

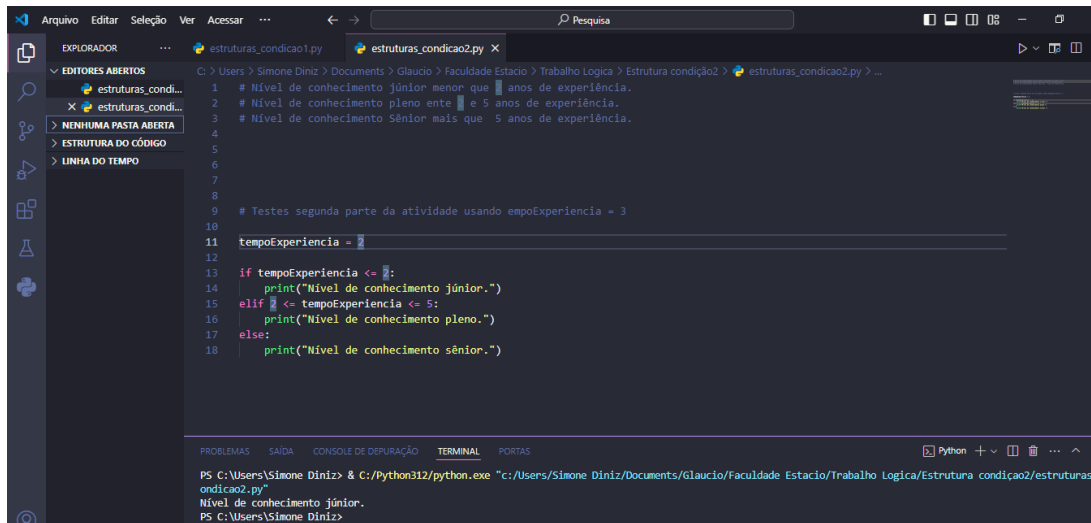
The image shows a code editor window with a dark theme. The top menu bar includes 'Arquivo', 'Editar', 'Seleção', 'Ver', 'Acessar', and 'Pesquisa'. The file path in the title bar is 'C:\Users\Simone Diniz\Documents\Glauco\Faculdade Estacio\Trabalho Logica\Estrutura condição1\estruturas_condicao1.py'. The code is a Python script with the following content:

```
1
2 #temperatura = 29
3
4
5 #if temperatura < 30:
6 #    print("A temperatura hoje está amena.")
7 #else:
8 #    print("Hoje está fazendo calor.")
9
10
11
12 # Segunda parte da atividade usando temperatura = 31
13
14 temperatura = 31
15
16 if temperatura < 30:
17     print("A temperatura hoje está amena.")
18 else:
19     print("Hoje está fazendo calor.")
```

The bottom of the window features a terminal panel with tabs for 'PROBLEMAS', 'SAÍDA', 'CONSOLE DE DEPURAÇÃO', 'TERMINAL' (selected), and 'PORTAS'. The terminal output shows the execution of the script:

```
PS C:\Users\Simone Diniz> & C:\Python312\python.exe "c:/Users/Simone Diniz/Documents/Glauco/Faculdade Estacio/Trabalho Logica/Estrutura condição1/estruturas_condicao1.py"
A temperatura hoje está amena.
PS C:\Users\Simone Diniz> & C:\Python312\python.exe "c:/Users/Simone Diniz/Documents/Glauco/Faculdade Estacio/Trabalho Logica/Estrutura condição1/estruturas_condicao1.py"
Hoje está fazendo calor.
```

Micro Atividade 2 - Condição else if (elif)



The screenshot shows the VS Code editor with a file named `estruturas_condicao2.py`. The code defines three experience levels based on years of experience: junior (less than 2), full (between 2 and 5), and senior (more than 5). A test is performed with `tempoExperiencia = 2`. The terminal output shows "Nível de conhecimento júnior."

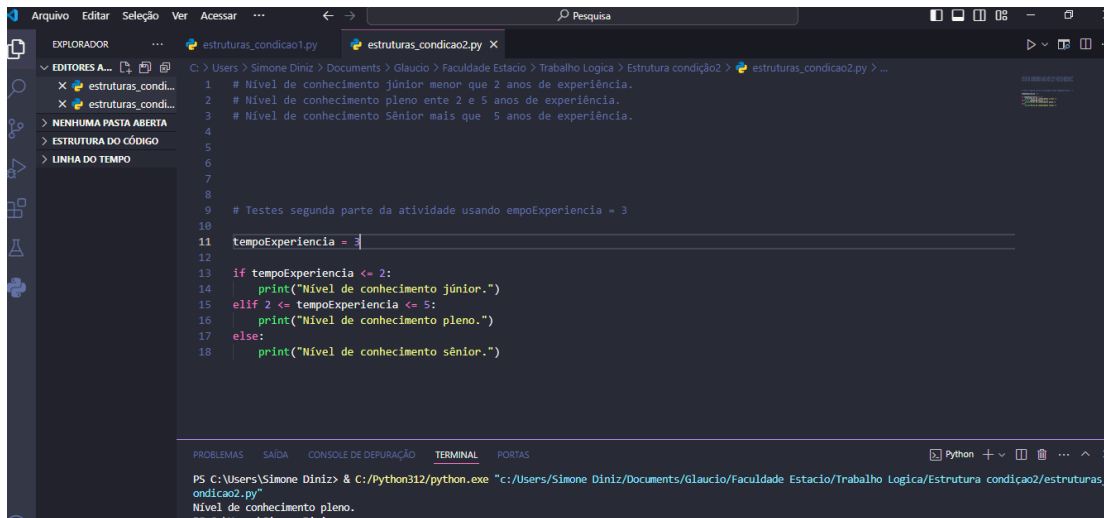
```
1 # Nível de conhecimento júnior menor que 2 anos de experiência.
2 # Nível de conhecimento pleno entre 2 e 5 anos de experiência.
3 # Nível de conhecimento Sênior mais que 5 anos de experiência.
4
5
6
7
8
9 # Testes segunda parte da atividade usando tempoExperiencia = 3
10
11 tempoExperiencia = 2
12
13 if tempoExperiencia <= 2:
14     print("Nível de conhecimento júnior.")
15 elif 2 <= tempoExperiencia <= 5:
16     print("Nível de conhecimento pleno.")
17 else:
18     print("Nível de conhecimento sênior.")
```

PROBLEMAS SAÍDA CONSOLE DE DEPURAÇÃO TERMINAL PORTAS

PS C:\Users\Simone Diniz> & C:/Python312/python.exe "c:/Users/Simone Diniz/Documents/Glaucio/Faculdade Estacio/Trabalho Logica/Estrutura condicao2/estruturas_condicao2.py"

Nível de conhecimento júnior.

PS C:\Users\Simone Diniz>



The screenshot shows the same VS Code editor with `estruturas_condicao2.py`. The test value is now `tempoExperiencia = 3`. The terminal output shows "Nível de conhecimento pleno."

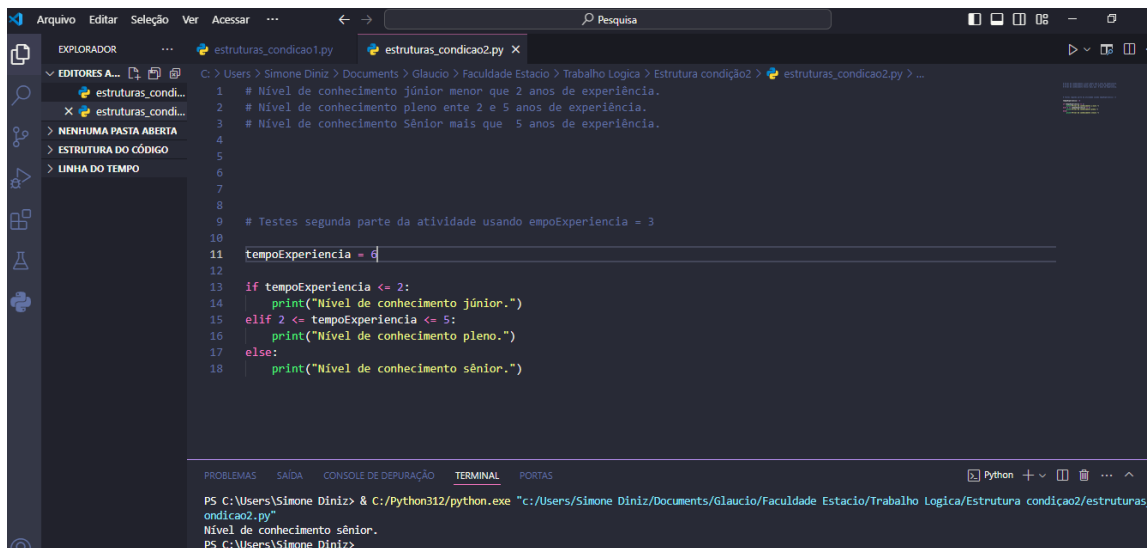
```
1 # Nível de conhecimento júnior menor que 2 anos de experiência.
2 # Nível de conhecimento pleno entre 2 e 5 anos de experiência.
3 # Nível de conhecimento Sênior mais que 5 anos de experiência.
4
5
6
7
8
9 # Testes segunda parte da atividade usando tempoExperiencia = 3
10
11 tempoExperiencia = 3
12
13 if tempoExperiencia <= 2:
14     print("Nível de conhecimento júnior.")
15 elif 2 <= tempoExperiencia <= 5:
16     print("Nível de conhecimento pleno.")
17 else:
18     print("Nível de conhecimento sênior.")
```

PROBLEMAS SAÍDA CONSOLE DE DEPURAÇÃO TERMINAL PORTAS

PS C:\Users\Simone Diniz> & C:/Python312/python.exe "c:/Users/Simone Diniz/Documents/Glaucio/Faculdade Estacio/Trabalho Logica/Estrutura condicao2/estruturas_condicao2.py"

Nível de conhecimento pleno.

PS C:\Users\Simone Diniz>



The screenshot shows the same VS Code editor with `estruturas_condicao2.py`. The test value is now `tempoExperiencia = 6`. The terminal output shows "Nível de conhecimento sênior."

```
1 # Nível de conhecimento júnior menor que 2 anos de experiência.
2 # Nível de conhecimento pleno entre 2 e 5 anos de experiência.
3 # Nível de conhecimento Sênior mais que 5 anos de experiência.
4
5
6
7
8
9 # Testes segunda parte da atividade usando tempoExperiencia = 3
10
11 tempoExperiencia = 6
12
13 if tempoExperiencia <= 2:
14     print("Nível de conhecimento júnior.")
15 elif 2 <= tempoExperiencia <= 5:
16     print("Nível de conhecimento pleno.")
17 else:
18     print("Nível de conhecimento sênior.")
```

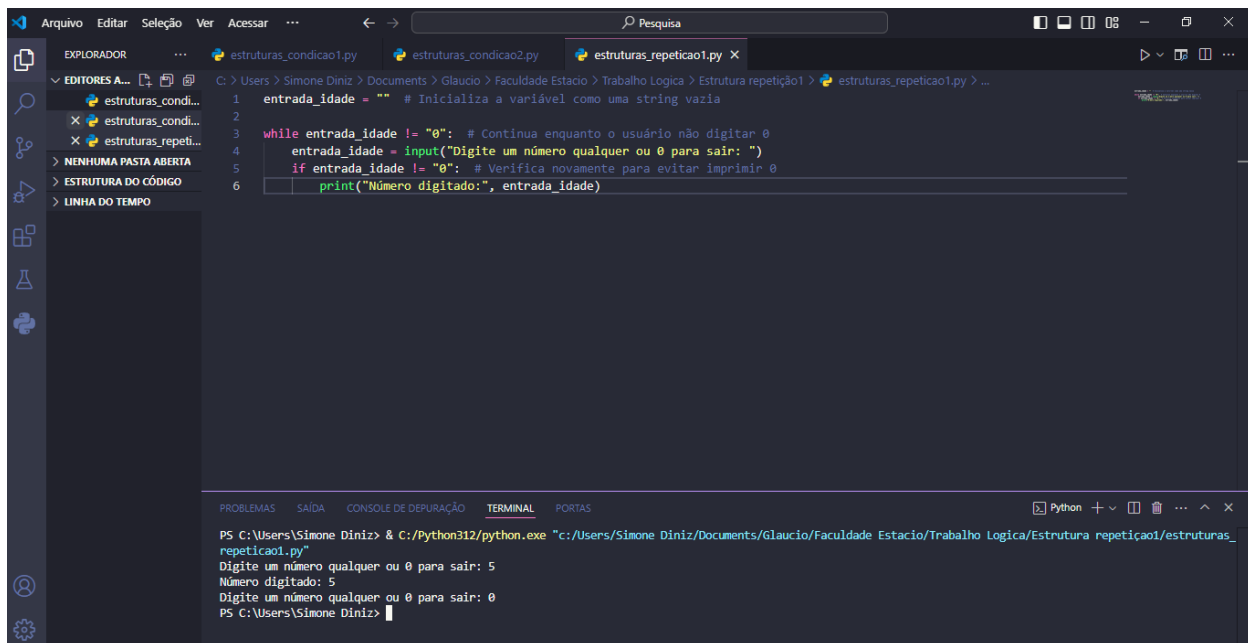
PROBLEMAS SAÍDA CONSOLE DE DEPURAÇÃO TERMINAL PORTAS

PS C:\Users\Simone Diniz> & C:/Python312/python.exe "c:/Users/Simone Diniz/Documents/Glaucio/Faculdade Estacio/Trabalho Logica/Estrutura condicao2/estruturas_condicao2.py"

Nível de conhecimento sênior.

PS C:\Users\Simone Diniz>

Micro Atividade 3 - Estrutura de repetição while



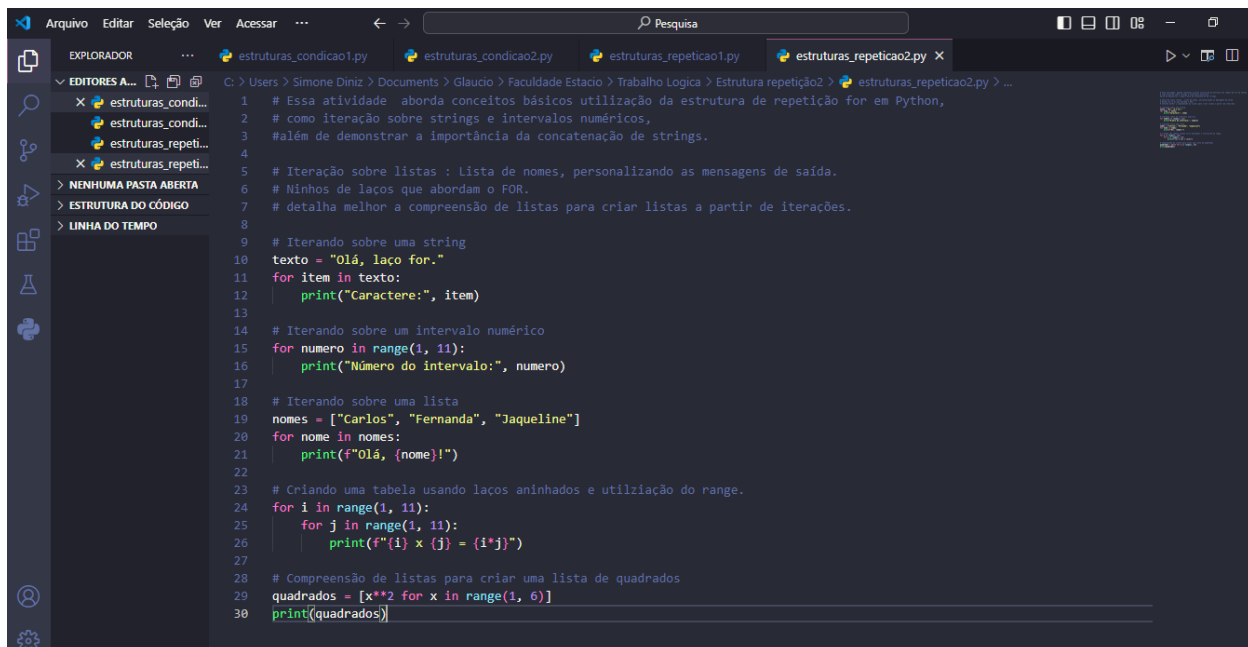
The screenshot shows a code editor with a file named `estruturas_repeticao1.py` open. The code implements a while loop that prompts the user to enter a number, checks if it's zero, and prints the entered number. The terminal output shows the program running successfully, with the user entering '5' and the program printing 'Número digitado: 5'.

```
1 entrada_idade = "" # Inicializa a variável como uma string vazia
2
3 while entrada_idade != "0": # Continua enquanto o usuário não digitar 0
4     entrada_idade = input("Digite um número qualquer ou 0 para sair: ")
5     if entrada_idade != "0": # Verifica novamente para evitar imprimir 0
6         print("Número digitado:", entrada_idade)
```

Terminal Output:

```
PS C:\Users\Simone Diniz> & C:\Python312\python.exe "c:\Users\Simone Diniz\Documents\Glaucio\Faculdade Estacio\Trabalho Logica\Estrutura repeticao1\estruturas_repeticao1.py"
Digite um número qualquer ou 0 para sair: 5
Número digitado: 5
Digite um número qualquer ou 0 para sair: 0
PS C:\Users\Simone Diniz>
```

Micro Atividade 4 - Estrutura de repetição for



The screenshot shows a code editor with a file named `estruturas_repeticao2.py` open. The code demonstrates various uses of the for loop, including iterating over a string, a range of numbers, a list of names, and a nested loop to create a multiplication table. It also shows a list comprehension to create a list of squares.

```
1 # Essa atividade aborda conceitos básicos utilização da estrutura de repetição for em Python,
2 # como iteração sobre strings e intervalos numéricos,
3 #além de demonstrar a importância da concatenação de strings.
4
5 # Iteração sobre listas : Lista de nomes, personalizando as mensagens de saída.
6 # Ninhos de laços que abordam o FOR.
7 # detalha melhor a compreensão de listas para criar listas a partir de iterações.
8
9 # Iterando sobre uma string
10 texto = "Olá, laço for."
11 for item in texto:
12     print("Caractere:", item)
13
14 # Iterando sobre um intervalo numérico
15 for numero in range(1, 11):
16     print("Número do intervalo:", numero)
17
18 # Iterando sobre uma lista
19 nomes = ["Carlos", "Fernanda", "Jaqueline"]
20 for nome in nomes:
21     print(f"Olá, {nome}!")
22
23 # Criando uma tabela usando laços aninhados e utilização do range.
24 for i in range(1, 11):
25     for j in range(1, 11):
26         print(f"{i} x {j} = {i*j}")
27
28 # Compreensão de listas para criar uma lista de quadrados
29 quadrados = [x**2 for x in range(1, 6)]
30 print(quadrados)
```

```
Arquivo  Editar  Seleção  Ver  Acessar  ...  ← →  Pesquisa  [Icons]  -  [Icons]  X
```

EXPLORADOR ... estruturas_condicao1.py estruturas_condicao2.py estruturas_repeticao1.py estruturas_repeticao2.py X

EDITORES ABERTOS C:\Users\Simone Diniz\Documents\Glaucio\Faculdade Estacio\Trabalho Logica\Estrutura repetição2 > estruturas_repeticao2.py > ...

- estruturas_condi...
- estruturas_condi...
- estruturas_repeti...
- X estruturas_repeti...

NENHUMA PASTA ABERTA

ESTRUTURA DO CÓDIGO

LINHA DO TEMPO

PROBLEMAS SAÍDA CONSOLE DE DEPURACÃO TERMINAL PORTAS Python + v [Icons] ... X

```
PS C:\Users\Simone Diniz> & C:/Python312/python.exe "c:/Users/Simone Diniz/Documents/Glaucio/Faculdade Estacio/Trabalho Logica/Estrutura repetição2/estruturas_repeticao2.py"
Caractere: 0
Caractere: 1
Caractere: á
Caractere: ,
Caractere: l
Caractere: a
Caractere: ç
Caractere: o
Caractere: f
Caractere: o
Caractere: r
Caractere: .
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
Número do intervalo: 10
```

```
Arquivo  Editar  Seleção  Ver  Acessar  ...  ← →  Pesquisa  [Icons]  -  [Icons]  [Icons]
```

EXPLORADOR ... estruturas_condicao1.py estruturas_condicao2.py estruturas_repeticao1.py estruturas_repeticao2.py X

EDITORES ABERTOS C:\Users\Simone Diniz\Documents\Glaucio\Faculdade Estacio\Trabalho Logica\Estrutura repetição2 > estruturas_repeticao2.py > ...

- estruturas_condi...
- estruturas_condi...
- estruturas_repeti...
- X estruturas_repeti...

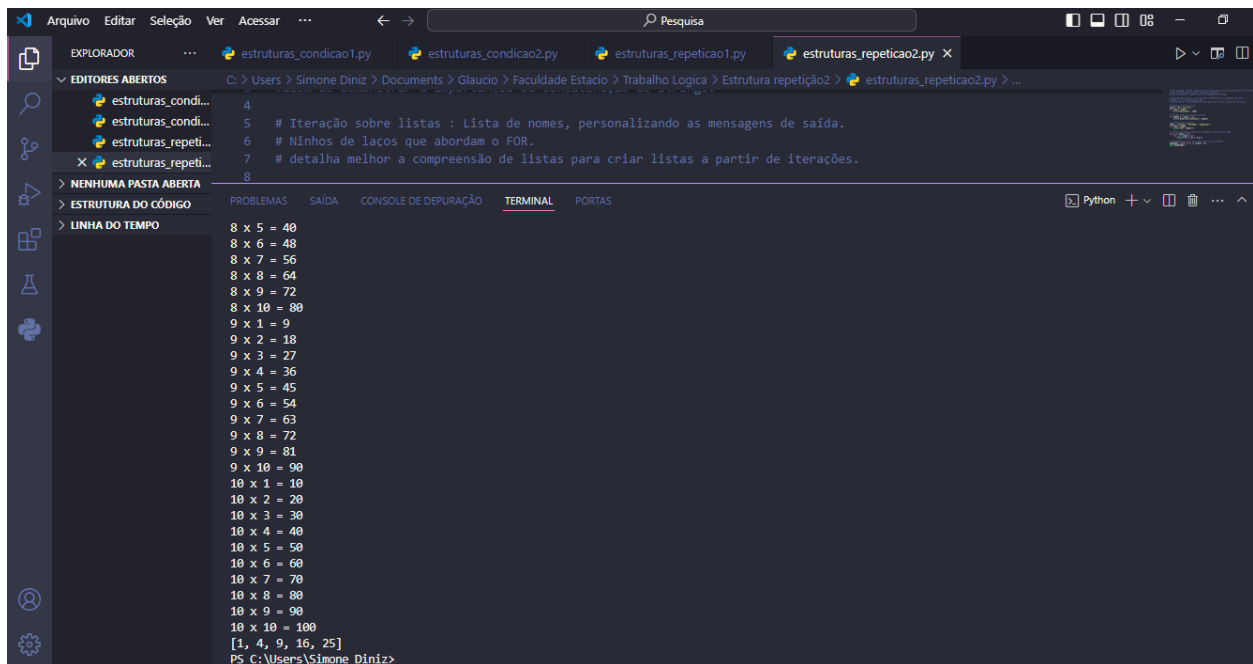
NENHUMA PASTA ABERTA

ESTRUTURA DO CÓDIGO

LINHA DO TEMPO

PROBLEMAS SAÍDA CONSOLE DE DEPURACÃO TERMINAL PORTAS Python + v [Icons] ... ^

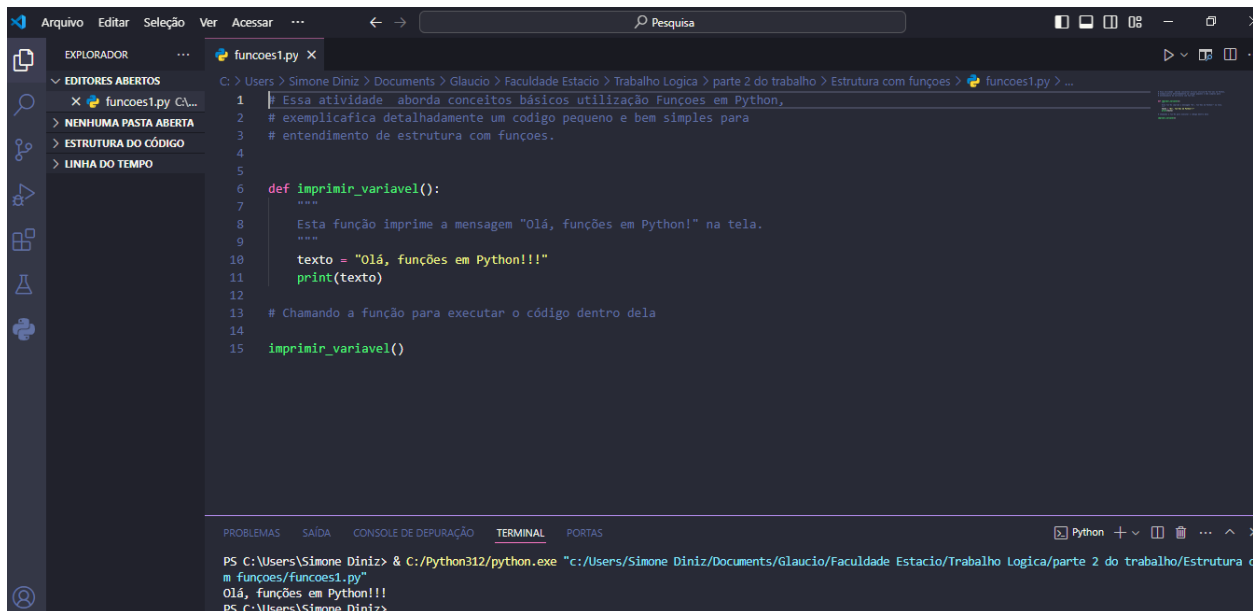
```
Número do intervalo: 9
Número do intervalo: 10
Olá, Carlos!
Olá, Fernanda!
Olá, Jaqueline!
1 x 1 = 1
1 x 2 = 2
1 x 3 = 3
1 x 4 = 4
1 x 5 = 5
1 x 6 = 6
1 x 7 = 7
1 x 8 = 8
1 x 9 = 9
1 x 10 = 10
2 x 1 = 2
2 x 2 = 4
2 x 3 = 6
2 x 4 = 8
2 x 5 = 10
2 x 6 = 12
2 x 7 = 14
2 x 8 = 16
2 x 9 = 18
2 x 10 = 20
3 x 1 = 3
3 x 2 = 6
3 x 3 = 9
```



The screenshot shows the Visual Studio Code interface with a Python file named `estruturas_repeticao2.py` open. The code defines a loop that iterates over a list of names, printing personalized messages. The terminal output shows the results of the loop, displaying multiplication-like patterns for each name and a final list of values.

```
4  
5 # Iteração sobre listas : Lista de nomes, personalizando as mensagens de saída.  
6 # Minhos de laços que abordam o FOR.  
7 # detalha melhor a compreensão de listas para criar listas a partir de iterações.  
8  
8 x 5 = 40  
8 x 6 = 48  
8 x 7 = 56  
8 x 8 = 64  
8 x 9 = 72  
8 x 10 = 80  
9 x 1 = 9  
9 x 2 = 18  
9 x 3 = 27  
9 x 4 = 36  
9 x 5 = 45  
9 x 6 = 54  
9 x 7 = 63  
9 x 8 = 72  
9 x 9 = 81  
9 x 10 = 90  
10 x 1 = 10  
10 x 2 = 20  
10 x 3 = 30  
10 x 4 = 40  
10 x 5 = 50  
10 x 6 = 60  
10 x 7 = 70  
10 x 8 = 80  
10 x 9 = 90  
10 x 10 = 100  
[1, 4, 9, 16, 25]  
PS C:\Users\Simone Diniz>
```

Micro Atividade 5 - Utilização de funções



The screenshot shows the Visual Studio Code interface with a Python file named `funcoes1.py` open. The code defines a function `imprimir_variavel()` that prints a message. The terminal output shows the command to run the script and the resulting output.

```
1 Essa atividade aborda conceitos básicos utilização Funções em Python,  
2 # exemplifica detalhadamente um código pequeno e bem simples para  
3 # entendimento de estrutura com funções.  
4  
5  
6 def imprimir_variavel():  
7     """  
8     Esta função imprime a mensagem "Olá, funções em Python!" na tela.  
9     """  
10    texto = "Olá, funções em Python!!!"  
11    print(texto)  
12  
13    # Chamando a função para executar o código dentro dela  
14  
15    imprimir_variavel()  
  
PS C:\Users\Simone Diniz> & C:/Python312/python.exe "c:/Users/Simone Diniz/Documents/Glaucio/Faculdade Estacio/Trabalho Logica/parte 2 do trabalho/Estrutura com funcoes/funcoes1.py"  
Olá, funções em Python!!!  
PS C:\Users\Simone Diniz>
```

Micro Atividade 6 - Utilização de funções

The screenshot shows a Windows 10 desktop with a dark-themed Visual Studio Code (VS Code) editor. The editor has two tabs open: 'EXPLORADOR' (Explorer) and 'funcoes2.py'. The 'EXPLORADOR' sidebar on the left shows a file tree with 'funcoes2.py' selected. The main editor area displays the contents of 'funcoes2.py', which is a Python script defining a 'loginUsuario' function. The script includes comments in Portuguese and a test call. Below the editor, a 'TERMINAL' window is open, showing the command prompt output for running the script. The terminal shows the command 'python.exe "c:/Users/Simone Diniz/Documents/Glaucio/Faculdade Estacio/Trabalho Logica/parte 2 do trabalho/Estrutura com funcoes/funcoes2.py"' and the output 'Bem-vindo, Administrador!'.

VS Code Explorer:

- EXPLORADOR
- funcoes2.py

VS Code Editor (funcoes2.py):

```

1  def loginUsuario(perfil):
2      """Verifica o perfil do usuário e imprime uma mensagem de boas-vindas.
3
4      Args:
5          perfil: O perfil do usuário (admin ou outro).
6      """
7
8      perfil_minusculo = perfil.lower()
9      if perfil_minusculo == "admin":
10         print("Bem-vindo, Administrador!")
11     else:
12         print("Bem-vindo, Usuário!")
13
14     # abaixo os testes Chamando a função com diferentes perfis.
15     loginUsuario("Admin")
16     #loginUsuario("admin")
17     #loginUsuario("User")
18     #loginUsuario("usuário")

```

VS Code Terminal:

```

PS C:\Users\Simone Diniz> & C:/Python312/python.exe "c:/Users/Simone Diniz/Documents/Glaucio/Faculdade Estacio/Trabalho Logica/parte 2 do trabalho/Estrutura
m funcoes/funcoes2.py"
Bem-vindo, Administrador!
PS C:\Users\Simone Diniz>

```

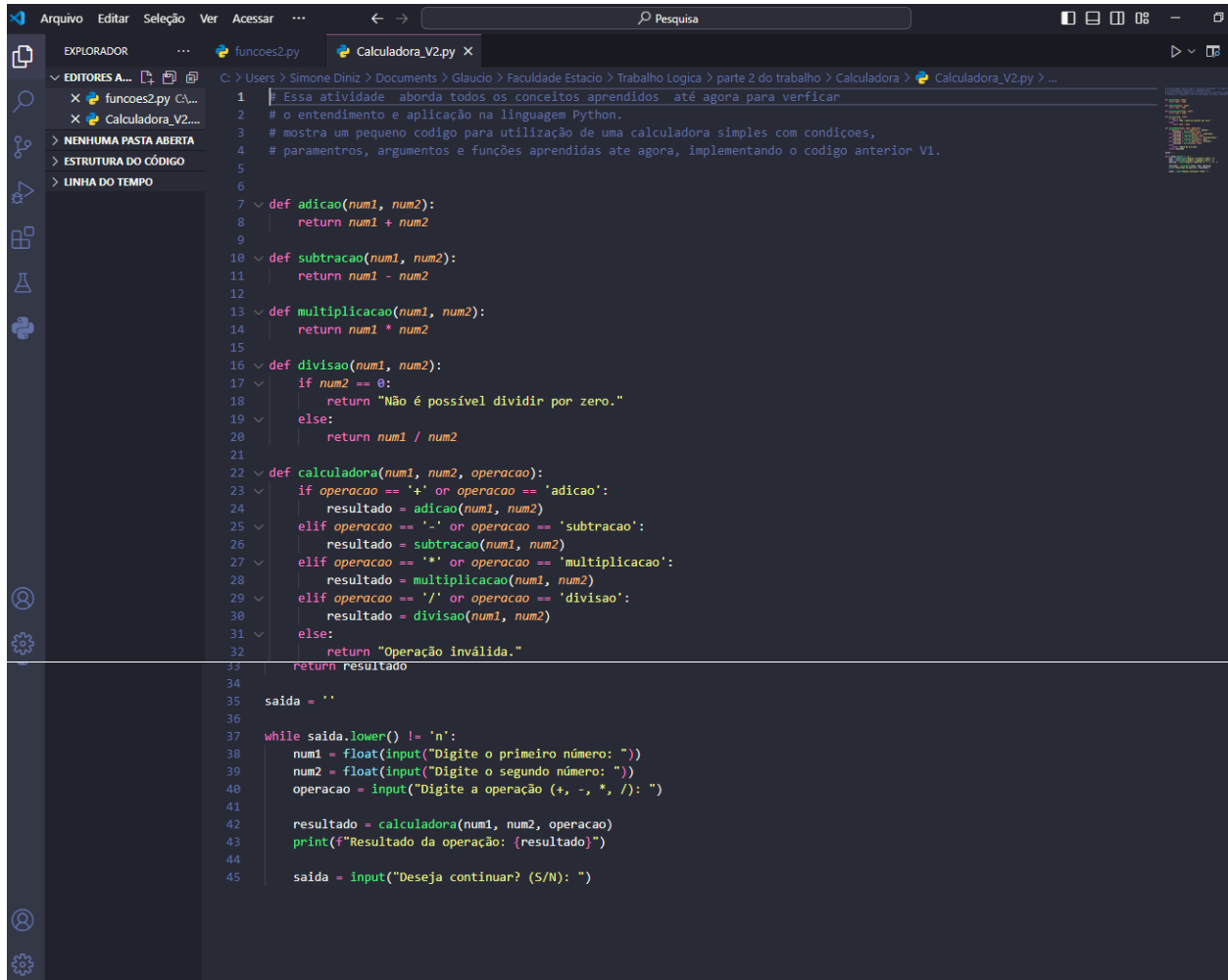
The screenshot shows a code editor with a dark theme. On the left, there's a sidebar with icons for Explorer, Search, Source Control, Run and Debug, and Extensions. The Explorer pane shows a file named 'funcoes2.py' under a folder structure. The main editor area displays the following Python code:

```
1 def loginUsuario(perfil):
2     """Verifica o perfil do usuário e imprime uma mensagem de boas-vindas.
3
4     Args:
5         perfil: O perfil do usuário (admin ou outro).
6     """
7
8     perfil_minusculo = perfil.lower()
9     if perfil_minusculo == "user":
10         print("Bem-vindo, Administrador!")
11     else:
12         print("Bem-vindo, Usuário!")
13
14 # abaixo os testes Chamando a função com diferentes perfis.
15 loginUsuario("Admin")
16 #loginUsuario("admin")
17 #loginUsuario("User")
18 #loginUsuario("usuário")
```

At the bottom, there's a terminal window with the following output:

```
PS C:\Users\Simone Diniz> & C:/Python312/python.exe "c:/Users/Simone Diniz/Documents/Glaucio/Faculdade Estacio/Trabalho Logica/parte 2 do trabalho/Estrutura com funcoes/funcoes2.py"
Bem-vindo, Usuário!
PS C:\Users\Simone Diniz>
```

Trabalho Prático - Calculadora V2



```
1  # Essa atividade aborda todos os conceitos aprendidos até agora para verificar
2  # o entendimento e aplicação na linguagem Python.
3  # mostra um pequeno código para utilização de uma calculadora simples com condições,
4  # parâmetros, argumentos e funções aprendidas até agora, implementando o código anterior V1.
5
6
7  def adicao(num1, num2):
8      return num1 + num2
9
10 def subtracao(num1, num2):
11     return num1 - num2
12
13 def multiplicacao(num1, num2):
14     return num1 * num2
15
16 def divisao(num1, num2):
17     if num2 == 0:
18         return "Não é possível dividir por zero."
19     else:
20         return num1 / num2
21
22 def calculadora(num1, num2, operacao):
23     if operacao == '+' or operacao == 'adicao':
24         resultado = adicao(num1, num2)
25     elif operacao == '-' or operacao == 'subtracao':
26         resultado = subtracao(num1, num2)
27     elif operacao == '*' or operacao == 'multiplicacao':
28         resultado = multiplicacao(num1, num2)
29     elif operacao == '/' or operacao == 'divisao':
30         resultado = divisao(num1, num2)
31     else:
32         return "Operação inválida."
33     return resultado
34
35 saida = ''
36
37 while saida.lower() != 'n':
38     num1 = float(input("Digite o primeiro número: "))
39     num2 = float(input("Digite o segundo número: "))
40     operacao = input("Digite a operação (+, -, *, /): ")
41
42     resultado = calculadora(num1, num2, operacao)
43     print(f"Resultado da operação: {resultado}")
44
45     saida = input("Deseja continuar? (S/N): ")
```

```
Arquivo  Editar  Seleção  Ver  Acessar  ...  Pesquisa

Explorador [Ctrl+Shift+E]
C:\Users\Simone Diniz\Documents\Glaucio\Faculdade Estacio\Trabalho Logica\parte 2 do trabalho\Calculadora > Calculadora_V2.py > ...
funcoes2.py C:\...
X Calculadora_V2...
NENHUMA PASTA ABERTA
ESTRUTURA DO CÓDIGO
LINHA DO TEMPO

funcoes2.py
10 def calculadora(num1, num2):
20     return num1 / num2
21
22 def calculadora(num1, num2, operacao):
23     if operacao == '+' or operacao == 'adicao':
24         resultado = adicao(num1, num2)
25     elif operacao == '-' or operacao == 'subtracao':
26         resultado = subtracao(num1, num2)
27     elif operacao == '*' or operacao == 'multiplicacao':
28         resultado = multiplicacao(num1, num2)
29     elif operacao == '/' or operacao == 'divisao':
30         resultado = divisao(num1, num2)
31     else:
32         return "Operação inválida."
33     return resultado
34
35 saida = ''
36
37 while saida.lower() != 'n':
38     num1 = float(input("Digite o primeiro número: "))
39     num2 = float(input("Digite o segundo número: "))
40     operacao = input("Digite a operação (+, -, *, /): ")
41
42     resultado = calculadora(num1, num2, operacao)

PROBLEMAS  SAÍDA  CONSOLA DE DEPURÇÃO  TERMINAL  PORTAS
Python + - [ ] [ ] ... ^ x

PS C:\Users\Simone Diniz> & C:/Python312/python.exe "c:/Users/Simone Diniz/Documents/Glaucio/Faculdade Estacio/Trabalho Logica/parte 2 do trabalho/Calculadora/Calculadora_V2.py"
Digite o primeiro número: 5
Digite o segundo número: 3
Digite a operação (+, -, *, /): *
Resultado da operação: 15.0
Deseja continuar? (S/N):
```

```
35 saida = ''
36
37 while saida.lower() != 'n':
38     num1 = float(input("Digite o primeiro número: "))
39     num2 = float(input("Digite o segundo número: "))
40     operacao = input("Digite a operação (+, -, *, /): ")
41
42     resultado = calculadora(num1, num2, operacao)

PROBLEMAS  SAÍDA  CONSOLA DE DEPURÇÃO  TERMINAL  PORTAS
Python + - [ ] [ ] ... ^ x

PS C:\Users\Simone Diniz> & C:/Python312/python.exe "c:/Users/Simone Diniz/Documents/Glaucio/Faculdade Estacio/Trabalho Logica/parte 2 do trabalho/Calculadora/Calculadora_V2.py"
Digite o primeiro número: 10
Digite o segundo número: 3
Digite a operação (+, -, *, /): /
Resultado da operação: 3.3333333333333335
```

```
35 saida = ''
36
37 while saida.lower() != 'n':
38     num1 = float(input("Digite o primeiro número: "))
39     num2 = float(input("Digite o segundo número: "))
40     operacao = input("Digite a operação (+, -, *, /): ")
41
42     resultado = calculadora(num1, num2, operacao)

PROBLEMAS  SAÍDA  CONSOLA DE DEPURÇÃO  TERMINAL  PORTAS
Python + - [ ] [ ] ... ^ x

PS C:\Users\Simone Diniz> & C:/Python312/python.exe "c:/Users/Simone Diniz/Documents/Glaucio/Faculdade Estacio/Trabalho Logica/parte 2 do trabalho/Calculadora/Calculadora_V2.py"
Digite o primeiro número: 20
Digite o segundo número: 5
Digite a operação (+, -, *, /): -
Resultado da operação: 15.0
Deseja continuar? (S/N):
```

```
37 while saida.lower() != 'n':
38     num1 = float(input("Digite o primeiro número: "))
39     num2 = float(input("Digite o segundo número: "))
40     operacao = input("Digite a operação (+, -, *, /): ")
41
42     resultado = calculadora(num1, num2, operacao)
43     print(f"Resultado da operação: {resultado}")
44
45     saida = input("Deseja continuar? (S/N): ")

PROBLEMAS  SAÍDA  CONSOLA DE DEPURÇÃO  TERMINAL  PORTAS
Python + - [ ] [ ] ... ^ x

PS C:\Users\Simone Diniz> & C:/Python312/python.exe "c:/Users/Simone Diniz/Documents/Glaucio/Faculdade Estacio/Trabalho Logica/parte 2 do trabalho/Calculadora/Calculadora_V2.py"
Digite o primeiro número: 50
Digite o segundo número: 26
Digite a operação (+, -, *, /): +
Resultado da operação: 76.0
Deseja continuar? (S/N):
```



```
35     saida =  
36  
37     while saida.lower() != 'n':  
38         num1 = float(input("Digite o primeiro número: "))  
39         num2 = float(input("Digite o segundo número: "))  
40         operacao = input("Digite a operação (+, -, *, /): ")  
41  
42         resultado = calculadora(num1, num2, operacao)  
43         print(f"Resultado da operação: {resultado}")  
44  
45         saida = input("Deseja continuar? (S/N): ")
```

PROBLEMAS SAÍDA CONSOLE DE DEPUÇÃO **TERMINAL** PORTAS

Python + - [] [] ... ^ x

```
P5 C:\Users\Simone Diniz> & C:/Python312/python.exe "c:/Users/Simone Diniz/Documents/Glaucio/Faculdade Estacio/Trabalho Logica/parte 2 do trabalho/Calculadora/  
Calculadora V2.py"  
Digite o primeiro número: 10  
Digite o segundo número: 0  
Digite a operação (+, -, *, /): /  
Resultado da operação: Não é possível dividir por zero.  
Deseja continuar? (S/N):
```