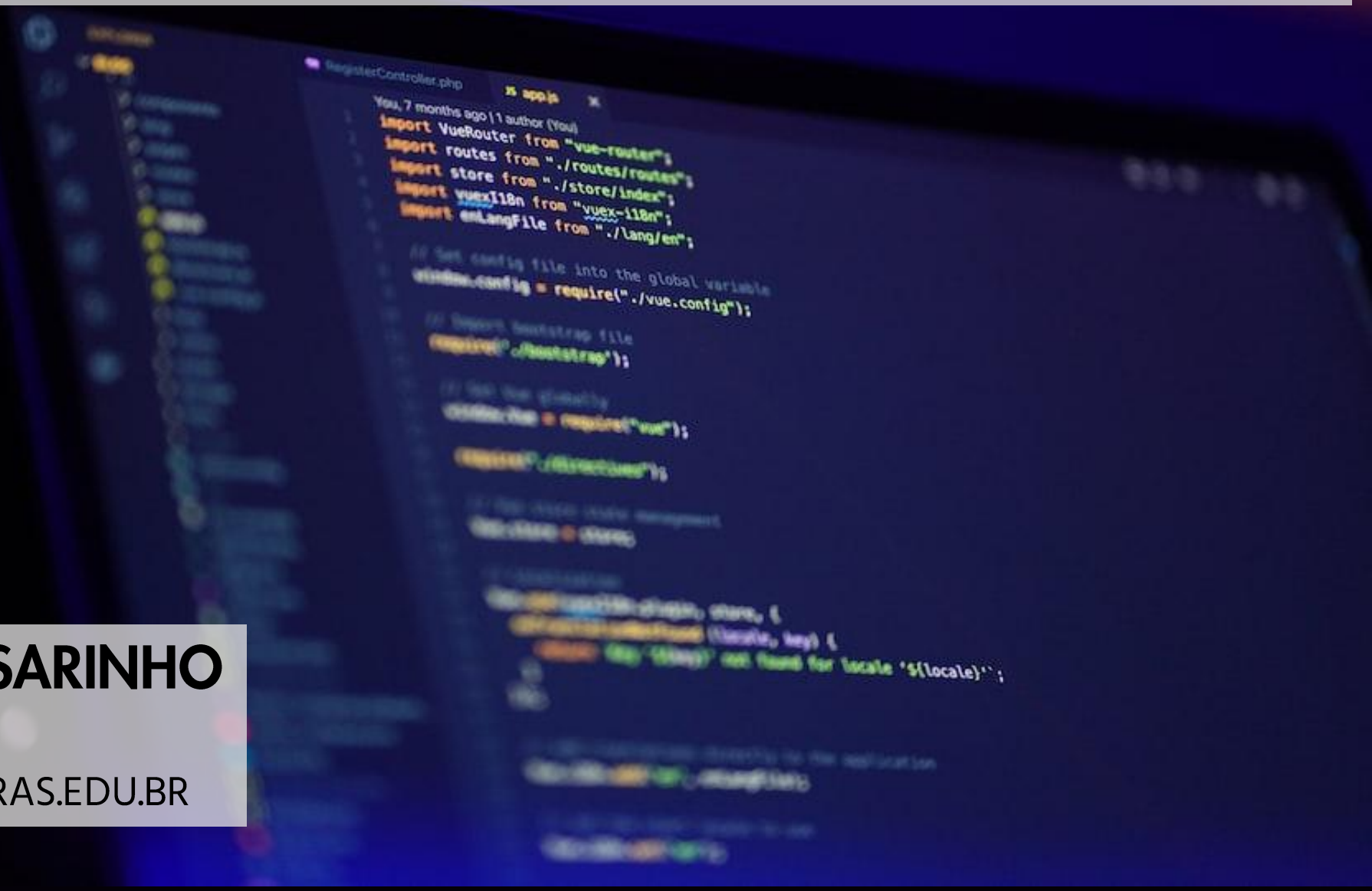
ALGORITMOS E LÓGICA DE PROGRAMAÇÃO

AULA 10 – MATRIZES E FUNÇÕES



WALTER TRAVASSOS SARINHO

@PROF.WALTERTRAVASSOS

WALTER.TRAVASSOS@SECTRAS.EDU.BR

Calendário de Aulas

- 13/11/2025 – Matrizes e Funções
- 20/11/2025 - Feriado
- 27/11/2025 – Manipulando Strings e Arquivos TXT.
- **04/11/2025 – 3ª Avaliação**
- **10/11/2025 – Apresentação do Fórum Integrador (Quarta)**
- **11/11/2025 – Avaliação Final**

Aula de Hoje

- Matrizes
- Funções

Qual a diferença entre Listas e Vetores?

ChatGPT

Qual a principal diferença entre vetores e listas no python?

A principal diferença entre vetores e listas no contexto de Python é que, em termos práticos, o conceito de "vetor" geralmente se refere a uma estrutura de dados homogênea, com elementos do mesmo tipo, enquanto "listas" são estruturas de dados heterogêneas que podem conter diferentes tipos de elementos.

Listas

- As listas em Python são estruturas de dados flexíveis e dinâmicas que podem conter elementos de diferentes tipos (**diferente do conceito da estrutura de dados vetores**).
- Elas podem ser alteradas (adicionar, remover, modificar elementos) e **são mais versáteis do que vetores**.
- Frequentemente utilizamos as Listas para representar vetores e matrizes.

Matrizes

Matrizes

- Listas aninhadas são frequentemente utilizadas para representar matrizes.
- Ex.: Dada a matriz:

- Poderíamos representa-la em Python da seguinte forma:

```
matriz = [ [1, 2, 3],[4, 5, 6],[7, 8, 9] ]
```

Matrizes

- Podemos selecionar **uma linha** (uma lista) de uma matriz:

```
matriz = [ [1, 2, 3],[4, 5, 6],[7, 8, 9] ]
print(matriz[1])
```

- Também podemos extrair **um único elemento** de uma matriz:

```
matriz = [ [1, 2, 3],[4, 5, 6],[7, 8, 9] ]
print(matriz[1][1])
```

Exemplo 1

Faça um programa que lê 4 números inteiros e adiciona cada um deles numa matriz 2x2.

Em seguida, o programa **deverá escrever cada número na tela.**

Exemplo 1 – Resposta 1

```
numeros = [
            [int(input("Digite a primeira idade:")),int(input("Digite a segunda idade:"))],
            [int(input("Digite a terceira idade:")),int(input("Digite a quarta idade:"))]
          ]

print(numeros)
```

Exemplo 1 – Resposta 2

```
numeros = [
            [0,0],
            [0,0]
          ]

for i in range(2):
    for j in range(2):
        numeros[i][j] = int(input('Digite um número: '))

print(numeros)
```

Exemplo 1 – Resposta 3

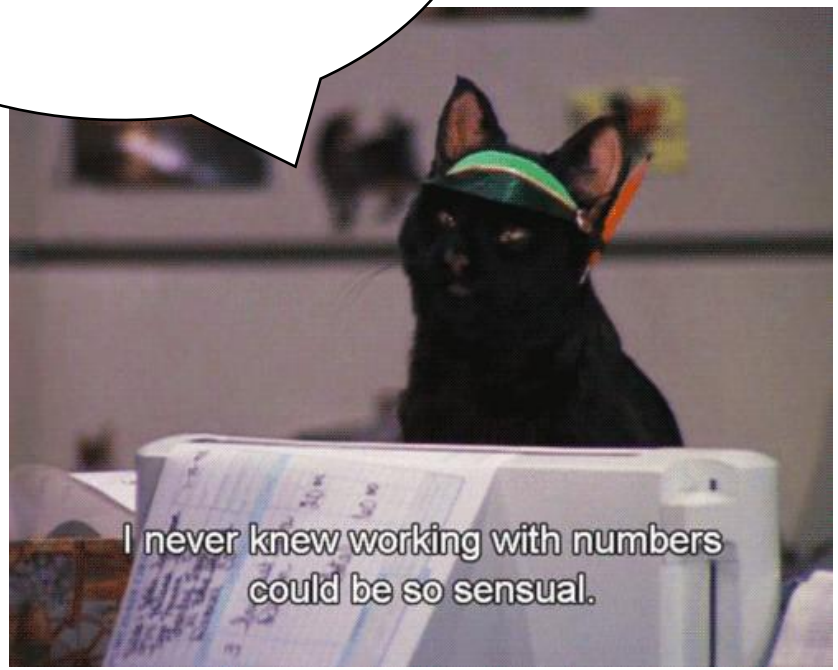
```
numeros = []

for i in range(2):
    linha = []
    for j in range(2):
        numero = int(input('Digite um número: '))
        linha.append(numero)
    numeros.append(linha)

print(numeros)
```

ACHO QUE NÃO ENTENDI MUITO BEM...

ABRAM O CHATGPT!



ChatGPT

Em 3 minutos:

- Peçam pro ChatGPT explicar o conceito do uso de matrizes em Python. Leiam!
- Conseguiram entender?
- Peça agora pra ele fazer **a mesma explicação utilizando o contexto de algum personagem que você gosta muito ou que marcou sua infância**. Por exemplo: “Explica a mesma coisa utilizando o universo do Dragonball Z”.

Exemplo 2

Faça um programa que lê 4 números inteiros e adiciona cada um deles numa matriz 2x2.

Em seguida, o programa deverá **escrever apenas a segunda linha**.

Exemplo 2 – Resposta 1

```
numeros = [
            [int(input("Digite a primeira idade:")),int(input("Digite a segunda idade:"))],
            [int(input("Digite a terceira idade:")),int(input("Digite a quarta idade:"))]
          ]

print(numeros[1])
```

Exemplo 2 – Resposta 2

```
numeros = [
            [0,0],
            [0,0]
          ]

for i in range(2):
    for j in range(2):
        numeros[i][j] = int(input('Digite um número: '))

print(numeros[1])
```

Exemplo 2 – Resposta 3

```
numeros = []

for i in range(2):
    linha = []
    for j in range(2):
        numero = int(input('Digite um número: '))
        linha.append(numero)
    numeros.append(linha)

print(numeros[1])
```

Exemplo 3

Faça um programa que lê 4 números inteiros e adiciona cada um deles numa matriz 2x2.

Em seguida, o programa **deverá escrever apenas o número localizado na primeira linha, segunda coluna**.

Exemplo 3 – Resposta 1

```
numeros = [
            [int(input("Digite a primeira idade:")),int(input("Digite a segunda idade:"))],
            [int(input("Digite a terceira idade:")),int(input("Digite a quarta idade:"))]
          ]

print(numeros[0][1])
```

Exemplo 3 – Resposta 2

```
numeros = [
            [0,0],
            [0,0]
          ]

for i in range(2):
    for j in range(2):
        numeros[i][j] = int(input('Digite um número: '))

print(numeros[0][1])
```

Exemplo 3 – Resposta 3

```
numeros = []

for i in range(2):
    linha = []
    for j in range(2):
        numero = int(input('Digite um número: '))
        linha.append(numero)
    numeros.append(linha)

print(numeros[0][1])
```

Exemplo 4

Faça um programa que receba o peso de 9 pessoas em uma matriz 3x3 e escreve o maior peso digitado.

Exemplo 4 – Resposta

```
pesos = [
            [0,0,0],
            [0,0,0],
            [0,0,0]
        ]

maior = 0

for i in range(3):
    for j in range(3):
        pesos[i][j] = float(input('Peso: '))
        if pesos[i][j] > maior:
            maior = pesos[i][j]

print(maior)
```

Exercícios

Exercício 1

Faça um programa que lê a altura de 4 pessoas em uma matriz 2x2, calcula e exibe **a quantidade de pessoas com altura superior a 1.50 metros.**

Exercício 1 - Resposta

```python
matriz = [
            [float(input("Primeiro altura?")), float(input("Segundo altura?"))],
            [float(input("Terceiro altura?")), float(input("Quarto altura?"))]
          ]
quantidade = 0
for i in range(2):
  for j in range(2):
    if matriz[i][j] > 1.5:
      quantidade+=1



print(quantidade)

```

Exercício 2

Faça um programa que armazena valores inteiros em uma matriz 2x3.

Em seguida, calcula e exibe a soma dos valores positivos contidos na matriz.

Exercício 4

Faça um programa que lê 6 números reais em uma matriz 3x2, calcula e exibe a quantidade de números negativos e a soma dos números positivos dessa mesma matriz.

Exercício 5

Faça um programa que receba o nome de 9 produtos em uma matriz 3x3.

Logo em seguida, o programa deverá solicitar ao usuário a linha e a coluna do produto a ser acessado, imprimindo seu nome.

Exercício 5 - Resposta

```python
matriz = [
            [0,0,0],
            [0,0,0],
            [0,0,0]
          ]

for i in range(3):
  for j in range(3):
    matriz[i][j] = input("Digite um produto: ")

linha = int(input("Qual a linha?"))
coluna = int(input("Qual a coluna?"))
print(matriz[linha][coluna])
```

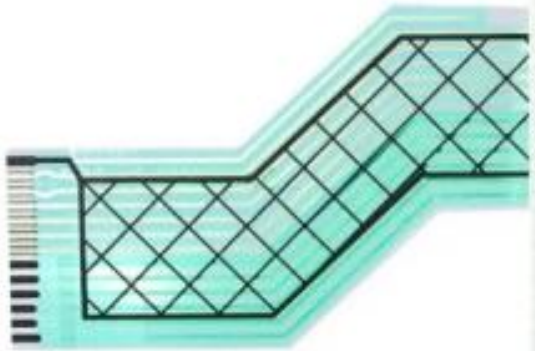
Exercício 6

Faça um programa que cadastre a senha de 4 usuários em uma matriz 2x2.

Em seguida, o programa deverá solicitar ao usuário sua senha. Caso a senha digitada esteja cadastrada na matriz de senhas, o programa deverá exibir a mensagem “Bem vindo!”. Caso contrário, deverá exibir a frase “Usuário não cadastrado”;

Intervalo: 15 minutos

Funções




TRAVA DE SEGURANÇA
TIRA ODOR
AQUECER PRATO PRONTO
DESCONGELAR
CARNE
FRANGO
PEIXE
FEIJÃO
DIA A DIA
ARROZ
MACARRÃO
BATATA
KIDS
B. CANECA
PIPOCA
BRIGADEIRO
LIGHT
FRUTAS
VEGETAIS
SOPAS
1
2
3
4
5
6
7
8
9
MANTER AQUECIDO
0
POTÊNCIA PORÇÃO
CANCELA PAUSA
INÍCIO +30 s

Funções

- Definem ações a serem tomadas na execução de um programa.
- São conjuntos de declarações de dados, instruções e expressões.
- Tratam-se de **blocos nomeados de código**, que exercem uma função específica (Ex.: soma, imprime, calcula ...).

Vantagens do uso de Funções

- Reduzem o tamanho do código-fonte de programas.
- Facilitam a visualização e compreensão de programas.
- Pensa-se na solução do problema por partes.
- É mais fácil corrigir e detectar erros.
- Se é preciso alterar, altera-se apenas uma vez.
- Uma mesma função poderá ser utilizada em outros scripts.

Funções famosas do Python

1. **print()**: Exibe mensagens na tela.
2. **input()**: Aceita entrada do usuário a partir do teclado.
3. **len()**: Retorna o tamanho (número de itens) de um objeto.
4. **range()**: Gera uma sequência de números (usada no for).
5. **int(), float(), str()**: Funções de conversão de tipos de dados.
6. **type()**: Retorna o tipo de um objeto.
7. **sum()**: Retorna a soma de todos os itens em um iterável.
8. **max(), min()**: Retorna o maior/menor item em um iterável.
9. **sorted()**: Retorna uma lista ordenada a partir de um iterável.
10. **zip()**: Combina elementos de iteráveis em tuplas.

Funções famosas do Python

11. **map()**: Aplica uma função a todos os itens em um iterável.
12. **filter()**: Filtra os itens de um iterável com base em uma função.
13. **open()**: Abre um arquivo.
14. **str.upper(), str.lower()**: Converte uma string para maiúsculas/minúsculas.
15. **str.strip()**: Remove espaços em branco do início e do fim de uma string.
16. **str.split()**: Divide uma string em uma lista de substrings.
17. **str.join()**: Junta elementos de uma lista em uma string usando um separador.
18. **os.path.exists()**: Verifica se um caminho de arquivo ou diretório existe.
19. **os.listdir()**: Retorna uma lista dos arquivos e diretórios no caminho especificado.
20. **random.choice(), random.randint()**: Funções do módulo random para seleção aleatória e geração de números inteiros aleatórios.

Definindo Funções

Além da possibilidade de utilizar as funções famosas que já estão prontas, nós podemos criar nossas próprias funções.

Definindo Funções

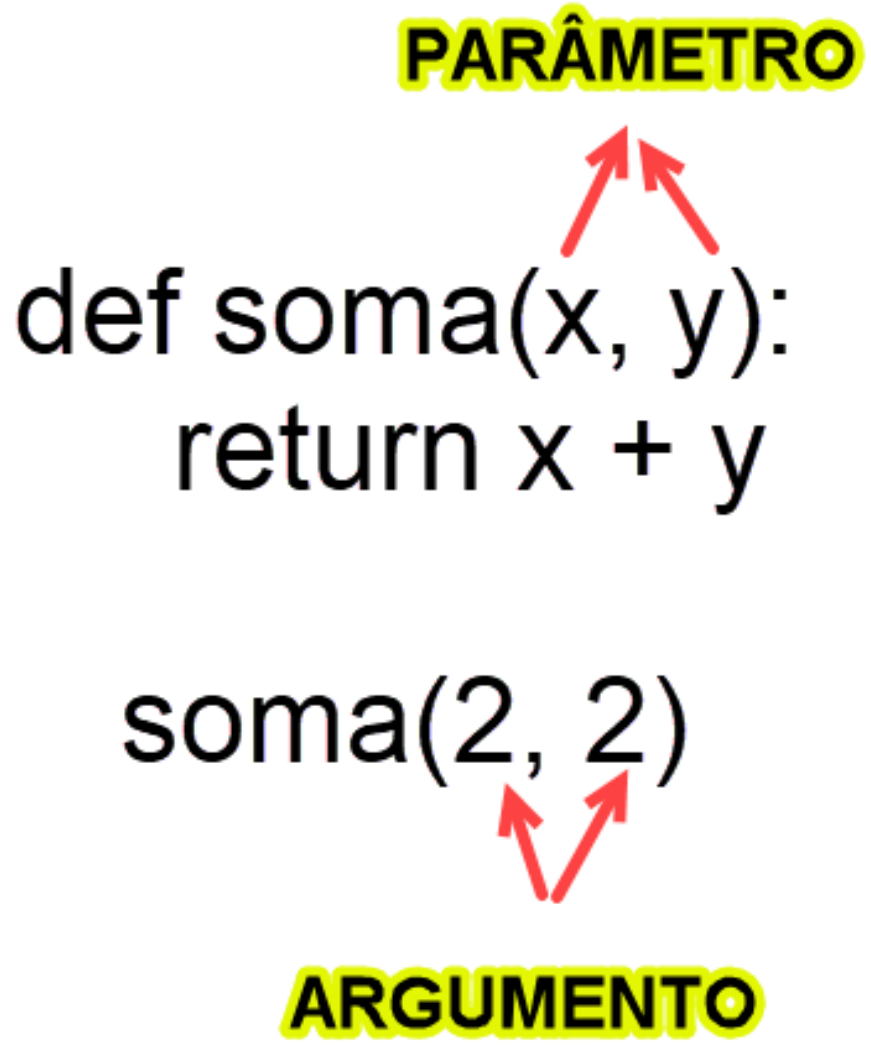
Sintaxe:

```
def nome_funcao(param1,param2,..., param_n):
    # Bloco de código da função

    return valor
```

Parâmetro ou Argumento?

- Os **parâmetros** estão dentro da função **soma**. São as variáveis x e y.
- Já os **argumentos**, são os **valores** que passamos quando utilizamos a função soma. No caso do exemplo ao lado, os argumentos são 2 e 2.



O que significa retornar valor?

Significa retornar algo para quem chamou a função. Por exemplo:

v1 = mediaAritmética(8, 10);

Observe como a função mediaAritmética(param1, param2) retorna um valor para a variável v1.

Exemplo 1

Função

- Faça um programa que possua uma função chamada **soma**.
- Esta função deverá retornar a soma entre duas variáveis sempre que for invocada.

Programa

- No mesmo módulo (mesmo arquivo), leia dois números inteiros e some-os a partir da função **soma** criada anteriormente.

Exemplo 1 - Resposta

```
def soma(x,y):
    return x+y

num1 = int(input("Digite o primeiro número: "))
num2 = int(input("Digite o segundo número: "))

resultado = soma(num1, num2)

print(resultado)
```

Exemplo 2

- Faça um programa que possua dois módulos: **funcoes.py** e **main.py**.

- No primeiro arquivo, crie a função soma. Esta função deverá retornar a soma entre duas variáveis sempre que for invocada.
- No segundo arquivo, importe a função soma, localizada no módulo **funcoes.py**, leia dois números inteiros e some-os a partir da função criada anteriormente.

Exemplo 2 – Resposta – parte 1

Criação do módulo **funcoes.py** (no arquivo de mesmo nome)

```
def soma(x,y):
    return x+y
```

Exemplo 2 – Resposta – parte 2

Criação do módulo **main.py** (aqui a gente importa o arquivo).

```
from funcoes import soma

num1 = int(input("Número 1: "))
num2 = int(input("Número 2: "))

resultado = soma(num1,num2)

print(resultado)
```

Exemplo 3

- Faça um programa que possua uma função chamada **aoQuadrado**.
- Essa função deverá retornar o valor ao quadrado de uma variável sempre que for invocada.

- No mesmo arquivo, leia um número inteiro e eleve-o ao quadrado a partir da função criada anteriormente.

Exemplo 3 - Resposta

```
def aoQuadrado(x):
    return x * x

num = int(input("Número: "))

resultado = aoQuadrado(num)

print(resultado)
```

Exemplo 4

- Faça um programa que possua dois módulos: **funcoes.py** e **main.py**.
- No primeiro arquivo, crie a função **aoQuadrado**. Esta função deverá retornar o valor ao quadrado de uma variável sempre que for invocada.
- No segundo arquivo, importe a função **aoQuadrado**, localizada no módulo funções, leia um número inteiros e execute a função criada anteriormente.

Exemplo 4 – Resposta – parte 1

Criação do módulo funcoes.py:

```
def aoQuadrado(x):
    return x * x
```

Exemplo 4 – Resposta – parte 1

Criação do módulo programa.py:

```
from funcoes import aoQuadrado

num = int(input("Número: "))

resultado = aoQuadrado(num)

print(resultado)
```

Exercícios

Exercício 1

Faça um programa que lê os lados de um retângulo e calcula o seu perímetro a partir de uma função.

Perímetro do retângulo = (2 * largura) + (2 * comprimento)

Exercício 2

Faça um programa que lê 3 notas de um aluno no semestre, calcula sua média **a partir de uma função** e informa se o aluno está aprovado (media >= 7) ou reprovado (media < 7).

A função não tem retorno, ela mostra no console o resultado.

Exemplo: **calculaMedia(7, 8, 5)**

Exercício 2 - Resposta

funcoes.py

```python
def calculaMedia(x,y,z):
    media = (x+y+z)/3
    if media>=7:
        print('Aprovado. A média foi ', media)
    else:
        print('Reprovado. A média foi ', media)
```

main.py

```python
from funcoes import calculaMedia

nota1 = float(input("Digite a primeira nota: "))
nota2 = float(input("Digite a segunda nota: "))
nota3 = float(input("Digite a terceira nota: "))

calculaMedia(nota1,nota2,nota3)
```

```python
def calculaMedia(x, y, z):
    media = (x+y+z)/3
    if media>=7:
        return "Aprovado ", media
    else:
        return "Reprovado ", media

n1 = int(input("1ª nota: "))
n2 = int(input("2ª nota: "))
n3 = int(input("3ª nota: "))

print(calculaMedia(n1,n2,n3))
```

Exercício 3

Faça um programa que leia a variação da distância percorrida por um carro e a variação de tempo que ele levou para percorrer o trajeto e calcula, a partir de uma função, a velocidade média do veículo.

Velocidade media = (Km final – km inicial) / (hora final – hora inicial).

Exercício 4

Faça um programa que lê o preço de um produto e a quantidade adquirida por um cliente.

O mesmo deverá calcular, a partir de uma função, o valor total a ser pago pelo cliente.

Exercício 5

Faça um programa que leia um número inteiro e o submeta para a função **checaPositivo** (crie a função), que deverá informar se o número digitado é positivo ou negativo.

Exercício 6

Faça um programa que leia dois números inteiros e informa, a partir de uma função, qual o maior número digitado.

Exercício 6 - Resposta

funcoes.py

```python
def maior(x, y):
    if x > y:
        return x
    else:
        return y
```

main.py

```python
from funcoes import maior

n1 = int(input("Digite um número"))
n2 = int(input("Digite outro número"))

print("O maior é ", maior(n1,n2))
```

Exercício 7

Faça um programa que leia o raio de uma esfera e submeta os dados para a função volume (crie a função), que deverá calcular o seu volume.

V = 4/3 * (R*R*R)

Exercício 8

Faça um programa que leia dois números reais e um símbolo que identifique uma operação matemática (+, -, *, /), submetendo-os para a função calculadora (crie a função).

A função deverá efetuar um cálculo entre os dois números submetidos, baseado no símbolo digitado.

Exercício 9

- Faça um programa que leia três inteiros que representam horas, minutos e segundos e submeta os dados para a função converte (crie a função), que deverá converter os três inteiros digitados para segundos.
- Por exemplo: 2h 40min e 10s correspondem a 9.610 segundos.

Exercício 9 - Resposta

```python
from funcoes import converte

h = int(input("Digite as horas: "))
m = int(input("Digite os minutos: "))
s = int(input("Digite os segundos: "))

segundos = converte(h,m,s)
print("O total de segundos é: ", segundos)
```

funcoes.py

```python
def converte(h,m,s):
    return s + (m*60) + (h*60*60)

```

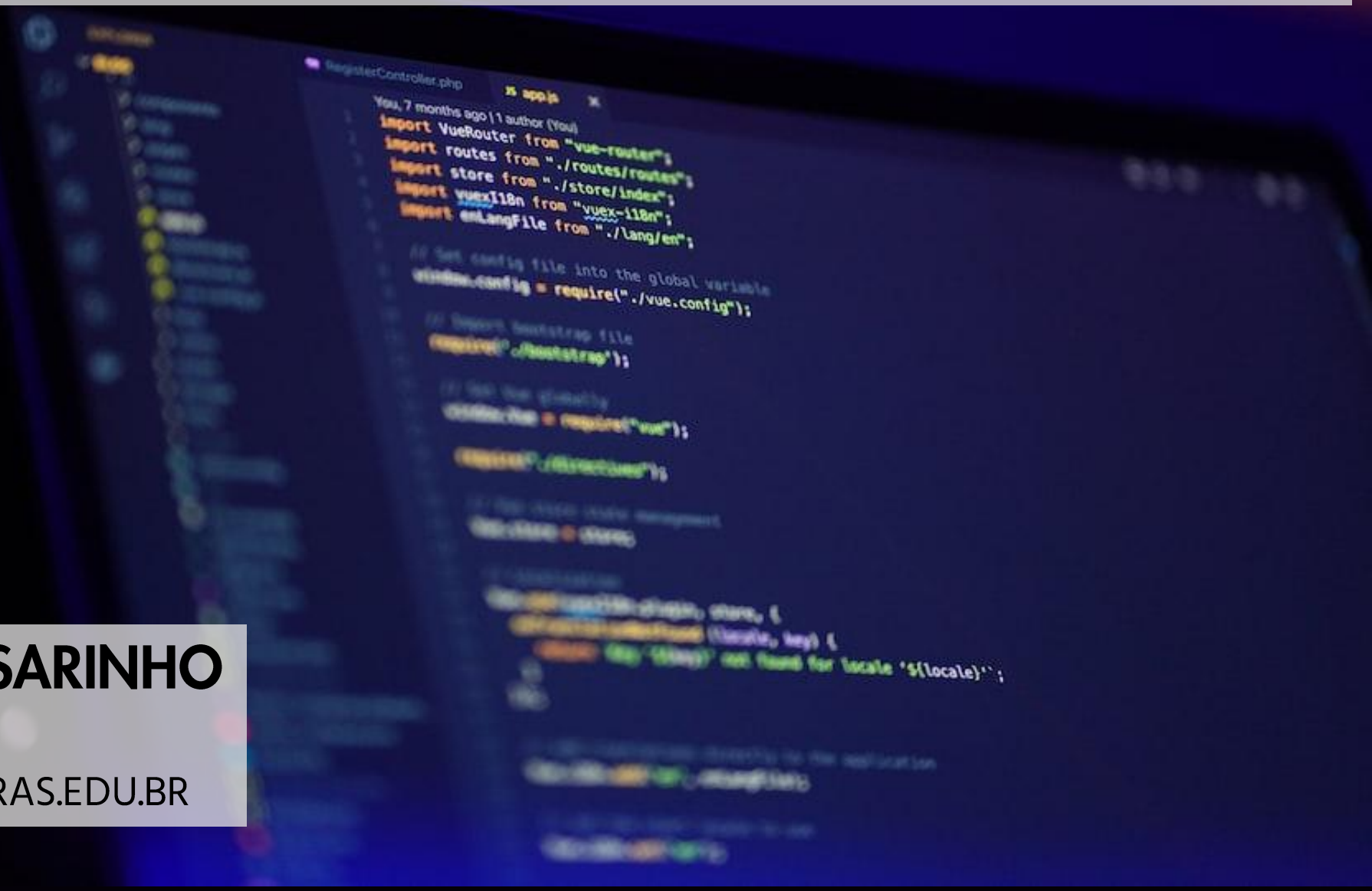
Exercício 10

Faça um programa que receba dois números e execute as operações listadas a seguir, de acordo com a escolha do usuário (crie uma função para cada opção).

CÓDIGO	OPERAÇÃO
1	Média entre os números digitados
2	Diferença do maior pelo menor
3	Produto entre os números digitados
4	Divisão do primeiro pelo segundo

ALGORITMOS E LÓGICA DE PROGRAMAÇÃO

AULA 10 – MATRIZES E FUNÇÕES



WALTER TRAVASSOS SARINHO

@PROF.WALTERTRAVASSOS

WALTER.TRAVASSOS@SECTRAS.EDU.BR