



Introdução à Programação

Aula 05: Desvio condicional com as Instruções if-else-elif

Prof. Eduardo Corrêa

Data 21/03/2024

Introdução (1/4)

- Até agora, nossos programas seguiram um mesmo padrão:
- Começam na **primeira instrução**.
- Executam **todas** as instruções ordenadamente até a **última**.
- **Ex.:** prog. que recebe 2 notas de um aluno e calcula e exibe a média.

```
print('Digite as suas 2 notas: ')\nnota1 = float(input())\nnota2 = float(input())\nmedia = (nota1 + nota2) / 2\nprint('Sua média é: ', media)
```

Introdução (2/4)

- E se fosse necessário determinar se aluno foi aprovado?
- Neste caso, é preciso utilizar as **estruturas de seleção** (ou **desvio condicional**) do Python: **if** e **else**.
- São usadas quando um programa precisar agir de forma diferente em situações diferentes.

```
print('Digite as suas 2 notas: ')
nota1 = float(input())
nota2 = float(input())
media = (nota1 + nota2) / 2
print('Sua média é: ', media)

if media >= 7.0:
    print('APROVADO!')
else:
    print('NÃO FOI APROVADO!')
```

Introdução (3/4)

- Se (**if**) a média for maior ou igual a 7.0, então a mensagem 'APROVADO' é exibida na tela.
- Senão (**else**), a mensagem 'NÃO FOI APROVADO' é exibida na tela.

```
print('Digite as suas 2 notas: ')\nnota1 = float(input())\nnota2 = float(input())\nmedia = (nota1 + nota2) / 2\nprint('Sua média é: ', media)
```

```
if media >= 7.0:\n    print('APROVADO!')\nelse:\n    print('NÃO FOI APROVADO!')
```

Introdução (4/4)

- Uma instrução de **desvio** toma uma decisão a partir da avaliação de uma **condição**.
- Uma condição representa uma comparação ou conjunto de comparações que irá resultar sempre em VERDADEIRO (**True**) ou FALSO (**False**).
- Como visto na última aula, podemos montar uma condição simples utilizando **operadores relacionais**:
 - **`==, !=, >, >=, <, <=`**
- E combinar diversas condições simples utilizando os **operadores lógicos**:
 - **`and, or`**

Blocos de Código (1/4)

- Toda instrução **if** e toda instrução **else** possuem um **bloco de código** associados.
 - Bloco do **if**: uma ou mais instruções que deverão ser executadas se a condição resultar em **True**.
 - Bloco do **else**: uma ou mais instruções que deverão ser executadas se a condição resultar em **False**.

```
print('Digite as suas 2 notas: ')\nnota1 = float(input())\nnota2 = float(input())\nmedia = (nota1 + nota2) / 2\nprint('Sua média é: ', media)
```

```
if media >= 7.0:
```

```
    print('APROVADO!')
```

```
else:
```

```
    print('NÃO FOI APROVADO!')
```

bloco do **if**

bloco do **else**

Blocos de Código (2/4)

- Em outras palavras
 - Bloco do **if**: instruções que são subordinadas ("obedecem") ao **if**.
 - Bloco do **else**: instruções que são subordinadas ("obedecem") ao **else**.

```
print('Digite as suas 2 notas: ')\nnota1 = float(input())\nnota2 = float(input())\nmedia = (nota1 + nota2) / 2\nprint('Sua média é: ', media)
```

```
if media >= 7.0:
```

```
    print('APROVADO!')
```


```
else:
```

```
    print('NÃO FOI APROVADO!')
```

bloco do **if**

bloco do **else**

Blocos de Código (3/4)

- **Aviso:** agora preste atenção, **muita atenção!!!** 
- Enquanto linguagens como C, Java e R utilizam os símbolos "{" e "}" para definir blocos de código, o Python utiliza **espaços em branco** ou **tabulações**.
- Isto significa que os comandos que formam um bloco precisam estar indentados (alinhados) da mesma forma para que o Python os reconheça como parte de um mesmo bloco.
 - A convenção é utilizar 4 espaços para indentação.
 - Mas é permitido usar outro contanto que este seja repetido para todos os comandos de um mesmo bloco.

Blocos de Código (4/4)

- **Veja o exemplo abaixo:**

Erro de indentação:	Indentação correta:
<pre>if (x > 0): a = 1</pre> <p><i>IndentationError: expected an indented block</i></p>	<pre>if (x > 0): a = 1</pre>

- Se a indentação não for feita, não tem como o Python saber qual é exatamente o bloco de código que deve obedecer a um comando de desvio ou repetição.

As instruções if e else (1/10)

- Agora vamos retornar para as instruções **if** e **else**.
 - Elas permitem com que o programa execute **desvios** em diferentes direções, de acordo com o resultado da **avaliação de uma condição**.
 - Ou seja, na prática, é nestas instruções que você montará as suas condições.
 - Por esta razão **if** e **else** são também chamadas de instruções de **desvio condicional**.
 - Os slides a seguir exemplificam diferentes formas de utilização.

As instruções if e else (2/10)

- **Categoria 1: desvio simples com if.**
- Programa executa bloco de código se a condição especificada no if for verdadeira.

```
print('Digite a sua Idade: ')\ni = int(input())\nif (i >= 18):\n    print(' Você não é menor de idade')\nprint('FIM!')
```

As instruções if e else (3/10)

- **Agora muita ATENÇÃO** – observe que:
 1. O comando **if termina com um sinal de dois pontos ":"**
 2. A instrução a ele subordinada está **indentada** (começa com 4 espaços em branco).
- Veja também que o programa executará a instrução **print('FIM!')**, independente do resultado do **if**.
 - Isso porque ela não está subordinada ao **if** (está no mesmo nível de indentação do **if**).

```
print('Digite a sua Idade: ')\ni = int(input())\nif (i >= 18):\n    print(' Você não é menor de idade')\nprint('FIM!')
```

As instruções if e else (4/10)

- Recomendação: teste o programa digitando diferentes entradas p/ que você entenda bem seu funcionamento.

```
1 print('Digite a sua Idade: ')
2 i = int(input())
3 if (i >= 18):
4     print(' Você não é menor de idade')
5 print('FIM!')
```

Shell ×

Python 3.10.4 (C:\Python310\python.exe)

>>> %Run if_sozinho.py

Digite a sua Idade:

21

Você não é menor de idade

FIM!

```
1 print('Digite a sua Idade: ')
2 i = int(input())
3 if (i >= 18):
4     print(' Você não é menor de idade')
5 print('FIM!')
```

Shell ×

>>> %Run if_sozinho.py

Digite a sua Idade:

17

FIM!

As instruções if e else (5/10)

- Se desejarmos mais de uma instrução subordinada ao if, precisamos coloca-las **todas indentadas**.
- No exemplo abaixo, temos 4 instruções subordinadas ao if (3 prints e 1 input).

```
print('Digite a sua Idade: ')\ni = int(input())\nif (i >= 18):\n    print('Você não é menor de idade')\n    print('Você já pode dirigir!')\n    print('Quer comprar um carro? (S=SIM, N=NÃO) ')\n    resposta = input()\nprint('FIM!')
```

As instruções if e else (6/10)

- Porém, existe **uma exceção**:
 - Quando o **if (ou else) possui apenas uma instrução subordinada**, esta pode ser indicada na mesma linha, depois do dois-pontos.
 - Sendo assim, a sintaxe do programa abaixo é válida.

```
print('Digite a sua Idade: ')\ni = int(input())\nif (i >= 18): print('Você não é menor de idade')\nprint('FIM!')
```

As instruções if e else (7/10)

- **Categoria 2: desvio if com else.**
- Em muitas situações práticas é preciso realizar **dois desvios**:
 - Um que seja executado se o resultado do teste lógico resultar em **True**; E outro se for **False**.
 - Nesse caso, você deve usar **if** com **else**:
 - Se a avaliação for **verdadeira**, será executado o bloco de código que acompanha a instrução **if**.
 - Se a avaliação for **falsa**, será executado o bloco de código que acompanha a instrução **else**.

As instruções if e else (8/10)

- **Exemplo:**
 - Se idade igual ou maior que 18, escreve "você não é menor de idade".
 - Senão, escreve "você é menor de idade".
- A mensagem "Viu como eu conheço a lei?" é escrita sempre.
 - Não faz parte do bloco do if nem do bloco do else.

```
print('Digite a sua Idade: ')\ni = int(input())\nif (i >= 18):\n    print('Você não é menor de idade')\nelse:\n    print('Você é menor de idade')\n\nprint('viu como eu conheço a lei?')
```

As instruções if e else (9/10)

- **** IMPORTANTE **** : Uma **regra sintática** do Python é que é preciso colocar o símbolo **dois-pontos** no final da linha do if e no final da linha do else.

```
print('Digite a sua Idade: ')\ni = int(input())\nif (i >= 18):\n    print('Você não é menor de idade')\nelse:\n    print('Você é menor de idade')\nprint('viu como eu conheço a lei?')
```

Dois-pontos

As instruções if e else (10/10)

- No exemplo a seguir, veja que tanto o **if** quanto o **else** possuem mais de um comando subordinado.

```
print('Digite a sua Idade: ')
i = int(input())
if (i >= 18):
    print('Você não é menor de idade')
    print('Você pode dirigir!')
else:
    print('Você é menor de idade')
    print('Você não pode dirigir')

print('viu como eu conheço a lei?')
```

```
>>> %Run if_com_else.py
Digite a sua Idade:
21
Você não é menor de idade
Você pode dirigir!
viu como eu conheço a lei?
```

```
>>> %Run if_com_else.py
Digite a sua Idade:
13
Você é menor de idade
Você não pode dirigir
viu como eu conheço a lei?
```

O par if e else

- Conforme visto, pode existir um **if** sem **else**.
- Situação em que o programa deve fazer alguma coisa se um teste der verdadeiro e **nada em especial** se der falso.
- Porém, **não existe else sem if !!!!**
 - Todo **else** está vinculado a alguma instrução **if** que veio antes dele.
 - Ou seja, todo **else** faz par com um **if** específico.
 - O **else** é usado em situações onde o programa deve fazer alguma coisa se o teste der verdadeiro e **outra coisa** se o teste der false.

Comparação com Pseudocódigo

- Uma comparação entre pseudocódigo (Aula 01) e Python.

<i>Pseudocódigo</i>	<i>Python</i>
se A > B então: imprimir ('A é maior') C = A ler (D) senão: imprimir ('A não é maior') C = B	if (A > B): print ('A é maior') C = A D = float (input (D)) else: print ('A não é maior') C = B

ifs aninhados (1/3)

- Alguns problemas requerem a utilização de **ifs aninhados** para serem resolvidos.
- Isso significa: “if dentro de if” ou “if dentro de else”.
- Não há qualquer tipo de restrição em fazer isso.
 - Você pode fazer if dentro de if dentro de if ... (quantos quiser)
 - Na verdade, ifs aninhados ocorrem comumente em programas.
- **Um exemplo é apresentado no slide a seguir**

ifs aninhados (2/3)

- **Exemplo:** ler 2 notas, calcular e exibir a média. Se a média for maior ou igual a 7, imprimir "aprovado". Se for menor do que 7 e maior ou igual a 3 imprimir "prova final". Se for menor do que 3 imprimir "Reprovado".

```
nota1 = float(input('Digite a nota1: '))
nota2 = float(input('Digite a nota2: '))

media = (nota1 + nota2) / 2
if media >= 7:
    print('Aprovado')
else:
    if media < 3:
        print('Reprovado')
    else:
        print('prova final')
```

ifs aninhados (3/3)

```
nota1 = float(input('Digite a nota1: '))
nota2 = float(input('Digite a nota2: '))

media = (nota1 + nota2) / 2
if media >= 7:
    print('Aprovado')
else:
    if media < 3:
        print('Reprovado')
    else:
        print('prova final')
```

```
>>> %Run ifs_aninhados.py
```

```
Digite a nota1: 7.5
Digite a nota2: 8
Aprovado
```

```
>>> %Run ifs_aninhados.py
```

```
Digite a nota1: 0.5
Digite a nota2: 2
Reprovado
```

```
>>> %Run ifs_aninhados.py
```

```
Digite a nota1: 6.5
Digite a nota2: 6.5
prova final
```


Instrução elif (1/7)

- Entretanto, há situações práticas em que existe a necessidade de avaliar **muitas possibilidades**.
- Neste caso, utilizando apenas **if** e **else**, o programa acaba ficando um **excessivamente indentado**
- Veja o exemplo no slide a seguir, onde 4 diferentes condições precisam ser avaliadas.

Instrução elif (2/7)

- **Exemplo:** A partir do valor de temperatura gerar a “classe” da seguinte forma:
 - Se menor ou igual a 20°C, classe deve ser 'FRIO'
 - De 20°C até 29°, o valor deve ser 'AGRADÁVEL'
 - 30°C a 34°, o valor será 'CALOR'
 - Acima de 34°C, o valor será 'CALOR ABSURDO'

```
temperatura = float(input('Qual a temperatura hoje? '))

# determina a classe da temperatura usando apenas if e else
if (temperatura <= 20):
    classe_temperatura = 'FRIO'
else:
    if (temperatura > 20) and (temperatura < 30):
        classe_temperatura = 'AGRADÁVEL'
    else:
        if (temperatura >= 30) and (temperatura <= 34):
            classe_temperatura = 'CALOR'
        else:
            classe_temperatura = 'CALOR ABSURDO'

print('Eu acho que isso é', classe_temperatura)
```

Instrução elif (3/7)

- A solução está correta. Mas os espaços em branco para alinhar os comandos subordinados aos if's e else's acabam causando um efeito visual desagradável.
- O programa fica excessivamente "largo" (ocupando muitas colunas), tornando-o mais difícil de ser examinado.

```
temperatura = float(input('Qual a temperatura hoje? '))

# determina a classe da temperatura usando apenas if e else
if (temperatura <= 20):
    classe_temperatura = 'FRIIO'
else:
    if (temperatura > 20) and (temperatura < 30):
        classe_temperatura = 'AGRADÁVEL'
    else:
        if (temperatura >= 30) and (temperatura <= 34):
            classe_temperatura = 'CALOR'
        else:
            classe_temperatura = 'CALOR ABSURDO'

print('Eu acho que isso é', classe_temperatura)
```

Instrução elif (4/7)

- Para contornar o problema, a linguagem Python oferece um comando chamado **elif**
- veja que o nome faz uma junção das palavras else e if.
- **elif** é usada para estruturar **diversos testes** em sequência, executando os comandos subordinados ao **primeiro teste avaliado como True**.
- Veja a seguir o exemplo que determina a classe de temperatura, agora usando **elif**.

Instrução elif (5/7)

- **Exemplo:** Obtém “classe” de temperatura da seguinte forma:
 - Se menor ou igual a 20°C, classe deve ser 'FRIO'
 - De 20°C até 29°, o valor deve ser 'AGRADÁVEL'
 - 30°C a 34°, o valor será 'CALOR'
 - Acima de 34°C, o valor será 'CALOR ABSURDO'

```
temperatura = float(input('Qual a temperatura hoje? '))

#determina a classe da temperatura usando if-else-elif
if (temperatura <= 20):
    classe_temperatura = 'FRIO'
elif (temperatura > 20) and (temperatura < 30):
    classe_temperatura = 'AGRADÁVEL'
elif (temperatura >= 30) and (temperatura <= 34):
    classe_temperatura = 'CALOR'
else:
    classe_temperatura = 'CALOR ABSURDO'

print('Eu acho que isso é', classe_temperatura)
```

Instrução elif (6/7)

■ EXPLICAÇÃO:

- Antes de mais nada, é importante entender que os comandos **if**, **elif** e **else** **trabalham em conjunto**.
- Quando utilizados, a primeira coisa que o computador vai checar, é se a condição associada ao comando **if** é verdadeira (resulta em True). Neste caso, os comandos (ou comando) subordinados ao **if** serão executados e nenhum dos **elifs** será processado.
- Porém, caso o teste do **if** resulte em False, o computador checará o **primeiro elif** da sequência. Se a condição deste **elif** resultar em True, os comandos (ou comando) a ele associados serão executados e nenhum dos **elifs** restantes será avaliado.

Instrução elif (7/7)

- EXPLICAÇÃO (cont...):
 - Entretanto, se o primeiro **elif** tem um teste que resulta em False, o computador analisa o segundo **elif**. Se o teste do segundo **elif** resultar em True, os seus comandos subordinados são executados e os **elifs** restantes não serão avaliados.
 - E assim ocorrerá para todos os comandos **elif** posteriores: o computador vai processando a sequência de cima pra baixo e quando ele acha algum **elif** que resulta em True, ele executa os seus comandos e não processa os **elifs** seguintes.
 - Se nem o **if** e nem qualquer dos **elifs** resultarem em True, os comandos subordinados ao **else** serão executados. Ou seja: o código que for colocado no **else** será executado apenas se todos os testes anteriores falharem. Por isso, o **else** precisa ser o último comando.
 - **Não** é obrigatório incluir o **else**. Você vai colocar apenas fizer sentido para resolver o seu problema.

A Sintaxe Econômica

- Sabemos que variáveis lógicas (bool) podem armazenar apenas dois valores: True ou False.
- Estas variáveis podem ser testadas de duas formas distintas em instruções **if** e **elif**: “convencional” e “econômica”.

```
x = True; y = False

# testa se é True do jeito convencional
if (x == True): print('x é True')

#Jeito econômico... não preciso especificar o "= True"
if x: print('x é True (teste econômico)')

#testa se é False do jeito convencional
if (y == False): print('y é False')

#Jeito econômico... usa-se "not variável"
if not y: print('y é False (teste econômico)')
```


Operação Ternária (1/3)

- É um recurso muito utilizado para realizar uma operação de atribuição (definir o valor de uma variável) a partir do resultado de um teste lógico.
- Esse teste lógico deve ser definido em uma única linha, possuindo a seguinte sintaxe:

v = valor1 if condição else valor2

- Neste exemplo, a variável **v** vai receber **valor1** caso a condição associada ao **if** resulte no valor True.
- Caso contrário, ela receberá o **valor2**.

Operação Ternária (2/3)

```
a = 10
```

```
b = 20
```

```
# usa o operador ternário para atribuir
```

```
# o valor às variáveis menor e maior
```

```
menor = a if a < b else b
```

```
maior = a if a > b else b
```

```
print(menor); print(maior)
```

```
>>>
```

```
10
```

```
20
```

- Nesse exemplo, a variável **menor** receberá o valor armazenado na variável **a** porque o resultado do teste **a < b** resulta em True.
- A variável **maior** recebe o valor armazenado em **b** pelo fato de o teste **a > b** resultar em False.

Operação Ternária (3/3)

- É importante registrar que a operação ternária não precisa necessariamente utilizada em uma atribuição.
- Ela é uma operação que retorna um valor, mas armazenar o valor em uma variável fica a critério do programador.
- Veja abaixo, onde foi utilizada dentro do `print()`.

```
a = 10; b = 20
```

```
print("menor valor: ", a if a < b else b)
```

```
print("maior valor: ", a if a > b else b)
```

```
>>>
```

```
menor valor: 10
```

```
maior valor: 20
```