Documento de Acompanhamento: Condicionais em Python

1. Introdução às Estruturas Condicionais

Explicação Textual:
As estruturas condicionais são fundamentais na programação, pois permitem que nossos programas tomem decisões. Imagine que você está criando um programa e precisa que ele se comporte de maneira diferente dependendo de certas situações. É exatamente para isso que servem os condicionais! Eles avaliam se uma condição é verdadeira ou falsa e, com base nisso, decidem qual bloco de código executar.

Por que usar condicionais?

- Tomada de decisão: O programa pode escolher entre diferentes caminhos
- Validação: Verificar se dados atendem a certos critérios
- Personalização: Adaptar o comportamento do programa ao contexto

2. A Importância da Identação em Python

Explicação Textual:
Python é uma linguagem que usa identação (espaços no início da linha) para definir blocos de código. Diferente de outras linguagens que usam chaves `{}`, em Python usamos espaços para mostrar quais comandos pertencem a qual bloco.

Regras importantes:

- Use 4 espaços por nível de identação
- NUNCA misture tabs e espaços
- A identação deve ser consistente em todo o código

Exemplo da importância:

```python
```

```
# CORRETO - com identação adequada
idade = 18
if idade >= 18:
    print("Maior de idade")    # 4 espaços
    print("Pode votar")        # 4 espaços - mesmo bloco

# ERRADO - sem identação
if idade >= 18:
print("Maior de idade")    # ERRO! Falta identação

# ERRADO - identação inconsistente
if idade >= 18:
    print("Primeira mensagem")  # 4 espaços
  print("Segunda mensagem")    # 2 espaços - ERRO!
```

3. Estruturas Básicas dos Condicionais

if (se)

Explicação Textual:
O `if` é a estrutura condicional mais básica. Ele verifica se uma condição é verdadeira. Se for, executa o bloco de código identado abaixo dele.

```
python
idade = 18

if idade >= 18:
    print("Você é maior de idade")  # Este código só executa se a condição for True
```

if-else (se-senão)

Explicação Textual:
O `if-else` nos permite definir dois caminhos: um para quando a condição é verdadeira e outro para quando é falsa. É como uma bifurcação na estrada - você vai por um caminho ou por outro.

```
python
idade = 16
```

```
if idade >= 18:
    print("Você é maior de idade")
else:
    print("Você é menor de idade")  # Este código executa se a condição for False
```

if-elif-else (se-senão se-senão)

Explicação Textual:
Quando temos múltiplas condições para verificar, usamos `if-elif-else`. O programa testa cada condição em ordem e executa apenas o bloco da primeira condição que for verdadeira.

```
python
nota = 85

if nota >= 90:
    print("Conceito A")
elif nota >= 80:  # Só testa esta condição se a primeira for falsa
    print("Conceito B")
elif nota >= 70:  # Só testa esta condição se as anteriores forem falsas
    print("Conceito C")
else:             # Executa apenas se todas as condições acima forem falsas
    print("Conceito D")
```

4. Operadores Disponíveis em Python

Operadores Aritméticos (já vistos)

Explicação Textual:
Estes operadores realizam cálculos matemáticos básicos.

```
python
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
print(a // b)  # 3 - Divisão inteira
print(a % b)   # 1 - Resto da divisão
print(a ** b)  # 1000 - Potência
```

Operadores de Comparação

Explicação Textual:
Estes operadores comparam dois valores e retornam `True` (Verdadeiro) ou `False` (Falso). São essenciais para as condições dos nossos `if`.

```
python
x = 10
y = 5

print(x == y)  # False - Igual a
print(x != y)  # True  - Diferente de
print(x > y)   # True  - Maior que
print(x < y)   # False - Menor que
print(x >= y)  # True  - Maior ou igual a
print(x <= y)  # False - Menor ou igual a
```

Operadores Lógicos

Explicação Textual:
Estes operadores combinam várias condições. São muito úteis quando precisamos verificar mais de uma coisa ao mesmo tempo.

```
python
idade = 25
tem_carteira = True
tem_carro = False

# AND - ambas condições devem ser True
if idade >= 18 and tem_carteira:
    print("Pode dirigir legalmente")

# OR - pelo menos uma condição deve ser True
if tem_carteira or tem_carro:
    print("Tem algo relacionado a veículos")
```

```
# NOT - inverte o valor (True vira False, False vira True)
if not tem_carro:
    print("Não tem carro")
```

5. Exemplos Práticos Detalhados

Exemplo 1: Verificador de Número

Explicação Textual:
Vamos criar um programa que analisa um número digitado pelo usuário e nos diz se é positivo/negativo e par/ímpar.

```
python
# Pedimos um número ao usuário
numero = int(input("Digite um número: "))

# Verificamos se é positivo, negativo ou zero
if numero > 0:
    print("Número positivo")
elif numero < 0:
    print("Número negativo")
else:
    print("Número zero")

# Verificamos se é par ou ímpar
# % é o operador resto da divisão
# Se o resto da divisão por 2 for 0, o número é par
if numero % 2 == 0:
    print("Número par")
else:
    print("Número ímpar")
```

Exemplo 2: Sistema de Login Simples

Explicação Textual:
Vamos simular um sistema de login bem básico para entender como os condicionais podem verificar múltiplas condições.

```
python
```

```python
# Definimos usuário e senha corretos
usuario_correto = "aluno"
senha_correta = "python123"

# Pedimos as credenciais ao usuário
usuario = input("Digite o usuário: ")
senha = input("Digite a senha: ")

# Verificamos se ambas estão corretas
if usuario == usuario_correto and senha == senha_correta:
    print("Login bem-sucedido!")
    print("Bem-vindo ao sistema!")
else:
    print("Usuário ou senha incorretos")
```

Exemplo 3: Verificador de Aprovação

Explicação Textual:
Vamos criar um sistema que verifica se um aluno foi aprovado baseado em sua nota e frequência.

```
python
# Coletamos os dados do aluno
nota = float(input("Digite a nota do aluno (0-10): "))
frequencia = float(input("Digite a frequência do aluno (0-100): "))

# Verificamos as condições para aprovação
if nota >= 7 and frequencia >= 75:
    print("Aluno APROVADO!")
elif nota >= 5 and frequencia >= 75:
    print("Aluno em RECUPERAÇÃO")
else:
    print("Aluno REPROVADO")
```

6. Exercícios Práticos

Exercício 1: Classificador de Idade

```
python
```

```
"""
Crie um programa que classifique a idade em categorias:
- 0-12: Criança
- 13-17: Adolescente
- 18-59: Adulto
- 60+: Idoso

Dica: Use input() para receber a idade e if-elif-else para classificar
"""
```

Exercício 2: Calculadora Simples

```
python
"""
Crie uma calculadora que:
1. Peça dois números ao usuário
2. Peça a operação (+, -, *, /)
3. Mostre o resultado
4. Se for divisão por zero, mostre "Erro: divisão por zero"

Dica: Use condições para verificar a operação escolhida
"""
```

Exercício 3: Verificador de Triângulo

```
python
"""
Peça 3 lados de um triângulo e determine se:
- É equilátero (todos os lados iguais)
- É isósceles (dois lados iguais)
- É escaleno (todos os lados diferentes)

Dica: Use == para verificar igualdade entre os lados
"""
```

Exercício 4: Sistema de Desconto

```
python
"""
```

```
Uma loja oferece descontos baseado no valor da compra:
- Até R$ 100: sem desconto
- R$ 100-200: 10% de desconto
- Acima de R$ 200: 20% de desconto

Calcule e mostre o valor final com desconto.
"""
```

Exercício 5: Verificador de Números

```
python
"""
Peça um número ao usuário e verifique se:
- É positivo, negativo ou zero
- É par ou ímpar
- É múltiplo de 5

Dica: Use % para verificar múltiplos (número % 5 == 0)
"""
```

7. Dicas Finais e Boas Práticas

1. Teste sempre seus condicionais com diferentes valores para garantir que todos os caminhos funcionam
2. Use nomes descritivos para variáveis booleanas: `é_maior_idade` em vez de `x`
3. Mantenha a identação consistente - sempre 4 espaços por nível
4. Comente seu código para explicar condições complexas
5. Comece com condições simples e vá aumentando a complexidade gradualmente

8. Erros Comuns de Iniciantes

```python
# ERRO: Esquecer os dois pontos
if idade >= 18   # FALTAM OS : (dois pontos)

# ERRO: Usar = em vez de == para comparação
```

```python
if idade = 18:   # ERRADO! = é para atribuição

# ERRO: Identação inconsistente
if idade >= 18:
    print("Primeira linha")
  print("Segunda linha")   # ERRO de identação!

# ERRO: Esquecer que input() retorna string
idade = input("Digite sua idade: ")
if idade >= 18:   # ERRO! Não pode comparar string com número
```

Lembrete: A prática é essencial! Resolva os exercícios e não tenha medo de errar - cada erro é uma oportunidade de aprendizado.