

SELEÇÃO CONDICIONAL BÁSICA

MC102 - Algoritmos e
Programação de
Computadores

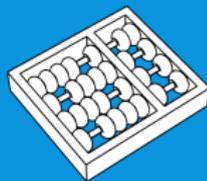
Santiago Valdés Ravelo
[https://ic.unicamp.br/~santiago/
ravelo@unicamp.br](https://ic.unicamp.br/~santiago/ravelo@unicamp.br)

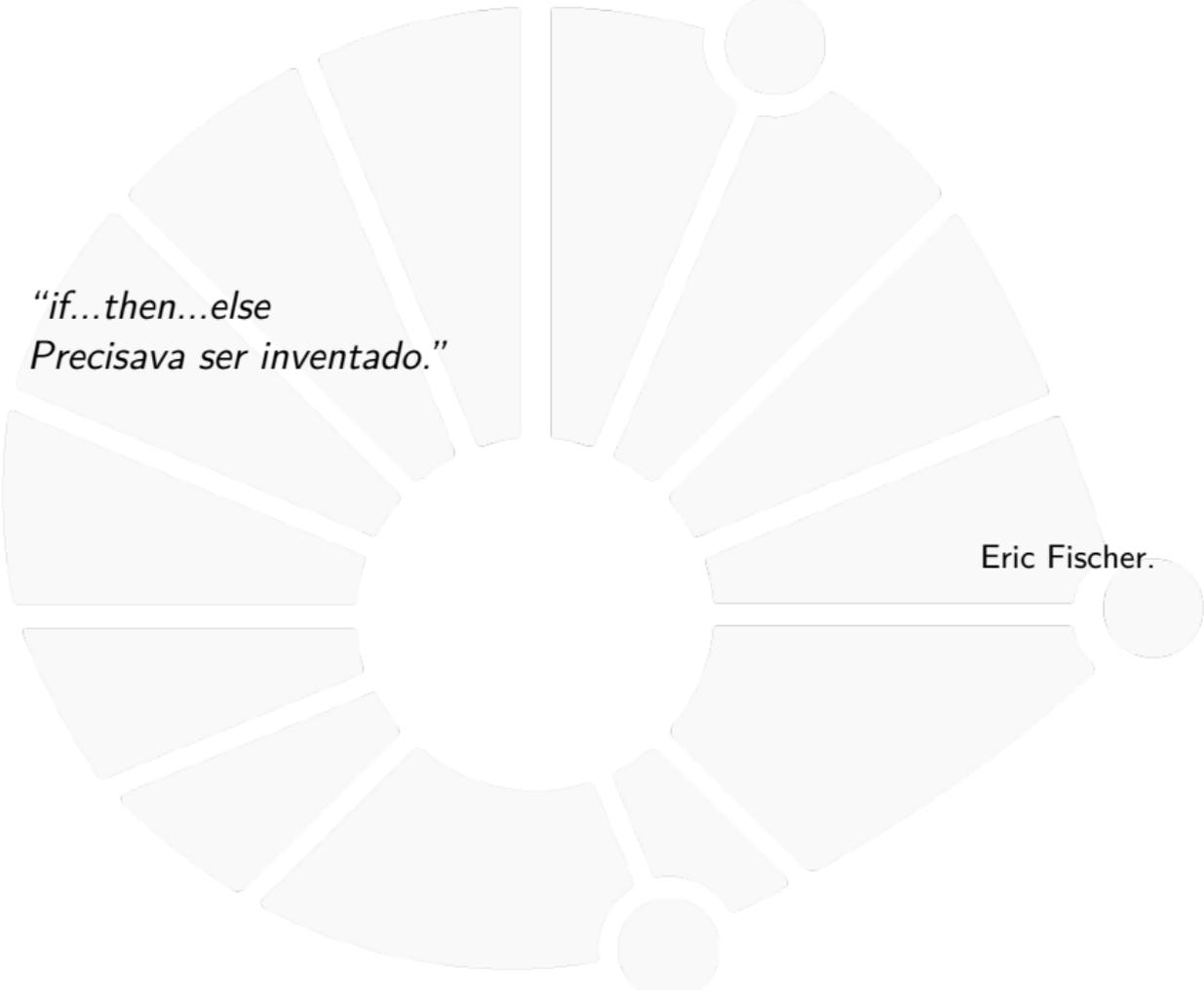
03/25

3



UNICAMP



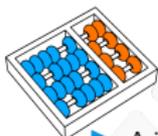


*"if...then...else
Precisava ser inventado."*

Eric Fischer.



DÚVIDAS DA AULA ANTERIOR



Dúvidas selecionadas

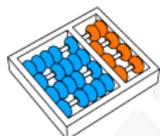
- ▶ A incerteza das operações com float é aleatória ou é definida? Se fizermos mais de uma operação para achar o resultado, podemos propagar a incerteza?
- ▶ É possível mudar o tipo de uma variável em python?
- ▶ Quais são as principais diferenças entre C e C++?
- ▶ Como rodar um código em python? Tentei fazer depois da aula e não consegui executar.
- ▶ Como poderíamos representar a raiz quadrada?
- ▶ Por que a linguagem Python é tão usada em AI e Machine Learning? Considerando o grande gasto energético para treinar os modelos, não seria melhor usar uma linguagem mais rápida?
- ▶ Por que, mesmo se o restante forem caracteres válidos, não se é possível nomear variáveis iniciando com um número? Por que se convencionou dessa forma?
- ▶ Tem uma explicação do porquê mesmo atribuindo $y=x$, o y não muda também pra 15?

```
1 >>>x=10
2 >>>x
3 10
4 >>>y=x
5 >>>y
6 10
7 >>>x=15
8 >>>y
9 10
```

- ▶ Em quais casos é mais útil usar o Python em modo interativo e no não interativo?
- ▶ Operações com floats na forma de notação científica também apresentam erros de precisão?
- ▶ Como mitigar a imprecisão de operações com floats decimais?
- ▶ O quão longe eu posso ir com Python? Isto é: dado o que eu irei aprender na disciplina, eu conseguirei evoluir ao ponto de fazer a minha carreira (se for o caso) apenas com essa linguagem de programação?
- ▶ Qual o critério para a seleção de dúvidas?



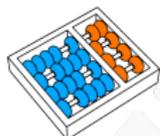
VERDADEIRO OU FALSO



Testando o valor de uma variável

No terminal do Python:

```
1 >>> x = 2
2 >>> x == 2
3 True
4 >>> x == 3
5 False
6 >>> type(x == 2)
7 <class 'bool'>
8 >>> type(x == 3)
9 <class 'bool'>
10 >>> type(True)
11 <class 'bool'>
12 >>> type(False)
13 <class 'bool'>
```

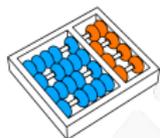


O tipo **bool**

O tipo **bool** define duas constantes:

- ▶ **True.**
- ▶ **False.**

E várias operações devolvem um **bool**.



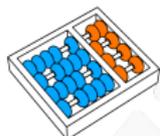
Testando se um número é par

Queremos definir se um número n dado é par ou ímpar:

- ▶ Isto é, $n = 2k$ ou $n = 2k + 1$ para k inteiro.

Problema (Par ou ímpar)

- ▶ **Entrada:** Um número n .
- ▶ **Saída:** Um texto que informe se n é par ou ímpar.



Pseudocódigo

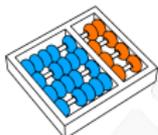
Antes do Python, vamos pensar abstratamente:

Algoritmo: PAR-OU-IMPAR

```
1 leia n
2 se n for par
3   | escreva 'é par'
4 senão
5   | escreva 'é ímpar'
```

Porém, precisamos ter cuidado para que:

- ▶ Cada passo seja suficientemente simples.
- ▶ E que possa ser executado pelo computador.



Pseudocódigo

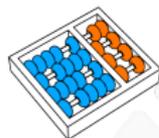
Qual é uma boa forma de testar se n é par ou não?

- ▶ Se n for par, então $n \% 2$ é 0.
- ▶ Se n for ímpar, então $n \% 2$ é 1.

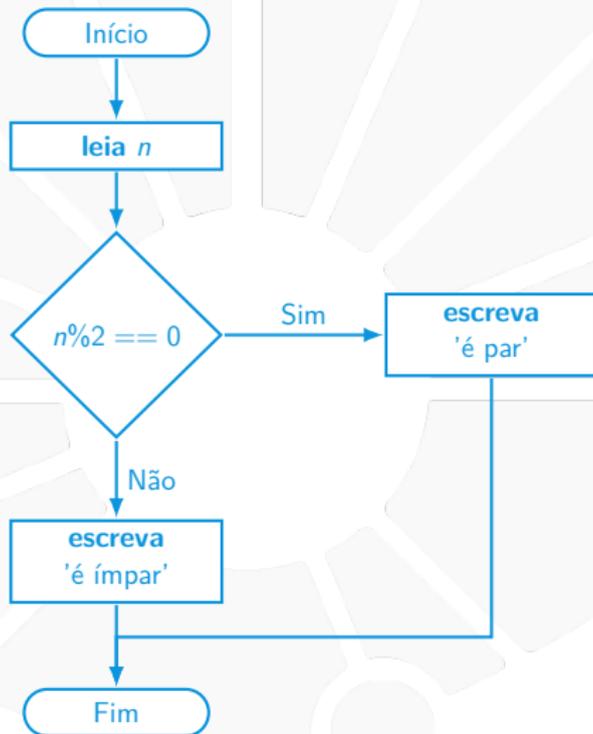
Um pseudocódigo mais claro seria:

Algoritmo: PAR-OU-IMPAR

- 1 **leia** n
 - 2 **se** $n \% 2 = 0$
 - 3 | **escreva** 'é par'
 - 4 **senão**
 - 5 | **escreva** 'é ímpar'
-



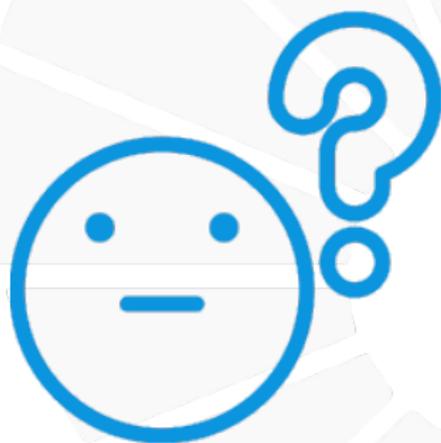
Fluxograma



Verdadeiro ou falso



Pergunta



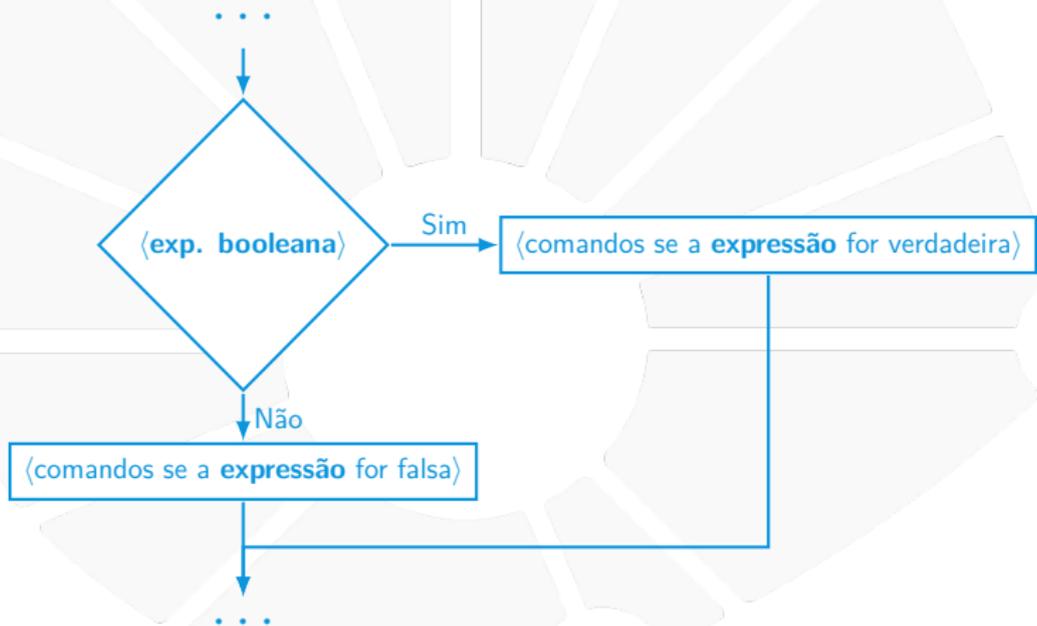
Como selecionar em Python?

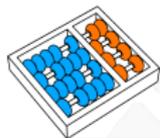


SELEÇÃO CONDICIONAL



Funcionamento





Em Python

```
1 ...
2 if <exp. booleana>:
3     <comandos se a expressão for verdadeira>
4 else:
5     <comandos se a expressão for falsa>
6 ...
```

O `if ... else`:

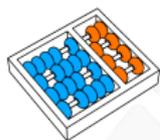
- ▶ Verifica o valor da expressão booleana.
- ▶ Se for **True**, executa o bloco de código do **if**.
- ▶ Se for **False**, executa o bloco de código do **else** (opcional).

O Python utiliza a indentação para criar **blocos de código**:

- ▶ Ela não é opcional como em outras linguagens.
- ▶ E precisa ser consistente (quatro espaços é o recomendado).

O `:` indica que a linha do **if/else** terminou:

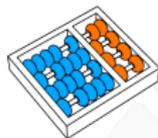
- ▶ Também indica que um bloco de código irá começar.
- ▶ Ele é usado em vários outros comandos.



Código em Python

Então, uma solução para determinar se um número é par ou ímpar seria:

```
1 n = int(input('Entre o número: '))
2 if n % 2 == 0:
3     print(n, 'é par')
4 else:
5     print(n, 'é ímpar')
```

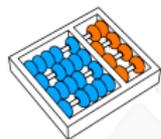


Outras comparações

Além de igualdade ($==$), podemos usar também:

- ▶ $<$ para saber se $a < b$.
- ▶ $>$ para saber se $a > b$.
- ▶ $<=$ para saber se $a \leq b$.
- ▶ $>=$ para saber se $a \geq b$.
- ▶ $!=$ para saber se $a \neq b$.

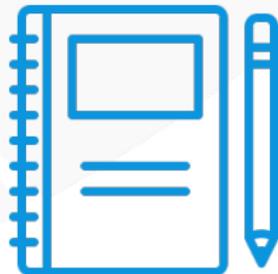
Juntamente com o $==$, são chamados de **operadores de comparação**.

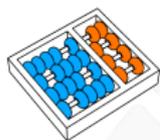


Seleção condicional



Vamos fazer alguns exercícios?



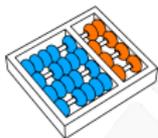


Exercício 1

Um inteiro n é divisível por um inteiro q se existe um inteiro a tal que $n = aq$.

▶ Isto é, se $n \% q = 0$.

Queremos verificar se n é divisível por 2 ou por 3.



Uma solução

Queremos verificar se n é divisível por **2** ou por **3**:

- ▶ Podemos usar dois **ifs** em sequência:

```
1 n = int(input())
2 # n ser divisível por 2 é o mesmo que o resto da divisão por 2 ser 0
3 if n % 2 == 0:
4     print(n, "é divisível por 2")
5 else:
6     print(n, "não é divisível por 2")
7 if n % 3 == 0:
8     print(n, "é divisível por 3")
9 else:
10    print(n, "não é divisível por 3")
```

A linha 2 é de comentários:

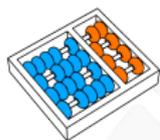
- ▶ Servem para entender melhor o código.
- ▶ São ignoradas pelo Python.
- ▶ Comentários **devem** ser usados, mas com **moderação**.



Exercício 2

Queremos verificar se n :

- ▶ É divisível por 2.
- ▶ E não é divisível por 3.



Uma solução

Queremos verificar se **n** é divisível por **2** e não é divisível por **3**.

Podemos usar um **if** dentro do outro!

```
1 n = int(input())
2 if n % 2 == 0:
3     if n % 3 != 0:
4         print(n, "é divisível por 2 e não por 3")
5     else:
6         print(n, "é divisível por 2 e por 3")
7 else:
8     print(n, "não é divisível por 2")
```



Exercícios 3, 4 e 5

1. Escreva um programa que lê dois números e encontra o maior dos dois.
2. Escreva um programa que lê dois números inteiros x e y , sendo que y tem apenas um dígito (na base 10) e verifica se y é o último dígito (na base 10) de x .
3. Escreva um programa que lê dois números inteiros x e y e diz em qual quadrante do plano o ponto (x, y) está.



Exercício 6

O tempo Unix nos diz quantos segundos se passaram desde a Época Unix (00:00 de 01 de Janeiro de 1970 — UTC).

Exemplo:

- ▶ Se o tempo Unix atual é 3600, então estamos em 01:00 de 01/01/1970 (UTC)
- ▶ Se o tempo Unix é 86400, então estamos em 00:00 de 02/01/1970 (UTC).

Escreva um programa que dado um tempo Unix diz qual é o dia da semana daquele tempo.

SELEÇÃO CONDICIONAL BÁSICA

MC102 - Algoritmos e
Programação de
Computadores

Santiago Valdés Ravelo
[https://ic.unicamp.br/~santiago/
ravelo@unicamp.br](https://ic.unicamp.br/~santiago/ravelo@unicamp.br)

03/25

3



UNICAMP

