



Lógica de Programação

Introdução ao Python

Checando o Aprendizado

- Na sua opinião o que é software?
- Existem diversos tipos de software. Cite ao menos uma tipo de software:
- Cite ao menos uma das 4 camadas de abstração do software:
- Quais são as 4 arquiteturas de software?



Introdução

Nesta aula completa, nós iremos embarcar em uma jornada fascinante pela lógica de programação utilizando a poderosa linguagem Python. Prepare-se para desvendar os segredos da sintaxe, **variáveis**, **tipos de dados** e **operadores**, os pilares fundamentais para construir nossos primeiros programas em Python.



Sintaxe Básica: A Arquitetura do Seu Código

A **sintaxe** é a linguagem que o **Python** utiliza para entender suas instruções. Imagine um código como uma bela **construção**: a sintaxe define a estrutura e organização, garantindo que tudo se encaixe perfeitamente.



Sintaxe Básica: A Arquitetura do Seu Código

- **Espaços Brancos:** No Python, os espaços em branco são essenciais para definir blocos de código, criando uma hierarquia visual que facilita a leitura e a compreensão do programa. Imagine que cada recuo com quatro espaços seja como um andar em um prédio: quanto mais recuado, mais profundo o nível de aninhamento.

Sintaxe Básica: A Arquitetura do Seu Código

- **Comentários:** Para deixar anotações no seu código que não serão executadas pelo Python, utilize o símbolo `#`. Pense neles como plaquinhas explicativas em um museu, ajudando você e outros programadores a entenderem o que cada parte do código faz.

Sintaxe Básica: A Arquitetura do Seu Código

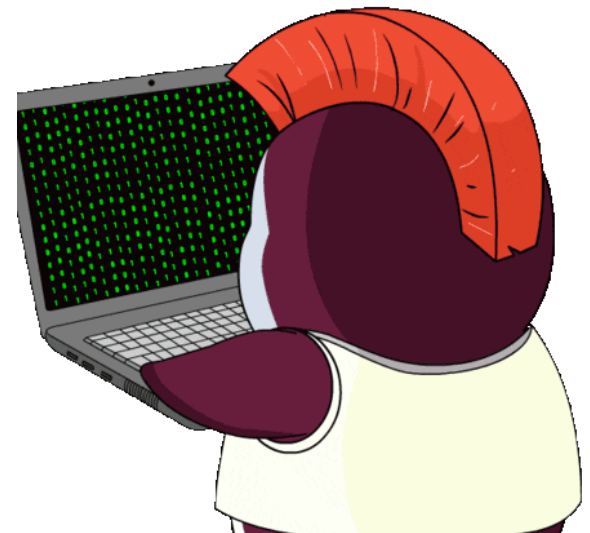
- **Instruções:** Cada linha do seu código geralmente contém uma instrução, como se cada uma fosse uma ordem que o Python precisa seguir. Imagine cada linha como um tijolo na construção: uma a uma, elas formam a base sólida do seu programa.

Variáveis: Armazenando as Informações Essenciais

As **variáveis** são **compartimentos** que **guardam** os **dados** que seu programa precisa para funcionar. Imagine-as como caixas organizadoras em um armário: cada uma guarda um tipo específico de informação, como **números**, **textos** ou **listas**.

Variáveis: Armazenando as Informações Essenciais

- **Declaração de Variáveis:** Para criar uma variável, basta utilizar a seguinte estrutura: `nome_da_variavel = valor`. Pense no `nome_da_variavel` como a etiqueta da caixa e no `valor` como o conteúdo que você deseja armazenar dentro dela.

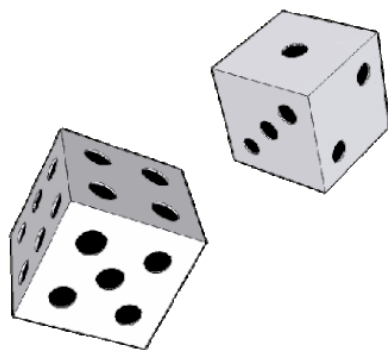


Variáveis: Armazenando as Informações Essenciais

- **Tipos de Dados:** O Python possui diversos tipos de dados para representar diferentes tipos de informações:
 - **int:** Números inteiros, como 1, 10, 200 (imagine caixas para guardar quantidades).
 - **float:** Números reais, como 3.14, 5.2, 100.0 (imagine caixas para guardar valores precisos).
 - **str:** Textos, como "Olá, mundo!", "Eu adoro programar!", "Python é incrível!" (imagine caixas para guardar mensagens e frases).

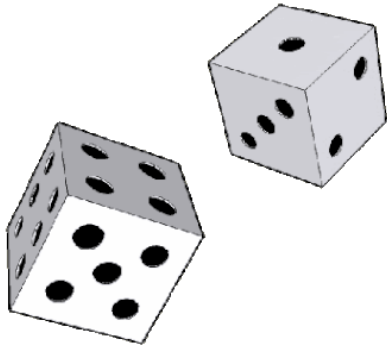
Variáveis: Armazenando as Informações Essenciais

- **Tipos de Dados:** O Python possui diversos tipos de dados para representar diferentes tipos de informações:
 - **bool:** Valores lógicos, **True** ou **False** (imagine caixas para guardar se algo é verdadeiro ou falso).
 - **list:** Listas ordenadas de valores, como **[1, 2, 3, 4, 5]**, **["banana", "maçã", "laranja"]**, **[True, False, True]**. Pense nelas como caixas que armazenam várias informações de um mesmo tipo, organizadas em uma sequência.



Variáveis: Armazenando as Informações Essenciais


- **Operações com Variáveis:** Utilize operadores matemáticos (+, -, *, /, %) para realizar operações com os valores armazenados nas variáveis. Imagine os operadores como ferramentas para manipular o conteúdo das caixas: você pode somar, subtrair, multiplicar, dividir e até mesmo tirar o resto da divisão entre os valores.



Tipos de Dados: Dominando as Diversas Formas de Informação

A solid green horizontal bar.

Cada tipo de dado no Python possui características únicas, permitindo que você represente diferentes tipos de informações com precisão.

Abstract geometric shapes in teal, yellow, and green in the bottom-left corner.

Tipos de Dados: Dominando as Diversas Formas de Informação

- Números (**int** e **float**):
 - Números Inteiros (**int**): Representam valores inteiros sem casas decimais, como 10, 25, -100. Imagine-os como moedas que só podem ter valores inteiros (1, 2, 5, 10...).
 - Números Reais (**float**): Representam valores com casas decimais, como 3.1415, 9.81, -50.23. Imagine-os como notas de dinheiro que podem ter valores fracionários (R\$ 1,50, R\$ 10,30...).

Tipos de Dados: Dominando as Diversas Formas de Informação

- Textos (`str`):
 - Armazenam sequências de caracteres, permitindo que você represente palavras, frases e parágrafos. Imagine-os como folhas de papel em que você pode escrever qualquer texto que desejar.

Tipos de Dados: Dominando as Diversas Formas de Informação

- Booleanos (`bool`):
 - Representam valores (Verdadeiro ou Falso)

Operadores: Dominando as Ferramentas da Programação

Os **operadores** são as ferramentas essenciais para realizar operações matemáticas, comparações e manipulações de dados em seus programas Python.

Operadores: Dominando as Ferramentas da Programação

- Operadores Matemáticos:



- `+`: Adição (soma dois valores).



- `-`: Subtração (diminui um valor de outro).



- `*`: Multiplicação (multiplica dois valores).



- `/`: Divisão (divide um valor por outro).
- `%`: Módulo (obtem o resto da divisão entre dois valores).
- `**`: Potenciação (eleva um valor à potência de outro).

Operadores: Dominando as Ferramentas da Programação



- Operadores de Comparação:
 - `==`: Igualdade (verifica se dois valores são iguais).
 - `!=`: Desigualdade (verifica se dois valores são diferentes).
 - `>`: Maior que (verifica se um valor é maior que outro).
 - `<`: Menor que (verifica se um valor é menor que outro).
 - `>=`: Maior ou igual que (verifica se um valor é maior ou igual a outro).
 - `<=`: Menor ou igual que (verifica se um valor é menor ou igual a outro).

Operadores: Dominando as Ferramentas da Programação

- Operadores Lógicos:
- **and**: E (verifica se duas condições são verdadeiras ao mesmo tempo).
- **or**: Ou (verifica se pelo menos uma condição é verdadeira).
- **not**: Não (inverte o valor de uma condição).

Exercícios Práticos: Colocando a Mão na Massa!

Operador AND

```
# Verifica se ambas as condições são verdadeiras
x = 5
y = 10
if x > 0 and y < 15:
    print("Ambas as condições são verdadeiras")
```

Exercícios Práticos: Colocando a Mão na Massa!

Operador OR

```
# Verifica se pelo menos uma das condições é verdadeira
a = 20
b = 30
if a > 25 or b < 25:
    print("Pelo menos uma das condições é verdadeira")
```

Exercícios Práticos: Colocando a Mão na Massa!

Operador NOT

```
# Inverte o resultado da condição
idade = 20
if not idade >= 18:
    print("Você é menor de idade")
else:
    print("Você é maior de idade")
```

Exercícios Práticos: Colocando a Mão na Massa!

Combinação de Operadores Lógicos

```
# Usando combinações de operadores lógicos
temperatura = 25
if temperatura > 30 and temperatura < 40:
    print("A temperatura está entre 30°C e 40°C")
elif temperatura <= 30 or temperatura >= 40:
    print("A temperatura está fora da faixa desejada")
```


Lembre-se:

- **A prática leva à perfeição.** Quanto mais você programar, mais aprimorará suas habilidades.
- **Não tenha medo de errar.** Erros fazem parte do processo de aprendizado e são oportunidades para identificar pontos de melhoria.