0. Preparação do ambiente

carros.csv

```
%%writefile carros.csv
id,valor_venda,valor_manutencao,portas,pessoas,porta_malas
1,vhigh,med,2,2,small
2,med,vhigh,2,2,small
3,low,vhigh,2,2,small
4,low,high,2,2,small
5,low,high,2,2,small
6,low,high,4,4,big
7,low,high,4,4,big
8,low,med,2,2,small
9,low,med,2,2,small
10,low,med,2,2,small
11,low,med,4,4,big
12,low,low,2,2,small
13,low,low,4,4,small
14,low,low,4,4,med
```

```
Writing carros.csv
```

musica.txt

```
%%writefile musica.txt
Roda Viva
Chico Buarque

Tem dias que a gente se sente
Como quem partiu ou morreu
A gente estancou de repente
Ou foi o mundo então que cresceu
A gente quer ter voz ativa
No nosso destino mandar
Mas eis que chega a roda viva
E carrega o destino pra lá

Roda mundo, roda-gigante
Roda moinho, roda pião

O tempo rodou num instante
Nas voltas do meu coração
A gente vai contra a corrente
Até não poder resistir
Na volta do barco é que sente
O quanto deixou de cumprir
Faz tempo que a gente cultiva
A mais linda roseira que há
Mas eis que chega a roda viva
E carrega a roseira pra lá

Roda mundo, roda-gigante
Roda moinho, roda pião
```

```
Writing musica.txt
```

---

1. Extração de coluna de arquivo csv

1.1 Extraia os valore valor_venda e armazene em uma lista

```
'''
  Recebendo arquivo carros.csv, lendo o cabeçalho e as linhas, quebrando as
  linhas na vígula (","), armazenando o valor venda na lista valores_venda.
```

```
  linhas na vigula (",") e armazenando o valor_venda na lista valores_venda.
  Recupera a lista.
'''

valores_venda = []
with open(file="carros.csv", mode="r", encoding="utf8") as file:
  line = file.readline()
  line = file.readline()
  while line:
    valores = line.split(",")
    valor_venda = valores[1]
    valores_venda.append(valor_venda)
    line = file.readline()
valores_venda
```

```
['vhigh',
 'med',
 'low',
 'low',
 'low',
 'low',
 'low',
 'low',
 'low',
 'low',
 'low',
 'low',
 'low',
 'low']
```

1.2 Complete a função abaixo para extrair uma coluna, do arquivo csv em uma lista

```
def extrai_coluna_csv(nome_arquivo: str, indica_coluna: str) -> list:
  """
    Função que recebe o caminho de uma rquivo e retorna uma lista com os
    valores de uma das colunas. A função retorna uma lista vazia, caso não
    encontre algumas das informações.

      :param nome_arquivo (str): Caminho do arquivo que se desja obter a coluna.
      :param indica_coluna (str): Nome da coluna à ser obtida.
      :return coluna (list): Caso o arquivo e a coluna sejam encontrados,
                             retorna os valores.
  """

  # Verifica e abre o arquivo
  try:
    coluna = []
    with open(file=nome_arquivo, mode="r", encoding="utf8") as file:
      line = file.readline()
      head = line.split(",")
      column_verify = False
      for column in range(len(head)):
        column_name = head[column].replace("\n","")
        if indica_coluna == column_name:
          column_verify = True
          break

      # Verifica se o nome da coluna existe no arquivo
      if not column_verify:
        print("Coluna indicada não está presente no arquivo!\n")
        print(f"Colunas disponíveis:\n{head}")
        return coluna
      else:
        line = file.readline()
        while line:
          line = line.replace("\n","")
          valores = line.split(",")
          valor = valores[column]
          coluna.append(valor)
          line = file.readline()

        return coluna
  except FileNotFoundError as file_error:
    print("Arquivo não encontrado!")
    return coluna
  except Exception as error:
    print(type(error))
    return coluna
```

```
print(extrai_coluna_csv.__doc__)
```

```
    Função que recebe o caminho de uma rquivo e retorna uma lista com os
    valores de uma das colunas. A função retorna uma lista vazia, caso não
    encontre algumas das informações.

      :param nome_arquivo (str): Caminho do arquivo que se desja obter a coluna.
      :param indica_coluna (str): Nome da coluna à ser obtida.
      :return coluna (list): Caso o arquivo e a coluna sejam encontrados,
                             retorna os valores.
```

```
# Testando erro no caminho do arquivo.

coluna_valor_manutencao = extrai_coluna_csv(
                                      nome_arquivo="carrs.csv",
                                      indica_coluna="porta_mal"
                                    )
coluna_valor_manutencao
```

```
Arquivo não encontrado!
[]
```

```
# Teste de erro no nome da coluna

coluna_valor_manutencao = extrai_coluna_csv(
                                      nome_arquivo="carros.csv",
                                      indica_coluna="coluna_errada"
                                    )
coluna_valor_manutencao
```

```
Coluna indicada não está presente no arquivo!

Colunas disponíveis:
['id', 'valor_venda', 'valor_manutencao', 'portas', 'pessoas', 'porta_malas\n']
[]
```

```
# Extrair a coluna valor_manutencao

coluna_valor_manutencao = extrai_coluna_csv(
                                      nome_arquivo="carros.csv",
                                      indica_coluna="valor_manutencao"
                                    )
coluna_valor_manutencao
```

```
['med',
 'vhigh',
 'vhigh',
 'high',
 'high',
 'high',
 'high',
 'med',
 'med',
 'med',
 'med',
 'low',
 'low',
 'low']
```

```
# Extrair a coluna porta_malas

coluna_porta_malas = extrai_coluna_csv(
                                      nome_arquivo="carros.csv",
                                      indica_coluna="porta_malas"
                                    )
coluna_porta_malas
```

```
['small',
 'small',
 'small',
 'small',
 'small',
 'big',
 'big',
 'small',
 'small',
 'small',
 'big',
 'small',
 'small',
 'med']
```

1.2.1 Os elementos devem ter o tipo de dado correto.

```
def extrai_coluna_csv(nome_arquivo: str, indica_coluna: str) -> list:
  """
    Função que recebe o caminho de uma rquivo e retorna uma lista com os
    valores de uma das colunas. Os parâmetros devem ter o tipo de dado correto.
    A função retorna uma lista vazia, caso não encontre algumas das informações.

      :param nome_arquivo (str): Caminho do arquivo que se desja obter a coluna.
      :param indica_coluna (str): Nome da coluna à ser obtida.
      :return coluna (list): Caso o arquivo e a coluna sejam encontrados,
                             retorna os valores.
  """
  # Instância a lista coluna para retorno
  coluna = []

  # Verifica se os parâmetros são dos tipos corretos
  if type(nome_arquivo) != str:
    print("O parâmetro nome_arquivo deve ser do tipo string!")
    return coluna
  elif type(indica_coluna) != str:
    print("O parâmetro indica_coluna deve ser do tipo string!")
    return coluna
  else:
    try:

      # Verifica e abre o arquivo
      with open(file=nome_arquivo, mode="r", encoding="utf8") as file:
        line = file.readline()
        head = line.split(",")
        column_verify = False
        for column in range(len(head)):
          column_name = head[column].replace("\n","")
          if indica_coluna == column_name:
            column_verify = True
            break

    # Verifica se o nome da coluna existe no arquivo e gera a lista de retorno
        if not column_verify:
          print("Coluna indicada não está presente no arquivo!\n")
          print(f"Colunas disponíveis:\n{head}")
          return coluna
        else:
          line = file.readline()
          while line:
            line = line.replace("\n","")
            valores = line.split(",")
            valor = valores[column]
            coluna.append(valor)
            line = file.readline()

          return coluna
    except FileNotFoundError as file_error:
      print("Arquivo não encontrado!")
      return coluna
    except Exception as error:
      print(type(error))
      return coluna
```

```
# Simulando erro de tipo Parâmetro

coluna_valor_venda = extrai_coluna_csv(
                                        nome_arquivo=1,
                                        indica_coluna="valor_venda"
                                      )
coluna_valor_venda
```

```
O parâmetro nome_arquivo deve ser do tipo string!
[]
```

```
# Simulando erro de Parâmetro

coluna_valor_venda = extrai_coluna_csv(
                                        nome_arquivo="carros.csv",
                                        indica_coluna=False
                                      )
coluna_valor_venda
```

```
O parâmetro indica_coluna deve ser do tipo string!
```

```
[]
```

```
# Extrair a coluna valor_venda

coluna_valor_venda = extrai_coluna_csv(
                                        nome_arquivo="carros.csv",
                                        indica_coluna="valor_venda"
                                       )
coluna_valor_venda
```

```
['vhigh',
 'med',
 'low',
 'low',
 'low',
 'low',
 'low',
 'low',
 'low',
 'low',
 'low',
 'low',
 'low']
```

```
# Extrair a coluna pessoas

coluna_pessoas = extrai_coluna_csv(
                                        nome_arquivo="carros.csv",
                                        indica_coluna="pessoas"
                                       )
coluna_pessoas
```

```
['2', '2', '2', '2', '2', '4', '4', '2', '2', '2', '4', '2', '4', '4']
```

---

2. Funções para arquivo txt

2.1 Complete a função abaixo para extrair uma a uma as palavras de uma linha do arquivo txt em uma lista.

```
def extrai_linha_txt(nome_arquivo: str, numero_linha: int) -> list:
  """
    Função para a recuperação das palavras contidas em uma das linhas de um
    arquivo txt.

    :param nome_arquivo (str): O caminho do arquivo à ser extraído.
    :param numero_linha (int): Número da linha do arquivo para extrair.
    :return lista_palavras (list): Lista contendo todas as palavras da linha
                                   extraída.
  """
  lista_palavras = []
  palavras_linha = []
  try:
    with open(file=nome_arquivo, mode="r", encoding="utf8") as file:
      line = file.readline()
      while line:
        palavras_linha.append(line)
        line = file.readline()
    linha = palavras_linha[numero_linha-1]
    linha = linha.replace("\n", "")
    lista_palavras = linha.split(" ")
  except Exception as error:
    print(error)
    return lista_palavras
  return lista_palavras
```

```
print(extrai_linha_txt.__doc__)
```

```

        Função para a recuperação das palavras contidas em uma das linhas de um
        arquivo txt.

        :param nome_arquivo (str): O caminho do arquivo à ser extraído.
```

```
        :param numero_linha (int): Número da linha do arquivo para extrair.
        :return lista_palavras (list): Lista contendo todas as palavras da linha
                                       extraída.
```

```
linha = extrai_linha_txt(
                              nome_arquivo="musica.txt",
                              numero_linha=1_000
)
linha
```

```
    list index out of range
    []
```

```
linha = extrai_linha_txt(
                              nome_arquivo="ica.txt",
                              numero_linha=10
)
linha
```

```
    [Errno 2] No such file or directory: 'ica.txt'
    []
```

```
linha = extrai_linha_txt(
                              nome_arquivo="musica.txt",
                              numero_linha=10
)
linha
```

```
    ['Mas', 'eis', 'que', 'chega', 'a', 'roda', 'viva']
```
