IEMA PLENO RIO ANIL
CURSO: Desenvolvimento de Sistemas
PROF(A).: Arthur Silva **DISCIPLINA:** PROGRAMAÇÃO ORIENTADA A OBJETOS
ALUNO(A): _____

Tutorial de Python
Este tutorial cobre os conceitos básicos da linguagem Python, incluindo variáveis, estruturas condicionais, loops, funções e muito mais.

1. Variáveis

Em Python, variáveis são usadas para armazenar dados. Não é necessário declarar o tipo de uma variável explicitamente; o Python infere o tipo com base no valor atribuído.

Exemplo:

```
# Atribuição de valores a variáveis
nome = "João"  # String/texto
idade = 25      # Inteiro
altura = 1.75   # Float
is_estudante = True  # Booleano (verdadeiro ou falso)

# Exibindo valores
print(nome)
print(idade)
print(altura)
print(is_estudante)
```

Tipos de Dados Comuns:

- Inteiro (`int`): Números inteiros, como `10`, `-5`, `0`.
- Float (`float`): Números decimais, como `3.14`, `-0.001`.
- String (`str`): Texto, como `"Olá, mundo!"`.
- Booleano (`bool`): `True` ou `False`.

2. Estruturas Condicionais: `if`, `elif`, `else`

Estruturas condicionais permitem que você execute diferentes blocos de código com base em condições.

Exemplo:

```
idade = 18

if idade < 18:
    print("Menor de idade")
elif idade == 18:
    print("Tem exatamente 18 anos")
else:
    print("Maior de idade")
```

Regras:

- `if`: Verifica a primeira condição.
- `elif`: Verifica condições adicionais se a condição do `if` for falsa.
- `else`: Executa se todas as condições anteriores forem falsas.

3. Loop `for`

O loop `for` é usado para iterar sobre uma sequência (como uma lista, string, etc.).

Exemplo:

```
# Iterando sobre uma lista
frutas = ["maçã", "banana", "laranja"]

for fruta in frutas:
        print(fruta)

# Iterando sobre uma string
for letra in "Python":
        print(letra)
```

Função `range()`:

A função `range()` gera uma sequência de números, útil para loops.

```
for i in range(5):  # 0, 1, 2, 3, 4
    print(i)
```

4. Loop `while`

O loop `while` executa um bloco de código enquanto uma condição for verdadeira.

Exemplo:

```
contador = 0

while contador < 5:
    print(contador)
    contador += 1  # Incrementa o contador
```

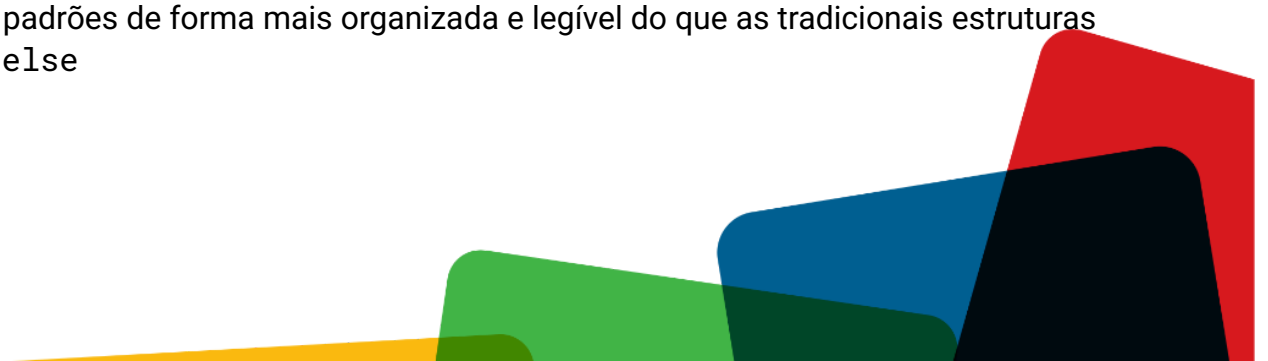
Cuidado com loops infinitos:

Certifique-se de que a condição do `while` eventualmente se torne falsa, caso contrário, o loop nunca terminará.

5. Estrutura `match` (Python 3.10+)

é uma estrutura de controle de fluxo que permite comparar uma variável com diferentes valores ou padrões de forma mais organizada e legível do que as tradicionais estruturas `if/elif/else`

Exemplo:



```
dia = "segunda"

match dia:
    case "segunda":
        print("Início da semana")
    case "sexta":
        print("Final da semana")
    case _:
        print("Dia não reconhecido")
```

Regras:

- `case`: Define os casos a serem verificados.
- `_`: Caso padrão (no caso de nenhuma das outras alternativas baterem).

6. Funções

Funções são blocos de código reutilizáveis que realizam uma tarefa específica. Em Python, funções são definidas usando a palavra-chave `def`.

Exemplo:

```
# Definindo uma função
def saudacao(nome):
    return f"Olá, {nome}!"

# Chamando a função
mensagem = saudacao("Maria")
print(mensagem)  # Olá, Maria!
```

Parâmetros e Argumentos:

- Parâmetros: Variáveis listadas na definição da função.

- Argumentos: Valores passados para a função quando ela é chamada.

Funções com Parâmetros Padrão:

```
def saudacao(nome="Visitante"):
        return f"Olá, {nome}!"

print(saudacao())  # Usa o valor padrão
print(saudacao("João"))  # Passa um argumento
```

Retorno de Múltiplos Valores:

```
def operacoes(a, b):
        soma = a + b
        diferenca = a - b
        return soma, diferenca

resultado = operacoes(10, 5)
print(resultado)  # (15, 5)
```

7. Introdução a Classes e Objetos

Python é uma linguagem de programação orientada a objetos (OOP). Classes são modelos para criar objetos, que são instâncias dessas classes.

Definindo uma Classe:

```
class Pessoa:
    # Método construtor
    def __init__(self, nome, idade):
        self.nome = nome
        self.idade = idade

    # Método da classe
    def apresentar(self):
        return f"Olá, meu nome é {self.nome} e tenho {self.idade} anos."
```

Criando Objetos:

```
# Criando uma instância da classe Pessoa
pessoa1 = Pessoa("João", 25)

# Acessando atributos e métodos
print(pessoa1.nome)  # João
print(pessoa1.apresentar())  # Olá, meu nome é João e tenho 25 anos.
```

Atributos e Métodos:

- **Atributos**: Variáveis que pertencem a um objeto (ex: `nome`, `idade`).
- **Métodos**: Funções que pertencem a um objeto (ex: `apresentar()`).