Linguagem de Programação Python

Comandos Condicionais

Professor: Ritomar Torquato

1

Condicionais

Objetivos: Identificar a necessidade de uso de comandos condicionais simples, compostas e com múltiplas escolhas e a importância dos operadores lógicos na tomada de decisão.

2

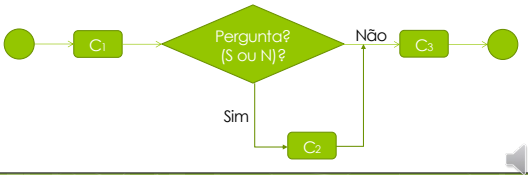
Estrutura Sequencial

- Até o momento, todos os programas foram sequenciais:

3

Estrutura Condicional

- Nem sempre nosso algoritmo é tão linear;
- Muitas vezes, precisamos tomar decisões



4

Condicional Simples

- Um bloco de comandos que é executado se a condição for verdadeira.

```
if < condição >:
    < bloco verdadeiro >
```

5

Condicional Simples

- Um bloco de comandos que é executado se a condição for verdadeira.

```
a = int(input("Primeiro valor: "))
b = int(input("Segundo valor: "))

if a > b:
    print ("O primeiro número é o maior!")

if b > a:
    print ("O segundo número é o maior!")
```

6

Condicional Simples

- Um bloco de comandos que é executado se a condição for verdadeira.

```
idade = int(input("Digite a idade do seu carro: "))

if idade <= 3:
    print("Seu carro é novo")

if idade > 3:
    print("Seu carro é velho")
```

7

Condicional Composta

- Um bloco de comandos que é executado se a condição for verdadeira; outro bloco é executado se a condição for falsa.

```
if < condição >:
    < bloco verdadeiro >
else:
    < bloco falso >
```

8

Condicional Composta

```
idade = int(input("Idade de seu carro: "))

if idade <= 3:
    print("Seu carro é novo")
else:
    print("Seu carro é velho")
```

Apenas um dos blocos é executado.

9

Comando `if/elif/else`

```
x = int(input("Entre com um número inteiro: "))
if x < 0:
    x = 0
    print('Numero negativo alterado para Zero')
elif x == 0:
    print('Zero')
elif x == 1:
    print('Um')
else:
    print('Maior que Um')
```

Não existe comando condicional de múltipla escolha em Python.

10

Caixa de ferramentas

```
# No mundo ideal...

# Se você ler um inteiro dessa forma...
idade = int(input("Qual sua idade? : "))

# O usuário entraria com um valor booleano assim...
tem_filhos = bool(input('Tem filhos? : '))

# Digitando exatamente True ou False

# Isso não acontece. Temos que interpretar uma resposta string.
tem_filhos = input('Tem filhos? : ')

# O problema é que o usuário pode dar várias respostas diferentes.
# Sim, SIM, sim, s, Tenho, não, N, S...

# Para facilitar nossa vida, temos que controlar a resposta do usuário
# indicando mais claramente como ele deve responder. Por exemplo...
tem_filhos = input('Tem filhos? (S - Sim ou N - Não): ')
tem_filhos = input('Tem filhos? (1 - Sim ou 2 - Não): ')
```

11

Caixa de ferramentas

```
>>> tem_filhos = input('Tem filhos? (S - Sim ou N - Não): ').upper()
Tem filhos? (S - Sim ou N - Não): Sim
>>> tem_filhos
'SIM'
>>> tem_filhos = input('Tem filhos? (S - Sim ou N - Não): ').lower()
Tem filhos? (S - Sim ou N - Não): Não
>>> tem_filhos
'não'
>>> tem_filhos.upper()
'NÃO'
>>> 'Em uma string qualquer...'.upper()
'EM UMA STRING QUALQUER...'
>>> tem_filhos[0]
'n'
>>> tem_filhos[0].upper()
'N'
>>>
```

O usuário ainda pode digitar maiúsculo (S, N) ou minúscula (s, n), "Sim" ou "Não"... Precisamos diminuir as chances de erro.

12

Caixa de ferramentas

```
# Fazendo dessa forma você aumenta muito a chance de acerto.
# O código vai funcionar mesmo o usuário digitando
# Sim, SIM, não, N, S, NAO..

tem_filhos = input('Tem filhos? (S - Sim ou N - Não): ')

if tem_filhos[0].upper() == 'S':
    print('Parabéns!')
else:
    print('É bom ter filhos.')
```

13

Caixa de ferramentas

```
# Ao invés de...
def e_par(n):
    if n % 2 == 0:
        return 'Sim'
    else:
        return 'Não'
```

```
# ou de...
def e_par(n):
    if n % 2 == 0:
        return True
    else:
        return False
```

```
# Prefira fazer...
def e_par(n):
    return n % 2 == 0
```

14

Caixa de ferramentas

```
# Ao invés de...
if e_par(n) == True:
    print('Par')
```

```
# Prefira fazer...
if e_par(n):
    print('Par')
```

15

Caixa de ferramentas

```
# Ao invés de...
if a == 5:
    if b == 10:
        print('a = 5 e b = 10')
```

```
# Prefira fazer...
if a == 5 and b == 10:
    print('a = 5 e b = 10')
```

Não tenha medo de usar **and**, **or** e **not**.

16

Caixa de ferramentas

```
# Ao invés de...
if a == 5:
    print('a = 5 ou b = 10')
else:
    if b == 10:
        print('a = 5 ou b = 10')
```

```
# Prefira fazer...
if a == 5 or b == 10:
    print('a = 5 ou b = 10')
```

Não tenha medo de usar **and**, **or** e **not**.

17

Caixa de ferramentas

Lembre-se que letras também são números para o computador.

```
letra = input('Digite uma letra: ')

if 'A' <= letra <= 'Z':
    print('Você digitou uma letra maiúscula.')
else:
    if 'a' <= letra <= 'z':
        print('Você digitou uma letra minúscula.')
    else:
        print('Você não digitou uma letra.')
```

18

Caixa de ferramentas

Operador **in**. Verifica se um valor está em um conjunto.

```
fruta = input('Qual sua fruta favorida? : ').upper()

if fruta in ('LARANJA', 'UVA', 'MORANGO'):
    print('Nosso gosto combina.')
else:
    print('Eu prefiro laranja.')
```

19

Caixa de ferramentas

Operador **in**. Verifica se um valor está em um conjunto.
O conjunto pode ser uma string.

```
nome = input('Qual seu nome? : ')

if 'Maria' in nome:
    print('Você tem Maria no nome')
else:
    print('Você NÃO tem Maria no nome')

if 'K' in nome.upper():
    print('Você tem a letra K no nome')
else:
    print('Você NÃO tem a letra K no nome')
```

20

Caixa de ferramentas

Operador **in**. Verifica se um valor está em um conjunto.
O conjunto pode ser uma string.

```
letra = input('Digite uma letra: ')

if letra in 'AaBbCc':
    print('A letra é A, B ou C.')
else:
    print('A letra NÃO é A, B ou C.')
```

21

Exemplo 1 de 2

- Crie um função que recebe a sigla de um estado e retorna uma das mensagens: CEARENSE, PAULISTA, MINEIRO ou OUTROS ESTADOS.

22

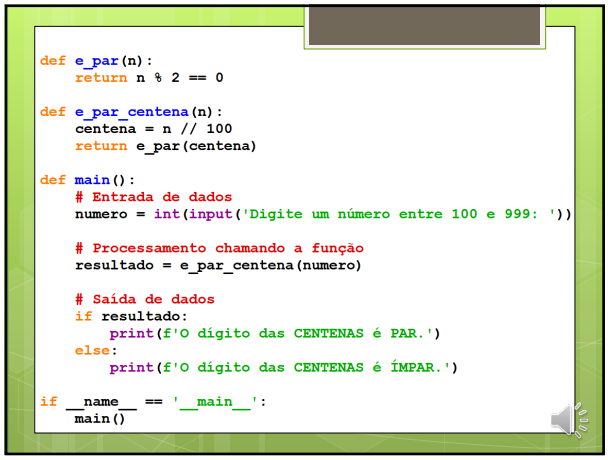
```
def naturalidade(sigla_estado):
    if sigla_estado.upper() == 'CE':
        return 'CEARENCE'
    elif sigla_estado.upper() == 'SP':
        return 'PAULISTA'
    elif sigla_estado.upper() == 'MG':
        return 'MINEIRO'
    else:
        return 'OUTROS ESTADOS'

def main():
    # Entrada de dados
    sigla = input('Digite a sigla do seu estado: ')

    # Processamento chamando a função
    resultado = naturalidade(sigla)

    # Saída de dados
    print(f'Você é {resultado}')

if __name__ == '__main__':
    main()
```

23

Exemplo 2 de 2

- Crie uma função que recebe um número inteiro de 3 dígitos e retorna o valor booleano True (verdadeiro) se a centena do número for par ou o valor booleano False (falso) caso contrário.
  - 100 ==> False
  - 200 ==> True

24



```python
def e_par(n):
    return n % 2 == 0

def e_par_centena(n):
    centena = n // 100
    return e_par(centena)

def main():
    # Entrada de dados
    numero = int(input('Digite um número entre 100 e 999: '))

    # Processamento chamando a função
    resultado = e_par_centena(numero)

    # Saída de dados
    if resultado:
        print(f'O dígito das CENTENAS é PAR.')
    else:
        print(f'O dígito das CENTENAS é ÍMPAR.')

if __name__ == '__main__':
    main()
```

25