Roteiro 3 - Aplicações da busca em profundidade e em largura

1. Faça uma função para encontrar um ciclo, se houver (Retorne a sequência de vértices e arestas do ciclo ou False se não houver ciclo)
    a. Assinatura da função:

```
def ha_ciclo(self)
```

    b. Retorno: Um valor booleano (False) que indica se não existe um ciclo ou uma lista no formato [v1, a1, v2, a2, v3, a3, …, an, v1] (onde vx são vértices e ax são arestas) indicando os vértices e arestas que formam o ciclo.

2. Faça uma função para encontrar um caminho de comprimento n (n passado como parâmetro), se houver. O caminho não pode ter vértices nem arestas repetidas.
    a. Assinatura da função:

```
def caminho(self, n)
```

    b. Retorno: Uma lista no formato [v1, a1, v2, a2, v3, a3, ...] onde vx são vértices e ax são arestas

3. Faça uma função para descobrir se um grafo é conexo.
    a. Assinatura da função:

```
def conexo(self):
```

    b. Retorno: Um valor booleano que indica se o grafo é ou não conexo

4. **<u>Você deve fazer um conjunto de casos de teste de unidade de acordo com os testes que foram passados no roteiro 1. A qualidade e quantidade de testes realizados será critério de avaliação do roteiro.</u>**

    Dica: Você pode fazer uma função auxiliar que verifica se há um caminho entre 2 vértices. Em seguida, chamar essa função para todos os pares de vértices do grafo.
    Assinatura da função auxiliar:
    `def caminho_dois_vertices(self, x, y)`