UNIVERSIDADE DA BEIRA INTERIOR  
CFIUTE – Centro de Formação Interação UBI Tecido Empresarial

TRABALHO II

Curso de Introdução à Programação em Python

LARA GUERREIRO PIRES

Abril de 2021

Resolução desenvolvida no editor de código Visual Code Editor – VSCode, em Pythom 3.  
Os códigos estão disponíveis no em <https://github.com/laraguerreiro/Python-Ubi/tree/main/trabalho2>

EXERCÍCIO 1

Uma imagem contendo Tabela

Descrição gerada automaticamente

Para desenvolver a solução dos três itens foi necessária a importação das bibliotecas *numpy*  e *numpy.linalg.*

from numpy import array, transpose

from numpy.linalg import det, inv, eig

D = array([[3.4, 0.5, 1.2, -0.8],

[-1.6, 6.8, 1.1, 2.3],

[0.1, -0.5, 2.5, 0.7],

[-0.9, 1.7, 0.2, 4.9]])

print("Matriz D")

print(D)

# Calcular a matriz A = D.Dt

DT = transpose(D)

print("Matriz DT")

print(DT)

A = D.dot(DT)

print("Matriz A")

print(A)

# 2. Calcular a determinante da matriz A. Qual a inversa da matriz A?

determinante\_A = det (A)

print ("A Determinante de A é igual a " + str(determinante\_A))

inversa\_A = inv(A)

print ("A Inversa de A é igual a =" + str(inversa\_A))

# 3. Calcular os valores e vetores próprios (autovalores e autovetores) da matriz A.

w, v= eig(A)

print( "Valores - autovalores - da matriz A = " + str(w))

print ("Os vetores próprios - autovetores - da matriz A = " +str(v))

**EXERCÍCIO 2**

Texto

Descrição gerada automaticamente

Para desenvolver a solução dos quatro itens foi necessária a importação das bibliotecas *math*  e *numpy*.

import math

from numpy import arange, max, min, array

#1. Escrever um programa que crie uma matriz M com os valores de z= f(x,y)

# para o intervalo [-2,2] comum passo de 0.1 e y no intervalo [-3, 3], com passo de 0.2 .

def fxy (x,y):

resultado = 0.1 + ( (1 + math.sin(2\*x + 3\*y)) / (3.5 + math.sin(x-y)) )

return resultado

def cria\_matriz ():

matriz\_M =[]

for x in arange (-2,2,0.1):

linha = []

for y in arange(-3,3,0.2):

valor = fxy(x,y)

linha.append(valor)

matriz\_M.append(linha)

return matriz\_M

M = cria\_matriz()

print("A matriz M é igual a ")

print (M)

# 2. Qual é o menor elemento da sexta coluna da Matriz M?

x = M[:][5]

xmin = min(x)

print( "O menor elemento de sexta coluna " + str(xmin) )

# 3. Qual o menor elemento da Matriz M

y = M[:][:]

ymin = min(y)

print ("O menor valor da Matriz M é igual a " + str(ymin))

# 4. Qual é o maior elemento da matriz M?

ymax = max(y)

print("O maior elemento da Matriz M é igual a " + str(ymax))

**EXERCÍCIO 3**

Escrever uma função que calcule a média e o desvio padrão de uma lista de dados

amostrais.

Para desenvolver a solução foi necessária a importação das bibliotecas *numpy* e *statistics.*

# Média

from numpy import array

import statistics as st

def media (amostra):

media = sum(amostra)/ len(amostra)

return media

teste = [10, 20, 30, 40]

print (media(teste) )

print (st.mean(teste))

# variancia

def variancia (amostra):

m = media(amostra)

somadoselementos = 0

for elemento in amostra:

calculo = (elemento - m) \*\* 2

somadoselementos = somadoselementos + calculo

variancia = somadoselementos / (len(amostra) - 1)

return variancia

def desvio\_padrao (amostra):

return variancia(amostra) \*\* (1/2)

print (variancia(teste))

print (st.variance(teste))

print (desvio\_padrao(teste))

print (st.stdev(teste))

def resposta\_exercicio(amostra):

print ("A média da amostra é igual a: ", media(amostra))

print ("O desvio padrão é igual a: ", desvio\_padrao(amostra))

resposta\_exercicio(teste)

**EXERCÍCIO 4**

Escrever um programa para efetuar a análise estatística dos dados populacionais

fornecidos em anexo (Anexo I). Comentar os resultados obtidos.

Os dados utilizados foram fornecidos pelo docente.

Para desenvolver a solução fez-se a importação das bibliotecas *pandas, statistics, matplotlib.pyplot, os, seaborn .*

*Pandas* foi usada para criação do *dataframe,* com base em um arquivo *.csv* disponibilizado.   
Statistics para análise dos dados, *Matplotlib* e *Seaborn* para construção dos gráficos.

A análise for feita selecionando apenas uma das colunas do *dataframe*, a “Close”*.*

# Exercicio 4 - Escrever um programa para efetuar a análise estatística da coluna "Close":

import pandas as pd

import statistics as st

import matplotlib.pyplot as plt

import os

import seaborn as sns

df = pd.read\_csv(os.path.join(os.path.dirname(\_\_file\_\_),"base\_ex4.csv"))

# Abaixo teste para verificar se a importação da base funcionou corretamente.

#print (df.head()) # 5 primeiros posições

# print (df.tail()) # -5 primeiros posições

media = st.mean(df['Close'])

print("Média =", media)

moda = st.mode(df['Close'])

print("Moda =", moda)

mediana = st.median(df['Close'])

print("Mediana =", mediana)

amplitude = max(df['Close']) - min(df['Close'])

print("Amplitude =", amplitude)

quartis = st.quantiles(df['Close'])

print("A lista dos quartis é:", quartis)

Q1 = quartis[0]; Q2 = quartis[1]; Q3 = quartis[2]

print(f"Q1 = {Q1}, Q2 = {Q2}, Q3 = {Q3}")

IQR = Q3 - Q1

print("IQR =", Q3-Q1)

# print (df['Close'].describe()) # descrição estatística dos dados

# Na análise comentada acima está toda a tabela, e serve como teste.

# O programa deve analisar a coluna close [4] apenas.

# O histograma mostra a quantidade de vezes que a moeda fechou em determinado valor.

#seaborn

sns.histplot(data = df, x = 'Close')

plt.title("Histograma")

plt.show()

# Boxplot (i.e. "CAixa de bigode")

plt.boxplot(df["Close"], notch = 'True')

plt.show ()

#Informa informação visual de dos quartis da amostra, bem como de sua mediana e dispersão.

EXERCÍCIO 5

Realizar um web scraping com base na descrição anexa (Anexo II).

Solução desenvolvida cm base nas indicações do docente. Dataframe formado com base em webscrapping realizado na página <https://www.reuters.com/companies/GOOG.O/people>.

Primeiro efetuou-se a raspagem dos dados da página para encontrar os dados relativo a idade das pessoas mencionadas, e depois foi efetuada a análise estatística desses mesmos dados usando o código desenvolvido no exercício anterior.

Para a raspagem de dados foram utilizadas as bibliotecas  *requests* e *pandas.*Para a análise estatística, as bibliotecas *statistics, seaborn* e *matplotlib.pyplot.*

# Exercício 5: Fazer a análise estatística dos dados da coluna 'Age'

# da tabela no seguinte link: https://www.reuters.com/companies/GOOG.O/people

import requests

import pandas as pd

import statistics as st

import seaborn as sns

import matplotlib.pyplot as plt

url = "https://www.reuters.com/companies/GOOG.O/people"

wb\_site\_pd = requests.get(url)

tables = pd.read\_html(wb\_site\_pd.text)

df = tables[0]

print(df.head())

media = st.mean(df['Age'])

print("Média =", media)

moda = st.mode(df['Age'])

print("Moda =", moda)

mediana = st.median(df['Age'])

print("Mediana =", mediana)

amplitude = max(df['Age']) - min(df['Age'])

print("Amplitude =", amplitude)

quartis = st.quantiles(df['Age'])

print("A lista dos quartis é:", quartis)

Q1 = quartis[0]; Q2 = quartis[1]; Q3 = quartis[2]

print(f"Q1 = {Q1}, Q2 = {Q2}, Q3 = {Q3}")

IQR = Q3 - Q1

print("IQR =", Q3-Q1)

sns.histplot(data = df, x = 'Age')

plt.title("Histograma")

plt.show()

plt.boxplot(df["Age"], notch = 'True')

plt.show ()