ATIVIDADE AVANT BOOTCAMP

Nome: Balduíno José da Silva
Curso: Machine Learning

ATIVIDADE 2

1- Função para receber números ímpares:

```
def filtrar_impares(lista):

    lista_impares = [num for num in lista if num % 2 != 0]
    return lista_impares

    lista_numeros = [3, 2, 4,5, 6, 7, 8, 10,11]
          impares = filtrar_impares(lista_numeros)
          print(ímpares)

          A saída vai ser -> [ 3, 5, 7,11]
```

2- Função que recebe uma lista de número e retorna apenas números primos:

```
def t_primo(num):

    if num <= 1:
        return False
    if num == 2:
        return True
    if num % 2 == 0:
        return False
    for i in range(3, int(num**0.5) + 1, 2):
        if num % i == 0:
            return False
    return True
def filtrar_primos(lista):

        lista_primos = [num for num in lista if t_primo(num)]
    return lista_primos

lista_numeros = [10, 15, 3, 4, 7, 9, 11, 13, 18, 23]
primos = filtrar_primos(lista_numeros)
print(primos)
```

A saída vai ser: [3, 7, 11, 13, 23]

3- Função que escreve conjunto diferença de de duas listas:

```
def elementos_unicos(lista1, lista2):

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

    unicos = list(conjunto1 ^ conjunto2)

    return unicos

lista1 = [1, 2, 3, 4, 5]
lista2 = [4, 5, 6, 7, 8]
unicos = elementos_unicos(lista1, lista2)
print(unicos)
#A saída vai sert: [1, 2, 3, 6, 7, 8]
```

4- Função recebe uma lista de números e retorna o segundo maior:

```
def segundo_maior(lista):
    if len(lista) < 2:
        return None  # Não há segundo maior valor se a lista tiver menos de 2
elementos

    # Converte a lista em um conjunto para eliminar duplicatas
    conjunto = set(lista)

    if len(conjunto) < 2:
        return None

    # Ordena os elementos do conjunto em ordem decrescente e retorna o segundo
valor
    lista_ordenada = sorted(conjunto, reverse=True)
    return lista_ordenada[1]

lista_numeros = [10, 4, 3, 30, 89, 30, 89, 10]
resultado = segundo_maior(lista_numeros)
print(resultado)
```

A saída vai ser: 30

5- A Função que recebe uma lista de tupla

```
def ordenar_por_nome(lista_pessoas):
        # Ordena a lista de tuplas pelo nome, que está no índice 0 de cada tupla
         lista_ordenada = sorted(lista_pessoas, key=lambda pessoa: pessoa[0])
        return lista_ordenada
```

Ex:

```
pessoas = [("René", 25), ("Angélica", 22), ("Balduino", 30), ("Bootcamp", 28)]
lista_ordenada = ordenar_por_nome(pessoas)
print(lista_ordenada)
```

A saída vai ser : [('Angélica', 22), ('Balduino', 30), ('Bootcamp', 28), ('René', 25)]

6- Código completo

```
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- Código completo:

```
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- A leitura de um arquivo CSV em um DataFrame usando o pandas e exibir as primeiras linhas, é feito seguindo os seguintes passos:

1º Importar a biblioteca pandas
EX:

“**import** pandas as pd”

2º Ler os arquivos CSV, pode ser usada a seguinte função:
EX:

“df = pd.read_csv('caminho do arquivos CSV’)”

3º Exibir as primeiras linhas do DataFrame, para tal usamos a função:

EX:

“print(df.head())”

9- Para selecionar uma coluna específica e filtrar linhas em um DataFrame do pandas com base em uma condição,

1º Selecionar uma coluna**:**
EX:

“coluna = df['nome da coluna']”

2º Filtrar linhas com base em uma condição:

EX:

“linhas_filtradas = df[df['nome da coluna'] > valor]**”**

10- Para lidar com valores ausentes (NaN) em um DataFrame usando pandas, podemos utilizar várias funções e métodos que a biblioteca oferece, como por exemplo:

1º Identificar Valores Ausentes:
EX:

```
import pandas as pd

data = {
  'A': [1, 2, None, 4],
  'B': [None, 2, 3, 4],
  'C': [1, None, 3, None]
}
df = pd.DataFrame(data)
print(df.isna()
```

2º Remover Linhas ou Colunas com Valores Ausentes:

3º Preencher Valores Ausentes Usando Métodos Específicos:

EX:

```
df_ffill = df.fillna(method='ffill')
df_bfill = df.fillna(method='bfill')
```

4º Verificar e Contar Valores Ausentes:

EX:

```
print(df.isna().sum())
```