

**CURSO: DESENVOLVIMENTO FULL STACK**  
**ALUNA: CAMILA DO CARMO PEREIRA EVANGELISTA -**  
**MATRÍCULA: 202408415567**

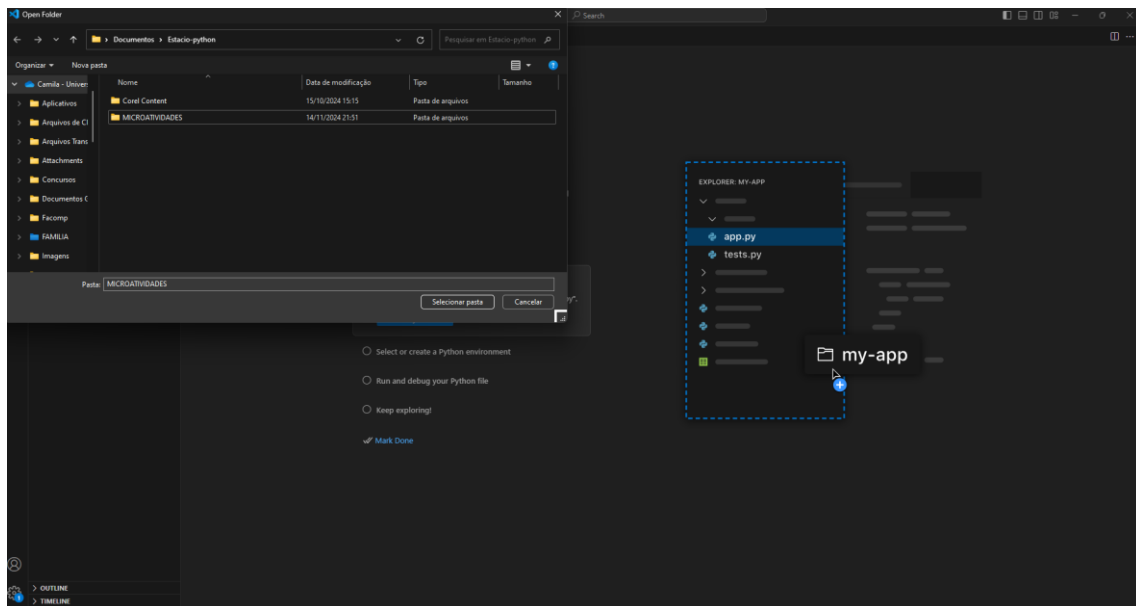
**Microatividade 1: Descrever a utilização das estruturas de condição  
if e else em Python**

**- Material necessário para a prática**

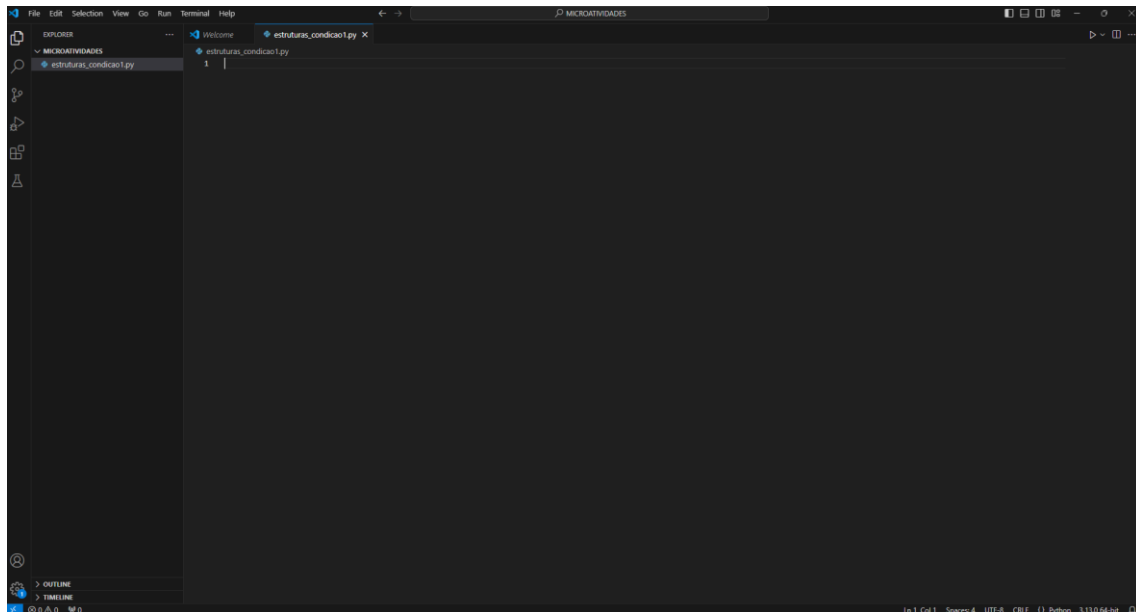
- Interpretador Python instalado no Sistema Operacional;
- IDE VS Code instalada no Sistema Operacional;

**- Procedimentos**

1. Abra a IDE VS Code;
2. No menu File, selecione a opção “Open Folder”;
3. Crie uma nova pasta em seu computador para armazenar os códigos desse conjunto de microatividades e a selecione a partir do VS Code;

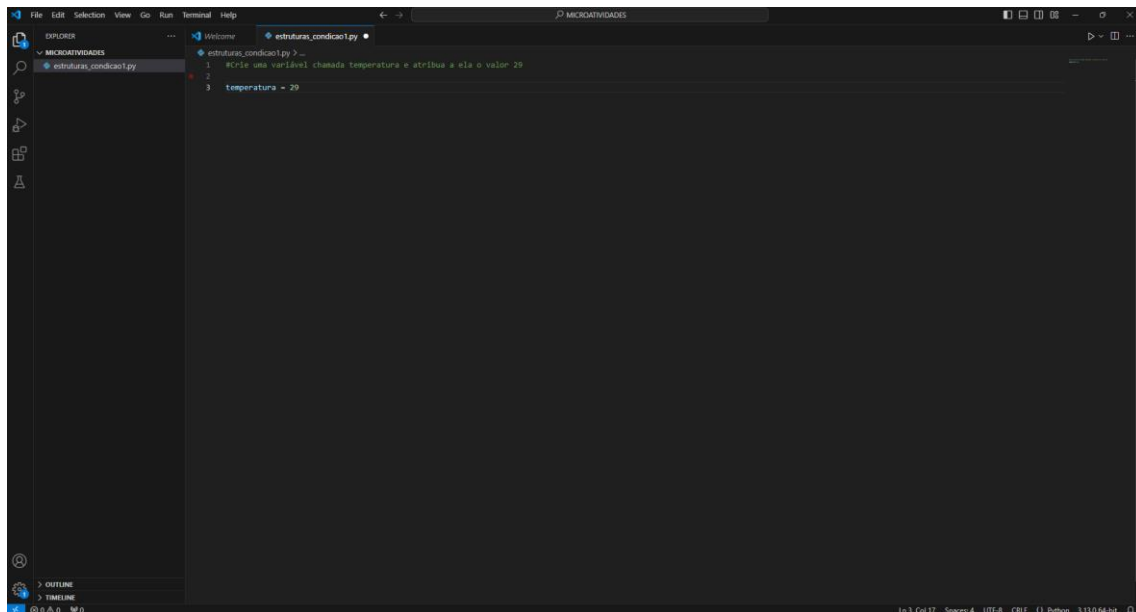


4. No VS Code, menu Explorer, clique na pasta do projeto e crie um novo arquivo/script chamado “estruturas\_condicao1.py”;



5. No script criado:

a. Crie uma variável chamada temperatura e atribua a ela o valor 29;

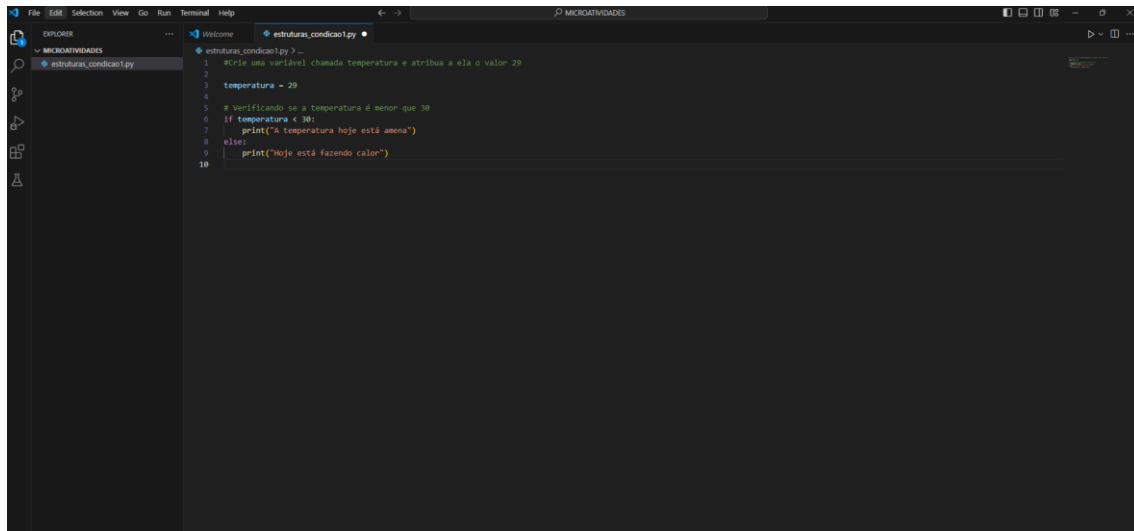


b. Crie uma verificação, utilizando a condição if, para checar se o valor da variável

c. temperatura é menor que 30;

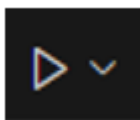
d. Caso positivo, imprima na tela o texto 'A temperatura hoje está amena';

e. Caso contrário, e utilizando a condição else, imprima na tela o texto 'Hoje está fazendo calor';

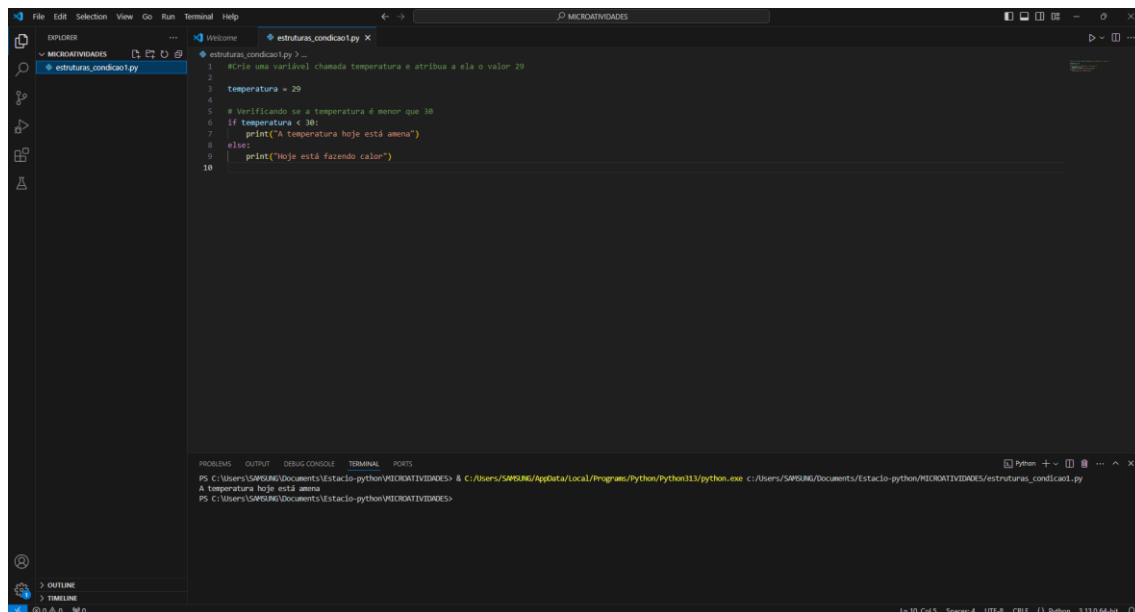


```
1 #Crie uma variável chamada temperatura e atribua o valor 29
2
3 temperatura = 29
4
5 # Verificando se a temperatura é menor que 30
6 if temperatura < 30:
7     print("A temperatura hoje está amena")
8 else:
9     print("Hoje está fazendo calor")
10
```

6. Salve o arquivo/script;
7. Na barra superior direita você verá um ícone no formato de um triângulo deitado, semelhante à imagem abaixo:



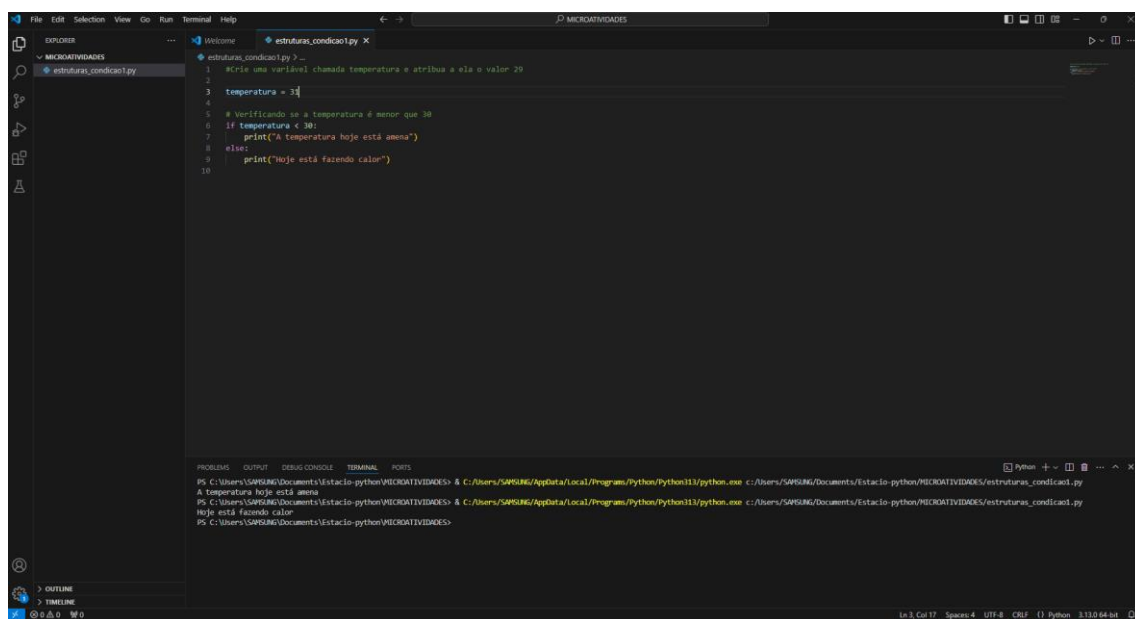
8. Clique no ícone. A seguir, deverá ser aberta uma janela, na parte inferior do VS Code, exibindo o terminal e a saída dos comandos acima, que é a exibição da frase digitada após o comando “print”;
9. Caso ocorra algum erro na execução do script, verifique o seu código, corrigindo eventuais erros, e tente executá-lo novamente;



```
File Edit Selection View Go Run Terminal Help
MICROATIVIDADES
estruturas_condicao1.py
1 # Crie uma variável chamada temperatura e atribua a ela o valor 29
2
3 temperatura = 29
4
5 # Verificando se a temperatura é menor que 30
6 if temperatura < 30:
7     print("A temperatura hoje está amena")
8 else:
9     print("Hoje está fazendo calor")
10
```

```
PS C:\Users\SAPRANG\Documents\Estacio-python\MICROATIVIDADES> & C:\Users\SAPRANG\AppData\Local\Programs\Python\Python311\python.exe c:\Users\SAPRANG\Documents\Estacio-python\MICROATIVIDADES\estruturas_condicao1.py
A temperatura hoje está amena
PS C:\Users\SAPRANG\Documents\Estacio-python\MICROATIVIDADES>
```

10. Ainda no script, altere o valor da variável temperatura para 31, salve a alteração e execute novamente o script.



```
File Edit Selection View Go Run Terminal Help
MICROATIVIDADES
estruturas_condicao1.py
1 # Crie uma variável chamada temperatura e atribua a ela o valor 29
2
3 temperatura = 31
4
5 # Verificando se a temperatura é menor que 30
6 if temperatura < 30:
7     print("A temperatura hoje está amena")
8 else:
9     print("Hoje está fazendo calor")
10
```

```
PS C:\Users\SAPRANG\Documents\Estacio-python\MICROATIVIDADES> & C:\Users\SAPRANG\AppData\Local\Programs\Python\Python311\python.exe c:\Users\SAPRANG\Documents\Estacio-python\MICROATIVIDADES\estruturas_condicao1.py
A temperatura hoje está amena
PS C:\Users\SAPRANG\Documents\Estacio-python\MICROATIVIDADES> & C:\Users\SAPRANG\AppData\Local\Programs\Python\Python311\python.exe c:\Users\SAPRANG\Documents\Estacio-python\MICROATIVIDADES\estruturas_condicao1.py
Hoje está fazendo calor
PS C:\Users\SAPRANG\Documents\Estacio-python\MICROATIVIDADES>
```

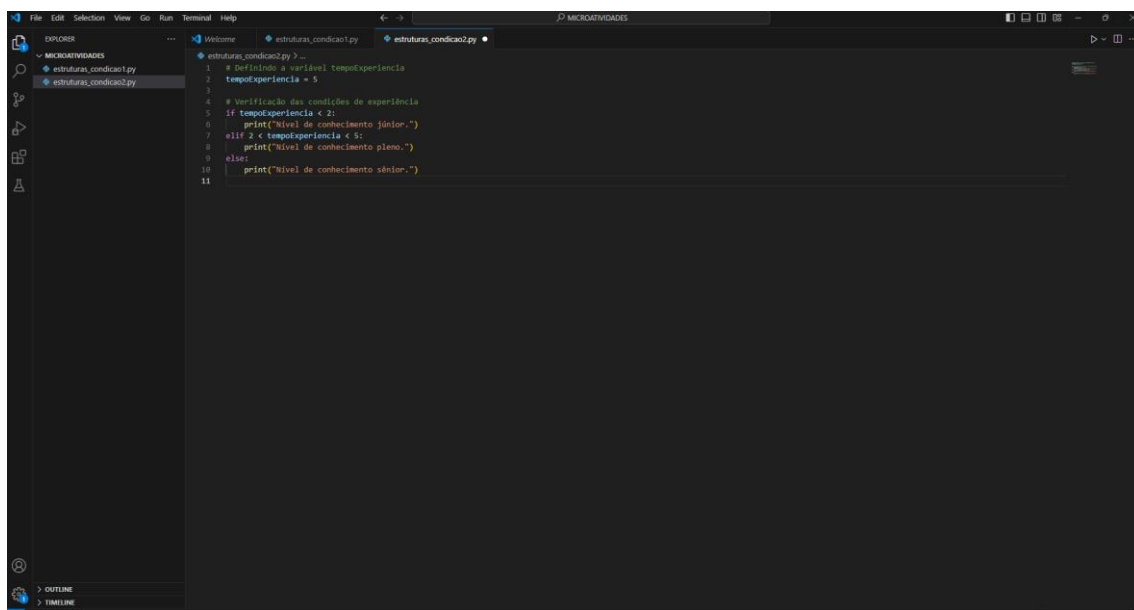
## Microatividade 2: Descrever a utilização da estrutura de condição else if (elif) em Python

### - Material necessário para a prática

- Interpretador Python instalado no Sistema Operacional;
- IDE VS Code instalada no Sistema Operacional;

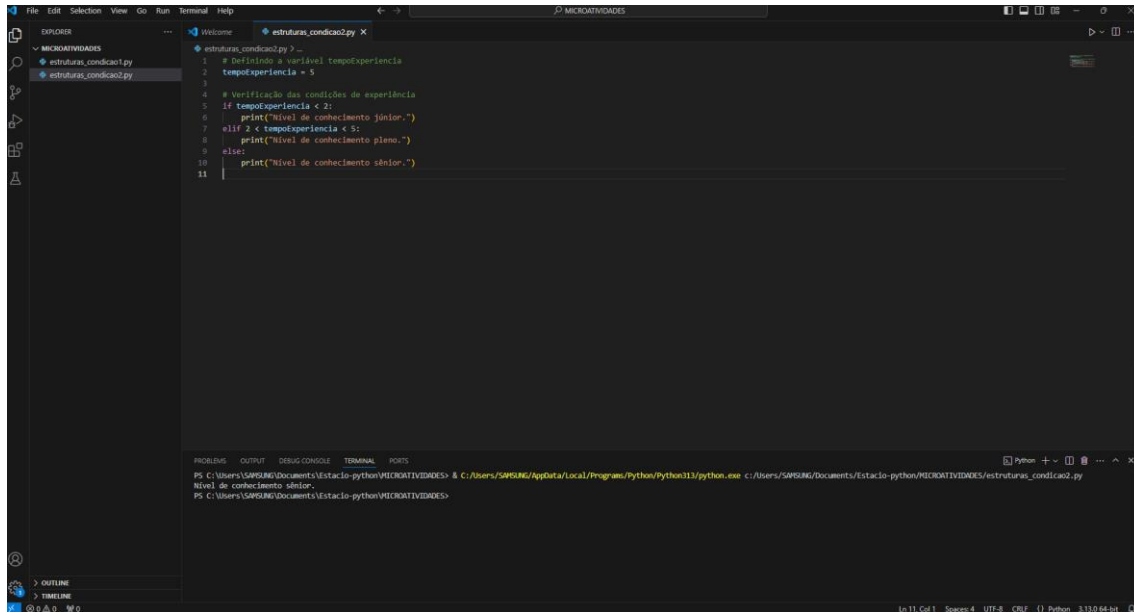
### - Procedimentos:

1. Abra a IDE VS Code;
2. Na mesma pasta utilizada na microatividade anterior, crie um novo arquivo/script chamado “estruturas\_condicao2.py”;
3. Nesse novo script:
  - a. Crie uma variável chamada tempoExperiencia e atribua a ela o valor 5;
  - b. Crie uma verificação, utilizando a condição if, para checar se o valor da variável tempoExperiencia é menor que 2;
  - c. Caso positivo, imprima na tela o texto ‘Nível de conhecimento júnior.’;
  - d. Após as instruções acima, crie uma outra condição utilizando elif (else if) para verificar se o valor da variável tempoExperiencia é maior que 2 e menor que 5. Em caso positivo, imprima o texto ‘Nível de conhecimento pleno.’
  - e. Por fim, crie uma condição else e imprima o texto ‘Nível de conhecimento sênior.’;



```
1 # Definindo a variável tempoExperiencia
2 tempoExperiencia = 5
3
4 # Verificação das condições de experiência
5 if tempoExperiencia < 2:
6     print("Nível de conhecimento júnior.")
7 elif 2 < tempoExperiencia < 5:
8     print("Nível de conhecimento pleno.")
9 else:
10    print("Nível de conhecimento sênior.")
11
```

4. Salve o arquivo/script e o execute;

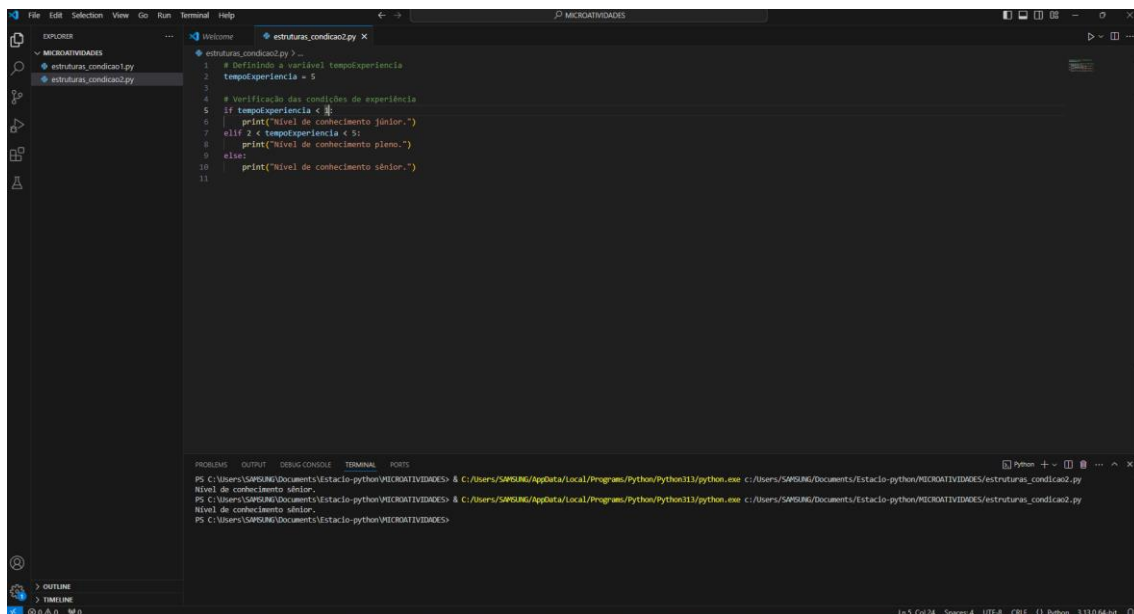


```
File Edit Selection View Go Run Terminal Help
MICROATIVIDADES
estrukturas_condicao1.py
estrukturas_condicao2.py
estrukturas_condicao2.py X

1 # Definindo a variável tempoExperiencia
2 tempoExperiencia = 5
3
4 # Verificação das condições de experiência
5 if tempoExperiencia < 2:
6     print("Nível de conhecimento júnior.")
7 elif 2 <= tempoExperiencia < 5:
8     print("Nível de conhecimento pleno.")
9 else:
10    print("Nível de conhecimento sênior.")
11

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
Python
PS C:\Users\SAP5UNG\Documents\Estacio-python\MICROATIVIDADES> & C:\Users\SAP5UNG\AppData\Local\Program\Python\Python311\python.exe c:\Users\SAP5UNG\Documents\Estacio-python\MICROATIVIDADES\estrukturas_condicao2.py
Nível de conhecimento sênior.
PS C:\Users\SAP5UNG\Documents\Estacio-python\MICROATIVIDADES>
```

5. Altere o script, modificando o valor da variável tempoExperiencia para 1. Salve e execute;

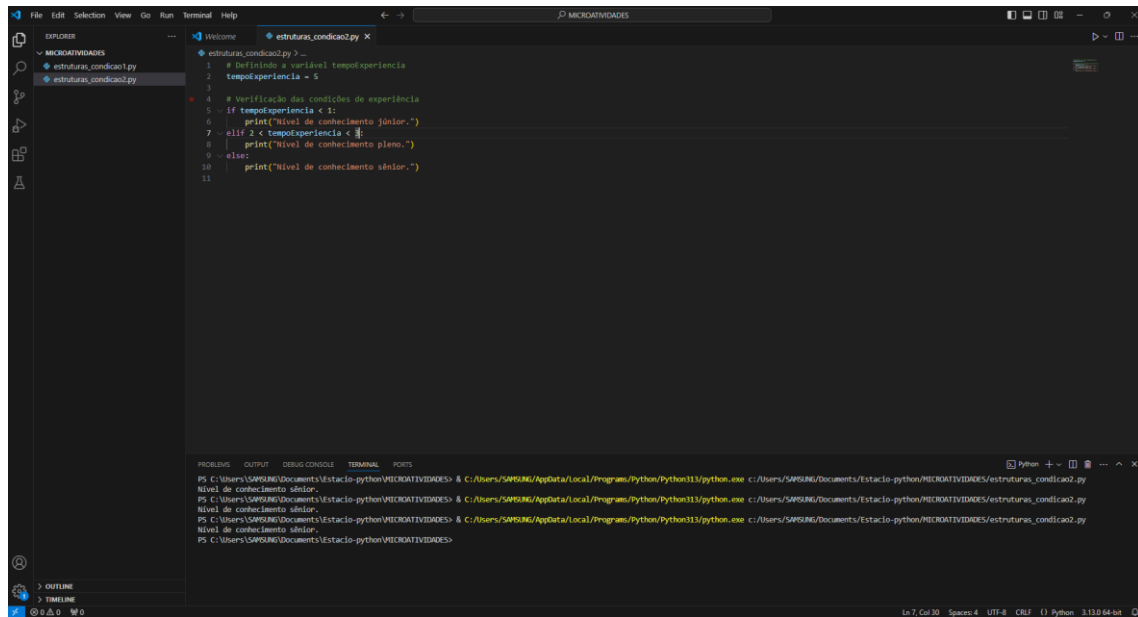


```
File Edit Selection View Go Run Terminal Help
MICROATIVIDADES
estrukturas_condicao1.py
estrukturas_condicao2.py
estrukturas_condicao2.py X

1 # Definindo a variável tempoExperiencia
2 tempoExperiencia = 1
3
4 # Verificação das condições de experiência
5 if tempoExperiencia < 2:
6     print("Nível de conhecimento júnior.")
7 elif 2 <= tempoExperiencia < 5:
8     print("Nível de conhecimento pleno.")
9 else:
10    print("Nível de conhecimento sênior.")
11

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
Python
PS C:\Users\SAP5UNG\Documents\Estacio-python\MICROATIVIDADES> & C:\Users\SAP5UNG\AppData\Local\Program\Python\Python311\python.exe c:\Users\SAP5UNG\Documents\Estacio-python\MICROATIVIDADES\estrukturas_condicao2.py
Nível de conhecimento júnior.
PS C:\Users\SAP5UNG\Documents\Estacio-python\MICROATIVIDADES>
```

6. tempoExperiencia para 3. Salve e execute.



### Microatividade 3: Descrever a utilização da estrutura de repetição

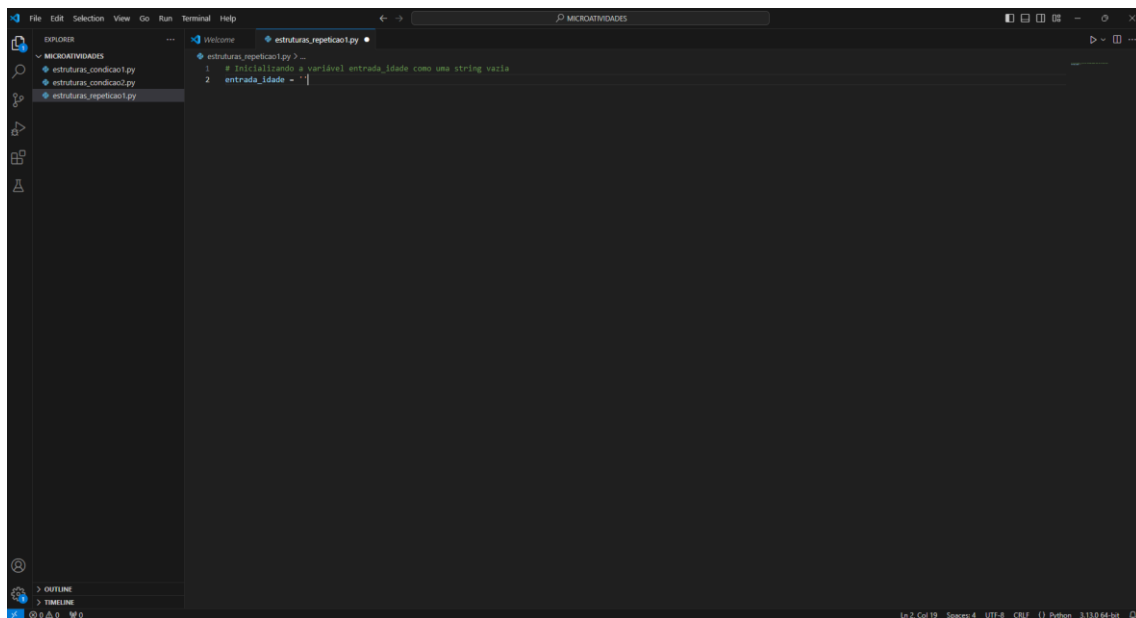
#### while em Python

##### - Material necessário para a prática

- Interpretador Python instalado no Sistema Operacional;
- IDE VS Code instalada no Sistema Operacional;

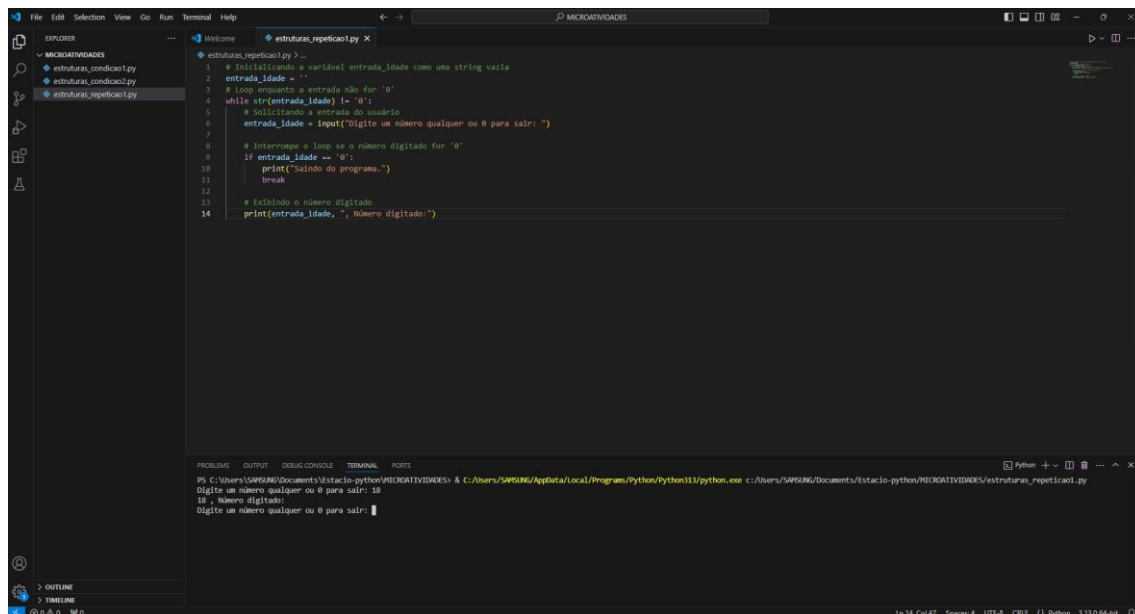
##### - Procedimentos

1. Abra a IDE VS Code;
2. Na mesma pasta utilizada nas microatividades anteriores, crie um novo arquivo/script chamado “estruturas\_repeticao1.py”;
3. Nesse novo script:
  - a. Crie uma variável chamada entrada\_idade e atribua a ela o valor ‘;



- b. Crie uma instrução while que verifique se o valor atribuído à variável entrada\_idade é diferente de 0 (como o valor inicial atribuído à variável é ‘’, isso a definiu como tipo string. Logo, a verificação no While deve ser feita com auxílio da instrução str );
- c. No escopo da instrução while, atribua à variável entrada\_idade um input de entrada de dados com o texto ‘Digite um número qualquer ou 0 para sair: ‘;
- d. Imprima, na tela, o número digitado pelo usuário precedido do texto ‘Número digitado: ‘;

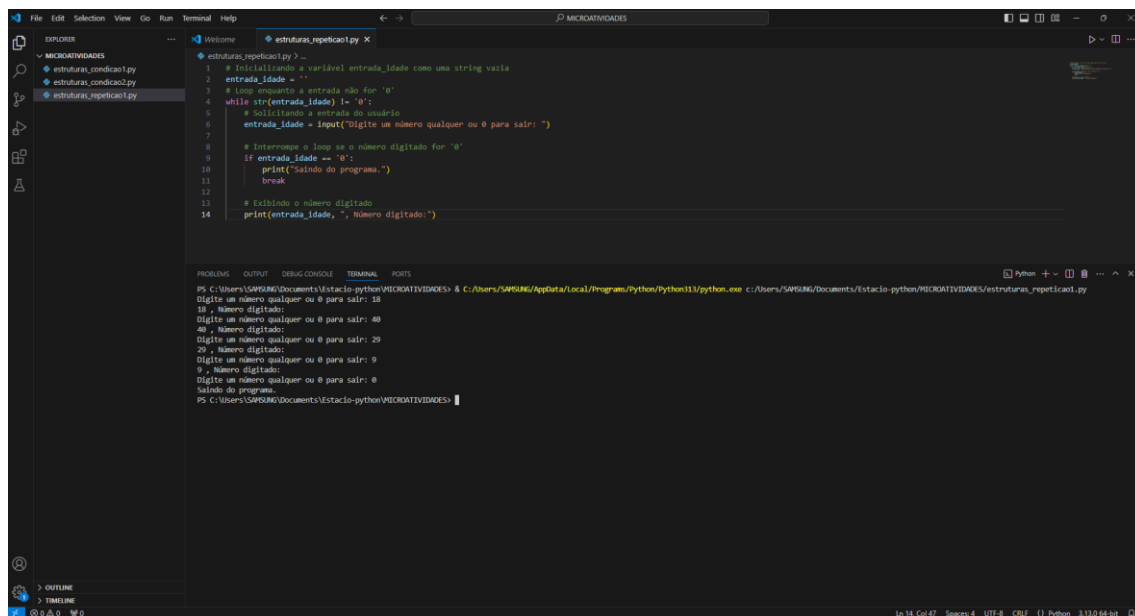




```
1 # Inicializando a variável entrada_idade como uma string vazia
2 entrada_idade = ""
3 # Loop enquanto a entrada não for '0'
4 while str(entrada_idade) != '0':
5     # Solicitando a entrada do usuário
6     entrada_idade = input("Digite um número qualquer ou 0 para sair: ")
7
8     # Interrompe o loop se o número digitado for '0'
9     if entrada_idade == '0':
10         print("Saindo do programa.")
11         break
12
13     # Exibindo o número digitado
14     print(entrada_idade, ", Número digitado:")
```

PS C:\Users\SAPUANG\Documents\Estacio-python\MICROATIVIDADES> & C:\Users\SAPUANG\AppData\Local\Program\Python\Python311\python.exe c:\Users\SAPUANG\Documents\Estacio-python\MICROATIVIDADES\estruturas\_repeticao1.py  
Digite um número qualquer ou 0 para sair: 18  
18 , Número digitado:  
Digite um número qualquer ou 0 para sair: █

4. Salve o arquivo/script e o execute;
5. Teste diferentes valores como entrada de dados, incluindo o número 0 - que deverá fazer com que a execução do programa seja interrompida. Caso isso não ocorra, verifique seu código – sobretudo a comparação na instrução While.



```
1 # Inicializando a variável entrada_idade como uma string vazia
2 entrada_idade = ""
3 # Loop enquanto a entrada não for '0'
4 while str(entrada_idade) != '0':
5     # Solicitando a entrada do usuário
6     entrada_idade = input("Digite um número qualquer ou 0 para sair: ")
7
8     # Interrompe o loop se o número digitado for '0'
9     if entrada_idade == '0':
10         print("Saindo do programa.")
11         break
12
13     # Exibindo o número digitado
14     print(entrada_idade, ", Número digitado:")
```

PS C:\Users\SAPUANG\Documents\Estacio-python\MICROATIVIDADES> & C:\Users\SAPUANG\AppData\Local\Program\Python\Python311\python.exe c:\Users\SAPUANG\Documents\Estacio-python\MICROATIVIDADES\estruturas\_repeticao1.py  
Digite um número qualquer ou 0 para sair: 18  
18 , Número digitado:  
Digite um número qualquer ou 0 para sair: 40  
40 , Número digitado:  
Digite um número qualquer ou 0 para sair: 29  
29 , Número digitado:  
Digite um número qualquer ou 0 para sair: 9  
9 , Número digitado:  
Digite um número qualquer ou 0 para sair: 0  
Saindo do programa.  
PS C:\Users\SAPUANG\Documents\Estacio-python\MICROATIVIDADES> █

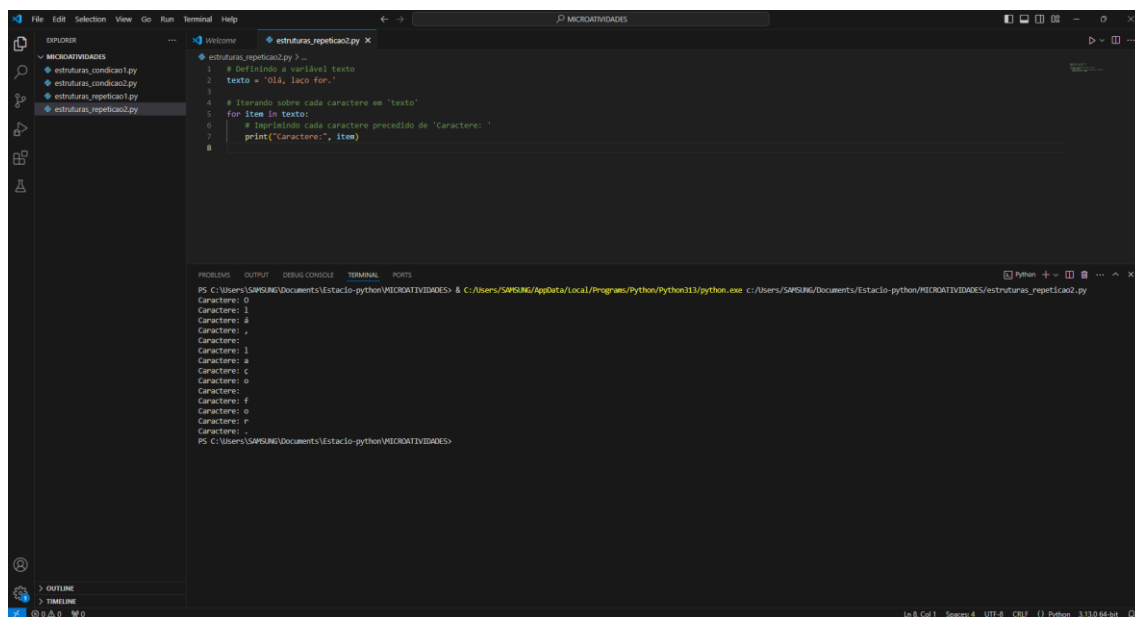
## Microatividade 4: Descrever a utilização da estrutura de repetição for em Python

### - Material necessário para a prática

- Interpretador Python instalado no Sistema Operacional;
- IDE VS Code instalada no Sistema Operacional;

### - Procedimentos

1. Abra a IDE VS Code;
2. Na mesma pasta utilizada nas microatividades anteriores, crie um novo arquivo/script chamado “estruturas\_repeticao2.py”;
3. Nesse novo script:
  1. Crie uma variável chamada texto e atribua a ela o valor ‘Olá, laço for.’;
  2. Crie uma instrução for que itere sobre a variável texto atribuindo cada um de seus caracteres a uma variável chamada item;
  3. Imprima, na tela, dentro do escopo do laço for, o valor da variável item precedido do texto ‘Caractere: ‘;
4. Salve o arquivo/script e o execute;



```
1 # Definindo a variável texto
2 texto = 'Olá, laço for.'
3
4 # Iterando sobre cada caractere em 'texto'
5 for item in texto:
6     # Imprimindo cada caractere precedido de 'Caractere: '
7     print('Caractere:', item)
8
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

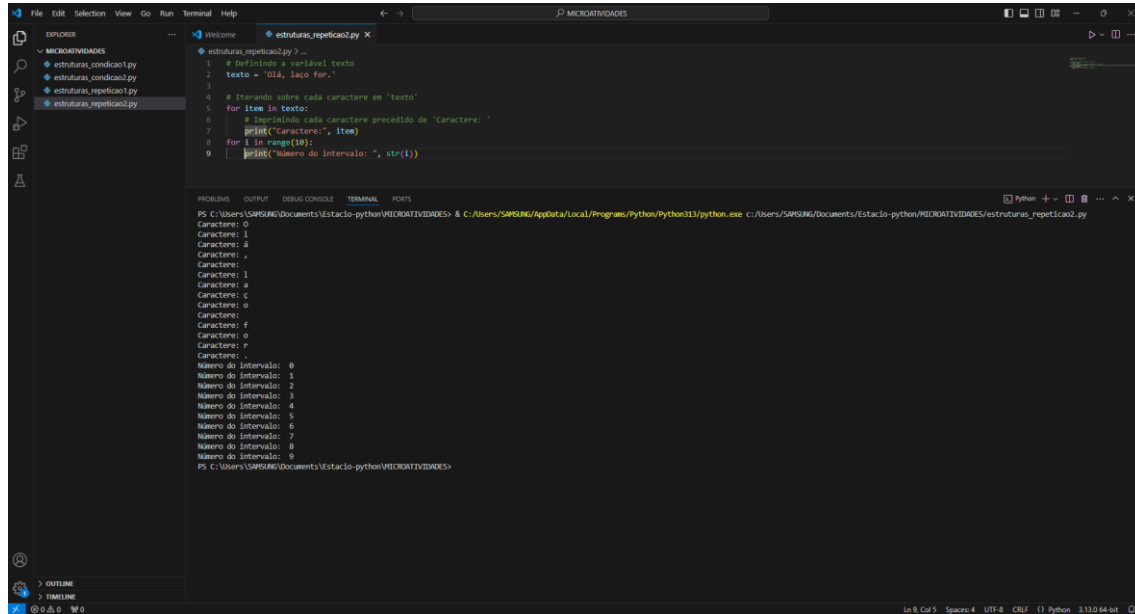
PS C:\Users\SAMSUNG\Documents\Estacio-python\MICROATIVIDADES> & C:\Users\SAMSUNG\AppData\Local\Programs\Python\Python313\python.exe c:\Users\SAMSUNG\Documents\Estacio-python\MICROATIVIDADES\estruturas\_repeticao2.py

Caractere: O  
Caractere: l  
Caractere: á  
Caractere: ,  
Caractere: ,  
Caractere: l  
Caractere: a  
Caractere: o  
Caractere: ,  
Caractere: f  
Caractere: o  
Caractere: r  
Caractere: .

PS C:\Users\SAMSUNG\Documents\Estacio-python\MICROATIVIDADES>

5. Crie, no mesmo script, uma nova instrução for que:
  - a. Itere sobre um intervalo numérico entre 1 e 10 (dica: use a instrução range);

- b. Imprima, na tela, dentro do escopo do laço for, o valor de cada número no intervalo acima precedido do texto ‘Número do intervalo: ‘ ;
- c. Lembre-se de utilizar a instrução str para concatenar o valor inteiro com a string no momento de imprimir o valor pedido na tela.



The screenshot shows a Visual Studio Code editor window with a Python file named `estruturas_repeticao2.py`. The code defines a string `texto = "Olá, laço for."` and iterates over each character in the string using a `for` loop. Inside the loop, it prints the character preceded by the text "Caractere: ". After the loop, it iterates over the range from 0 to 9 using another `for` loop, printing the number preceded by the text "Número do intervalo: ".

```
1 # Definindo a variável texto
2 texto = "Olá, laço for."
3
4 # Iterando sobre cada caractere em 'texto'
5 for item in texto:
6     # Imprimindo cada caractere precedido de 'Caractere: '
7     print("Caractere: ", item)
8
9 for i in range(10):
10    print("Número do intervalo: ", str(i))
```

The terminal output shows the execution of the script, displaying the characters of the string and the numbers from 0 to 9, each preceded by the specified text.

```
PS C:\Users\SAPSONG\Documents\Estacio-python\MICROATIVIDADES> & C:\Users\SAPSONG\AppData\Local\Programs\Python\Python111\python.exe c:\Users\SAPSONG\Documents\Estacio-python\MICROATIVIDADES\estruturas_repeticao2.py
Caractere: O
Caractere: l
Caractere: á
Caractere: ,
Caractere: l
Caractere: a
Caractere: ç
Caractere: o
Caractere: .
Caractere: "
Caractere: "
Caractere: "
Número do intervalo: 0
Número do intervalo: 1
Número do intervalo: 2
Número do intervalo: 3
Número do intervalo: 4
Número do intervalo: 5
Número do intervalo: 6
Número do intervalo: 7
Número do intervalo: 8
Número do intervalo: 9
PS C:\Users\SAPSONG\Documents\Estacio-python\MICROATIVIDADES>
```

# PROJETO FINAL

1. Abra a IDE VS Code;
2. Na mesma pasta onde criou os scripts utilizados nas microatividades, crie um novo script chamado “calculadora\_v2.py”;
3. No script:
  - a. Crie uma variável chamada `saida` e atribua a ela o valor “”;
  - b. Crie uma função chamada `adicao` . Tal função deverá receber dois parâmetros e retornar a soma entre ambos;
  - c. Crie uma função chamada `subtracao` . Tal função deverá receber dois parâmetros e retornar a subtração entre ambos;

```
1 # Variável para armazenar mensagens de saída
2 saida = ''
3
4 # Função para soma
5 def adicao(a, b):
6     return a + b
7
8 # Função para subtração
9 def subtracao(a, b):
10    return a - b
```

- d. Crie uma função chamada `multiplicacao` . Tal função deverá receber dois parâmetros e retornar a multiplicação entre ambos;
- e. Crie uma função chamada `divisao`. Tal função deverá receber dois parâmetros, verificar se um deles é igual a 0. Em caso positivo, deverá retornar a mensagem “Não foi possível realizar a divisão por 0”. Em caso negativo, deverá retornar a divisão entre ambos;

```
1 # Variável para armazenar mensagens de saída
2 saida = ''
3
4 # Função para soma
5 def adicao(a, b):
6     return a + b
7
8 # Função para subtração
9 def subtracao(a, b):
10    return a - b
11
12 # Função para a multiplicação
13 def multiplicacao(a, b):
14     return a * b
15
16 # Função para divisão
17 def divisao(a, b):
18     if b == 0:
19         return "Não foi possível realizar a divisão por 0"
20     return a / b
```

- f. Crie uma função chamada calculadora. Tal função deverá receber três parâmetros, sendo eles: os dois números que serão usados para os cálculos e a operação matemática que se deseja realizar. Sobre esse último parâmetro, você poderá utilizar tanto o sinal da operação quanto o seu nome;

```
calculadora_v2.py X
calculadora_v2.py > divisao
1 # Variável para armazenar mensagens de saída
2 saida = ''
3
4 # Função para soma
5 def adicao(a, b):
6     return a + b
7
8 # Função para subtração
9 def subtracao(a, b):
10    return a - b
11
12 # Função para a multiplicação
13 def multiplicacao(a, b):
14    return a * b
15
16 # Função para divisão
17 def divisao(a, b):
18     if b == 0:
19         return "Não foi possível realizar a divisão por 0"
20     return a/b
21
22 # Função principal da calculadora
23 def calculadora(a, b, operacao):
24     if operacao in ('+', 'adicao'):
25         return adicao(a, b)
26     elif operacao in ('-', 'subtracao'):
27         return subtracao(a, b)
28     elif operacao in ('*', 'multiplicacao'):
29         return multiplicacao(a, b)
30     elif operacao in ('/', 'divisao'):
31         return divisao(a, b)
32     else:
33         return "Operação inválida. Use '+', '-', '*', '/' ou os nomes correspondentes."
```

- g. No corpo da função calculadora você deverá verificar qual a operação desejada pelo usuário, checando o valor do parâmetro correspondente. Utilize estruturas de condição para isso e, dependendo da operação desejada, você deverá chamar a função relativa a ela, passando as variáveis contendo os dois números para serem utilizados no cálculo. Armazene o resultado da chamada às funções de cálculo numa variável chamada resultado. Ao final da função calculadora você deverá retornar a variável resultado;
- h. Crie um laço while e, como condição do mesmo, verifique se o valor da variável saída é diferente de n. Lembre-se de que o usuário poderá inserir tanto N quanto n;
- i. No escopo do laço while peça ao usuário para digitar o primeiro número e armazene seu valor numa variável. Faça o mesmo para o segundo número e para a operação matemática. Passe essas três variáveis para o método calculadora, armazenando o retorno dessa chamada numa variável também chamada resultado. Imprima na tela o valor da variável resultado precedido pelo texto 'Resultado da operação: '. Por fim, pergunte ao usuário se ele deseja continuar

- ou não executando o programa. Armazene tal input na variável `saida`;
- j. Tome cuidado com a condição de verificação do laço `for` em relação à entrada do usuário armazenada na variável `saida`. Em outras palavras, deixe claro para o usuário as respostas possíveis para a pergunta se ele deseja sair. Use, por exemplo, S/N. Com isso você poderá considerar um desses dois valores na verificação do laço para saber se deve continuar executando o programa ou se deve encerrá-lo.
  - k. Salve as alterações no script e o execute via VS Code;
  - l. Teste o aplicativo interagindo com ele através do prompt, fornecendo os dados
  - m. necessários para a sua execução.
  - n.

```
21 # Função principal da calculadora
22 def calculadora(a, b, operacao):
23     if operacao in ('+', 'adicao'):
24         return adicao(a, b)
25     elif operacao in ('-', 'subtracao'):
26         return subtracao(a, b)
27     elif operacao in ('*', 'multiplicacao'):
28         return multiplicacao(a, b)
29     elif operacao in ('/', 'divisao'):
30         return divisao(a, b)
31     else:
32         return "Operação inválida. Use '+', '-', '*', '/' ou os nomes correspondentes."
33 # Laço principal da calculadora
34 while saida.lower() != 'n':
35     try:
36         # Entrada do usuário
37         num1 = float(input("Digite o primeiro número: "))
38         num2 = float(input("Digite o segundo número: "))
39         operacao = input("Digite a operação desejada (+, -, *, / ou o nome da operação: ").lower()
40
41         # Chamada à função calculadora
42         resultado = calculadora(num1, num2, operacao)
43         # Exibição do resultado
44         print(f"Resultado da operação: {resultado}")
45     except ValueError:
46         print("Por favor, insira valores numéricos válidos.")
47     # Verificar se o usuário deseja continuar
48     saida = input("Deseja continuar? (S/N): ").lower()
49     if saida not in ['s', 'n']:
50         print("Opção inválida! Digite 'S' para continuar ou 'N' para sair.")
```

# Teste do Código

## 1. Subtração

```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
PS C:\Users\SAMSUNG\Documents\Estacio-python\MICROATIVIDADES> & C:/Users/SAMSUNG/AppData/Local/Programs/Python/Python313/python.exe c:/Users/SAMSUNG/Documents/Estacio-python/MICROATIVIDADES/calculadora_v2.py
Digite o primeiro número: 10
Digite o segundo número: 3
Digite a operação desejada (+, -, *, / ou o nome da operação): -
Resultado da operação: 7.0
Deseja continuar? (S/N): n
PS C:\Users\SAMSUNG\Documents\Estacio-python\MICROATIVIDADES>
PS C:\Users\SAMSUNG\Documents\Estacio-python\MICROATIVIDADES>
```

## 2. Adição

```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
PS C:\Users\SAMSUNG\Documents\Estacio-python\MICROATIVIDADES> & C:/Users/SAMSUNG/AppData/Local/Programs/Python/Python313/python.exe c:/Users/SAMSUNG/Documents/Estacio-python/MICROATIVIDADES/calculadora_v2.py
Digite o primeiro número: 15
Digite o segundo número: 6
Digite a operação desejada (+, -, *, / ou o nome da operação): +
Resultado da operação: 21.0
Deseja continuar? (S/N):
```

## 3. Multiplicação e Verificação se o Usuário Continuará

```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
PS C:\Users\SAMSUNG\Documents\Estacio-python\MICROATIVIDADES> & C:/Users/SAMSUNG/AppData/Local/Programs/Python/Python313/python.exe c:/Users/SAMSUNG/Documents/Estacio-python/MICROATIVIDADES/calculadora_v2.py
Digite o primeiro número: 9
Digite o segundo número: 11
Digite a operação desejada (+, -, *, / ou o nome da operação): *
Resultado da operação: 99.0
Deseja continuar? (S/N): s
Digite o primeiro número:
```

## 4. Divisão

```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
PS C:\Users\SAMSUNG\Documents\Estacio-python\MICROATIVIDADES> & C:/Users/SAMSUNG/AppData/Local/Programs/Python/Python313/python.exe c:/Users/SAMSUNG/Documents/Estacio-python/MICROATIVIDADES/calculadora_v2.py
Digite o primeiro número: 21
Digite o segundo número: 7
Digite a operação desejada (+, -, *, / ou o nome da operação): /
Resultado da operação: 3.0
Deseja continuar? (S/N):
```