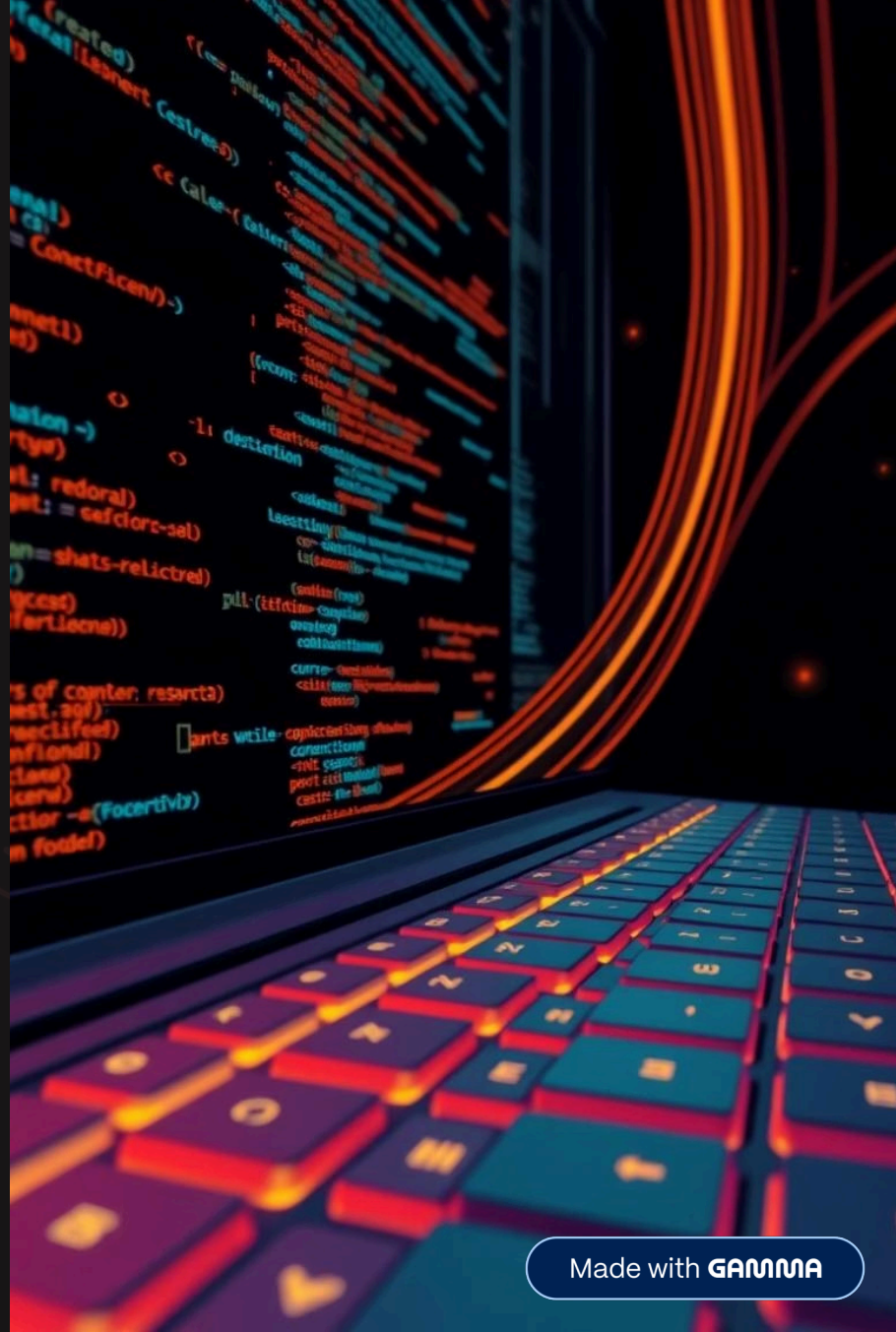
Desvendando os Operadores em Python

Olá, futuros programadores! Hoje vamos explorar as ferramentas essenciais que permitem ao Python “pensar” e “decidir”: os operadores. Preparem-se para dar superpoderes aos seus códigos!



Encontro 3

Operadores Aritméticos, Comparação e Lógicos

Dominar

Operadores aritméticos (+, -, *, /, //, %, **)

Aplicar

Operadores de comparação (==, !=, <, >, <=, >=)

Usar

Operadores lógicos (and, or, not)

Combinar

Operadores para resolver problemas práticos

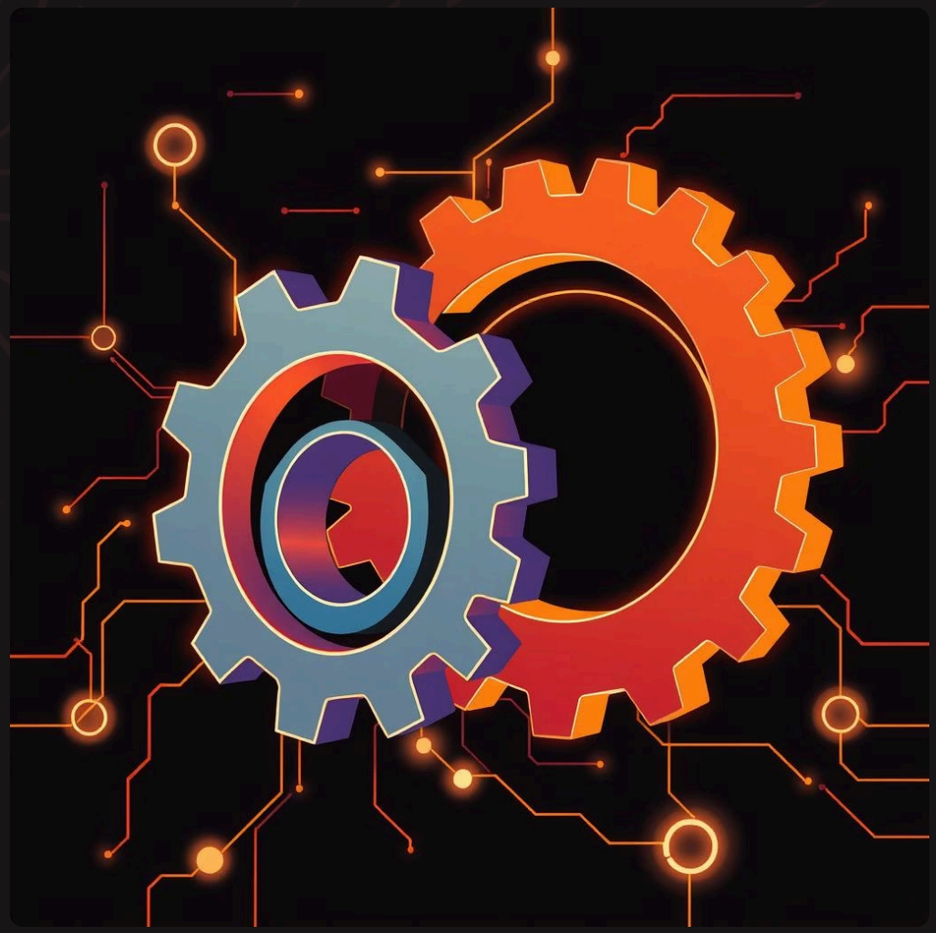
Revisão: O que já sabemos?

Dos encontros anteriores:

- **Entrada/Saída**: `input()` e `print()`
- **Tipos de Dados**: `int`, `float`, `str`, `bool`
- **Conversões**: `int()`, `float()`, `str()`, `bool()`

Hoje: Operadores para criar lógica!

Vamos fazer os **dados "conversarem" entre si**, realizando cálculos e tomando decisões. Essa é a base para programas inteligentes!



➕ Operadores Aritméticos: As Bases dos Cálculos

```
a = 10
b = 3

print(a + b)  # 13 (adição)
print(a - b)  # 7 (subtração)
print(a * b)  # 30 (multiplicação)
print(a / b)  # 3.333... (divisão real)
print(a // b) # 3 (divisão inteira)
print(a % b)  # 1 (resto/módulo)
print(a ** b) # 1000 (potenciação)
```

Esses operadores são a espinha dorsal de qualquer cálculo. Eles permitem que você some, subtraia, multiplique, divida e até eleve números a potências, formando a base para interações matemáticas complexas em seus programas.

Dica Essencial:

O operador **/** sempre retorna um número com casas decimais (float), enquanto **//** sempre retorna um número inteiro, descartando as casas decimais. O operador **%** é crucial para verificar divisibilidade.

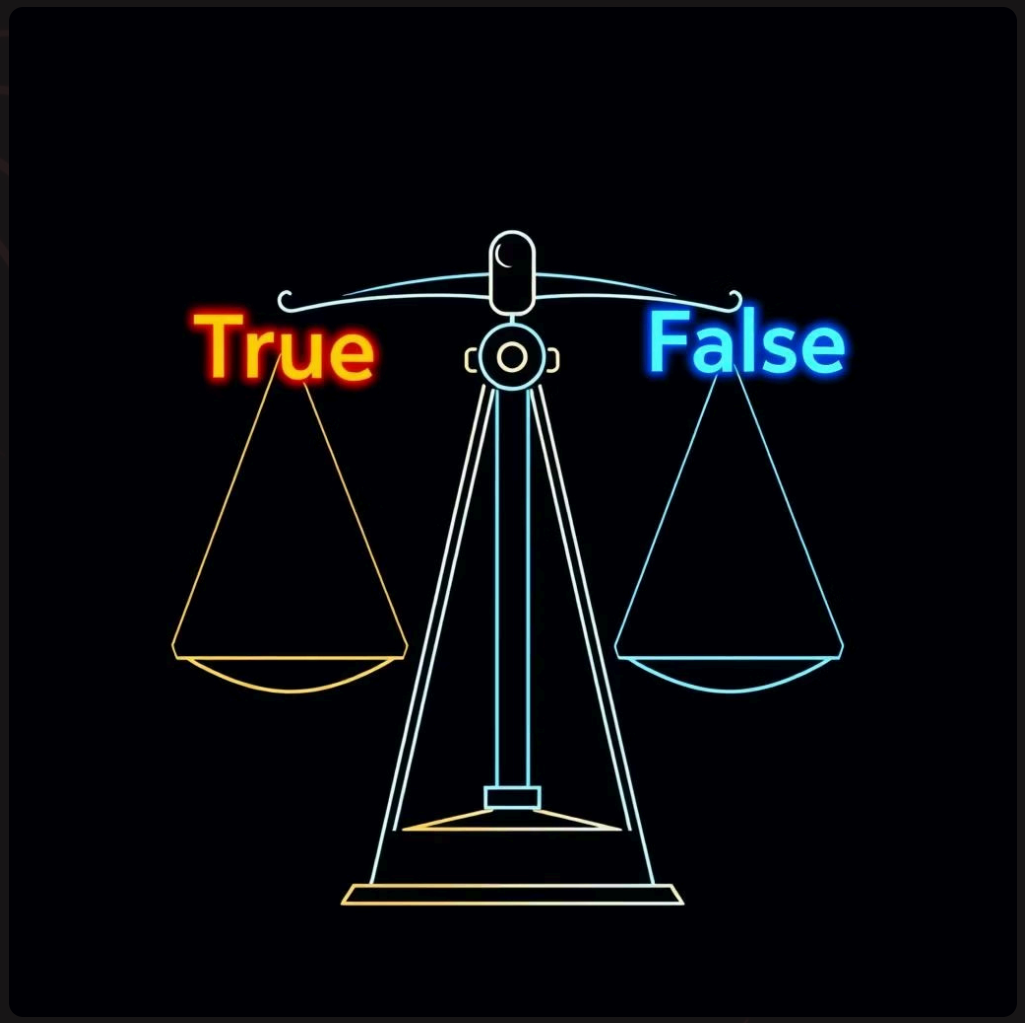
🔍 Operadores de Comparação: Fazendo Perguntas ao Python

```
idade = 20

print(idade == 18)  # False (É igual a?)
print(idade != 18)  # True (É diferente de?)
print(idade < 25)   # True (É menor que?)
print(idade > 15)   # True (É maior que?)
print(idade <= 20)  # True (É menor ou igual a?)
print(idade >= 18)  # True (É maior ou igual a?)
```

Estes operadores permitem que seu código "compare" valores, fazendo perguntas como "isso é igual a aquilo?" ou "isso é maior que aquilo?".

> **Resultado: Sempre True ou False!**
>
> Eles são a base para a tomada de decisões em seu programa, direcionando o fluxo da execução.



🧠 Operadores Lógicos: Conectando Ideias

```
maior_idade = True
tem_cnh = False

# AND: Ambos precisam ser verdadeiros
pode_dirigir = maior_idade and tem_cnh  # False

# OR: Pelo menos um precisa ser verdadeiro
pode_votar = maior_idade or tem_cnh     # True

# NOT: Inverte o resultado
nao_pode_dirigir = not pode_dirigir     # True
```

Operadores lógicos são como conjunções na gramática, permitindo que você combine múltiplas condições para formar lógicas mais complexas. Eles são a chave para decisões elaboradas em seus programas.

AND (E Lógico)

Ambas as condições devem ser **Verdadeiras**.

OR (OU Lógico)

Pelo menos uma condição deve ser **Verdadeira**.

NOT (NÃO Lógico)

Inverte o valor de verdade de uma condição.

⚡ Precedência de Operadores: A Ordem Importa!

Assim como na matemática, Python segue uma ordem para resolver operações. Conhecer a precedência evita surpresas e garante que seu código faça o que você espera.

> ⚠ **Dica Valiosa:**
>
> Use **parênteses** `()` para forçar a ordem de execução e deixar seu código mais claro, mesmo que a precedência já esteja a seu favor. Clareza é fundamental na programação!

```
resultado = (2 + 3) * 4  # Isso dá 20
                         # Sem parênteses daria 14!
```

Ordem de Execução (do mais prioritário ao menos):

1. ****** (Potenciação)
2. ***, /, //, %** (Multiplicação, Divisões, Resto)
3. **+, -** (Adição, Subtração)
4. **<, >, <=, >=, ==, !=** (Comparações)
5. **not** (Negação Lógica)
6. **and** (E Lógico)
7. **or** (OU Lógico)

💡 Exemplo Prático: **Par ou Ímpar**


```
numero = int(input("Digite um número: "))

resto = numero % 2
eh_par = resto == 0

print(f"Número: {numero}")
print(f"Resto ÷ 2: {resto}")
print(f"É par: {eh_par}")

# Versão mais direta:
if numero % 2 == 0:
 print("
```

Usando o operador **módulo (%)**, podemos facilmente determinar se um número é par ou ímpar. Se o resto da divisão por 2 for 0, é par!

⚠️ Fique de Olho: **Pegadinhas Comuns!**

Confundir `=` (Atribuição) com `==` (Comparação)

`if idade = 18:` ❌

`if idade == 18:` ✅

Divisão por Zero

`10 / 0` ❌

Sempre **valide** o divisor: `if divisor != 0:` ✅

Precedência sem Parênteses

`2 + 3 * 4` ❌ (resultado 14)

`(2 + 3) * 4` ✅ (resultado 20)

Pequenos detalhes fazem toda a diferença! Evitar essas armadilhas comuns tornará seu código mais robusto e previsível.

🎯 Recado Final: Seus Superpoderes Lógicos!

Operadores Aritméticos

Para fazer todos os seus cálculos e contas.

Operadores de Comparação

Para testar condições e tomar decisões simples.

Operadores Lógicos

Para combinar decisões e criar lógicas complexas.

Dominar esses três grupos de operadores é como ganhar **superpoderes** na programação. Eles são a base para construir programas que **calculam**, **decidem** e **interagem** de forma inteligente.

Vamos transformar matemática em lógica! 🚀