

ATIVIDADE PRÁTICA LINGUAGEM DE PROGRAMAÇÃO

ROBSON CRUZ DE MELO – 3773638

Professora Me. Mariane G. Bergamini

GRAVATAÍ

2023

1 EXEMPLO DE RESOLUÇÃO

Exercício 01 exemplo: Realizar uma classe veículo que apresente o tipo do carro, modelo, fabricante e quantidade de passageiros. Além disso, utilizar os métodos GETTER e SETTER para incluir as classes Modelo retornando o modelo e o fabricante do carro e outra classe Quantidade de Passageiro no carro.

RESPOSTA DO ALUNO

COLE AQUI O SEU CÓDIGO FONTE (usar os comandos Ctrl + c / Ctrl + v) :

```
class Veiculo:
    def __init__(self, tipo, modelo, fabricante, qtd_passageiro):
        self.tipo = tipo
        self.modelo = modelo
        self.fabricante = fabricante
        self.qtd_passageiro = qtd_passageiro

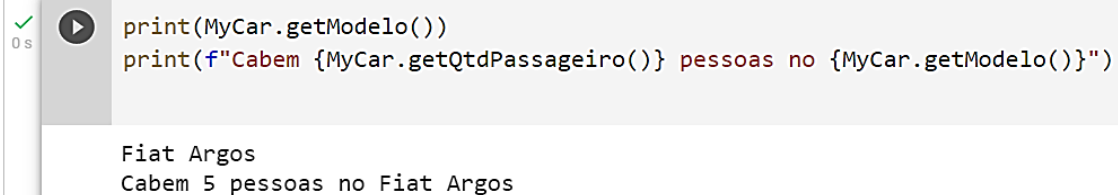
    # getter: adicionando
    def getModelo(self):
        return f"{self.fabricante} {self.modelo}"

    def getQtdPassageiro(self):
        return f"{self.qtd_passageiro}"

MyCar = Veiculo('carro', modelo = 'Argos', fabricante='Fiat', qtd_passageiro=5)
MyBus = Veiculo('Ônibus', modelo = 'Bus202', fabricante='Mercedes', qtd_passageiro=40)

print(MyCar.getModelo())
print(f"Cabem {MyCar.getQtdPassageiro()} pessoas no {MyCar.getModelo()}")
```

COLE AQUI IMAGEM(NS)/PRINT(S) DO TERMINAL SENDO EXECUTADO SEM ERRO:



```
print(MyCar.getModelo())
print(f"Cabem {MyCar.getQtdPassageiro()} pessoas no {MyCar.getModelo()}")

Fiat Argos
Cabem 5 pessoas no Fiat Argos
```

2 EXERCÍCIOS A SEREM SOLUCIONADOS PELO ALUNO :

Exercício 1 – Desenvolver uma **classe calculadora** que faça operações matemáticas utilizando dois números inteiros, sendo os **dois últimos números de seu RU**. Os dois números inteiros devem ser solicitados para o usuário digitar. Se o seu **RU** for **zero**, **substituí-lo(s) pelo número 5**. Sendo as possíveis operações matemáticas: **soma(+)**, **subtração(-)**, **multiplicação(*)**, **divisão(/)**, **expoente (^)**, **resto(%)** e **raíz quadrada da soma dos dois números** (**$\sqrt{\text{Num1} + \text{Num2}}$**). Além destas funcionalidades, o algoritmo deverá ter um **MENU** que possibilite ao usuário escolher qual o tipo de operação que se deseja realizar e que possibilite ao usuário a digitar os dois números. Apresentar todas as operações matemáticas da calculadora funcionando!

RESPOSTA DO ALUNO:

```
#CLASSE CALCULADORA
#IMPORT PARA REALIZAR OS CALCULOS MATEMATICOS
import math

#CLASSE CALCULADORA E SEUS METODOS COM AS OPERAÇÕES MATEMÁTICAS
class Calculadora:

    #METODO DE INICIALIZAÇÃO DA CLASSE
    def __init__(self):
        self.n1 = 0
        self.n2 = 0
        self.res = 0

    def somar(self, n1, n2):
        self.n1 = n1
        self.n2 = n2
        self.res = self.n1 + self.n2
        return self.res

    def subtrair(self, n1, n2):
        self.n1 = n1
        self.n2 = n2
        self.res = self.n1 - self.n2
        return self.res

    def multiplicar(self, n1, n2):
        self.n1 = n1
        self.n2 = n2
        self.res = self.n1 * self.n2
        return self.res

    def dividir(self, n1, n2):
```

```
        self.n1 = n1
        self.n2 = n2
        self.res = self.n1 / self.n2
        return self.res

    def expoente(self, n1, n2):
        self.n1 = n1
        self.n2 = n2
        self.res = self.n1 ** self.n2
        return self.res

    def resto(self, n1, n2):
        self.n1 = n1
        self.n2 = n2
        self.res = self.n1 % self.n2
        return self.res

    def raiz(self, n1, n2):
        self.n1 = n1
        self.n2 = n2
        self.res = math.sqrt(self.n1 + self.n2)
        return self.res

#CLASSE PRINCIPAL
#IMPORTAR DA CLASSE CALCULADORA
from calculadora import Calculadora

class Principal:

    #CONSTRUTOR DA CLASSE PRINCIPAL
    #INICIALIZA A INSTANCIA DA CLASSE CALCULADORA COMO UM
    #ATRIBUTO DA CLASSE PRINCIPAL
    def __init__(self):
        self.calculadora = Calculadora()

    #METODO PARA REALIZAR OS CALCULOS
    def executar_calculos(self):

        # MENU INICIAL
        print("SEJA BEM VINDO A CALCULADORA!!!!")
        print("1 - ADIÇÃO")
        print("2 - SUBTRAÇÃO")
        print("3 - MULTIPLICAÇÃO")
        print("4 - DIVISÃO")
        print("5 - POTENCIAÇÃO")
        print("6 - MÓDULO (RESTO)")
        print("7 - RAÍZ QUADRADA (SOMA DOS DOIS NÚMEROS)")
```

```
print("0 - SAIR")

# LAÇO DE REPETIÇÃO PARA A ESCOLHA DA OPÇÃO
while True:
    op = input("ESCOLHA UMA DAS OPÇÕES PARA REALIZAR  
UMA OPERAÇÃO: ")

    #CONDICIONAL COM AS OPÇÕES ESCOLHIDAS E SUAS DEVI-  
DAS OPREÇÕES
    if op == '1':

        print("***OPERAÇÃO DE ADIÇÃO SELECIONADA ***")
        print("***DIGITE OS DOIS ÚLTIMOS NÚMEROS DO  
SEU RU***")

        # LAÇO DE REPETIÇÃO PARA NÚMERO VÁLIDO
        while True:
            num1 = input("DIGITE O PENÚLTIMO NÚMERO DO  
SEU RU: ")

            if num1.isdigit():
                num1 = int(num1)
                break
            else:
                print("Erro: DIGITE UM NÚMERO VÁ-  
LIDO!")

        while True:
            num2 = input("DIGITE O ÚLTIMO NÚMERO DO  
SEU RU: ")

            if num2.isdigit():
                num2 = int(num2)
                break
            else:
                print("Erro: DIGITE UM NÚMERO VÁ-  
LIDO!")

        resultado = self.calculadora.somar(n1=num1,  
n2=num2)

        print("RESULTADO DA SOMA:", resultado)

    if op == '2':

        print("***OPERAÇÃO DE SUBTRAÇÃO SELECIONADA  
***")

        print("***DIGITE OS DOIS ÚLTIMOS NÚMEROS DO  
SEU RU***")

        # LAÇO DE REPETIÇÃO PARA NÚMERO VÁLIDO
```

```
while True:
    num1 = input("DIGITE O PENÚLTIMO NÚMERO DO
SEU RU: ")
    if num1.isdigit():
        num1 = int(num1)
        break
    else:
        print("Erro: DIGITE UM NÚMERO VÁ-
LIDO!")

while True:
    num2 = input("DIGITE O ÚLTIMO NÚMERO DO
SEU RU: ")
    if num2.isdigit():
        num2 = int(num2)
        break
    else:
        print("Erro: DIGITE UM NÚMERO VÁ-
LIDO!")

resultado = self.calculadora.subtrair(n1=num1,
n2=num2)
print("RESULTADO DA SUBTRAÇÃO:", resultado)

if op == '3':
    print("***OPERAÇÃO DE MULTIPLICAÇÃO SELECIO-
NADA ***")
    print("***DIGITE OS DOIS ÚLTIMOS NÚMEROS DO
SEU RU***")

    # LAÇO DE REPETIÇÃO PARA NÚMERO VÁLIDO
    while True:
        num1 = input("DIGITE O PENÚLTIMO NÚMERO DO
SEU RU: ")
        if num1.isdigit():
            num1 = int(num1)
            break
        else:
            print("Erro: DIGITE UM NÚMERO VÁ-
LIDO!")

        while True:
            num2 = input("DIGITE O ÚLTIMO NÚMERO DO
SEU RU: ")
            if num2.isdigit():
                num2 = int(num2)
                break
```

```
else:
    print("Erro: DIGITE UM NÚMERO VÁ-
LIDO!")

    resultado = self.calculadora.multiplicar(n1=num1, n2=num2)
    print("RESULTADO DA MULTIPLICAÇÃO:", resultado)

    if op == '4':

        print("***OPERAÇÃO DE DIVISÃO SELECIONADA
***")
        print("***DIGITE OS DOIS ÚLTIMOS NÚMEROS DO
SEU RU***")

        # LAÇO DE REPETIÇÃO PARA NÚMERO VÁLIDO
        while True:
            num1 = input("DIGITE O PENÚLTIMO NÚMERO DO
SEU RU: ")

            if num1.isdigit():
                num1 = int(num1)
                break
            else:
                print("Erro: DIGITE UM NÚMERO VÁ-
LIDO!")

        while True:
            num2 = input("DIGITE O ÚLTIMO NÚMERO DO
SEU RU: ")

            if num2.isdigit():
                num2 = int(num2)
                break
            else:
                print("Erro: DIGITE UM NÚMERO VÁ-
LIDO!")

        resultado = self.calculadora.dividir(n1=num1,
n2=num2)
        print("RESULTADO DA DIVISÃO:", resultado)

        if op == '5':

            print("***OPERAÇÃO DE EXPONENCIAÇÃO SELECIO-
NADA ***")
            print("***DIGITE OS DOIS ÚLTIMOS NÚMEROS DO
SEU RU***")
```

```
# LAÇO DE REPETIÇÃO PARA NÚMERO VÁLIDO
while True:
    num1 = input("DIGITE O PENÚLTIMO NÚMERO DO
SEU RU: ")
    if num1.isdigit():
        num1 = int(num1)
        break
    else:
        print("Erro: DIGITE UM NÚMERO VÁ-
LIDO!")

while True:
    num2 = input("DIGITE O ÚLTIMO NÚMERO DO
SEU RU: ")
    if num2.isdigit():
        num2 = int(num2)
        break
    else:
        print("Erro: DIGITE UM NÚMERO VÁ-
LIDO!")

resultado = self.calculadora.expoente(n1=num1,
n2=num2)
print("RESULTADO DA POTENCIAÇÃO:", resultado)

if op == '6':
    print("***OPERAÇÃO DE MÓDULO(RESTO) SELECIO-
NADA ***")
    print("***DIGITE OS DOIS ÚLTIMOS NÚMEROS DO
SEU RU***")

# LAÇO DE REPETIÇÃO PARA NÚMERO VÁLIDO
while True:
    num1 = input("DIGITE O PENÚLTIMO NÚMERO DO
SEU RU: ")
    if num1.isdigit():
        num1 = int(num1)
        break
    else:
        print("Erro: DIGITE UM NÚMERO VÁ-
LIDO!")

while True:
    num2 = input("DIGITE O ÚLTIMO NÚMERO DO
SEU RU: ")
    if num2.isdigit():
        num2 = int(num2)
```



```
                break
            else:
                print("Erro: DIGITE UM NÚMERO VÁ-
LIDO!")

        resultado = self.calculadora.resto(n1=num1,
n2=num2)
        print("RESULTADO DO MÓDULO(RESTO):", resul-
tado)

        if op == '7':

            print("***OPERAÇÃO DE RAIZ QUADRADA SELECIO-
NADA ***")
            print("***DIGITE OS DOIS ÚLTIMOS NÚMEROS DO
SEU RU***")

            # LAÇO DE REPETIÇÃO PARA NÚMERO VÁLIDO
            while True:
                num1 = input("DIGITE O PENÚLTIMO NÚMERO DO
SEU RU: ")

                if num1.isdigit():
                    num1 = int(num1)
                    break
                else:
                    print("Erro: DIGITE UM NÚMERO VÁ-
LIDO!")

            while True:
                num2 = input("DIGITE O ÚLTIMO NÚMERO DO
SEU RU: ")

                if num2.isdigit():
                    num2 = int(num2)
                    break
                else:
                    print("Erro: DIGITE UM NÚMERO VÁ-
LIDO!")

            resultado = self.calculadora.raiz(n1=num1,
n2=num2)
            print("RESULTADO DA RAIZ(SOMA DOS DOIS NÚMEROS
DIGITADOS):", resultado)

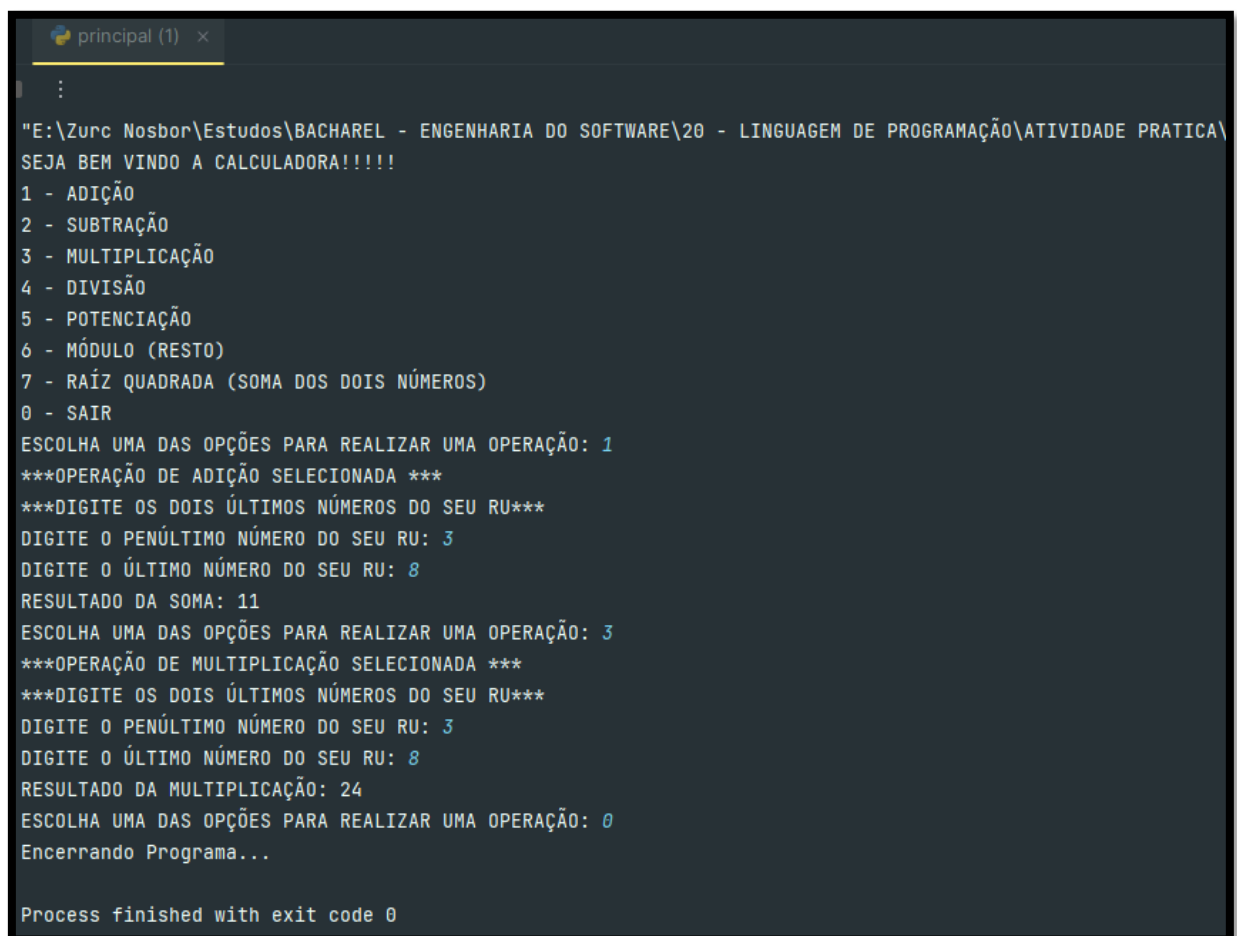
            # OPÇÃO 0: ENCERRA O PROGRAMA
            elif op == '0':
                print('Encerrando Programa...')
                return
```

```
# OPÇÃO INVÁLIDA
elif not op.isdigit() or int(op) not in range(8):
    print('Erro: Digite uma opção válida!!')

# INSTANCIA DA CLASSE PRINCIPAL
p = Principal()

# EXECUÇÃO DOS CALCULOS
p.executar_calculos()
```

CÓDIGO EXERCÍCIO – 01



```
principal (1) x
:
"E:\Zurc Nosbor\Estudos\BACHAREL - ENGENHARIA DO SOFTWARE\20 - LINGUAGEM DE PROGRAMAÇÃO\ATIVIDADE PRÁTICA\
SEJA BEM VINDO A CALCULADORA!!!!
1 - ADIÇÃO
2 - SUBTRAÇÃO
3 - MULTIPLICAÇÃO
4 - DIVISÃO
5 - POTENCIAÇÃO
6 - MÓDULO (RESTO)
7 - RAÍZ QUADRADA (SOMA DOS DOIS NÚMEROS)
0 - SAIR
ESCOLHA UMA DAS OPÇÕES PARA REALIZAR UMA OPERAÇÃO: 1
***OPERAÇÃO DE ADIÇÃO SELECIONADA ***
***DIGITE OS DOIS ÚLTIMOS NÚMEROS DO SEU RU***
DIGITE O PENÚLTIMO NÚMERO DO SEU RU: 3
DIGITE O ÚLTIMO NÚMERO DO SEU RU: 8
RESULTADO DA SOMA: 11
ESCOLHA UMA DAS OPÇÕES PARA REALIZAR UMA OPERAÇÃO: 3
***OPERAÇÃO DE MULTIPLICAÇÃO SELECIONADA ***
***DIGITE OS DOIS ÚLTIMOS NÚMEROS DO SEU RU***
DIGITE O PENÚLTIMO NÚMERO DO SEU RU: 3
DIGITE O ÚLTIMO NÚMERO DO SEU RU: 8
RESULTADO DA MULTIPLICAÇÃO: 24
ESCOLHA UMA DAS OPÇÕES PARA REALIZAR UMA OPERAÇÃO: 0
Encerrando Programa...

Process finished with exit code 0
```

TERMINAL EXERCÍCIO - 01

Exercício 2 – Dada uma determinada equação linear $y = ax + bx - c$, sendo que os valores para **a**, **b** e **c** serão os **três primeiros números de seu RU** (**a = NUM1**, **b = NUM2**, **c = NUM3**). Caso, **algum número do RU seja igual a zero**, substituí-lo(s) pelo **número 5**. Além disso, será preciso criar um vetor aleatório de tamanho 10, onde cada posição do vetor conterá os valores de **x para a equação linear**. Feito isto, fazer um gráfico para mostrar os pontos obtidos pela equação linear dentro do plano cartesiano. Por fim, nomear os eixos Y e X do gráfico, colocar cores diferentes para os pontos e colocar legenda. **Dica:** você vai ter no total, 10 pontos no seu plano cartesiano, ou seja, 10 pontos serão ilustrados no gráfico.

RESPOSTA DO ALUNO:

```
# IMPORT PARA A GERAÇÃO DE GRAFICOS
import matplotlib.pyplot as plt

# IMPORT DO RANDOM PARA GERAR NUMEROS ALEATORIOS
import random

# MEU RU - COMO PEDE NO ENUNCIADO, UTILIZEI OS 3 PRIMEIROS DI-
GITOS
RU = 3773638
a = 3
b = 7
c = 7

# GERA VETOR COM 10 POSIÇÕES COM NÚMEROS ALEÁTORIOS DO NÚMERO
1 AO 25 COMO EXEMPLO
vetorX = random.sample(range(1, 25), 10)

# CALCULA OS VALORES DE Y COM A EQUAÇÃO PARA VALOR DE X NO VE-
TOR
y = [a * vetorXi + b * vetorXi - c for vetorXi in vetorX]

# ORDENA OS PONTOS EM ORDEM CRESCENTE COM BASE NO VETOR
pontos_ordenados = sorted(zip(vetorX, y))

vetorX_ordenado, y_ordenado = zip(*pontos_ordenados)

# PLOTAR OS PONTOS COM CORES
for i in range(len(vetorX_ordenado)):
    plt.plot(vetorX_ordenado[i], y_ordenado[i], marker="o",
markersize=12)

# NOMEAR EIXO X
plt.xlabel('EIXO X')

# NOMEAR EIXO Y
plt.ylabel('EIXO Y')

# LEGENDA DO GRÁFICO
legend_labels = [f'POSIÇÃO X={vetorX_ordenado[i]} POSIÇÃO
Y={y_ordenado[i]}' for i in range(len(vetorX_ordenado))]
plt.legend(legend_labels)

# GRADE NO GRÁFICO
plt.grid()

# TÍTULO DO GRÁFICO
```

```
plt.title('Exercício 02 - EQUAÇÃO LINEAR')

# MOSTRAR GRÁFICO
plt.show()
```

CÓDIGO EXERCÍCIO – 02

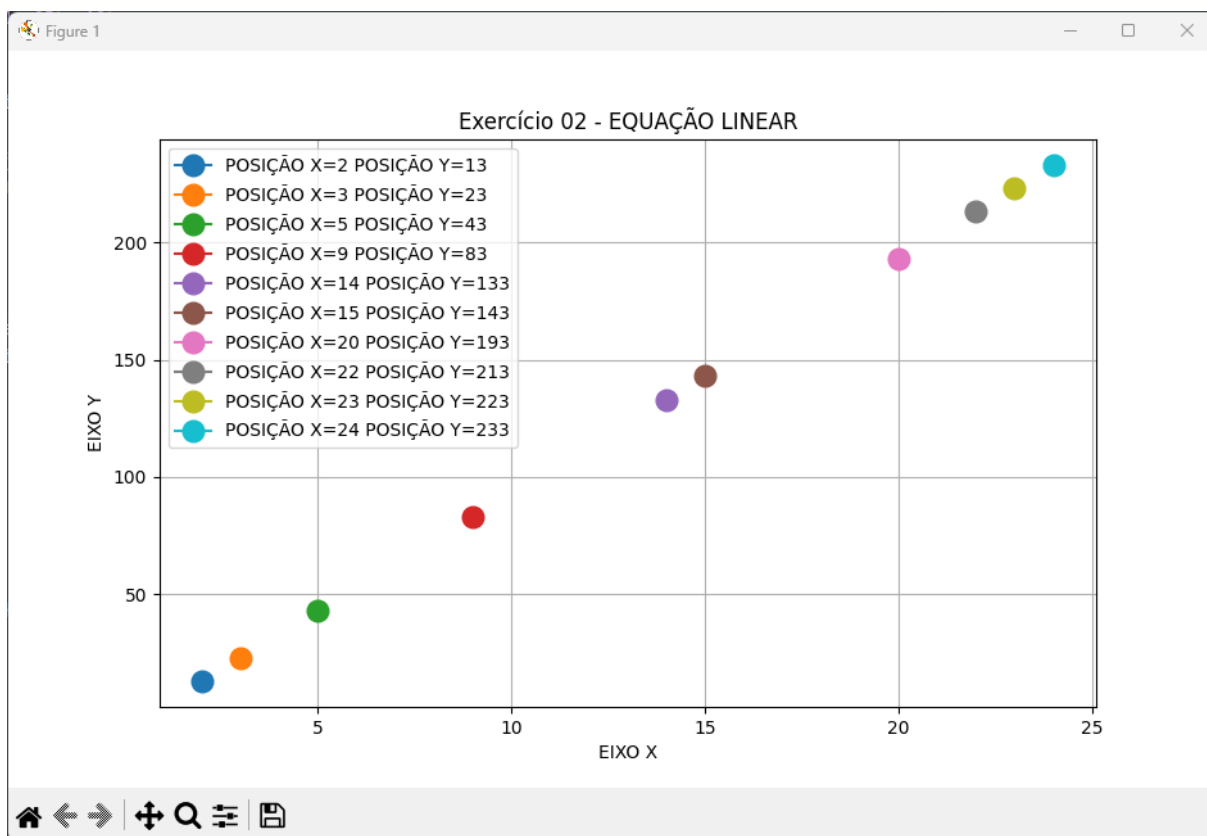


GRÁFICO EXERCÍCIO - 02

Exercício 3 – Realizar o upload do arquivo **STORES.csv**. Renomear todas as colunas do arquivo STORES.csv, onde os respectivos nomes sejam compactados ([Exemplo: Daily Customer Count foi renomeado para Visitantes](#)). Após isto, para se analisar o desempenho das lojas de supermercado/mercado do arquivo STORES.csv encontre os valores **mínimo, máximo, médio e desvio padrão** das seguintes colunas: **"Items_Available"**; **"Daily_Customer_Count"**; e **"Store_Sales"**. Posto isto, realizar três gráficos com as seguintes informações: Items Available, Daily Customer Count e Store Sales. Não se esqueça de colocar: nomes para os eixos Y e X do gráfico, colocar cores diferentes para os pontos e colocar legenda.

Algumas informações extras sobre a tabela do arquivo **STORES.csv**:

- *ID da loja: (Índice) ID da loja específica.*
- *Store da loja: Área Física da loja em pátio.*
- *Itens Avaliados: Número de itens diferentes disponíveis na loja correspondente.*

- *Contagem diária de clientes: Número de clientes que visitaram as lojas em média ao longo do mês.*
- *Histórico de vendas: Vendas em (US\$) que as lojas realizaram.*

RESPOSTA DO ALUNO:

```
# IMPORT DA BIBLIOTECA PANDAS PARA MANIPULAÇÃO E ANÁLISE DE DADOS
import pandas as pd

# IMPORT PARA A GERAÇÃO DE GRAFICOS
import matplotlib.pyplot as plt

# LER O ARQUIVO "Stores.csv" NA PASTA
df = pd.read_csv("Stores.csv", sep=',', encoding='ISO 8859-1')

# RENOMEAR TODAS AS COLUNAS DO ARQUIVO
df = df.rename(columns={
    df.columns[0]: "ID_Loja",
    df.columns[1]: "Produtos",
    df.columns[2]: "Produtos Disponíveis",
    df.columns[3]: "Visitantes",
    df.columns[4]: "Vendas (DOLAR)",
})

# VER O ARQUIVO NO TERMINAL
print(df)

# SELECIONAR AS COLUNAS PARA BUSCAR OS DADOS: PRODUTOS DISPONÍVEIS, VISITANTES E VENDAS DA LOJA
col_interesse = ["Produtos Disponíveis", "Visitantes", "Vendas (DOLAR)"]
df_col_interesse = df[col_interesse]

# BUSCAR OS DADOS MÍNIMO, MÁXIMO E DESVIO PADRÃO
dados = df_col_interesse.agg(["min", "max", "std", "mean"])

# VER O ARQUIVO NO TERMINAL
print(dados)

# AMOSTRA DE 30% DOS DADOS DA COLUNA "PRODUTOS DISPONÍVEIS" DE ACORDO COM A MENSAGEM DA TUTORIA
amostra = df["Produtos Disponíveis"].sample(frac=0.3)

# GRÁFICO DE PRODUTOS DISPONÍVEIS, ACABEI NÃO COLOCANDO CORES ALEATÓRIAS PELA FATO QUE GEROU TRAVAMENTOS NO PLANO CARTESIANO
plt.scatter(amostra.index, amostra, color='blue', label='Produtos Disponíveis')
```

```
plt.xlabel('Índice')
plt.ylabel('Produtos Disponíveis')
plt.title('Análise de Produtos Disponíveis')
plt.legend()
plt.grid()
plt.show()

# AMOSTRA DE 30% DOS DADOS DA COLUNA "VISITANTES" DE ACORDO
COM A MENSAGEM DA TUTORIA
amostra_2 = df["Visitantes"].sample(frac=0.3)

#GRÁFICO DE PRODUTOS DISPONÍVEIS, ACABEI NÃO COLOCANDO CORES
ALEATÓRIAS PELA FATO QUE GEROU TRAVAMENTOS NO PLANO CARTESIANO
plt.scatter(amostra_2.index, amostra_2, color='green', la-
bel='Visitantes')

plt.xlabel('Índice')
plt.ylabel('Visitantes')
plt.title('Análise de Visitantes')
plt.legend()
plt.grid()
plt.show()

# AMOSTRA DE 30% DOS DADOS DA COLUNA "Vendas (DOLAR)" DE
ACORDO COM A MENSAGEM DA TUTORIA
amostra_3 = df["Vendas (DOLAR)"].sample(frac=0.3)

#GRÁFICO DE PRODUTOS DISPONÍVEIS, ACABEI NÃO COLOCANDO CORES
ALEATÓRIAS PELA FATO QUE GEROU TRAVAMENTOS NO PLANO CARTESIANO
plt.scatter(amostra_3.index, amostra_3, color='red', la-
bel='Vendas da Loja (DOLAR)')

plt.xlabel('Índice')
plt.ylabel('Vendas da Loja (DOLAR)')
plt.title('Análise de Vendas da Loja (DOLAR)')
plt.legend()
plt.grid()
plt.show()
```

```
"E:\Zurc Nosbor\Estudos\BACHAREL - ENGENHARIA DO SOFTWARE\20 - LINGUAGEM DE PROGRAMAÇÃO\ATIVIDADE PRÁTICA\
ID_Loja  Produtos  Produtos Disponíveis  Visitantes  Vendas (DOLAR)
0        1        1659                1961        530        66490
1        2        1461                1752        210        39820
2        3        1340                1609        720        54010
3        4        1451                1748        620        53730
4        5        1770                2111        450        46620
..      ...      ...                ...        ...        ...
891     892     1582                1910       1080       66390
892     893     1387                1663        850       82080
893     894     1200                1436       1060       76440
894     895     1299                1560        770       96610
895     896     1174                1429       1110       54340

[896 rows x 5 columns]
      Produtos  Disponíveis  Visitantes  Vendas (DOLAR)
min           932.000000    10.000000    14920.000000
max          2667.000000   1560.000000   116320.000000
std           299.872053    265.389281    17190.741895
mean          1782.035714    786.350446    59351.305804
```

TERMINAL EXERCÍCIO - 03

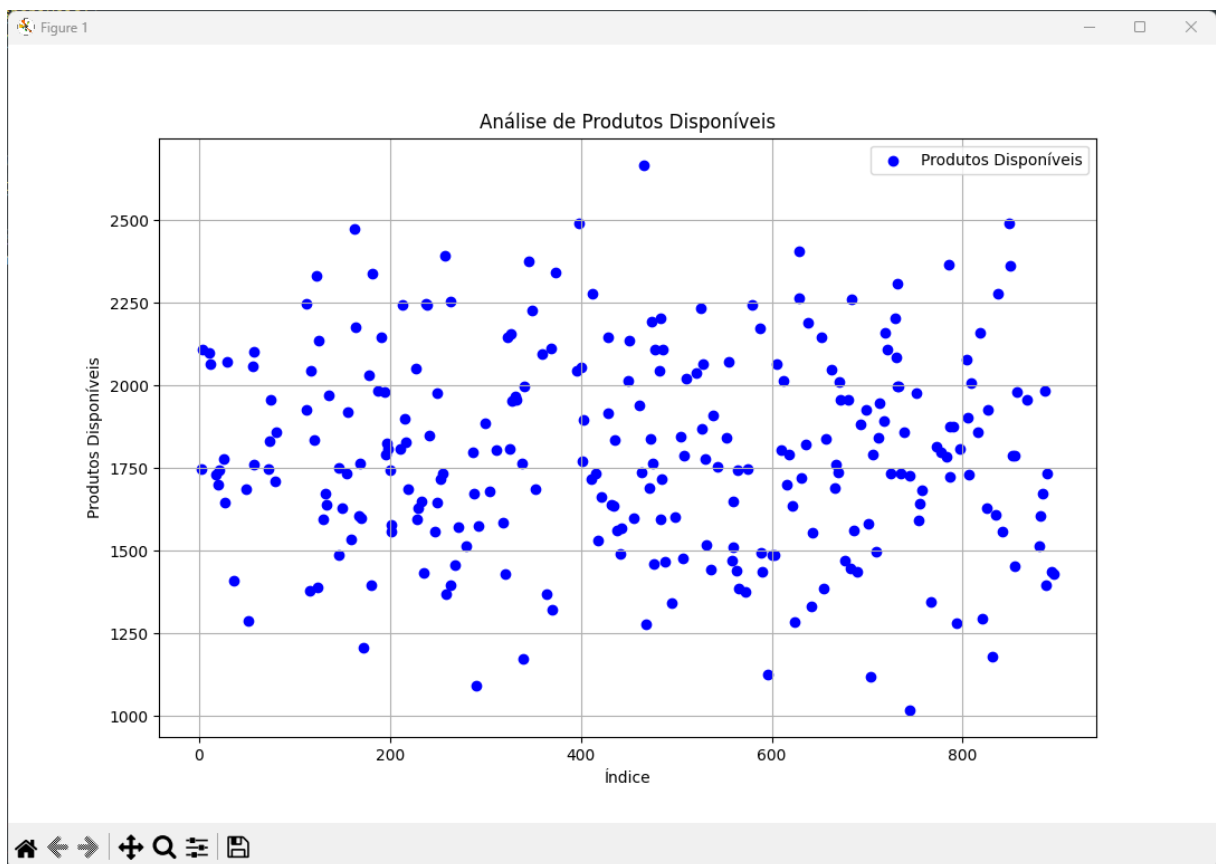


GRÁFICO 01 EXERCÍCIO - 03

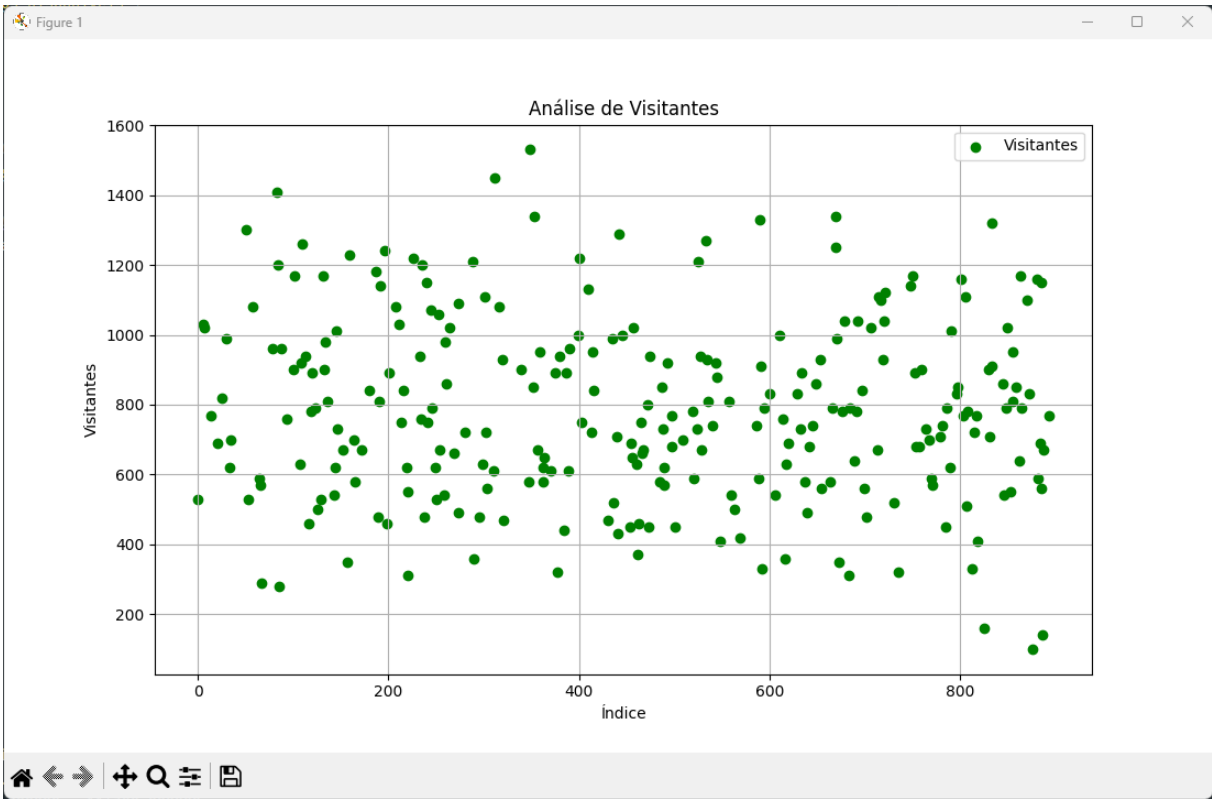


GRÁFICO 02 EXERCÍCIO – 03

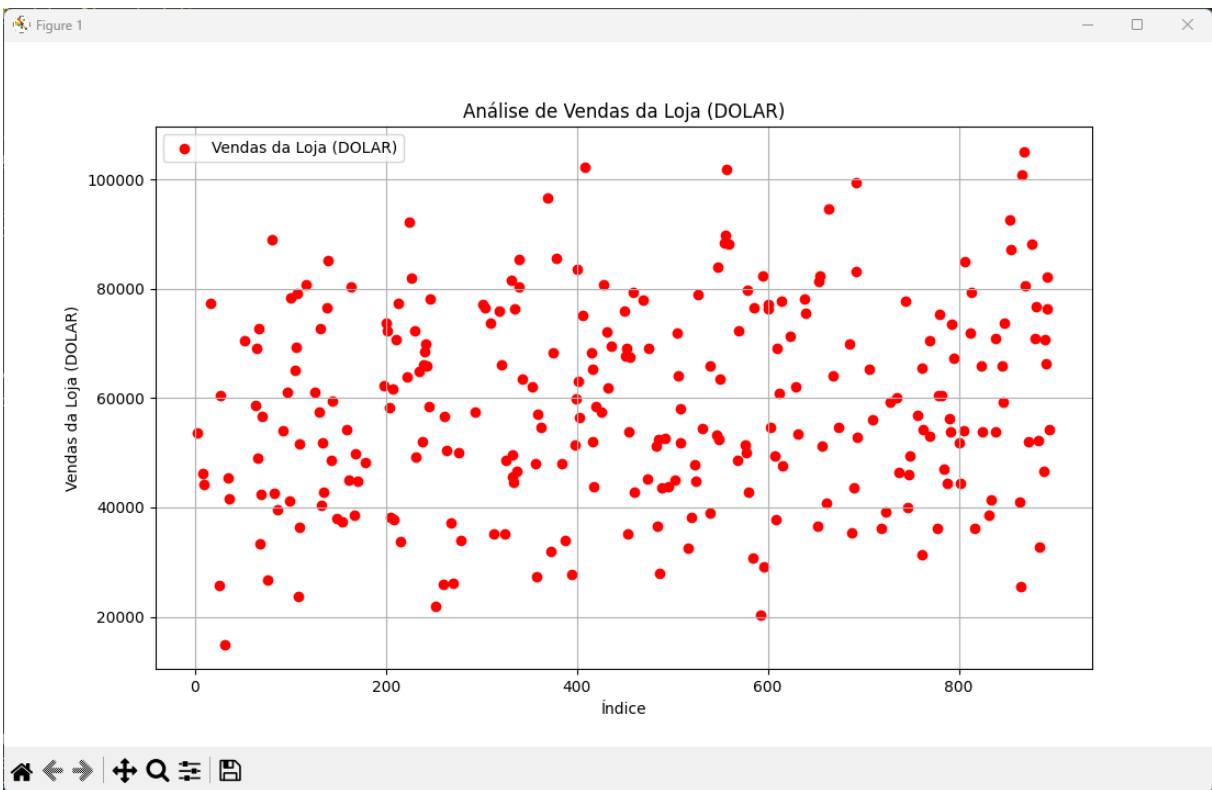


GRÁFICO 03 EXERCÍCIO - 03