MACHINE LEARNING

Atividade 02

Paloma Pinheiro de Lima

1. Escreva uma função que receba uma lista de números e retorne outra lista com os números ímpares.

```
def filtrar_impares(lista):
    return [num for num in lista if num % 2 != 0]

numeros = [610, 229, 432, 343, 11, 580, 627, 710, 84, 99, 112, 249, 573]
impares = filtrar_impares(numeros)
print(impares)
```

2. Escreva uma função que receba uma lista de números e retorne outra lista com os números primos presentes.

```
def numero_primo(num):

    if num <= 1:
        return False
    for i in range(2, int(num ** 0.5) + 1):
        if num % i == 0:
            return False
    return True

def filtrar_primos(lista):
    return [num for num in lista if numero_primo(num)]


numeros = [610, 229, 432, 343, 11, 580, 627, 710, 84, 99, 112, 249, 573]
primos = filtrar_primos(numeros)
print(primos)
```

3. Escreva uma função que receba duas listas e retorne outra lista com os elementos que estão presentes em apenas uma das listas.

```
def elementos_unicos(lista1, lista2):

    conjunto1 = set(lista1)
    conjunto2 = set(lista2)

    # Elementos únicos em cada conjunto
    unicos_lista1 = conjunto1 - conjunto2
    unicos_lista2 = conjunto2 - conjunto1

    # Unindo os elementos únicos
    elementos_unicos = list(unicos_lista1.union(unicos_lista2))

    return elementos_unicos

lista1 = [11, 26, 3, 41, 55]
lista2 = [42, 55, 26, 7, 8]
unicos = elementos_unicos(lista1, lista2)
print(unicos)
```

4. Dada uma lista de números inteiros, escreva uma função para encontrar o segundo maior valor na lista.

```
def segundo_maior(lista):
    if len(lista) < 2:
        return None  # lista com menos de dois elementos

    conjunto = set(lista) # removendo duplicatas

    if len(conjunto) < 2:
        return None  # conjunto com menos de dois elementos

    conjunto.remove(max(conjunto))  # remove o maior elemento

    return max(conjunto)

numeros = [610, 229, 432, 343, 11, 580, 627, 710, 84, 99, 112, 249, 573]
print(segundo_maior(numeros))
```

5. Crie uma função que receba uma lista de tuplas, cada uma contendo o nome e a idade de uma pessoa, e retorne a lista ordenada pelo nome das pessoas em ordem alfabética.

```
def ordem_alfabetica(lista):
    return sorted(lista, key=lambda x: x[0])

pessoas = [("Rita", 70), ("Cássia", 45), ("Elis", 35), ("Bethânia", 80)]
ordenadas = ordem_alfabetica(pessoas)
print(ordenadas)
```

6. Observe os espaços sublinhados e complete o código.

```
import matplotlib.pyplot as plt
import numpy as np
fig, axs = plt.subplots(ncols=2, nrows=2, figsize=(5.5, 3.5),
                        layout="constrained")

for row in range(2):
  for col in range(2): axs[row, col].annotate(f'axs[{row}, {col}]', (0.5, 0.5),
                          transform=axs[row, col].transAxes,
                          ha='center', va='center', fontsize=18,
                          color='darkgrey')
fig.suptitle('plt.subplots()')
```

7. Observe os espaços sublinhados e complete o código.

```
import numpy as np
import matplotlib as mpl
import matplotlib.pyplot as plt

x = np.linspace(-2 * np.pi, 2 * np.pi, 100)
y = np.sin(x) fig, ax = plt.subplots() ax.plot(x, y)
```

8. Utilizando pandas, como realizar a leitura de um arquivo CSV em um DataFrame e exibir as primeiras linhas?

Deve-se importar a biblioteca Pandas:

```
import pandas as pd
```

Utilizar a função pd.read_csv() para carregar os dados do arquivo CSV:

```
df = pd.read_csv(caminho_arquivo)
```

Utilizar o método .head() do para exibir as primeiras linhas do arquivo:

```
print(df.head())
```

9. Utilizando pandas, como selecionar uma coluna específica e filtrar linhas em um "DataFrame" com base em uma condição?

Deve-se importar a biblioteca Pandas:

```
import pandas as pd
```

Criar ou carregar um DataFrame, utilizando pd.DataFrame() ou pd.read_csv():

```
df = pd.read_csv(caminho_arquivo)
```

Selecionar a coluna usando: df['nome_da_coluna'] ou df.nome_da_coluna

Usar colchetes [ ] para aplicar uma condição e filtrar as linhas desejadas.

Exemplo:

```
import pandas as pd

# criação de DataFrame (pode ser carregado de um arquivo CSV)
data = {
    'Nome': ['Cássia', 'Elis', 'Rita', 'Bethânia'],
    'Idade': [45, 32, 70, 80],
    'Cidade': ['São Paulo', 'Rio de Janeiro', 'Rio de Janeiro', 'Salvador']
}

df = pd.DataFrame(data)
```

```
# Seleciona uma coluna específica
coluna_idade = df['Idade']
print("Coluna Idade:")
print(coluna_idade)
print()

# Filtra linhas com base em uma condição (por exemplo, idade maior que 30)
filtro_idade_maior_que_30 = df[df['Idade'] > 30]
print("Linhas com idade maior que 30:")
print(filtro_idade_maior_que_30)
```

10.Utilizando pandas, como lidar com valores ausentes (NaN) em um DataFrame?

Algumas técnicas que podem ser utilizadas para lidar com valores ausentes:

1 - É possível remover linhas e colunas que contenham o valor ausente usando o método ‘dropna( )’.

2 - É possível preencher os valores ausentes com valores específicos usando o método ‘fillna( )’.

3 - É possível utilizar máscaras booleanas para seleção condicional, filtrando dados e aplicando critérios específicos para selecionar partes do DataFrame que atendam a condições que forem determinadas.