# MC-102 — Aula 03
# Expressões Relacionais, Lógicas e Comandos Condicionais

Jacques Wainer

Instituto de Computação – Unicamp

8 de Agosto de 2019

# O Tipo bool

Em python o tipo **bool** especifica os valores booleanos falso (`False`) e verdadeiro (`True`).

- Podemos criar variáveis associadas a booleanos mas o uso mais comum é na verificação de resultados de expressões relacionais e lógicas.

```
>>> a = True
>>> type(a)
<class 'bool'>
```

# Expressões relacionais

Expressões relacionais são aquelas que realizam uma **comparação** entre duas expressões e retornam

1. **False**, se o resultado é falso
2. **True**, se o resultado é verdadeiro.

# Operadores Relacionais

Os operadores relacionais da linguagem Python são:

- $==$ : igualdade.
- $!=$ : diferente.
- $>$ : maior que.
- $<$ : menor que.
- $>=$ : maior ou igual que.
- $<=$ : menor ou igual que.

# Expressões relacionais

- *expressão* == *expressão* : Retorna verdadeiro quando as expressões forem iguais.

```
>>> 9 == 9
True
>>> 9 == 9.0   #<-- note este exemplo - compara OK int e float
True
>>> 9 == 10
False
>>> 9 == 9.0000001
False
>>>
```

- *expressão* != *expressão* : Retorna verdadeiro quando as expressões forem diferentes.
- *expressão* $>$ *expressão* : Retorna verdadeiro quando a expressão da esquerda tiver valor maior que a expressão da direita.
- *expressão* $<$ *expressão* : Retorna verdadeiro quando a expressão da esquerda tiver valor menor que a expressão da direita.

# Expressões relacionais

- *expressão* >= *expressão* : Retorna verdadeiro quando a expressão da esquerda tiver valor maior ou igual que a expressão da direita.
- *expressão* <= *expressão* : Retorna verdadeiro quando a expressão da esquerda tiver valor menor ou igual que a expressão da direita.

O que será impresso pelo programa?

```
print(9 > 3)

print( (3*4)/2 != (2*3) )

a = 1;
b = -1;
print(a!=b);
```

# Expressões lógicas

Expressões lógicas são aquelas que realizam uma operação lógica (**ou**, **e**, **não**, etc...) e retornam verdadeiro ou falso (como as expressões relacionais).

# Operadores Lógicos

Na linguagem Python temos os seguintes operadores lógicos:

- `and`: operador E.
- `or`: operador OU.
- `not`: operador NÃO.

# Expressões lógicas

- *expressão* `and` *expressão* : Retorna verdadeiro quando ambas as expressões são verdadeiras. Sua tabela verdade é:

| $Op_1$ | $Op_2$ | $Ret$ |
|---|---|---|
| V | V | V |
| V | F | F |
| F | V | F |
| F | F | F |

Qual o resultado da expressão lógica abaixo?

```
a = 0
b = 0
a == 0 and b == 0
```

# Expressões lógicas

- *expressão* `or` *expressão* : Retorna verdadeiro quando pelo menos uma das expressões é verdadeira. Sua tabela verdade é:

| $Op_1$ | $Op_2$ | $Ret$ |
|---|---|---|
| V | V | V |
| V | F | V |
| F | V | V |
| F | F | F |

Qual o resultado da expressão lógica abaixo?

```
a = 0
b = 1
a == 0 or b == 0
```

# Expressões lógicas

- `not` *expressão* : Retorna verdadeiro quando a expressão é falsa e vice-versa. Sua tabela verdade é:

| $Op_1$ | *Ret* |
|---|---|
| V | F |
| F | V |

Qual o resultado da expressão lógica abaixo?

```
a = 0
b = 1
not (a != b)
```

O que será impresso pelo programa?

```
print((8>9) and (10!=2))

print((14 > 100) or (2 > 1))

print(not (14>100)  and not (1>2)  )
```

# Precedência

Precedência é a ordem na qual os operadores serão avaliados quando o programa for executado. Em Python, os operadores são avaliados na seguinte ordem:

- **
- *, /, //, na ordem em que aparecerem na expressão.
- %
- + e -, na ordem em que aparecerem na expressão.
- comparações na ordem que aparecem
- **not**
- **and**
- **or**

Normalmente essa precedencia para os operadores lógicos é a mais intuitiva, mas na dúvida coloque parenteses.

Cuidado com testar se uma variável esta entre 2 valores
em Matemática $1 < x \leq 10$
Em Python NÃO podemos escrever

```
1 < x <= 10
```

mas temos que escrever assim:

```
1 < x and x <= 10
```

# Comandos condicionais

Um comando condicional é aquele que permite decidir se um determinado bloco de comandos deve ou não ser executado, a partir do resultado de uma expressão relacional ou lógica.

# Bloco de comandos

- É um conjunto de instruções agrupadas.
- Os comandos agrupados do bloco devem estar **indentados** dentro de um comando anterior seguido de **dois pontos**.
- A indentação é feita em geral com 2 espaços em branco antes de cada comando que deve estar dentro do bloco.

# Comandos condicionais - if

- O principal comando condicional é o **if**, cuja sintaxe é:

```
if expressão relacional ou lógica :
    comando1
    comando2
    ...
    comandon executados se a expressão é verdadeira

proximoComando
```

- Os comandos são executados somente se a expressão relacional/lógica for verdadeira.
- o `proximoComando` sera sempre executado, já que ele é o comando de *se segue* ao **if**
- os comandos precisam estar alinhados - mesmo número de brancos antes de cada um

# Comandos condicionais

O programa abaixo determina se um valor é par.

```
a = int(input())
if a%2 == 0:
  print("O número digitado é par")
```

# Comandos condicionais - if-else

- Uma variação do comando **if** é o **if/else**, cuja sintaxe é:

```
if expressão relacional ou lógica :
    ...
    ...
    comandos executados se a expressão é verdadeira
else:
    ...
    ...
    comandos executados se a expressão é falsa

proximoComando
```

# Comandos condicionais

Exemplo: Determinando o menor de dois números:

```
a = int(input("Digite um número:"))
b = int(input("Digite um número:"))
if a < b:
    print("O menor número é: ", a )
else:
    print("O menor número é: ", b )
```

# Comandos condicionais

- Note que o **if** é um comando, e como tal pode aparecer dentro do bloco de comandos de outro **if**.

Exemplo: Usando apenas **operadores relacionais e aritméticos**, vamos escrever um programa que lê um número e verifica em qual dos seguintes casos o número se enquadra:

- Par e menor que 100.
- Par e maior ou igual a 100.
- Ímpar e menor que 100.
- Ímpar e maior ou igual a 100.

# Comandos condicionais

```
a = int(input("Digite um número:"))
if a % 2 == 0:     #<-- Se número for par, executa bloco abaixo
    if a<100:
        print("O número é par e menor do que 100")
    else:
        print("O número é par e maior ou igual que 100")
else:              #<-- Se número for ímpar, executa bloco abaixo
    if a<100:
        print("O número é ímpar e menor do que 100")
    else:
        print("O número é ímpar e maior ou igual que 100")
```

Note o novo uso do **input**

Usando operadores lógicos, refazer este programa

# Comandos condicionais

```
print("Digite um número:")
a = int(input())
if (a % 2 == 0) and (a < 100):
    print("O número é par e menor do que 100")
if (a % 2 == 0) and (a >= 100):
    print("O número é par e maior ou igual que 100")
if (a % 2 != 0) and (a < 100):
    print("O número é ímpar e menor do que 100")
if (a % 2 != 0) and (a >= 100):
    print("O número é ímpar e maior ou igual que 100")
```

# Comandos condicionais

Lembre-se que o que define a qual bloco de comandos um comando pertence é a sua indentação!

```
if cond1:
  if cond2:
    comando1
else:
  comando2
```

Quando o **comando2** é executado?

# Exemplo; Comandos condicionais

Lembre-se que o que define a qual bloco de comandos um comando pertence é a sua indentação!

```
if cond1:
  if (cond2):
    comando1
else:
  comando2
```

Quando o **comando2** é executado?
Resposta: quando **cond1** for falsa.

# Exemplo: Comandos condicionais

```
if cond1:
  if cond2:
    comando1
  else:
    comando2
```

Quando o `comando2` é executado?

# Exemplo: Comandos condicionais

```
if cond1:
  if cond2:
    comando1
  else:
    comando2
```

Quando o `comando2` é executado?

Resposta: quando a `cond1` for verdadeira e `cond2` for falsa.

# Exemplo: Comandos condicionais

```
if cond1:
    if cond2:
        comando1
    else:
        comando2
else:
    if cond3:
        comando3
    else:
        comando4
```

Quando o **comando4** é executado?

# Exemplo: Comandos condicionais

```
if cond1:
    if cond2:
        comando1
    else:
        comando2
else:
    if cond3:
        comando3
    else:
        comando4
```

Quando o **comando4** é executado?
Resposta: quando a **cond1** for falsa e **cond3** for falsa.

# Exemplo

```
a = 5
if a > 3:
    if a < 7:
        print("a")
else:
    if a > -10:
        print("b")
    else:
        print("c")
```

O que será impresso?

# Exemplo

```
a = -12
if a > 3:
    if a < 7:
        print("a")
else:
    if a > -10:
        print("b")
    else:
        print("c")
```

O que será impresso?

# Exemplo

```
a = 9
if a > 3:
    if a < 7:
        print("a")
else:
    if a > -10:
        print("b")
    else:
        print("c")
```

O que será impresso?

# Exercícios

A solução abaixo está correta para classificar um número como par e menor que 100, ou par e maior ou igual a 100, etc, como no exemplo visto anteriormente?

```
print("Digite um número:")
a = int(input())
if (a % 2 == 0) and (a < 100):
  print("O número é par e menor do que 100")
else:
  if (a >= 100):
    print("O número é par e maior ou igual que 100")
if (a % 2 != 0) and (a < 100):
  print("O número é ímpar e menor do que 100")
else:
  if a >= 100:
    print("O número é ímpar e maior ou igual que 100")
```

# Exercícios

- Escreva um programa que lê um número inteiro do teclado e imprime "SIM"se o número for par e maior do que 10, ou for ímpar e menor do que 50. Caso contrário o programa deve imprimir "NAO".

# Exercícios

- Escreva um programa lê três números e imprime o maior deles.

# Exercícios

- Escreva um programa lê três números distintos e os imprime em ordem (ordem decrescente).

# Comandos Condicionais **if-else**

- Vamos fazer um programa que calcula a área de três tipos de objetos geométricos: quadrado, retângulo e círculo.
- Primeiramente deve ser lido um caractere que indica o tipo de objeto a ter a área calculada: 'q' para quadrado, 'r' para retângulo e 'c' para círculo.
- Em seguida deverá ser lido as dimensões do objeto:
  - Para um quadrado deve ser lido o tamanho de um lado.
  - Para um retângulo devem ser lidos os tamanhos de cada lado.
  - Para um círculo, deve ser lido o raio.
- Em seguida o programa faz o cálculo da área e a imprime.
- Se o usuário digitar um caractere diferente de 'q', 'r', e 'c' o programa deverá imprimir uma mensagem de erro.

# Alternativa: um **if** debaixo do outro

```
print("Digite uma opção (q, r, ou c):")
a = input()
if a == "q":
  ...
if a == "r":
  ...
if a == "c":
  ...
if a != "q" and a != "r" and a != "c":
  print("Opção inválida!")
```

- O programa lê um caractere e testa se este corresponde a cada uma das opções válidas.
- O **if** final testa se o caractere lido não corresponde a nenhuma opção.
- Basta agora, dentro de cada opção, implementar a leitura dos dados e o cálculo da área.

# Alternativa: um **if** debaixo do outro

```
print("Digite uma opção (q, r, ou c):")
a = input()
if a == "q":
  x = input("Digite o tamanho do lado do quadrado: ")
  lado = float(x)
  print("A área é", lado*lado)
if a == "r":
  ...
if a == "c":
  ...
if a != "q" and a != "r" and a != "c":
  print("Opção inválida!")
```

Vamos chamar essa solução de *um if debaixo do outro*

# Alternativa: um **if** debaixo do outro

```
print("Digite uma opção (q, r, ou c):")
a = input()
if a == "q":
    ...
if a == "r":
    ...
if a == "c":
    ...
if a != "q" and a != "r" and a != "c":
    print("Opção inválida!")
```

- para que um if debaixo do outro funcione, cada condição do if só pode ser verdade uma vez
- a ultima condição diz: se a for diferente de "q" e de "r" e de "c" então:
- um outro geito de escrever essa condicão é

```
if not (a == "q" or a == "r" or a == "c"):
    print("Opção inválida!")
```

# Alternativa: **if-else** encaixados

Refazendo o programa utilizando **if-else**:

```
print("Digite uma opção (q, r, ou c):")
a = input()
if a == "q":
    ...
else:
    if a == "r":
        ...
    else:
        if a == "c":
            ...
        else:
            ...
```

Na versão de if-else encaixados, a parte then sabe o que fazer, as outras perguntas ficam apenas na parte **else**
O ultimo caso, ficou no ultimo **else** - não precisamos escreve-lo explicitamente.

# Comparação

- A versão **if** debaixo do outro é visualmente mais simples, mas todos os testes precisam ser excludentes, quando um da certo os outros não podem dar certo, o que tornou o ultimo teste meio complexo
- a versão do **if-else** encaixados só faz os testes necessários até achar a alternativa correta, mas o programa fica "escorregando" para a direita.
- Há uma alternativa que combina a eficiência do **if-else** encaixado com a simplicidade visual do **if** debaixo do outro. O **if-elif**.
- o **if-elif** combina os comandos de **else** e o **if** seguinte do **if-else** encaixados, num único comando **elif**
- a ultima alternativa da sequencia é o **else** (o mesmo que no **if-else** encaixado.

# Alternativa: **if-elif**

```
print("Digite uma opção (q, r, ou c):")
a = input()
if a == "q":
    ...
elif a == "r":
    ...
elif a == "c":
    ...
else:
    ...
```

# Comando **if-elif**

A versão completa do programa:

```
print("Digite uma opção (q, r, ou c):")
a = input()
if a == "q":
  x = input("Digite o tamanho do lado do quadrado: ")
  l = float(x)
  print("A área é :", l*l)
elif a == "r":
  x1 = input("Digite o tamanho de um lado do retângulo: ")
  l1 = float(x1)
  x2 = input("Digite o tamanho do outro lado do retângulo: ")
  l2 = float(x2)
  print("A área é :", l1*l2)
elif a == "c":
  x = input("Digite o tamanho do raio: ")
  r = float(x)
  print("A área é :", 3.1415*r*r)
else:
  print("Opção inválida!")
```

# Comandos **if-elif**

Outro exemplo:

- No brasileirão, 20 times disputam o título em dois turnos. No primeiro turno todos os times jogam entre si uma única vez. Os jogos do segundo turno ocorrem na mesma ordem que no primeiro, apenas invertendo-se o mando de campo.
- Os times são classificados por pontos. Caso dois times atinjam o mesmo número de pontos, eles são desempatados aplicando-se os seguintes critérios nesta ordem:
  1. número de vitórias (maior melhor)
  2. saldo de gols (maior melhor)
  3. gols marcados (maior melhor)
  4. número de cartões vermelho (menor melhor)
  5. número de cartões amarelos (menor melhor)

Faça um programa que leia o número de pontos de cada time, e as cinco informações acima de dois times e decida qual time venceu o campeonato.

# Comando **if-elif**

Abaixo temos o código que faz a leitura das informações necessárias.

```
print("Lendo dados do time 1")
pontos1 = int(input("Número de pontos:"))
vitorias1 = int(input("Número de vitórias:"))
saldo1 = int(input("Saldo de gols:"))
gols1 = int (input("Gols marcados:"))
vermelho1 = int(input("Número de cartões vermelhos:"))
amarelo1 = int(input("Número de cartões amarelos:"))

print("Lendo dados do time 2")
pontos2 = int(input("Número de pontos:"))
vitorias2 = int(input("Número de vitórias:"))
saldo2 = int(input("Saldo de gols:"))
gols2 = int (input("Gols marcados:"))
vermelho2 = int(input("Número de cartões vermelhos:"))
amarelo2 = int(input("Número de cartões amarelos:"))
```

# Comando **if-elif**

Começamos então a testar quem possui mais vitórias para decidir o vencedor:

```
print("Lendo dados do time 1")
...
...

if pontos1 > pontos2:
    print("Time 1 ganha do Time 2")
elif pontos1 < pontos2:
    print("Time 2 ganha do Time 1")
```

O que podemos deduzir se as duas condições dos **ifs** acima forem falsas?
**Resposta:** O número de pontos dos dois times é igual. Devemos então continuar testando as outras informações...

# Comando **if-elif**

```
if pontos1 > pontos2:
    print("Time 1 ganha do Time 2")
elif pontos1 < pontos2:
    print("Time 2 ganha do Time 1")
if vitorias1 > vitorias2:
    print("Time 1 ganha do Time 2")
elif vitorias1 < vitorias2:
    print("Time 2 ganha do Time 1")
elif saldo1 > saldo2:
    print("Time 1 ganha do Time 2")
elif saldo1 < saldo2:
    print("Time 2 ganha do Time 1")
elif gols1 > gols2:
    print("Time 1 ganha do Time 2")
elif gols1 < gols2:
    print("Time 2 ganha do Time 1")
elif vermelho1 < vermelho2:
    print("Time 1 ganha do Time 2")
elif vermelho1 > vermelho2:
    print("Time 2 ganha do Time 1")
elif amarelo1 < amarelo2:
    print("Time 1 ganha do Time 2")
elif amarelo1 > amarelo2:
    print("Time 2 ganha do Time 1")
```

# Comando **if-elif**

```
...
...
elif vermelho1 < vermelho2:
    print("Time 1 ganha do Time 2")
elif vermelho1 > vermelho2:
    print("Time 2 ganha do Time 1")
elif amarelo1 < amarelo2:
    print("Time 1 ganha do Time 2")
elif amarelo1 > amarelo2:
    print("Time 2 ganha do Time 1")
else:
    print("Times continuam empatados!")
```

Pela regra do campeonato, se os times continuarem empatados então o desempate se dará por sorteio!

# Exercícios

Quando ações são vendidas ou compradas por meio de um corretor, a comissão do corretor é muitas vezes calculada usando uma escala que depende do valor das ações negociadas. Escreva um programa que calcule o valor da comissão a partir do valor da transação informado pelo usuário, sabendo-se que o corretor cobra os valores indicados abaixo e que a **comissão mínima é de R$ 39,00**:

- Até R$ 2.500,00, comissão de R$30+1,7%
- R$2.500,01 até R$6.250,00, comissão de R$56 + 0,66%
- R$6.250,01 até R$20.000,00, comissão de R$76 + 0,34%
- R$20.000,01 até R$50.000,00, comissão de R$100 + 0,22%
- R$50.000,01 até R$500.000,00, comissão de R$155 + 0,11%
- Mais que R$ 500.000,00, comissão de R$255 + 0,09%