



INSTITUTO FEDERAL

São Paulo

Câmpus São Carlos

Introdução ao Python

ALG – Algoritmos e Programação

Aula 9

Curso Técnico em Informática para Internet – Integrado ao Ensino Médio



INSTITUTO FEDERAL

São Paulo

Câmpus São Carlos

Objetivos da aula

- Apresentar as principais características da Linguagem de Programação Python
- Apresentar o ambiente de programação
- Apresentar os comandos de entrada e saída
- Desenvolver os primeiros programas em Python utilizando estruturas sequenciais

O que é Python?

- Linguagem de Programação de alto nível criada em 1990 por Guido Van Rossum, no Instituto Nacional de Pesquisa para Matemática e Ciência da Computação da Holanda (CWI).
- Tinha originalmente foco em usuários como físicos e engenheiros.
- Foi concebida a partir de outra linguagem existente na época, chamada ABC.
- A implementação oficial da linguagem é mantida pela *Python Software Foundation* (PSF)
- Python foi escrito em C e por isso também é conhecida como CPython.

Principais Características

- Software de código aberto (licença compatível com a *General Public License* – GPL)
- Sintaxe clara e concisa que favorece a legibilidade do código-fonte, tornando a linguagem mais produtiva
- Inclui diversas estruturas de alto nível (listas, dicionários, data/hora, etc.)
- Possui uma vasta coleção de módulos prontos para uso, além de *frameworks* de terceiros que podem ser adicionados
- Multiparadigma, a linguagem suporta programação modular, funcional e orientada a objetos
- Linguagem Interpretada através do *bytecode* pela máquina virtual Python
 - Código portátil (multiplataforma)

Quem usa Python atualmente?

- A linguagem é bem aceita na indústria por empresas de alta tecnologia, tais como:
 - Google (aplicações web)
 - Yahoo (aplicações web)
 - Microsoft (IronPython: Python para .Net)
 - Nokia (disponível para as linhas recentes de celulares e PDAs)
 - Disney (animações 3D)

Primeiros Passos

- Para a plataforma Windows, basta executar o instalador.
- Versão estável mais recente está disponível em:
<http://www.python.org/download>
- Para outras plataformas, como Linux, geralmente o Python já faz parte do sistema.

Primeiros exemplos

Imprimindo “Hello World” na tela

```
>>> print('Hello World')  
Hello World
```

Outra forma de fazer a mesma coisa

```
>>> a = 'Hello World'  
>>> a  
'Hello World'
```

Outra maneira...

```
>>> frase = input('Entre com uma frase:')  
Entre com uma frase:Hello World  
>>> frase  
'Hello World'
```

Primeiros exemplos

Imprimindo “Hello World” na tela

```
>>> print('Hello World')  
Hello World
```

Comando de saída: **print**

Outra forma de fazer a mesma coisa

```
>>> a = 'Hello World'  
>>> a  
'Hello World'
```

a é uma variável que armazena a string *Hello World*

Outra maneira...

```
>>> frase = input('Entre com uma frase:')  
Entre com uma frase:Hello World  
>>> frase  
'Hello World'
```

Comando de entrada: **input**

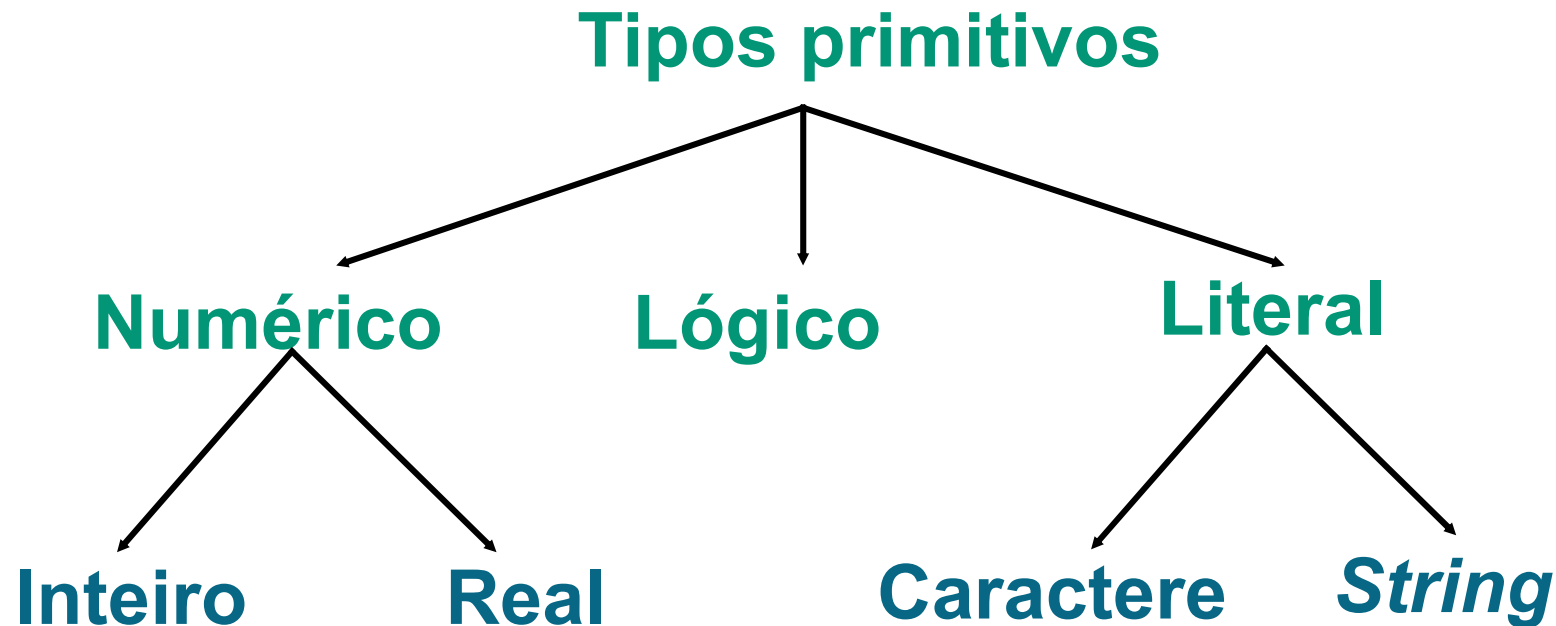
frase é uma variável que armazena a string digitada pelo usuário

Relembrando: Informação, Dados e Instruções

- O computador manipula **informações** contidas em sua **memória**.
- Elas são classificadas em dois tipos:
 - **Instruções**: comandam o funcionamento da máquina e determinam a maneira como os dados devem ser tratados.
 - **Dados**: informação que devem ser manipulada pelo computador.
- Exemplo de DADOS:
 - nomes de pessoas, de ruas, salário, idade, preço de um produto, peso, altura, etc.
- Existem vários **tipos de dados**
 - cada tipo é representado e processado de forma diferente
- Definir um tipo de dados tem dois propósitos:
 - classificar os dados de acordo com as informações contidas neles.
 - indicar quanto espaço de memória deve ser alocado

Relembrando: Informação, Dados e Instruções (cont.)

- Tipos de dados primitivos



Tipos de Dados em Python

- Existem vários tipos simples de dados predefinidos em Python, tais como:
 - Números (inteiros, reais, complexos, ...)
 - Texto
- Alguns tipos numéricos:
 - **int** (inteiro): $i = 10$
 - **float** (real de ponto flutuante): $f = 3.14$
 - **complex** (complexo): $c = 3 + 4j$
- Além dos inteiros convencionais também existem os inteiros longos, que têm dimensão arbitrária e são limitados pela memória disponível.
 - A conversão entre inteiro e longo é automática.
- A função **int()** é usada para converter outros tipos para inteiro, incluindo mudança de base.

Tipos de Dados em Python (cont.)

■ Exemplos:

- Convertendo de real para inteiro:
 - `print("int(3.14) = ", int(3.14))`
- Convertendo de inteiro para real:
 - `print("float(3) = ", float(3))`
- Inteiros em outras bases:
 - `print("int('20', 8) = ", int('20', 8)) # base 8`
 - `print("int('20', 16) = ", int('20', 16)) # base 16`
- Cálculo entre inteiro e real resulta em real:
 - `print("5.0 / 2 + 3 = ", 5.0 / 2 + 3)`

Tipos de Dados em Python (cont.)

■ Dado do tipo **Texto**:

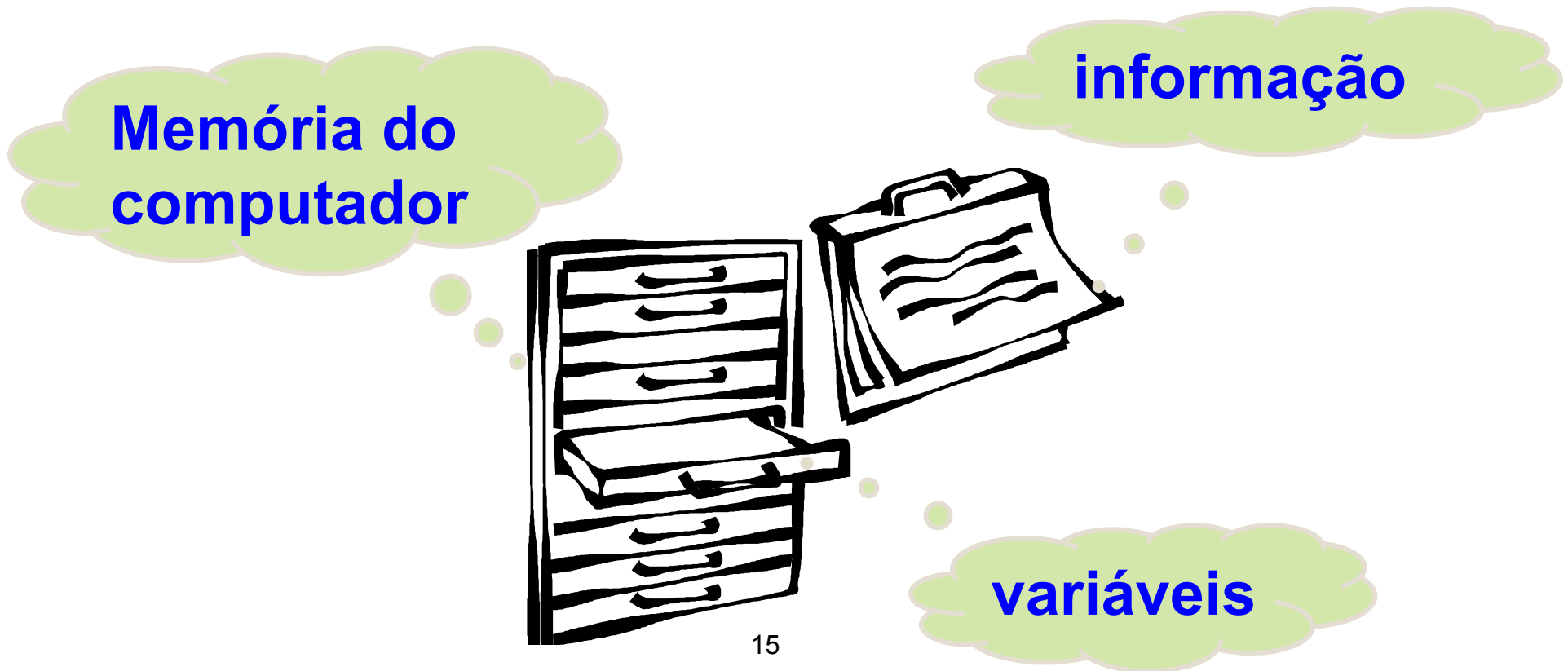
- **string** é o tipo especializado para armazenar texto.
- É imutável, isto é, não permite alterações no conteúdo da variável.
- Para adicionar, remover ou mesmo alterar algum caractere de uma string, o Python precisa criar uma nova string.
- Suporta acentos e caracteres especiais.
- A inicialização de uma string pode ser com aspas simples ou duplas.
- Exemplos:
 - nome = 'Led Zeppelin'
 - frase = "Eu adoro programar."
 - Sexo = 'F'

Relembrando: Variáveis

- Durante a execução do programa os **dados** estão sendo manipulados
- Para que o computador não esqueça das informações **contidas em um dado** é necessário guardá-las em sua **memória**.
- As **variáveis** guardam informações sobre os dados (o seu conteúdo) que estão sendo manipulados.

Relembrando: Variáveis (cont.)

- Armazenamento das variáveis na memória do computador.



15

Relembrando: Variáveis (cont.)

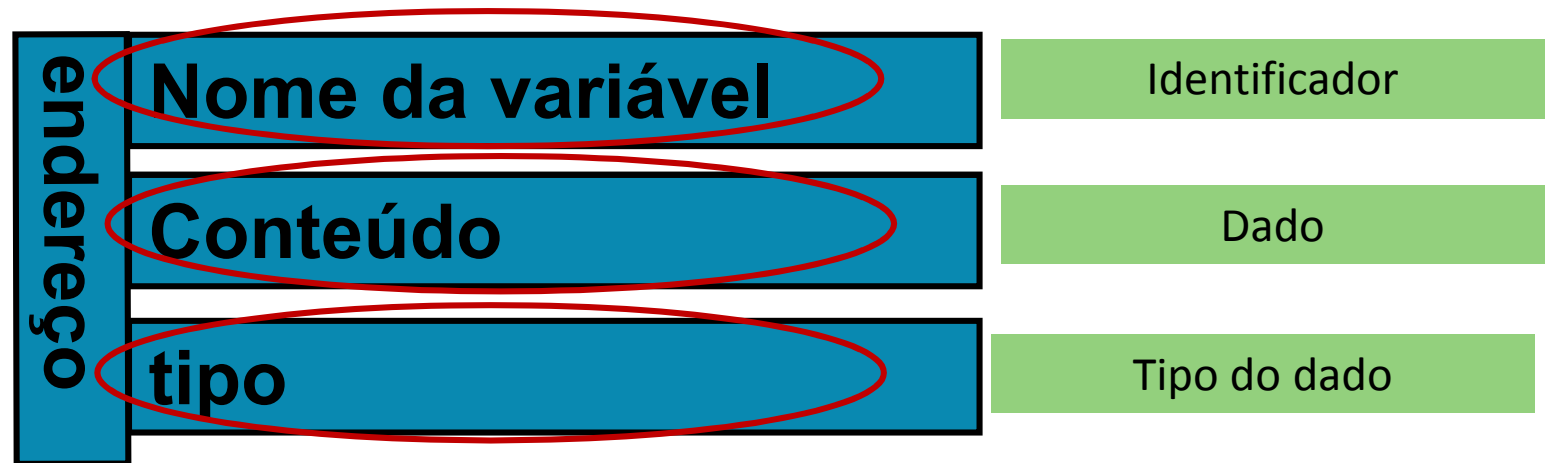
- Armazenamento das variáveis na memória do computador.



Overflow!!!!

Relembrando: Variáveis (cont.)

- Armazenamento das variáveis na memória do computador.



Variáveis em Python

- As variáveis são criadas através de atribuição e são destruídas pelo coletor de lixo (*garbage collector*), quando não existem mais referências a elas.
- Os nomes das variáveis devem começar com uma letra ou sublinhado (`_`), seguido por letras, dígitos ou sublinhados.
 - Exemplos válidos: `_nome`, `endereço`, `Preço_atual`, `nota1`, `peso_1`
- Não podem conter **espaços** nem **caracteres especiais** (`!`, `%`, `$`, `#`, etc.)
 - Exemplos inválidos: `Idade#`, `nome do cliente`, `@produto`
- Não podem iniciar com **números**.
- Letras **maiúsculas** e **minúsculas** são consideradas diferentes
 - Exemplo: `Numero`, `numero`, `NUMERO`

Variáveis em Python (cont.)

- Atenção: Procure utilizar sempre nomes significativos para as variáveis.
 - Exemplos:
 - Que informação é armazenada na variável **idade**?
 - E na variável **endereço**?
 - E na variável **x12aa**?

Instruções Primitivas em Programação

- Comando de atribuição
- Comando de entrada
- Comando de saída

Comando de Atribuição

- Permite que se **atribua um valor** a uma certa **variável**.
- A **natureza** desse valor deve ser **compatível** com o tipo da variável na qual está sendo armazenado.
- **Operador de atribuição em Python:** operador **=** (sinal de igual)
 - Exemplos:
 - `A = 312` # a variável A recebe o valor inteiro 312
 - `sexo = 'M'` # a variável sexo recebe o caractere M
 - `Rua = "13 de Maio"` # a variável Rua recebe a *string* "13 de Maio"
- **Obs:** Equivale ao comando **mude** do Scratch



Relembrando: Comando de Entrada

- O comando de entrada é utilizado para receber dados digitados pelo usuário (**DADOS DE ENTRADA**) e armazená-los em **variáveis**.
- Os dados de entrada são fornecidos ao sistema por meio de uma **unidade de entrada**, por exemplo o **teclado**.



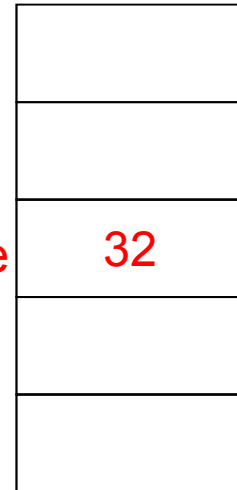
Digite sua idade em anos: 32

Programa em execução

22

idade

32



Memória

Relembrando: Comando de Entrada (cont.)

- Atenção: Na execução de um comando de entrada, o processamento é **interrompido**, até que sejam fornecidos, via unidade de entrada, **valores** para os dados de entrada.
- Os valores digitados pelo teclado devem ser **separados** pela digitação da tecla **<ENTER>**

Comando de Entrada em Python – input()

- O comando **input()** obtém dados informados pelo usuário via teclado
- O retorno do comando **input()** será sempre uma **string**.
- Para retornar dados do tipo numérico é necessário converter para **int** (inteiro) ou **float** (real)
- Exemplos:

```
>>> num = int(input("Entre com um número: "))
Entre com um número: 147
>>> num
147
```


Relembrando: Comando de Saída

- O comando de saída é utilizado para que o sistema forneça, numa **unidade de saída**, os **resultados** do processamento e **mensagens**.



Digite sua idade em anos: 32

Você nasceu em 1980.

Programa em execução

dados de entrada

dados de saída

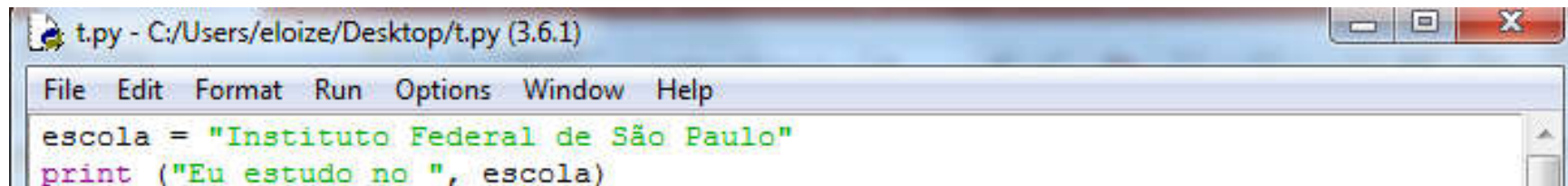
Relembrando: Comando de Saída (cont.)

- **Resultados de Processamento:** podem ser fornecidos através de conteúdos de **variáveis**, conteúdos de **constantes** e resultados de **expressões aritméticas** e **lógicas**.
- **Mensagens:** são utilizadas para que o programa dê **informações** ao usuário a respeito do **processamento** que está se realizando. Podem ser fornecidas através de conteúdo de **variáveis**, ou constantes do tipo **string** ou da **mensagem** propriamente dita.

Comando de Saída em Python – print()

- O comando print() é usado para imprimir dados (strings ou conteúdos de variáveis) na tela
- Exemplos:

```
>>> print('Eu adoro programar!')  
Eu adoro programar!
```

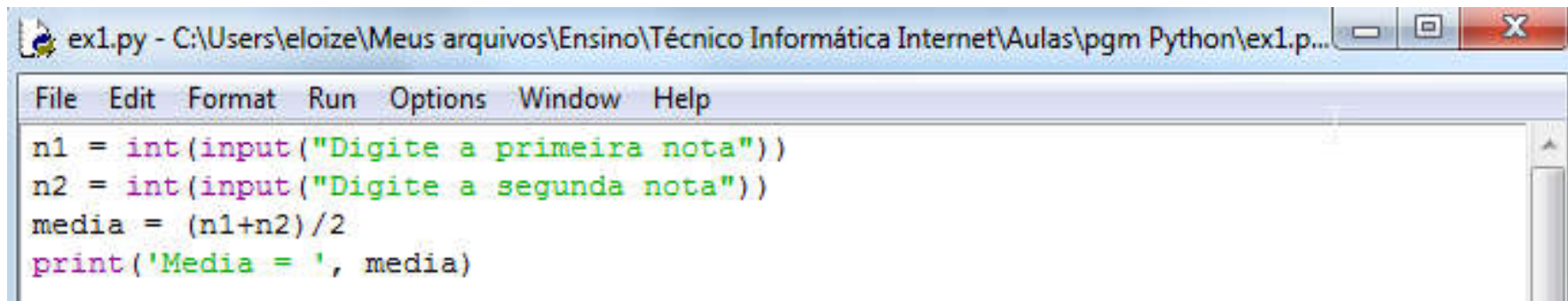


A screenshot of a Python IDE window titled 't.py - C:/Users/eloize/Desktop/t.py (3.6.1)'. The window has a menu bar with 'File', 'Edit', 'Format', 'Run', 'Options', 'Window', and 'Help'. The code editor shows the following Python code:

```
escola = "Instituto Federal de São Paulo"  
print ("Eu estudo no ", escola)
```

Exemplo de Programa em Python

- Programa para calcular a média de um aluno a partir de 2 notas de provas



The image shows a screenshot of a Python script editor window. The title bar reads "ex1.py - C:\Users\eloize\Meus arquivos\Ensino\Técnico Informática Internet\Aulas\pgm Python\ex1.p...". The menu bar includes "File", "Edit", "Format", "Run", "Options", "Window", and "Help". The code in the editor is as follows:

```
n1 = int(input("Digite a primeira nota"))
n2 = int(input("Digite a segunda nota"))
media = (n1+n2)/2
print('Media = ', media)
```

Relembrando: Expressão Aritmética

- Denomina-se **expressão aritmética** aquela cujos **operadores** são **aritméticos** e cujos operandos são constantes ou variáveis do tipo numérico (inteiro ou real).

- Exemplo:

$$\frac{45.6}{A} - |D - \cos(B)|$$

Operadores Aritméticos em Python

+	adição
-	subtração
*	multiplicação
/	divisão
%	resto da divisão
**	potência

Relembrando: Expressão Lógica

- Denomina-se expressão lógica aquela cujos operadores são **lógicos** ou **relacionais** e cujos operandos são relações ou variáveis ou constantes do tipo lógico.
- Exemplo: $(A+B == 0) \text{ and } (C \neq 1)$

O que são Operadores Relacionais?
E Operadores Lógicos?

Relembrando: Operadores relacionais

- São usados para fazer comparações entre dois valores de mesmo tipo primitivo.
- Tais valores são representados por constantes, variáveis ou expressões aritméticas
- O resultado obtido é sempre um **valor lógico**.

Operadores Relacionais em Python

<	Menor que
<=	Menor ou igual
>	Maior que
>=	Maior ou igual
==	Igual
!=	Diferente

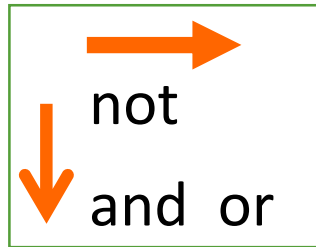
Operadores Lógicos em Python

- Utilizam-se três conectivos básicos para a formação de novas proposições lógicas compostas a partir de outras proposições lógicas simples:

and	e
or	ou
not	não

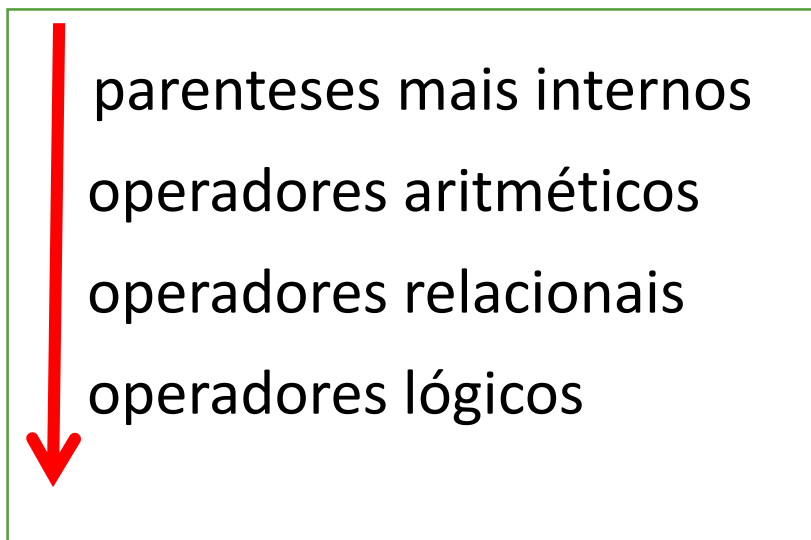
Operadores Lógicos em Python

- Na resolução das expressões lógicas, os operadores lógicos guardam uma hierarquia entre si:



ATENÇÃO:

- Na resolução das expressões lógicas, os diversos operadores guardam uma hierarquia entre si:

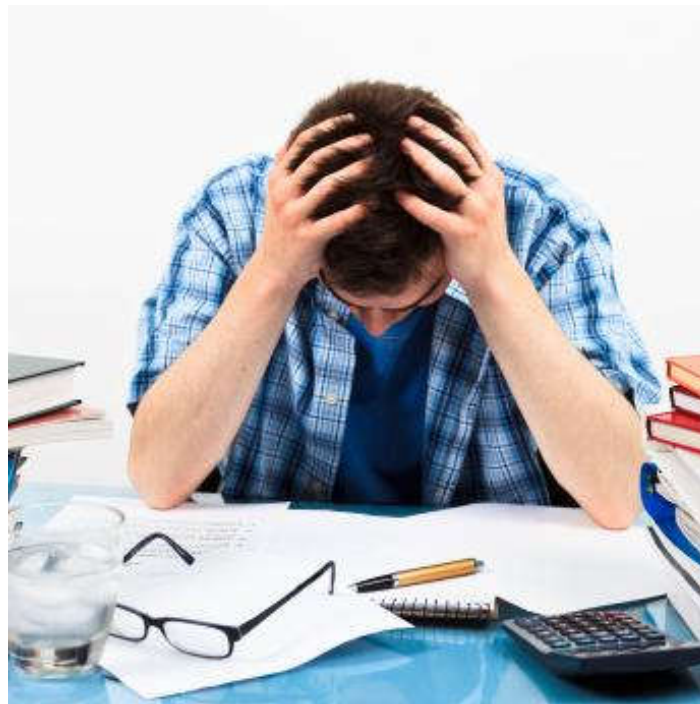


Expressão Lógica:

- Exemplo:

$$\begin{aligned}
 &\text{not} (\quad 5 \neq \underline{10 / 2} \quad \text{or } V \quad \text{and} \quad \underline{2 - 5} > \underline{5 - 2} \quad \text{or } V) \\
 &\text{not} (\quad 5 \neq \quad 5 \quad \text{or } V \quad \text{and} \quad \underline{-3} > \underline{3} \quad \text{or } V) \\
 &\text{not} (\quad F \quad \text{or } V \quad \text{and} \quad \quad F \quad \text{or } V) \\
 &\text{not} (\quad F \quad \text{or} \quad F \quad \text{or } V) \\
 &\text{not} (\quad F \quad \text{or } V) \\
 &\text{not} (\quad V \quad) \\
 &\quad F
 \end{aligned}$$

Exercícios



Exercícios

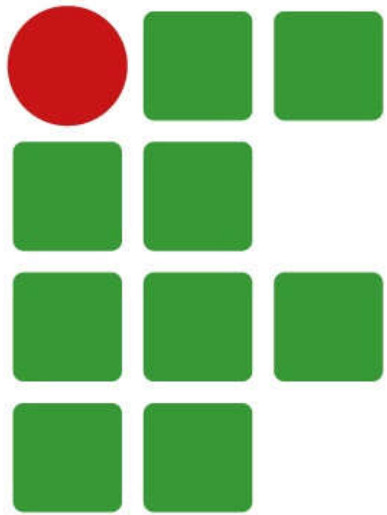
1. Faça um programa em Python que leia um número X do usuário e escreva ele na tela no seguinte formato: “O número escolhido foi X”
2. Faça um programa em Python que leia do usuário dois números. Faça a multiplicação dos dois números e mostre o resultado.
3. Faça um programa que leia do usuário um número e escreva o seu sucessor e o seu antecessor.
4. Faça um programa que leia 2 notas de um aluno, onde a primeira nota possui peso um, a segunda possui peso dois. Calcule a média ponderada do aluno baseada nos pesos e exiba.

Exercícios (cont.)

5. Faça um programa que receba dois inteiros x e y e calcule o valor de z:

$$z = \frac{(x^2 + y^2)}{(x - y)^2}$$

6. Faça um programa que receba o salário de um funcionário, reajusta o salário em 25% e apresenta o novo salário após o reajuste.



INSTITUTO FEDERAL

São Paulo

Câmpus São Carlos