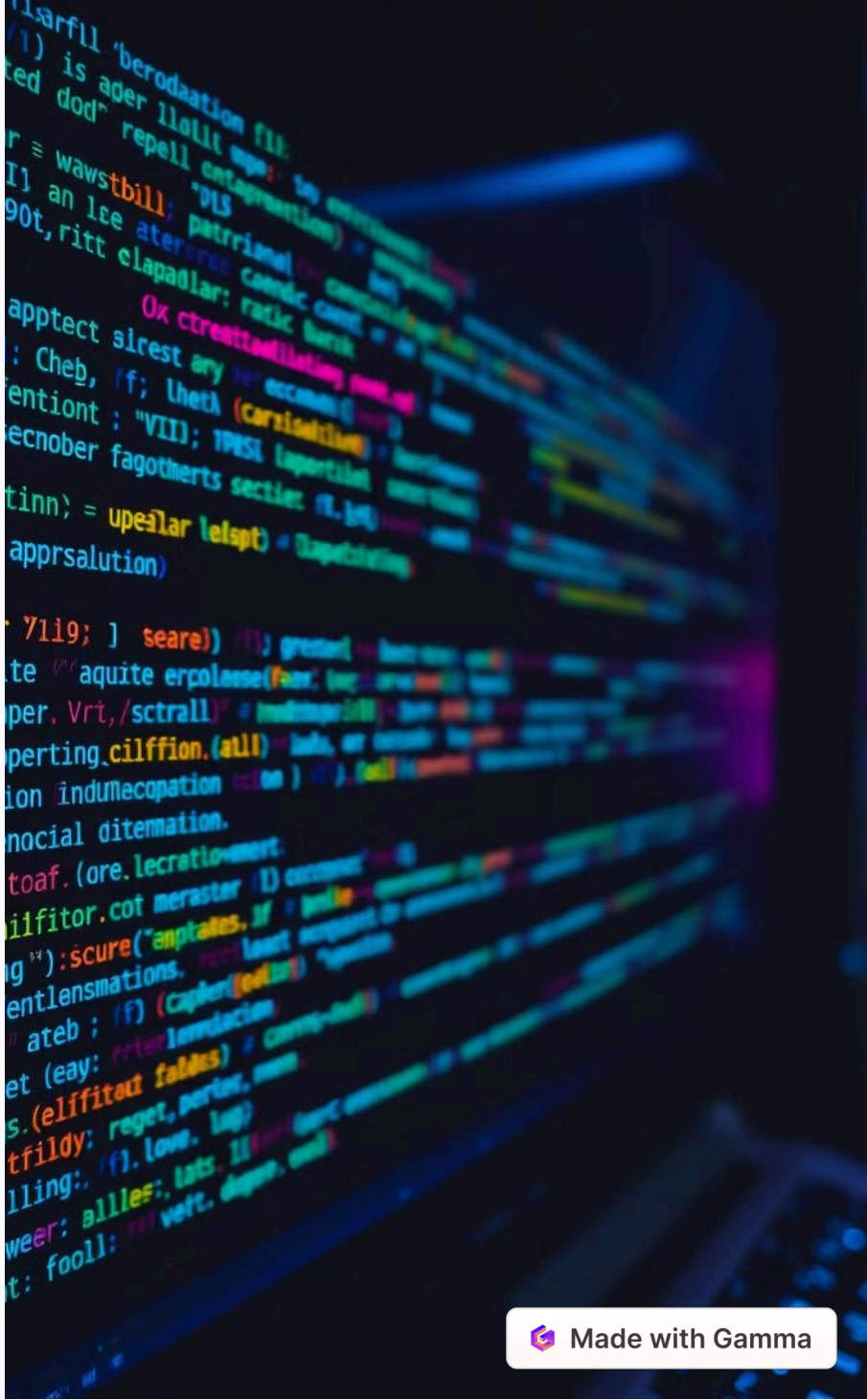
TRABALHO Microatividades 1,2,3,4,5,6; DGT2817 - Logica, Algoritmos e Programação de Computadores

Este trabalho visa explorar a programação em Python. Ele aborda estruturas de controle de fluxo e funções, ferramentas essenciais para a lógica de programação e desenvolvimento de soluções eficientes.

by Claudia Duarte



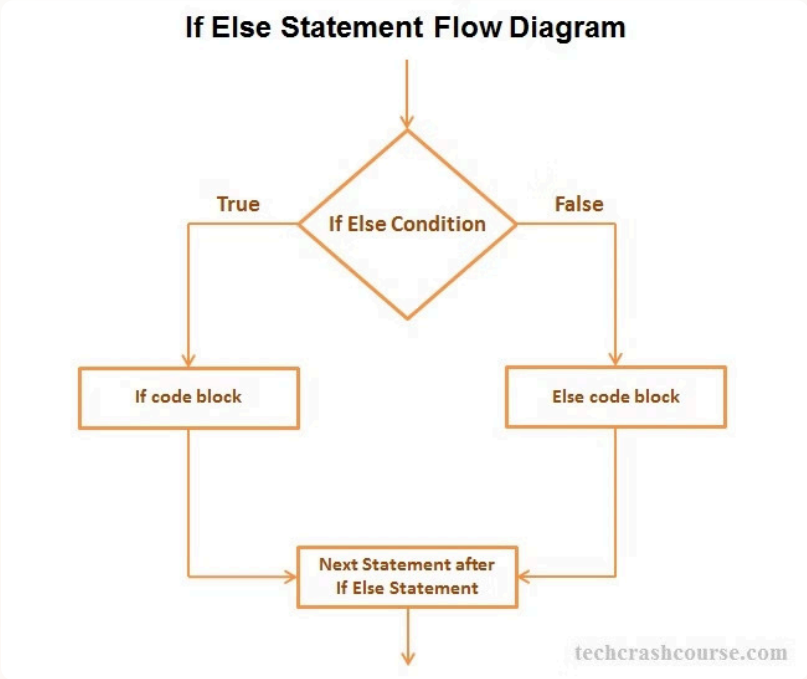
Estruturas de Condição if e else em Python

if

A estrutura if é utilizada para executar um bloco de código apenas se uma condição for verdadeira. Por exemplo, "if idade >= 18:" - nesse caso, o bloco de código será executado apenas se a idade for maior ou igual a 18.

else

A estrutura else é utilizada para executar um bloco de código caso a condição do if não seja verdadeira. Por exemplo, "else: print('Você é menor de idade')" - nesse caso, o bloco de código será executado apenas se a idade for menor que 18.



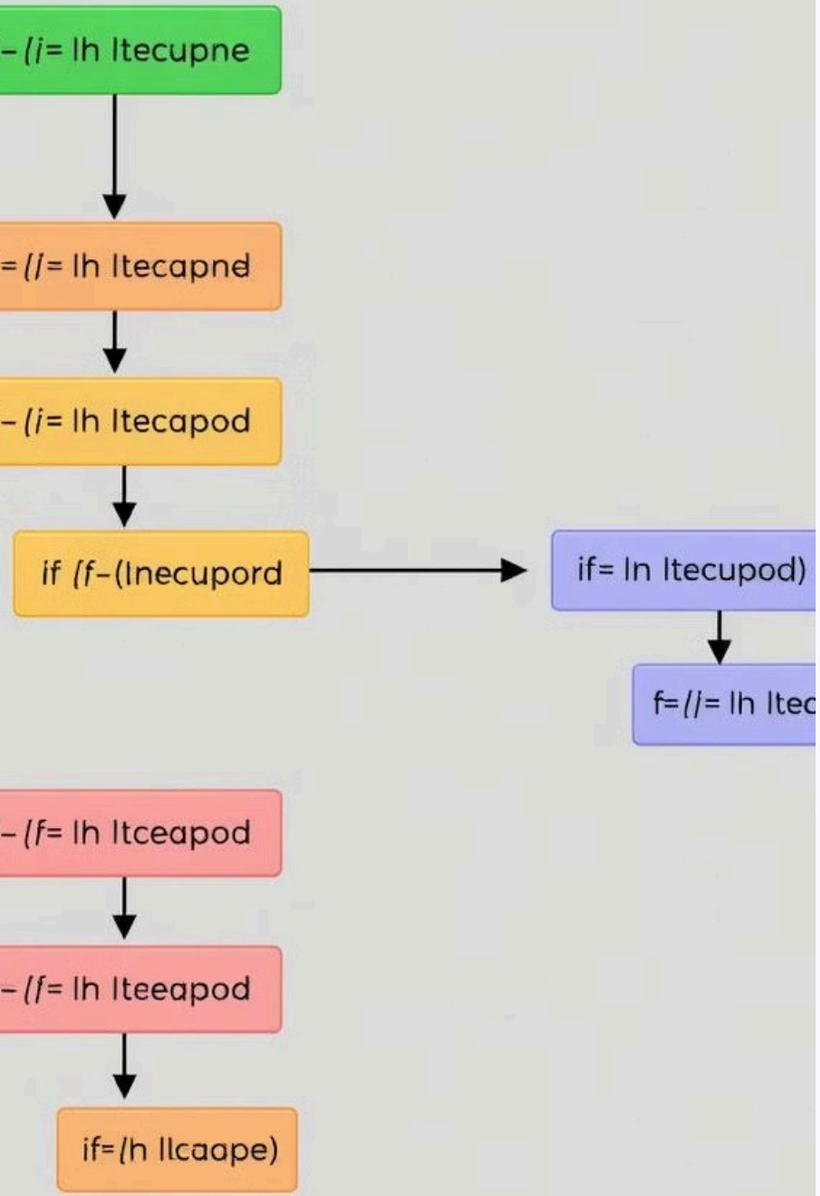
Estrutura de Condição else if (elif) em Python

1 elif

A estrutura elif permite verificar várias condições em sequência. É uma forma concisa de testar diversas possibilidades dentro de um mesmo bloco de código.

2 Exemplo

Podemos verificar a idade de uma pessoa e imprimir uma mensagem específica para cada faixa etária: "if idade < 18: ... elif idade < 60: ... else: ..."



Estruturas de condição if e else em Python

1 Controle de Fluxo

As estruturas if e else são fundamentais para o controle de fluxo em um programa. Elas permitem que você execute blocos de código específicos, dependendo de uma condição que pode ser verdadeira ou falsa.

2 Sintaxe

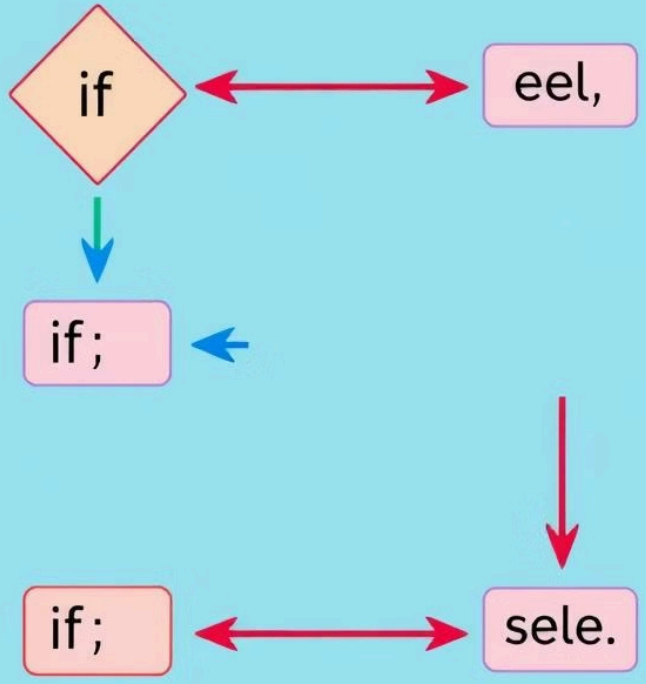
Em Python, a sintaxe para if e else é clara e concisa. Você inicia com a palavra-chave if seguida da condição, um dois pontos (:) e, em seguida, o bloco de código a ser executado se a condição for verdadeira. O bloco else, opcional, é executado se a condição for falsa.

3 Exemplo

Imagine um programa que verifica a idade do usuário e determina se ele pode dirigir. Usando if e else, você pode escrever código que imprime uma mensagem diferente, dependendo da idade do usuário.

4 Aplicações

As estruturas if e else são amplamente utilizadas em diversas situações, desde a validação de dados até a tomada de decisões dentro de um programa.



Estruturas de condição if e else em Python

1 Controle de Fluxo

As estruturas if e else permitem que o código tome decisões com base em condições, direcionando o fluxo de execução.

2 Verificação de Condições

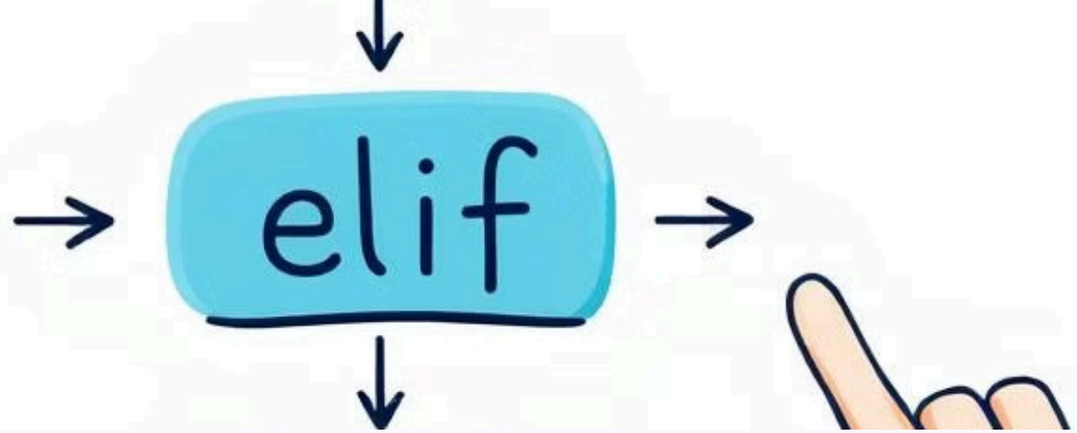
O if avalia uma condição booleana e executa um bloco de código se a condição for verdadeira.

3 Alternativa

O else executa um bloco de código se a condição do if for falsa, fornecendo uma alternativa.

4 Exemplo

Se a nota for maior ou igual a 7, o aluno é aprovado; caso contrário, ele é reprovado.



Estrutura de condição else if (elif) em Python

1

Condições Encadeadas

O elif permite adicionar mais condições a um bloco if-else, verificando-as em sequência.

2

Múltiplas Opções

Se a condição do if for falsa, o código verifica as condições dos elifs em ordem.

3

Alternativa Final

O else é executado somente se todas as condições anteriores forem falsas.

Estrutura de condição else if (elif) em Python

Múltiplas Condições

A estrutura elif (abreviação de "else if") permite que você avalie múltiplas condições em sequência, fornecendo mais flexibilidade para seus programas. Cada condição elif é verificada apenas se a condição anterior for falsa.

Sintaxe

A sintaxe é semelhante ao if e else. Você usa elif seguido da condição e um bloco de código. Se uma condição elif for verdadeira, seu bloco de código é executado e a verificação de outras condições elif é interrompida.

Exemplo

Imagine um programa que classifica a nota de um aluno em diferentes categorias: A, B, C, D ou F. Você pode usar elif para verificar cada faixa de nota e imprimir a classificação correspondente.

Estruturas de repetição while e for em Python

Repetição Condicionada

O while executa um bloco de código enquanto uma condição for verdadeira, repetindo o processo até a condição ser falsa.

Repetição Definida

O for executa um bloco de código para cada elemento de uma sequência, como uma lista ou uma string.

Exemplo

O while pode ser usado para ler números do usuário até que ele digite zero. O for pode ser usado para iterar sobre cada letra de uma palavra.

Estrutura de repetição while em Python

1 | 2 | 3 | 4

Repetição Condicionada

A estrutura while é usada para repetir um bloco de código enquanto uma condição específica for verdadeira. O loop continua a ser executado até que a condição se torne falsa.

Sintaxe

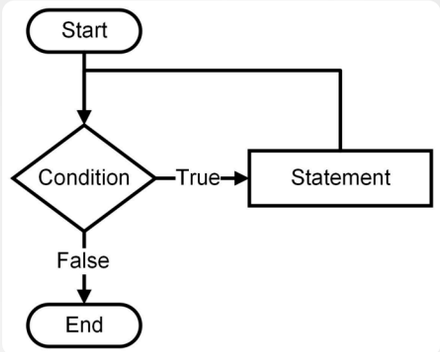
A sintaxe da estrutura while é simples: while seguido da condição e um bloco de código indentado.

Exemplo

Você pode usar um loop while para solicitar que o usuário insira um número até que ele digite um número específico. O loop continua a ser executado até que a condição de parada seja satisfeita.

Aplicações

Os loops while são úteis em cenários onde o número de iterações não é conhecido de antemão, como quando você precisa continuar processando dados até atingir um critério específico.



Estrutura de Repetição while em Python

Condição

O bloco de código dentro do loop while é executado repetidamente enquanto a condição for verdadeira.

1

Execução

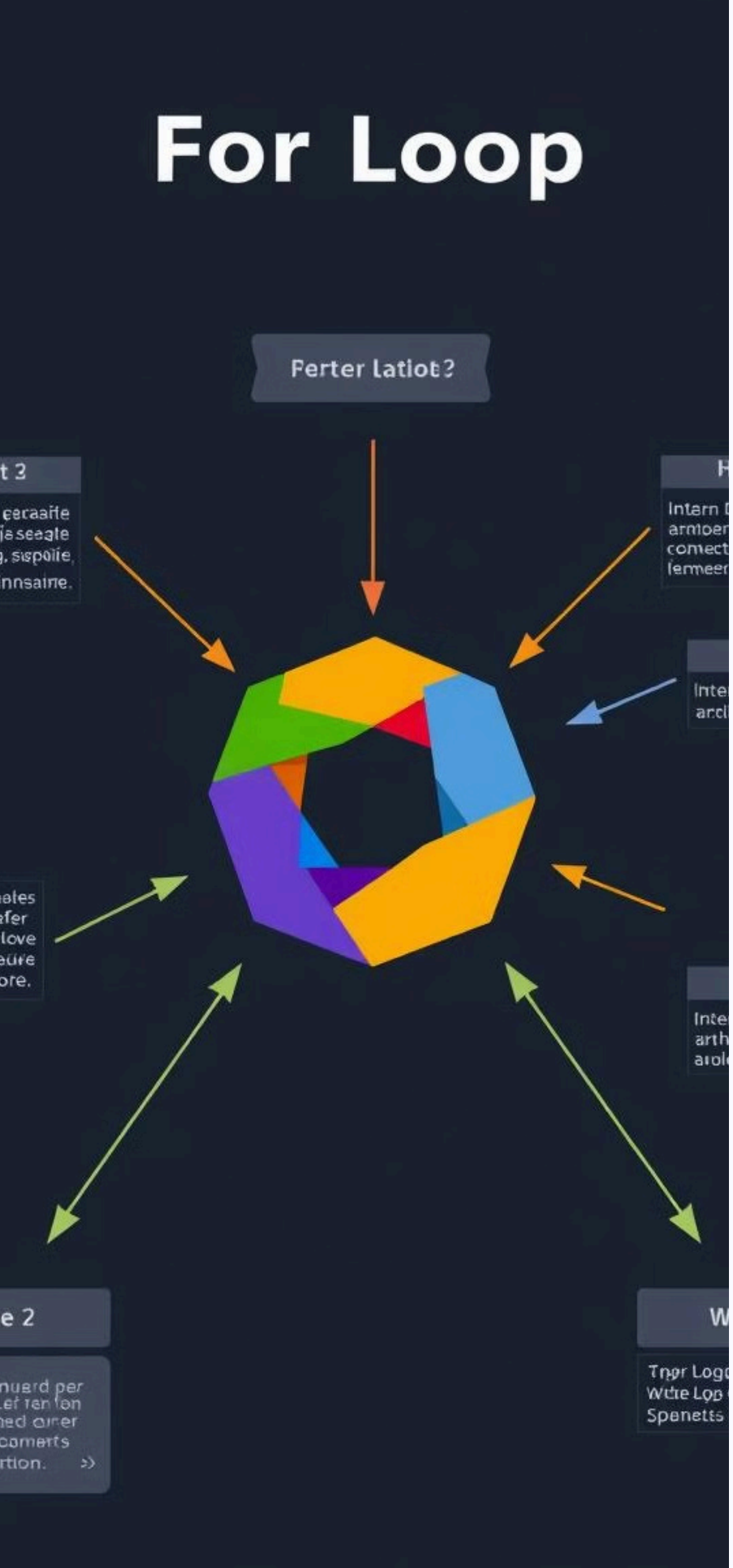
A cada iteração, a condição é verificada novamente. Se a condição for verdadeira, o loop continua a ser executado.

2

Parada

O loop while termina quando a condição se torna falsa.

3



Estrutura de repetição for em Python

1

Iteração em Sequências

O loop for em Python é usado para iterar sobre os elementos de uma sequência (listas, tuplas, strings, etc.) de forma eficiente, executando um bloco de código para cada elemento.

2

Sintaxe

A sintaxe é for seguido da variável de iteração, a palavra-chave in e a sequência. O bloco de código indentado é executado para cada elemento da sequência.

3

Exemplo

Você pode usar um loop for para imprimir cada item de uma lista. O loop itera sobre cada elemento da lista, executando o código dentro do bloco do loop.

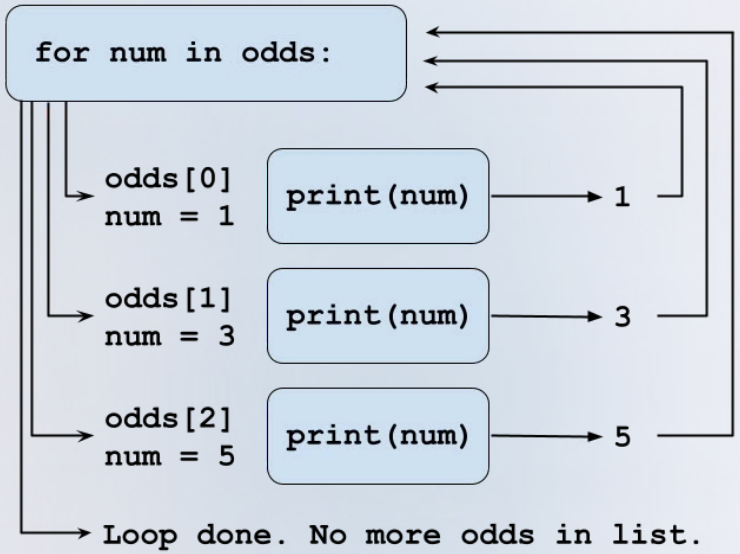
4

Aplicações

Os loops for são muito úteis para trabalhar com conjuntos de dados, processar listas, strings e realizar operações iterativas.

Estrutura de Repetição for em Python

```
odds = [1, 3, 5]
for num in odds:
    print(num)
```



1

Iterável

O loop for é utilizado para iterar sobre um iterável, como uma lista, tupla ou string.

2

Variável de Iteração

A cada iteração, o valor do elemento atual do iterável é atribuído à variável de iteração.

3

Bloco de Código

O bloco de código dentro do loop for é executado para cada elemento do iterável.

Funções e Argumentos em Python

Blocos Reutilizáveis

Funções em Python são blocos de código reutilizáveis que executam uma tarefa específica. Elas ajudam a organizar seu código, tornando-o mais modular e fácil de manter.

Sintaxe

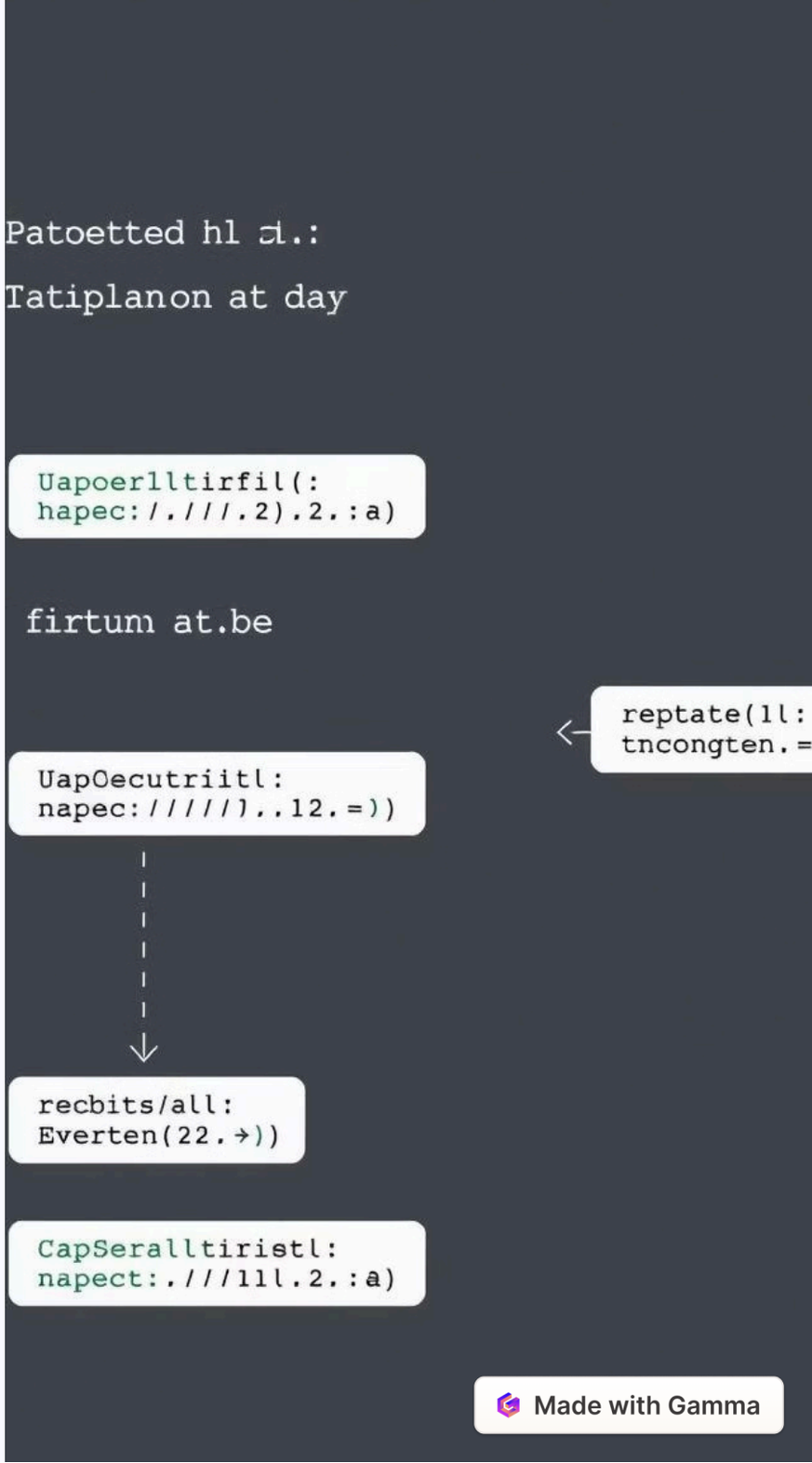
Você define uma função usando a palavra-chave def, seguida do nome da função, parênteses (que podem conter parâmetros) e dois pontos (:). O bloco de código da função é indentado.

Argumentos

Os argumentos são valores que você passa para uma função quando a chama. Eles permitem que você personalize o comportamento da função para diferentes entradas.

Retorno

Uma função pode retornar um valor usando a palavra-chave return. Isso permite que você use o resultado da função em outras partes do seu código.



Funções e Argumentos em Python

Funções

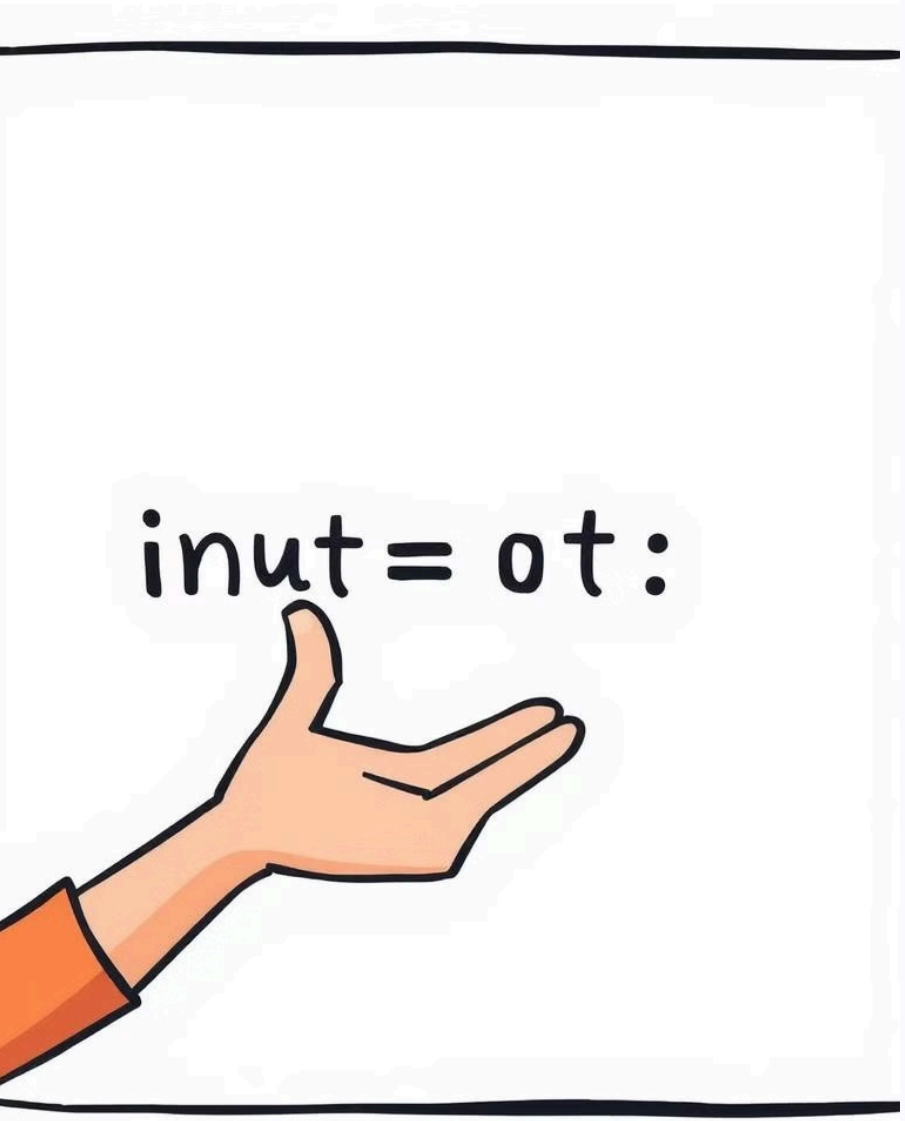
Funções são blocos de código reutilizáveis que executam tarefas específicas. Elas são definidas com a palavra-chave "def".

Argumentos

Argumentos são valores que são passados para uma função quando ela é chamada. Eles são utilizados para fornecer dados à função.

Exemplo

"def soma(a, b): return a + b" - esta função recebe dois argumentos, "a" e "b", e retorna a soma dos dois.



Funções e argumentos de funções em Python

Blocos Reutilizáveis

As funções agrupam um conjunto de instruções que podem ser chamadas várias vezes no código.

Entrada e Saída

As funções podem receber argumentos como entrada e retornar um valor como saída.

Reutilização

Funções permitem organizar o código, evitar repetições e tornar o código mais legível e modular.

Exemplo

Uma função chamada 'soma' poderia receber dois números como entrada e retornar a soma deles.

Refazer a calculadora utilizando estrutura condicional e funções

Operação	Descrição
Soma	Adiciona dois números.
Subtração	Subtrai um número do outro.
Multiplicação	Multiplica dois números.
Divisão	Divide um número pelo outro.