



PABLO VASCONCELOS DA CRUZ

LABORATÓRIO DE ALGORITMO E ESTRUTURA DE DADOS II:
Prática 05 - Implementação TAD MatrizAdj

Professor: Thiago de Souza Rodrigues

Belo Horizonte
2022

1 - CLASSES

- XGrafo

```
1
2  public class XGrafo {
3
4  >    private static class XCiclo { ...
43
44    private int mat[][]; // pesos do tipo inteiro
45    private int numVertices;
46    private int pos[]; // posição atual ao se percorrer os adjs de um vértice v
47
48  >    public XGrafo(int numVertices) { // matriz (n x n) bits ...
61
62  >    public void insereAresta(int v1, int v2, int peso) { ...
65
66  >    public boolean existeAresta(int v1, int v2) { ...
69
70  >    public boolean listaAdjVazia(int v) { ...
76
77  >    public Aresta primeiroListaAdj(int v) { ...
83
84  >    public Aresta proxAdj(int v) { ...
96
97  >    public Aresta retiraAresta(int v1, int v2) { ...
106
107  >    public void imprime() { ...
125
126  >    public int numVertices() { ...
129
130  >    public void verificaCiclo() { ...
137  }
```

- XCiclo

```
4  private static class XCiclo {
5      private int[][] grafo;
6      private int numVertices;
7
8      public XCiclo(int[][] grafo, int numVertices) {
9          this.grafo = grafo;
10         this.numVertices = numVertices;
11     }
12
13     private boolean verifyPath(int linha[], int elementoInicio) {
14         for (int i = 0; i < numVertices; i++) {
15             if (linha[i] != 0) {
16                 if (i == elementoInicio) {
17                     return true;
18                 } else {
19                     return verifyPath(grafo[i], elementoInicio); // Trilha as ligacoes dos elementos do grafo a fim
20                                                                    // de encontrar um Vk = Vo
21                 }
22             }
23         }
24         return false;
25     }
26
27     public boolean isCicle() {
28         for (int i = 0; i < numVertices; i++) {
29             for (int j = 0; j < numVertices; j++) {
30                 if (grafo[i][j] != 0) { // Verifica a linha se o elemento do grafo faz
31                     boolean retornoLinha = verifyPath(grafo[j], i); // ligacao com algum outro elemento
32                     if (retornoLinha) {
33                         return retornoLinha;
34                     }
35                 }
36             }
37         }
38         return false;
39     }
40 }
```

2 - RESULTADOS

- Grafo A

```
-----  
  0 1 2 3 4 5 6 7  
0 0 1 0 1 1 0 0  
1 1 0 1 0 0 1 0  
2 0 1 0 1 0 0 1  
3 1 0 1 0 0 0 1  
4 1 0 0 0 0 1 0  
5 0 1 0 0 1 0 1  
6 0 0 1 0 0 1 0  
7 0 0 0 1 1 0 1  
  
0 grafo analisado possui ciclos!  
-----
```

- Grafo B

```
-----  
  0 1 2 3 4 5 6 7 8 9  
0 0 1 1 1 0 1 0 0 0  
1 0 0 1 0 0 0 0 0 0  
2 0 0 0 1 1 0 0 0 0  
3 0 0 0 0 0 0 0 0 0  
4 0 0 0 0 0 0 1 0 0  
5 0 0 0 0 1 0 1 0 0  
6 0 0 0 0 0 0 0 1 1  
7 0 0 0 0 0 0 0 0 1  
8 0 0 0 0 0 0 0 0 0  
9 0 0 0 0 0 0 1 0 0  
  
0 grafo analisado nao possui ciclos!  
-----
```