

TESTE DE ESTÁGIO - ATECH

Explicação ex 1

No problema tive que encontrar o retângulo maior de “números 1” em uma matriz 2D qualquer.

Para resolver este problema, fiz uma classe MaximalRectangle com duas funções. A primeira com um método para calcular a maior área dentro da matriz.

```
1 public class MaximalRectangle { 2 usages
2     // Método que calcula a maior área
3     public int maximalRectangle(char[][] matrix) { 1 usage
4         if (matrix == null || matrix.length == 0) return 0;
5
6         int maxArea = 0;
7         int[] heights = new int[matrix[0].length];
8
9         for (char[] row : matrix) {
10             for (int i = 0; i < row.length; i++) {
11                 heights[i] = (row[i] == '1') ? heights[i] + 1 : 0;
12             }
13             maxArea = Math.max(maxArea, largestRectangleHistogram(heights));
14         }
15         return maxArea;
16     }
```

Nesse método, a função pega cada número e verifica se, em ordem, existe um retângulo.

E a segunda função para calcular o maior retângulo em histograma.

```
@ // Método para calcular maior retângulo em histograma
private int largestRectangleHistogram(int[] heights) { 1 usage
    java.util.Stack<Integer> stack = new java.util.Stack<>();
    int maxArea = 0;

    for (int i = 0; i <= heights.length; i++) {
        int currentHeight = (i == heights.length) ? 0 : heights[i];

        while (!stack.isEmpty() && currentHeight < heights[stack.peek()]) {
            int height = heights[stack.pop()];
            int width = stack.isEmpty() ? i : i - 1 - stack.peek();
            maxArea = Math.max(maxArea, height * width);
        }
        stack.push(i);
    }
    return maxArea;
}
```

Explicação EX 2

No segundo exercício criei um contador simples, que conta cada parênteses, se existir, com um se e senão.

```
1 public class ParenthesesChecker { 2 usages
2 // Método que verifica se os parênteses estão balanceados
3 @ public int isBalanced(String str) { 3 usages
4     int count = 0;
5
6     for (char c : str.toCharArray()) {
7         if (c == '(') {
8             count++;
9         } else if (c == ')') {
10             if (count == 0) return 0;
11             count--;
12         }
13     }
14
15     return count == 0 ? 1 : 0;
16 }
```

Por fim, criei um método principal, que “chama” as duas classes e “entregam” a elas um certo dado onde a função irá “analisar” e retornar o resultado.

```
public class Main {
    public static void main(String[] args) {
        System.out.println("Ex 1");
        char[][] matrix = {
            {'1','0','1','0','0'},
            {'1','0','1','1','1'},
            {'1','1','1','1','1'},
            {'1','0','0','1','0'}
        };

        MaximalRectangle obj = new MaximalRectangle();
        System.out.println("Maior área encontrada: " + obj.maximalRectangle(matrix));

        System.out.println("\nEx 2");
        ParenthesesChecker checker = new ParenthesesChecker();

        System.out.println(checker.isBalanced(str: "(hello (world))"));
        System.out.println(checker.isBalanced(str: "(hello (world))"));
        System.out.println(checker.isBalanced(str: "hello world"));
    }
}
```