 UEPB	UNIVERSIDADE ESTADUAL DA PARAÍBA CENTRO DE CIÊNCIAS E TECNOLOGIA CURSO DE BACHARELADO EM CIÊNCIA DA COMPUTAÇÃO	
ATIVIDADE DE ESTUDO IV^{ANO} 2021.1		
CURSO	Ciência da Computação	
DISCIPLINA	Linguagem de Programação II	
PROFESSOR TITULAÇÃO		
Jucelio Soares dos Santos Mestrado		
NOME	Lucas de Lucena Siqueira	
MATRÍCULA	201080354 CONCEITO	
DATA		

DESCRIÇÃO DA ATIVIDADE

1. Implemente o programa Labirinto, de acordo com a versão mostrada, só para apresentar o tabuleiro vazio. Tente não copiar o código mostrado, mas fazer sem olhar o material da aula. Relate o que você fez de diferente em termos de implementação ou se a sua implementação ficou igual à mostrada na aula.

R/ A diferença do código que fizemos para o demonstrado foi na utilização dos laços, acabamos utilizando um laço a mais, porém a solução que foi apresentada é muito mais interessante. Fora isso não criamos uma função para realizar o print do tabuleiro

```

1 package Atividade04;
2 import java.util.*;
3
4 /*
5  * Implemente o programa Labirinto, de acordo com a versão mostrada, só para apresentar o tabuleiro vazio.
6  * Tente não copiar o código mostrado, mas fazer sem olhar o material da aula.
7  * Relate o que você fez de diferente em termos de implementação ou se a sua implementação ficou igual à mostrada na aula.*/
8 */
9 public class Labirinto {
10     private static final int lado = 10;
11
12     public static void main (String [] args) {
13         Scanner entrada = new Scanner(System.in);
14         String tabuleiro[][] = new String[lado][lado];
15
16         //Montagem do tabuleiro
17         for (int i = 0; i < lado; i++) {
18             tabuleiro[i][0] = "+";
19             tabuleiro[i][lado - 1] = "+";
20             tabuleiro[0][i] = "+";
21             tabuleiro[lado - 1][i] = "+";
22         }
23         for (int i = 1; i < lado - 1; i++) {
24             for (int j = 1; j < lado - 1; j++) {
25                 tabuleiro[i][j] = " ";
26             }
27         }
28         //Print do tabuleiro
29         for (int i = 0; i < lado; i++) {
30             System.out.print("\n");
31             for (int j = 0; j < lado; j++) {
32                 System.out.print(tabuleiro[i][j] + " ");
33             }
34         }
35     }
36 }

```

2. Altere:

- a. O valor da constante representa o tamanho do tabuleiro. Relate o que houve com o tabuleiro mostrado pelo programa e explique como a alteração dessa constante afetou o resto do programa.

R/ Alterar a constante que é utilizada para definir as dimensões da matriz (tabuleiro) vai influenciar diretamente no tamanho da matriz (tabuleiro) em si.

- b. A implementação do programa Labirinto para apresentar o tabuleiro com as paredes internas (obstáculos), como mostrado nessa segunda parte da aula. Tente não copiar o código mostrado, mas fazer sem olhar o material da aula. Relate o que você fez de diferente em termos de implementação ou se a sua implementação ficou igual à mostrada na aula.

R/ Havíamos pensado na possibilidade de pôr o usuário para fazer a montagem dos obstáculos, mas ficaria algo extenso para a digitação. Gostamos da utilização da função "Math.random()" que foi aplicada no vídeo.

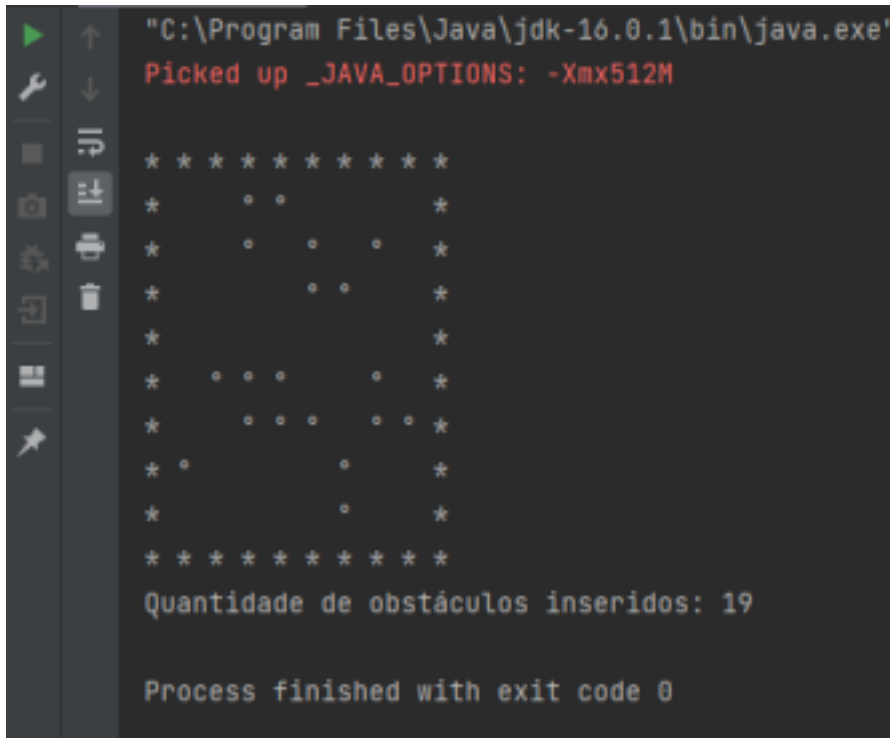
3. Execute várias vezes o programa Labirinto e relate se houve mudanças na posição e quantidade de obstáculos que formam as paredes internas do labirinto.

R/ Sim, exceto uma vez, todas as outras houveram mudanças nas posições dos

obstáculos e na quantidade dos mesmos.

4. Teste o uso de outros caracteres para as constantes usadas no programa e indique se você encontrou valores mais adequados que os mostrados na aula (deixam o tabuleiro mais bonito, são mais simples de serem visualizados etc.).

R/ A utilização do caractere “*” para montar as laterais do tabuleiro e o caractere “o” para os obstáculos deixa o mesmo muito mais bonito.



```
"C:\Program Files\Java\jdk-16.0.1\bin\java.exe"  
Picked up _JAVA_OPTIONS: -Xmx512M  
* * * * *  
*   o   o   *  
*   o   o   o   *  
*       o   o   *  
*       *   *   *  
*   o   o   o   *  
*   o   o   o   o   *  
* o       o   *  
*       o   *  
* * * * *  
Quantidade de obstáculos inseridos: 19  
Process finished with exit code 0
```

5. Altere o valor da constante PROBABILIDADE e relate o que acontece quando aumentamos ou diminuimos o seu valor.

R/ Quando o valor da constante PROBABILIDADE é diminuído é possível notar que a quantidade de obstáculos gerados no labirinto aumenta, já quando o valor da constante é aumentado a quantidade de obstáculos é diminuída.

6. Altere sua implementação do programa Labirinto para apresentar o tabuleiro com as posições de início e destino, como mostrado nesta parte da aula. Tente não copiar o código mostrado, mas fazer sem olhar o material da aula. Relate se você conseguiu fazer a implementação sem copiar, se você fez algo de diferente em termos de implementação ou se a sua implementação ficou igual à mostrada na aula.

R/ O código ficou um pouco diferente, porém segue a mesma finalidade. A diferença que optamos por permanecer é a respeito da geração aleatória do Início e do Destino do labirinto, para que ele fique mais dinâmico, fora essa utilizamos métodos e funções diferentes para realizar a construção do que foi proposto.

```

1 package Atividades04;
2 import java.util.*;
3 /**
4  * Implemente o programa Labirinto, de acordo com a versão mostrada, só para apresentar o tabuleiro vazio.
5  * Tente não copiar o código mostrado, mas fazer sem olhar o material da aula.
6  * Relate o que você fez de diferente em termos de implementação ou se a sua implementação ficou igual é mostrada na aula.
7  */
8 public class Questao02 {
9     private static final int Lado = 10;
10    private static final double probabilidade = 0.7;
11    private static String[][] tabuleiro;
12    private static int randomIniciol;
13    private static int randomColonsInicio;
14    private static int randomInioFim;
15    private static int randomColonsFim;
16
17    public static void main (String [] args) {
18        Scanner entrada = new Scanner(System.in);
19        tabuleiro = new String[Lado][Lado];
20
21        //Montagem do tabuleiro
22        construirTabuleiro();
23
24        //Inicio - Fim
25        inicioFimTabuleiro();
26
27        //Print do tabuleiro
28        imprimirTabuleiro();
29    }
30

```

Console:

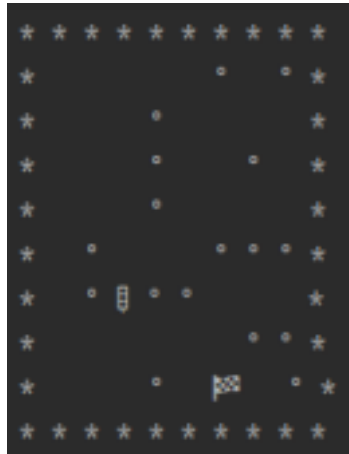
7. Execute várias vezes o programa Labirinto e relate se houve mudanças na posição dos pontos de início e destino do labirinto.

R/ Sim, sempre houve essa mudança, como é mostrado na questão anterior.

8. Teste:

- o uso de outros caracteres para as constantes de INICIO e DESTINO usadas no programa e indique se você encontrou valores mais adequados (para deixar o tabuleiro mais bonito, mais simples de ser visualizado etc. que os mostrados na aula).

R/ Utilizamos alguns emojis pra ficar mais interessante



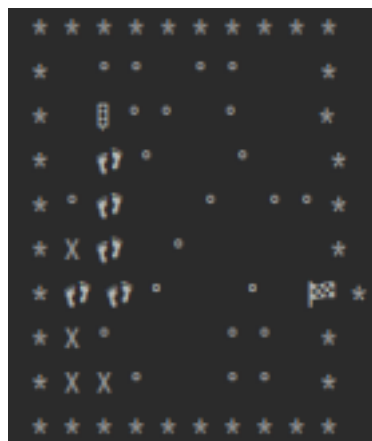
hehehe.

Semáforo: Início.

Bandeira: Destino.

- b. o uso de outros caracteres para as constantes de CAMINHO e SEM_SAIDA usadas no programa e indique se você encontrou valores mais adequados (para deixar o tabuleiro mais bonito, mais simples de ser visualizado etc.) que os mostrados na aula.

R/ Para sinalizar um CAMINHO utilizamos um emoji (pegada) também, ficou bem interessante.



9. Altere:

- a. A função "inicializarMatriz()" para posicionar o caractere de início na parte inferior à esquerda do tabuleiro, e para posicionar o caractere de destino na parte superior à direita do tabuleiro. Execute e confira se sua implementação está correta.

```

public static void iniciarTabuleiro() {
    //lateralis
    for (int i = 0; i < tamanho; i++) {
        tabuleiro[i][0] = parede_vertical;
        tabuleiro[i][tamanho - 1] = parede_vertical;
        tabuleiro[0][i] = parede_horizental;
        tabuleiro[tamanho - 1][i] = parede_horizental;
    }

    for(int i = 1; i < tamanho -1; i++) {
        for (int j = 1; j < tamanho -1; j++) {
            if (Math.random() > probabilidade) {
                tabuleiro[i][j] = obstaculo;
            }
            else {
                tabuleiro[i][j] = vazio;
            }
        }
    }

    linhaInicio = gerarNumero(1, tamanho/2-1);
    colunaInicio = gerarNumero(1, tamanho/2-1);
    tabuleiro[tamanho - 2][1] = inicio;
    linhaDestino = gerarNumero(tamanho/2, tamanho-2);
    colunaDestino = gerarNumero(tamanho/2, tamanho-2);
    tabuleiro[1][tamanho - 2] = destino;
}

```

- b. O tempo em "milissegundo"s do comando "Thread.sleep() para mais e para menos. Relate o que acontece com essas mudanças, e qual seria o melhor valor para você poder acompanhar a execução do algoritmo de busca.

R/ Caso o valor inserido seja aumentado, o tempo de espera entre a impressão dos tabuleiros após as ações de procura do destino será maior, caso seja diminuído, será menor. Achamos melhor deixar com o valor de 800 milissegundos.

10. Reflexão:

- a. Qual a estrutura de dados que se pode utilizar para representar o tabuleiro do jogo de labirinto?

R/ Uma matriz bidimensional.

- b. Como podemos preencher os obstáculos (paredes internas) no tabuleiro do jogo, de forma que toda vez que se jogue, um novo tabuleiro seja mostrado?

R/ Uma solução que foi vislumbrada, apesar de talvez não ser uma das mais eficientes, pode cumprir o proposto. É possível que cada tabuleiro seja armazenado em uma estrutura que servirá de referência para comparar com o novo tabuleiro gerado, que caso seja idêntico, é descartado e em seguida gerado um novo.

c. Qual a ideia do uso da recursão utilizada na programação do programa Labirinto.

R/ A de trazer uma solução mais eficiente e resumida à proposição do problema proposto de realizar os movimentos no tabuleiro até que se chegue ao resultado esperado.