

Oficina de Programação Básica em Python



PET - Programa de Educação Tutorial

Oficina de Python - Aula 3

Conteúdo Aula 3:

- **FUNÇÕES**
- **BIBLIOTECAS**



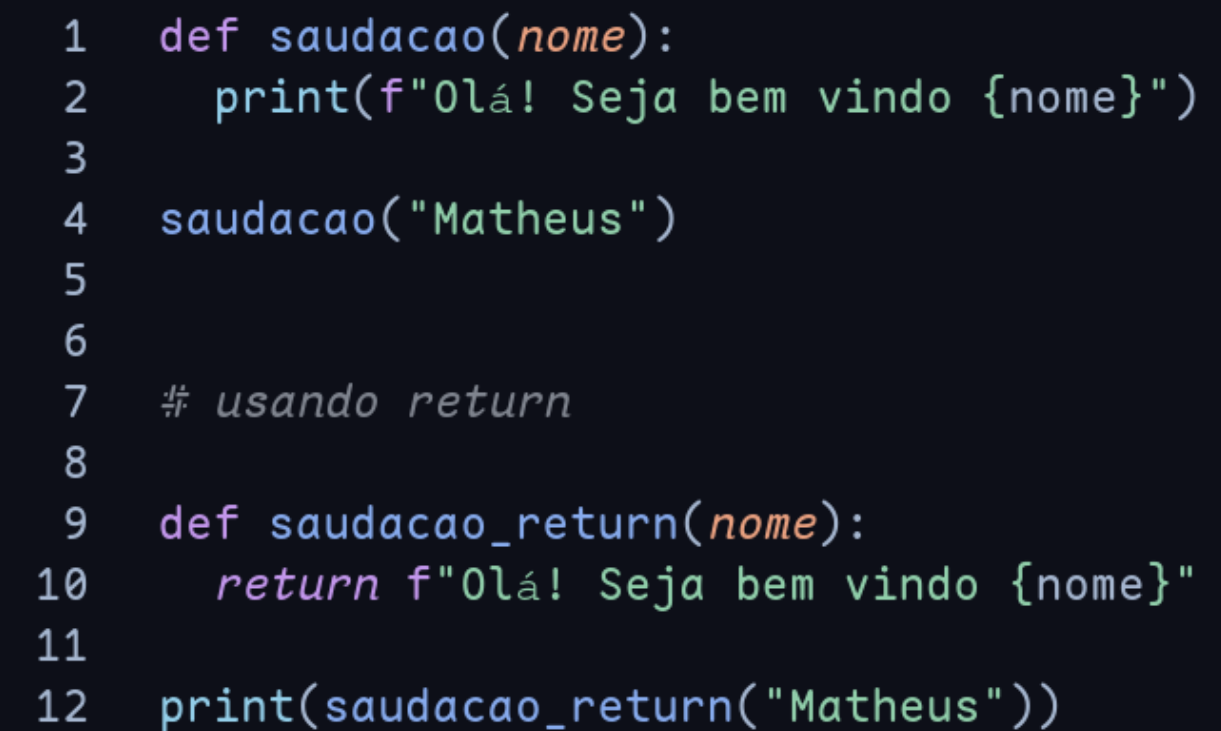
FUNÇÕES

- FUNÇÃO
- PARÂMETROS
- PARÂMETROS OPCIONAIS
- LAMBDA
- RECURSIVIDADE

FUNÇÕES

Uma função em programação é um bloco de código autônomo que executa uma tarefa específica. Ela recebe entrada, processa-a e retorna um resultado. Em Python, você define uma função com a palavra-chave `def`, seguida do nome da função, parâmetros entre parênteses e um bloco de código indentado.

Exemplo de uma função simples em Python:



```
1  def saudacao(nome):  
2      print(f"Olá! Seja bem vindo {nome}")  
3  
4  saudacao("Matheus")  
5  
6  
7  # usando return  
8  
9  def saudacao_return(nome):  
10     return f"Olá! Seja bem vindo {nome}"  
11  
12  print(saudacao_return("Matheus"))
```

Exercícios

- **Escreva uma função de uma calculadora que vai receber dois numeros e uma operação e como resultado deve retornar o resultado da operação**

LAMBDA

As funções lambda em Python são funções anônimas e de uma única expressão. Elas são criadas usando a palavra-chave lambda, seguida por uma lista de argumentos, dois pontos (:) e a expressão que será avaliada quando a função for chamada. Essas funções são úteis quando se precisa de uma função temporária para uma operação simples.

A sintaxe básica de uma lambda function em Python é a seguinte:



```
1 soma = lambda x, y: x + y
2 print(soma(2, 3)) # Saída: 5
```

Exercícios

- **Escreva uma função lambda que receba um número e retorne se ele é par ou não. (True | False)**
- **Escreva uma função lambda que receba um número e retorne se ele é impar ou não. (True | False)**

FUNÇÕES: PARÂMETROS OPCIONAIS

Parâmetros opcionais são aqueles que podem ser fornecidos ou não ao chamar uma função em programação. Eles oferecem flexibilidade na forma como você pode chamar uma função, permitindo que você forneça valores apenas para os parâmetros que são relevantes para a execução específica que você está realizando.



```
1 def saudacao(nome, saudacao_personalizada="Olá"):
2     print(saudacao_personalizada + ", " + nome + "!")
3
4 # Chamando a função com ambos os parâmetros
5 saudacao("João", "Oi") # Saída: Oi, João!
6
7 # Chamando a função com apenas um parâmetro
8 saudacao("Maria") # Saída: Olá, Maria!
9
```


RECURSIVIDADE

Recursão é um conceito em programação onde uma função chama a si mesma durante sua execução. Em outras palavras, uma função recursiva é uma função que se define em termos de si mesma. Isso pode ser útil para resolver problemas que podem ser divididos em subproblemas menores e mais simples.



```
1  def fatorial(n):  
2      # Caso base  
3      if n == 0 or n == 1:  
4          return 1  
5      # Caso recursivo  
6      else:  
7          return n * fatorial(n-1)  
8  
9  # Exemplo de uso  
10 resultado = fatorial(5)  
11 print(resultado) # Saída: 120  
12
```

Exercícios

- **Escreva uma função que receba um número e a potência desejada e utilize de recursividade para calcular a potência de um numero. Exemplo: potencia_recursiva(2, 3) # Saída: 8**
- **Escreva uma função que receba um número e utilize de recursão para somar todos os algarismos do número. Exemplo: soma_algarismos(1234) # Saída: 10**

BIBLIOTECAS

Uma biblioteca em programação é um conjunto de funções e rotinas pré-escritas que podem ser utilizadas por programas para realizar tarefas específicas. Essas funções são geralmente organizadas em módulos, proporcionando uma maneira eficiente de reutilizar código, economizando tempo e esforço na implementação de funcionalidades comuns.



```
1  import math
2
3  # Agora, você pode usar funções da biblioteca math
4  raiz_quadrada = math.sqrt(25)
```

Exercícios

- **Utilizando a biblioteca random, crie um mini-jogo de adivinhação, onde você deve gerar um número aleatório de 1 a 10 e deve dar 3 tentativas para usuário acertar.**