive); // Essa função irá pintar a celula escolhida com a cor do meu cpf
        i++; // o contador irá servir para controlar a estrutura de repetição
    }
} while (i < perc1); // Ele parará de pintar as celulas até que chegue na porcentagem requerida
```

Código

```
// CRIANDO VETORES PARA PLOTAR OS GRAFICOS
double[] vectviva = new double[slices]; // vetor para guardar numero de celulas vivas por geração
double[] vectmorto = new double[slices];// vetor para guardar numero de celulas vivas por geração
double[] vectsemvida = new double[slices];// vetor para guardar numero de celulas vivas por geração
double[] geracoes = new double[slices];// vetor que armazena as gerações
// contador para celulas sem vida.

// ESTRUTURA DE REPETIÇÃO PARA VARRER AS IMAGENS
for (int G = 1; G < slices; G++) {// Fazendo um for para rodar varias gerações de imagens - GIF
    imp1 = ims.getProcessor(G + 1);// O stack vai acessar o proecessor de todas as outras imagens, a partir da
                                   // segunfa
    // System.out.println(G) para teste;
    geracoes[G] = G;
    int countviva = 0;// contador para celulas vivas
    int countmorto = 0;// contador para celulas mortas
    int countsemvida = 0;
    // ESTRUTURA DE REPETIÇÃO PARA VARRER OS PIXELS
    for (x = 0; x < w; x++) {// Varrendo as linhas até w-1
        for (y = 0; y < h; y++) {// Varrendo as colunas até h-1
            // System.out.println(vidas[x][y]);

            if (ims.getProcessor(G).getPixel(x, y) == alive) {
                countviva++;
            }
            if (ims.getProcessor(G).getPixel(x, y) == dead) {
                countmorto++;
            }
            if (ims.getProcessor(G).getPixel(x, y) == lifeless) {
                countsemvida++;
            }

            vectviva[G] = countviva;// armazena o numero de vivos no vetor para cada geração
            vectmorto[G] = countmorto;// armazena o numero de morto no vetor para cada geração
            vectsemvida[G] = countsemvida;// armazena o numero de celulas sem vida no vetor para cada geração
```

Código

```
if (ims.getProcessor(G).getPixel(x, y) == alive) {// Se a celula a ser analisada os vizinhos for
                                                  // viva entrará no if e o contador será iniciado
                                                  // com -1;

    count1 = -1;
    // ESTRUTURA DE REPETIÇÃO PARA VARRER OS VIZINHOS- P(x) e Q(y)
    for (int x1 = x - 1; x1 <= x + 1; x1++) {// Sempre adicionando 1 no eixo x para mais e para
                                             // menos.
        for (int y1 = y - 1; y1 <= y + 1; y1++) {// Sempre adicionando 1 no eixo x para mais e para
                                                 // menos.
            p = x1;// Não queremos que p mude para sempre o seu valor, por isso xi e y1 funcionam
                   // como uma variavel auxilar
            q = y1;
            if (p < 0) {// Condição1- Borda esquerda
                p = p + w; // apenas mais w porquq p=-1, Soma-se a largura
            }
            if (q < 0) {// Condição2- Borda superior
                q = q + h; // Soma-se a altura

            }
            if (q >= h) {// Condição 3- Borda Direita
                q = q - h;// Subtrai a altura
            }
            if (p > w) {// Condição 4- Borda inferior
                p = p - w;// Subtrai a largura.

            }

            if (p >= 0 && p < w && q >= 0 && q < h) {// Sobrará apenas os pixels com conectavidade 8

                if (ims.getProcessor(G).getPixel(p, q) == alive) {
                    count1++;
```

Código

```
}
System.out.println(Arrays.toString(vectsemvida));
System.out.println(Arrays.toString(vectmorto));
System.out.println(Arrays.toString(vectviva));
System.out.println(Arrays.toString(geracoes));

Plot plotar = new Plot("Grafico de células sem vida", "Gerações", "Número de células");
plotar.setColor(new Color(255, 0, 0));// plota de vermelho as sem vida
plotar.add("Celula viva", geracoes, vectsemvida);
plotar.setColor(new Color(0, 0, 0));// plota de preto as vivas
plotar.add("Celula morta", geracoes, vectviva);
plotar.setColor(new Color(0, 0, 255));// plota de azul as mortas
plotar.add("Célula sem vida", geracoes, vectmorto);
plotar.show();
plotar.addLegend("Vermelho- Sem vidas / Preto -Vivas / Azul- Mortas ");
```

Desafios

- Utilização do Image Stack, Image Plus e Image Processor;
- Pintar a imagem aleatoriamente sem que o pixel fosse repetiodo;
- Varrer os vizinhos;
- Condições de borda;

Conclusões

- O programa implementou corretamente o código e efeitou todos os requisitos pedidos
- Para cpf com digitos >85, o jogo não funciona bem por ter muitas celulas pintadas inicialmente, causando uma alta mortalidade na geração seguinte.
- Imagens com muitos pixels e muitas gerações torna o programa lento.

Referências

- https://www.devmedia.com.br/forum/matriz-e-sua-declaracao-e-inicializacao/56710
- https://www.geeksforgeeks.org/java-util-random-nextint-java/
- Jogo da Vida – Wikipedia. Endereço: http://pt.wikipedia.org/wiki/Jogo_da_vida.
- Videos de Harison

Obrigada!

Vitória Bezerrra
vitoria0406@gmai.com

Nunca desista!