Fundamentos de Lógica de Programação

Linguagem: Python 3

1. Algoritmos: O Passo a Passo

Um algoritmo é uma **sequência lógica e finita** de instruções. No Python, a execução segue a ordem linear (de cima para baixo). Todo programa geralmente segue o seguinte fluxo básico:

- **Entrada (input)**: Coleta de dados fornecidos pelo usuário ou por um sistema.
- **Processamento**: Manipulação dos dados (cálculos, validações, filtragem).
- **Saída (print)**: Apresentação do resultado final para o usuário.

2. Variáveis e Tipos de Dados

No Python, a tipagem é dinâmica: a variável assume automaticamente o tipo do valor que recebe.

Tipo (Conceito)	Nome no Python	Exemplo Prático
Texto	`str`	`"Olá Mundo"`
Inteiro	`int`	`25`
Real (Decimal)	`float`	`1.75`
Booleano	`bool`	`True` ou `False`
Lista (Mutável)	`list`	`[1, 2, 3]`
Tupla (Imutável)	`tuple`	`(10, 20)`
Dicionário	`dict`	`{"id": 1, "nome": "Ana"}`

3. Operadores

➕ Aritméticos

`+` Soma
`-` Subtração
`*` Multiplicação
`/` Divisão
`//` Divisão Inteira
`%` Resto (Módulo)
`**` Potência

⚖️ Relacionais

`==` Igual
`!=` Diferente
`>` Maior
`<` Menor
`>=` Maior ou igual
`<=` Menor ou igual

📥 Atribuição

`=` Atribui valor simples.
`+=` Soma ao valor atual (ex: x += 1).
`-=` Subtrai do valor atual.

4. Funções Essenciais

Manipulação e Inspeção

- **print()**: Exibe dados na tela.
- **input()**: Recebe dados do teclado (retorna sempre como texto/str).
- **len()**: Retorna o tamanho/comprimento de um objeto (ex: caracteres de uma string).
- **sum()**: Soma elementos de uma coleção numérica.
- **type()**: Retorna o tipo da variável.

Conversão (Casting)

- **int()**: Converte para número inteiro.
- **float()**: Converte para número decimal.
- **str()**: Converte qualquer dado em texto.

5. Estruturas de Decisão

Estruturas que controlam o fluxo do programa baseadas em condições:

- **Simples**: Apenas um bloco if.
- **Composta**: Blocos if e else.
- **Aninhada**: Uso de elif para múltiplas verificações encadeadas.

```
:     nota = 7.5

      if nota >= 9:
          print("Excelente!")
      elif nota >= 7:
          print("Aprovado")
      else:
          print("Recuperação")
```

```
Aprovado
```

6. Tabela Verdade (Operadores Lógicos)

Operador and (E)

Resulta em **True** apenas se **TODOS** forem verdadeiros.

Condição A	Condição B	Resultado
True	True	**True**
True	False	False
False	True	False
False	False	False

Operador or (OU)

Resulta em **True** se **PELO MENOS UM** for verdadeiro.

Condição A	Condição B	Resultado
True	True	**True**
True	False	**True**
False	True	**True**
False	False	False

7. Estruturas de Repetição (Loops)

As estruturas de repetição permitem executar um bloco de código várias vezes. São fundamentais para evitar repetição de código manual e para processar coleções de dados.

Loop for

Utilizado quando **sabemos previamente quantas vezes** queremos repetir o código ou quando precisamos percorrer uma coleção de itens (como uma lista ou os caracteres de um texto).

```
:     # Repete 3 vezes (os valores de i serão 0, 1 e 2)
      for i in range(3):
          print(f"Repetição número {i}")
```

```
Repetição número 0
Repetição número 1
Repetição número 2
```

Loop while (Enquanto)

Utilizado quando **não sabemos o número exato** de repetições, mas temos uma **condição** que deve permanecer verdadeira para que o loop continue rodando.

```
:     contador = 0

      while contador < 2:
          print(f"Contador está em: {contador}")
          contador += 1  # Importante para evitar um loop infinito!
```

```
Contador está em: 0
Contador está em: 1
```

E o do... while?

Diferente de linguagens como C, Java ou JavaScript, **o Python não possui uma estrutura do while nativa**. No `do while`, o código é executado *pelo menos uma vez* antes de verificar a condição. Em Python, simulamos esse comportamento criando um loop infinito (`while True:`) e usando o comando `break` para interrompê-lo por dentro.

```
:     while True:
          # Este bloco vai executar obrigatoriamente a primeira vez
          senha = input("Digite a senha: ")

          if senha == "123":
              print("Acesso liberado!")
              break  # Quebra o loop e sai da repetição
          else:
              print("Senha incorreta, tente novamente.")
```

```
Digite a senha: abc
Senha incorreta, tente novamente.
Digite a senha: 123
Acesso liberado!
```