Universidade Estadual do Maranhão (UEMA)
Caixa Postal 09 – 65055-310 –São Luís – MA – Brazil

[2] Departamento de Engenharia de Computação
Universidade Estadual do Maranhão (UEMA) – São Luis, SC – Brazil

Wesleson Souza Silva

1. Introdução

Apresentação de um projeto Teoria dos Grafos com a implementação de uma estrutura de dados em uma matriz adjacente a qual vai ser carregada a partir de um arquivo .txt qual será formado 1 ª Linha ::“ ou “ Onde: D = Dirigido ND = Não Dirigido, 2 ª até a linha oo v 1 ,v 2. Após a leitura do arquivo e armazenamento na estrutura de dados foi desenvolvida algumas funções.

2. Desenvolvimento

Esse projeto foi desenvolvido com a linguagem de programação Python, no início do projeto foi preciso fazer a importação de uma biblioteca Numpy ela é um pacote para a linguagem Python responsável por da suporte a matrizes multidimensionais, além disso ela possui várias funções matemáticas prontas, mais para fazer uso desse pacote é preciso rodar o comando pip3 install nympy para fazer a instalação das dependências para assim ser possível fazer o seu uso a partir da sua importação como podemos ver na ilustração da imagem abaixo.

```
2   import numpy as np
```

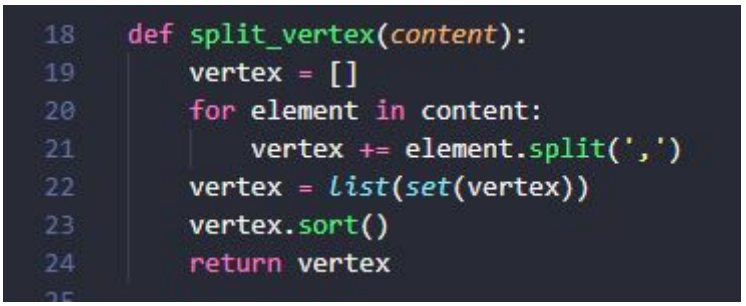
Figure 1. exemplo code 1

A figura abaixo representa a função onde é feito a abertura do arquivo, criação de uma aresta vazia e em seguida vai adicionando linha a linha dessa lista e se tiver um \n no final ele é removido. além disso, ela é responsável por dizer se ele é dirigido ou não e faz a exibição do conteúdo que está lá no arquivo.

```
5   def read_file(string,method):
6       string = str(string)
7       method = str(method)
8       with open(string,method) as file:
9           content = []
10          for line in file:
11              content.append(line.replace('\n',''))
12          file.close()
13          graph_type = content[0]
14          del(content[0])
15          print(content)
16      return content, graph_type
17
```

Figure 2. exemplo code 2

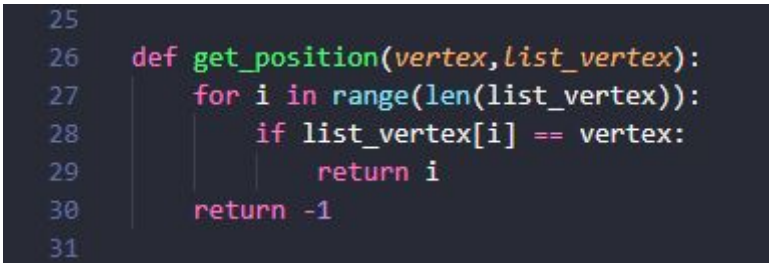
A função split_vertex ilustrada na figura 3 é responsável por pegar todo o conteúdo do txt e transformá na lista de vértice, onde é feito remoção dois index repetidos e ordenação.



```
18  def split_vertex(content):
19      vertex = []
20      for element in content:
21          vertex += element.split(',')
22      vertex = list(set(vertex))
23      vertex.sort()
24      return vertex
```

Figure 3. exemplo code 3

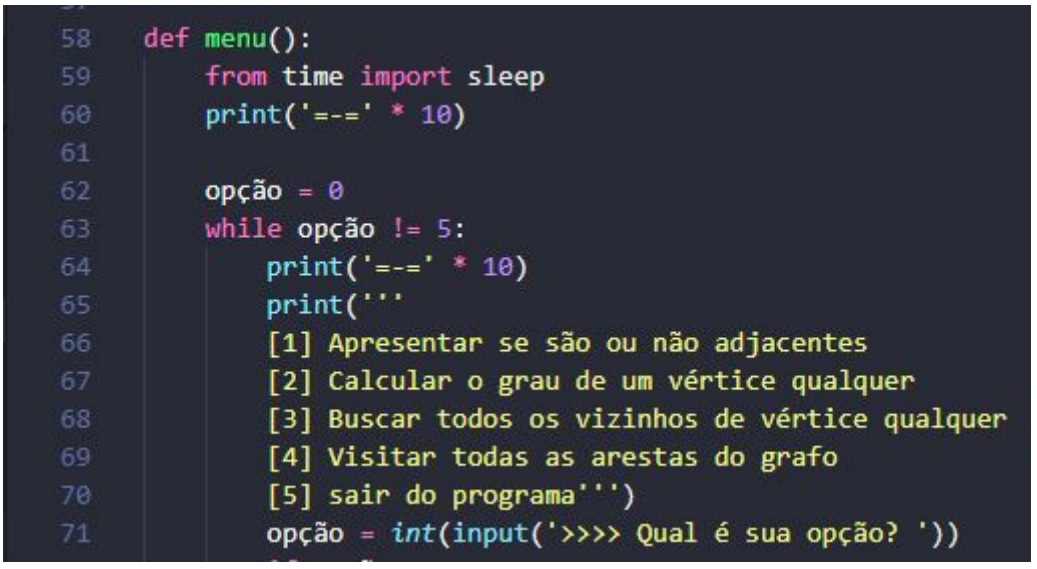
A função get_position ilustrada na figura 4 vai ser responsável por fazer cada letra do txt se torne uma posição na lista.



```
25
26  def get_position(vertex,list_vertex):
27      for i in range(len(list_vertex)):
28          if list_vertex[i] == vertex:
29              return i
30      return -1
31
```

Figure 4. exemplo code 4

A função



```
58  def menu():
59      from time import sleep
60      print('=-=' * 10)
61
62      opção = 0
63      while opção != 5:
64          print('=-=' * 10)
65          print('''
66          [1] Apresentar se são ou não adjacentes
67          [2] Calcular o grau de um vértice qualquer
68          [3] Buscar todos os vizinhos de vértice qualquer
69          [4] Visitar todas as arestas do grafo
70          [5] sair do programa''')
71          opção = int(input('>>>> Qual é sua opção? '))
```

Figure 5. exemplo code 5

exibição da lista de vértice

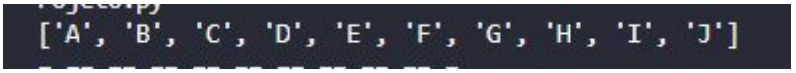


Figure 6. exemplo 1

exibição da lista de aresta

Figure 7. exemplo 2

Referencias

Contribuidores da Wikipédia, "Teoria dos grafos," *Wikipédia, a enciclopédia livre,* https://pt.wikipedia.org/w/index.php?title=Teoria_dos_grafos&oldid=59014368 (accessed agosto 11, 2020).

Colaboradores da Wikipedia, "Adjacency matrix", *Wikipedia, The Free Encyclopedia,* https://en.wikipedia.org/w/index.php?title=Adjacency_matrix&oldid=979433676 (acessado em 1 de outubro de 2020).