



# **PARADIGMAS DE LINGUAGENS DE PROGRAMAÇÃO EM PYTHON**

## **VARIÁVEIS EM PYTHON E O BÁSICO DA LINGUAGEM**

Prof. Dr. Daniel Caetano

2020 - 2

# Compreendendo do problema

- **Missão:** desenvolver software para um drone
  - Giroscópio, acelerômetro e barômetro
  - Sensores: informação momentânea



- 0 e 1 são suficientes para tudo?



<https://www.menti.com/>

# Compreendendo do problema

- **Missão:** desenvolver software para um drone
  - Giroscópio, acelerômetro e barômetro
  - Importante: precisão dos dados



ON



OFF



Mentimeter

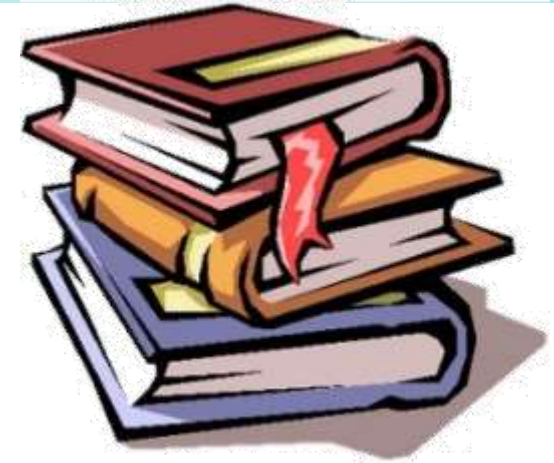
- Existe melhor forma de armazenar? <https://www.menti.com/>

# Objetivos

- Compreender o conceito de variável
  - Compreender a declaração de variáveis
  - Conhecer as bases da linguagem Python
  - Compreender os problemas dos binários
  - Treinar construção de programas
- 
- **Desafio Aula 04**



# Bibliografia da Aula



---

## Material

## Acesso ao Material

Apresentação

<https://www.caetano.eng.br/>  
(Paradigmas de Programação – Aula 4)

Livro Texto

Capítulo 5, páginas 197 a 215

Aprenda Mais!

- Texto: “As falhas numéricas que podem causar desastres”

[https://www.bbc.com/portuguese/noticias/2015/05/150513\\_vert\\_fut\\_bug\\_digital\\_ml#:~:t](https://www.bbc.com/portuguese/noticias/2015/05/150513_vert_fut_bug_digital_ml#:~:t)

# A MEMÓRIA DO COMPUTADOR



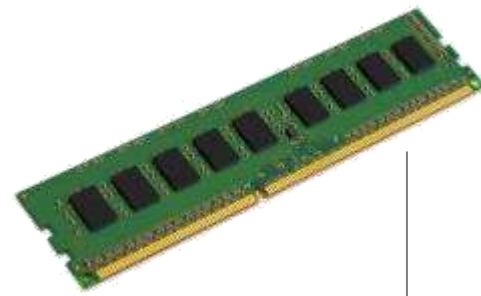
Mentimeter

<https://www.menti.com/>

Prof. Dr. Daniel Caetano

# As Memórias do Computador

- Armazenamento Temporário
  - Memória Principal / Memória Cache
  - RAM: Random Access Memory (escrita e leitura)



- Armazenamento “Permanente”
  - ROM: Read Only Memory (Só escrita)
  - Discos Magnéticos (HDDs)
  - Disco Ópticos (CDs, DVDs...)
  - Flash RAM (SDs, SSDs etc)
  - Legados (fitas magnéticas...)



# A Memória Principal

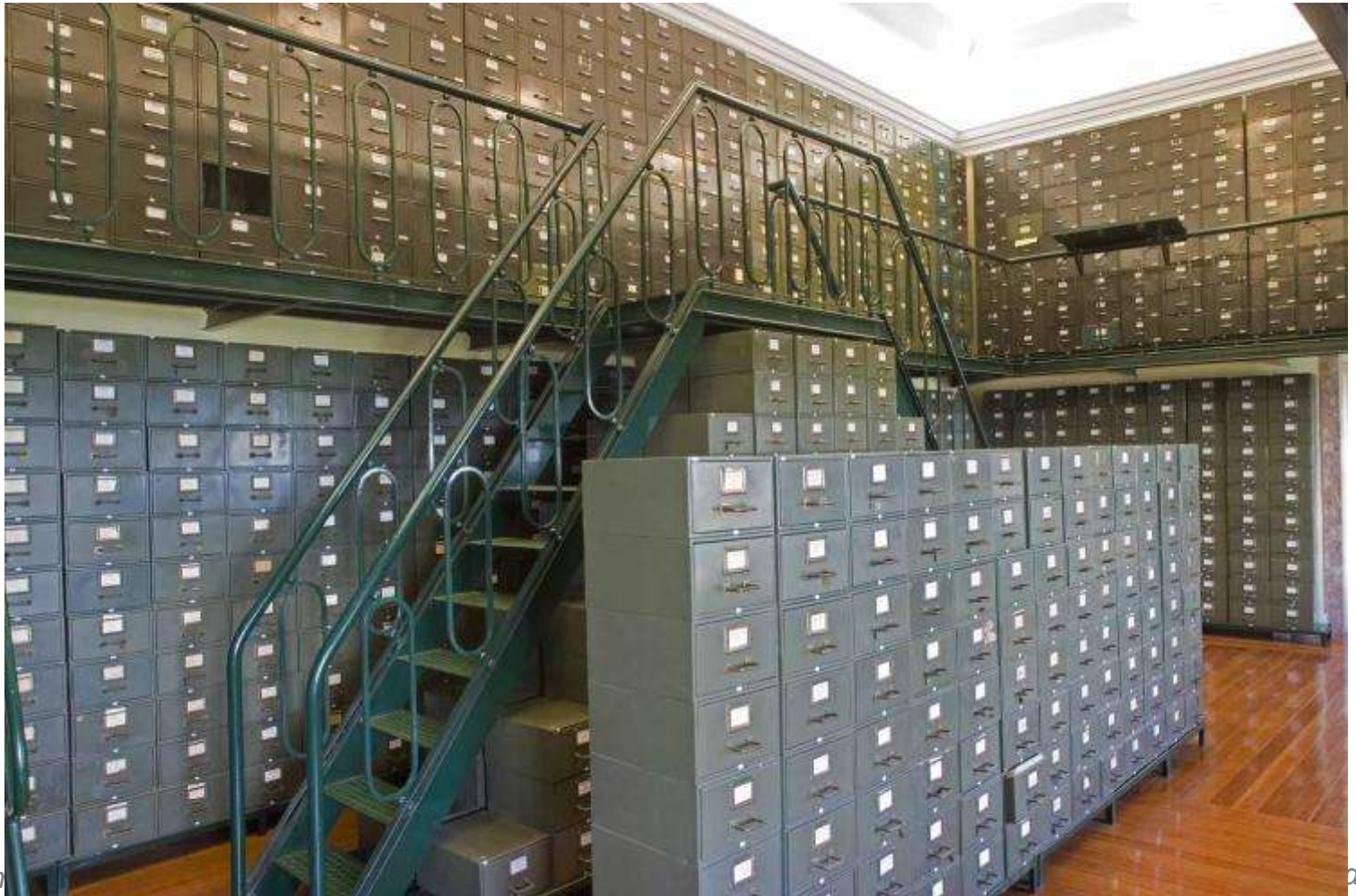
- A memória é como um arquivo de fichas
- Cada gaveta é chamada **posição de memória**
- Em cada uma cabe um número fixo de “dígitos”
- Cada posição de memória é identificada por um número, o **endereço de memória**



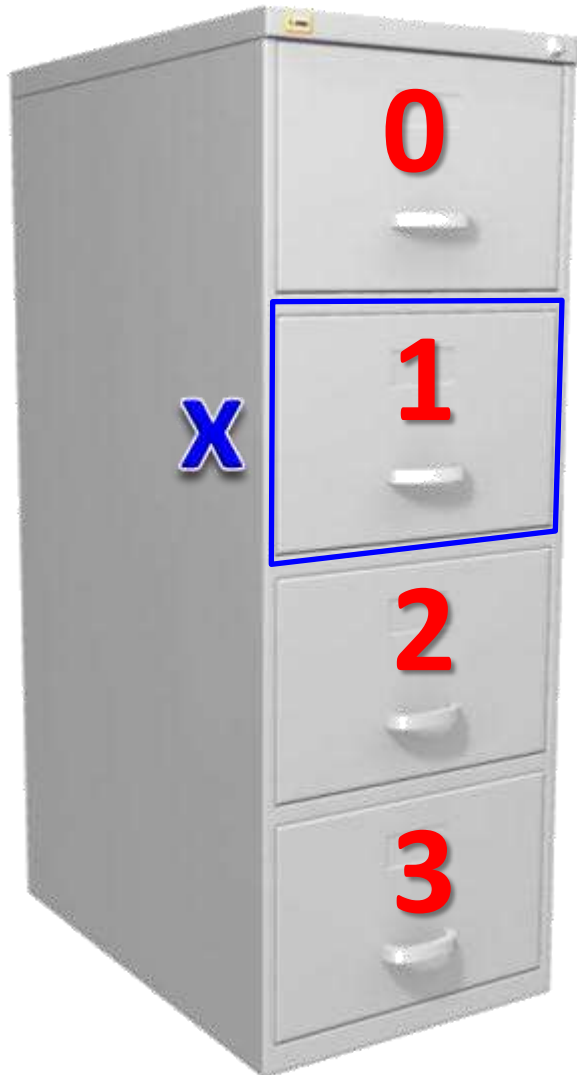


# A Memória Principal

- A memória principal (RAM) é...



# “Apelidos” para Endereços



- **Variáveis** representam essas gavetas
- As variáveis recebem nomes, como “**x**”, para não termos que decorar seu endereço
- **Declaração de Variável** é o processo de reservar gavetas e dar nomes a elas.



# **VARIÁVEIS: GUARDANDO DADOS NA MEMÓRIA DO COMPUTADOR**

# Identificadores de Variáveis (Python)

Há regras para os NOMES das variáveis:

1. SEM espaços
2. Há algumas palavras “proibidas”  
(print, por exemplo)
3. Não iniciar com números

São nome válidos?

- ✓ nome
- ✓ IDADE
- ✗ data de nascimento
- ✓ limite1
- ✗ 1dado

- Como lidar com espaços?
  - Sublinhados  
data\_de\_nascimento
  - Notação Camelo (Camel Case)  
dataDeNascimento

# Criando variáveis: tipos

- Linguagens em geral: bem mais que 0s e 1s...
  - Inteiros
  - Reais
  - Lógicos (booleanos)
  - Texto (strings)



**Falaremos mais sobre tipos no futuro!**

- Declaração explícita ou implícita
- **Acompanhe: criação de variáveis** (C++/Python)

# Vinculação de Tipos

- Fortemente Tipadas
  - Dados armazenados: possuem um tipo específico
  - Operação c/ tipos misturados: possível problema.
- Fracamente Tipadas
  - Dados armazenados: são “genéricos”
  - Operações c/ tipos misturados: auto conversões.



# Momento da Vinculação de Tipo

- Vinculação de Tipo Estática
  - O tipo da variável é definido e verificado em tempo de compilação
  - Tipo da variável não pode mudar.
- Vinculação de Tipo Dinâmica
  - O tipo da variável é definido e verificado em tempo de execução
  - Tipo da variável pode mudar.

Estático



Dinâmico

# Tempo de Vida da Variável

- Variáveis **Estáticas**:
  - Posição é definida antes da execução do programa
    - “Existem” durante todo o tempo de execução.
- Variáveis **Dinâmicas**:
  - **De pilha**: tipo definido na compilação, mas espaço durante a execução, no momento da declaração.
    - Existem de acordo com o escopo
  - **Explícitas**: tipo e espaço alocado durante a execução
    - Existem até que sejam liberadas (manual x coleta de lixo)
  - **Implícitas**: tipo e espaço alocado ao atribuir valores
    - Existem de acordo com escopo e uso (coleta de lixo).





# COMO OS NÚMEROS SÃO ARMAZENADOS

# Números no Computador

- Vimos que computador trabalha em “binário”



0101001010111b

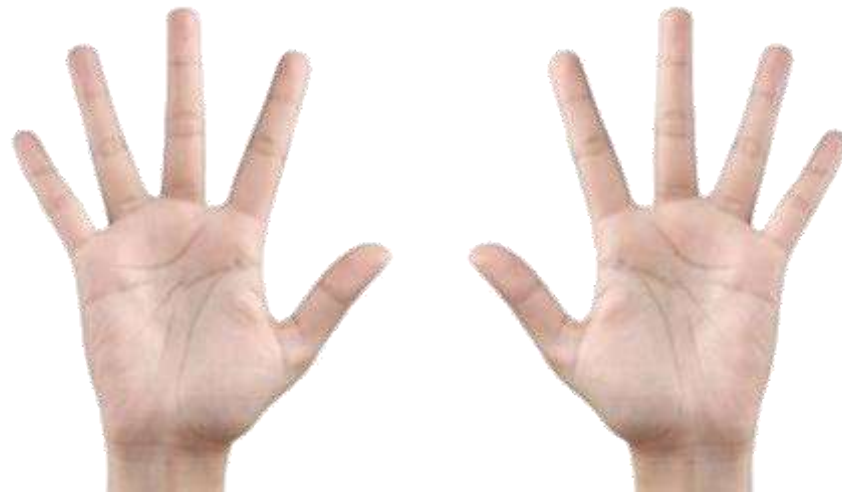
- Por que ele trabalha em binário?
- Qual a consequência disso?



<https://www.menti.com/>

# Humanos x Processadores

- Humanos aprendem a contar com os dedos;
- Quantos dedos temos nas mãos?



- Nossa contagem usa o sistema **DECIMAL**
- Cada **dígito**: “ocupado” por 1 de 10 símbolos:  
**0, 1, 2, 3, 4, 5, 6, 7, 8, 9**

# Humanos x Processadores

- Como indicar  $n^{os}$  decimais para o processador?



# Humanos x Processadores

- Como indicar  $n^{os}$  decimais para o processador?

**FIOS**



# Humanos x Processadores

- Como indicar  $n^{\text{os}}$  decimais para o processador?



0

# Humanos x Processadores

- Como indicar  $n^{\text{os}}$  decimais para o processador?



**1**

# Humanos x Processadores

- Como indicar  $n^{\text{os}}$  decimais para o processador?



2



# Humanos x Processadores

- Como indicar  $n^{\text{os}}$  decimais para o processador?



**3**

# Humanos x Processadores

- Como indicar  $n^{\text{os}}$  decimais para o processador?



4

# Humanos x Processadores

- Como indicar  $n^{\text{os}}$  decimais para o processador?



5

# Humanos x Processadores

- Como indicar  $n^{\text{os}}$  decimais para o processador?



**94.614**

# Como saber isso?

- Cada fio: um “dígito” numérico, chamado **bit**
- Esse fio pode estar **desligado** ou **ligado**
- Com fio “desligado” → 0 e fio “ligado” → **1**, temos o número que o computador entende:



# Como saber isso?

- Cada fio: um “dígito” numérico chamado **bit**
- Esse fio pode estar **desligado** ou **ligado**



**10111000110010110b = 94.614**

10111000110010110



**94.614**



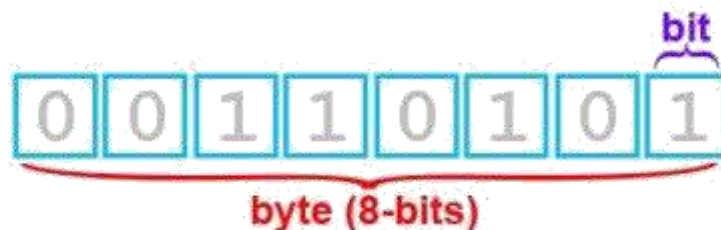
# Os Números Binários

- Como cada **bit** pode ser apenas 0 ou 1...



...o nome dessa representação é “**binária**”.

- Um único bit armazena pouca informação
- Usualmente, os bits aparecem agrupados



# Os múltiplos e submúltiplos

- Memória do computador: muitos dados!



- Quibibyte (**KiB**): 1024 bytes
- Mebibyte (**MiB**): 1024 KiB (~ 1mi de bytes)
- Gibibyte (**GiB**): 1024 MiB (~ 1bi de bytes)
- Tebibyte (**TiB**): 1024 GiB (~ 1tri de bytes)
- Pebibyte (**PiB**): 1024 TiB (~ 1tetra de bytes)



# Os múltiplos e submúltiplos

- Memória permanente: notação convencional



- Quilobyte (**KB**): 1000 bytes
- Megabyte (**MB**): 1000 KB (= 1mi de bytes)
- Gigabyte (**GB**): 1000 MB (= 1bi de bytes)
- Terabyte (**TB**): 1000 GB (= 1tri de bytes)
- Petabyte (**PB**): 1000 TB (= 1tetra de bytes)

# Qual o Problema com os Binários?

- **Inteiros:** representados em binário exato
- **Fracionários:** nem sempre binários exatos
- Ex.: 0,1 decimal, em binário fica...:

0,000110011001100110011001100110011001100110011...

- Mas o computador guarda infinitas casas?
- **NÃO!**
  - Nem em números inteiros...
  - Nem em números fracionários!

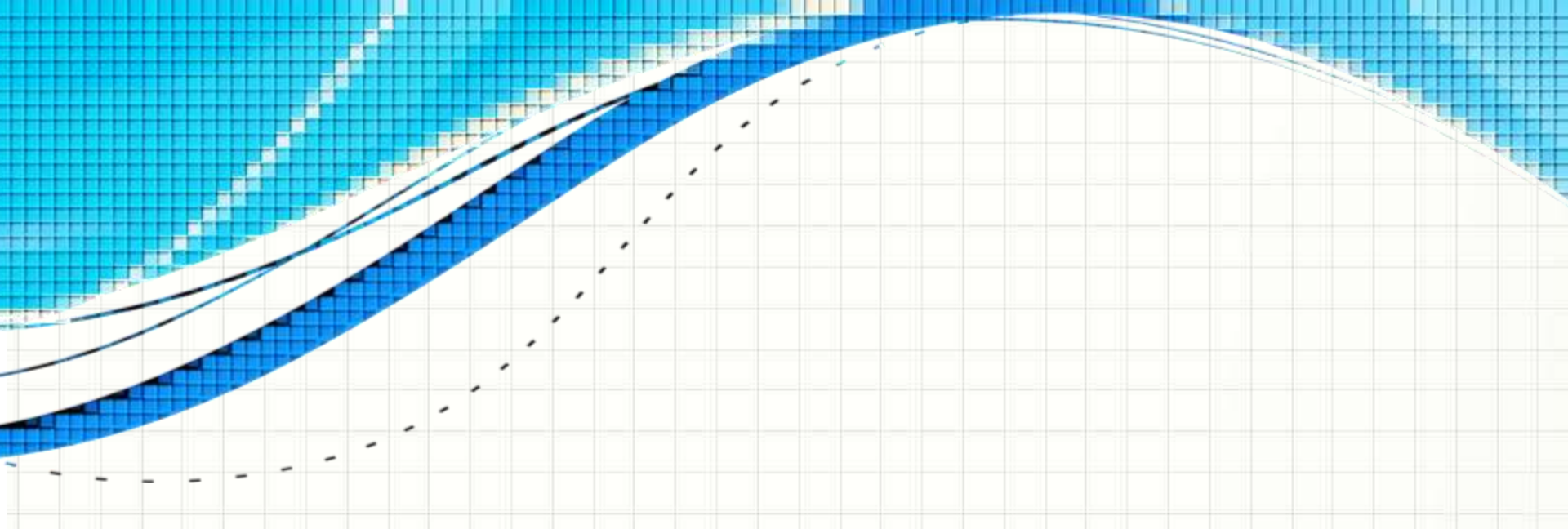


# Qual o Problema com os Binários?

- Se ele guardar apenas 16 bits, por exemplo...  
0,0001100110011001100110011001100110011001100110011...
- Que em decimal é... 0,099976  
0,1  $\neq$  0,09976
- Problemas!
  - Valores fracionários (reais) ocupam mais espaço
  - Valores fracionários não são exatos: erros!



**Artigo: Erros que Causam Desastres**



# **CONHECENDO O BÁSICO DA LINGUAGEM PYTHON**

# Usando Python com o IDLE

- Permite digitar e testar comandos diretamente
- Digite **a parte indicada em azul:**

IDLE

```
>>> print("Alô mundo!")
```

Aperte a tecla Enter



<https://www.python.org/shell/>

# Usando Python com o IDLE

- Permite digitar e testar comandos diretamente
- Tente agora:

Aperte a tecla Enter

IDLE

```
>>> Print("Alô mundo!")
```

Por enquanto, não  
coloque espaços antes  
do do comando!

**Python diferencia maiúsculas de  
minúsculas!**

# Usando Python com o IDLE

- Permite digitar e testar comandos diretamente
- Tente agora:

Não esqueça do Enter

IDLE

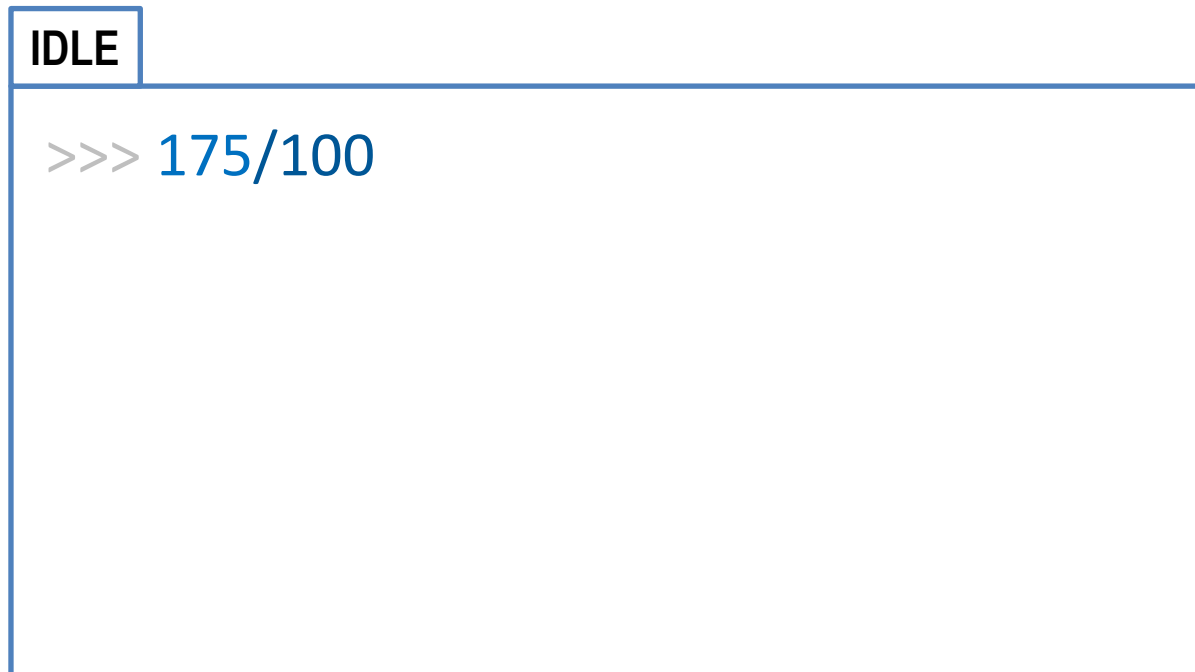
```
>>> 2+2
```

## Operadores:

Soma:	+	Divisão “para baixo”:	//
Subtração:	-	Resto da Divisão:	%
Multiplicação:	*	Potência:	**
Divisão:	/	Parênteses	()

# Usando Python com o IDLE

- Permite digitar e testar comandos diretamente
- Tente agora:

A screenshot of the Python IDLE (Integrated Development and Learning Environment) shell. The window title is "IDLE". The prompt ">>>" is followed by the expression "175/100".

```
IDLE
>>> 175/100
```



# Usando Python com o IDLE

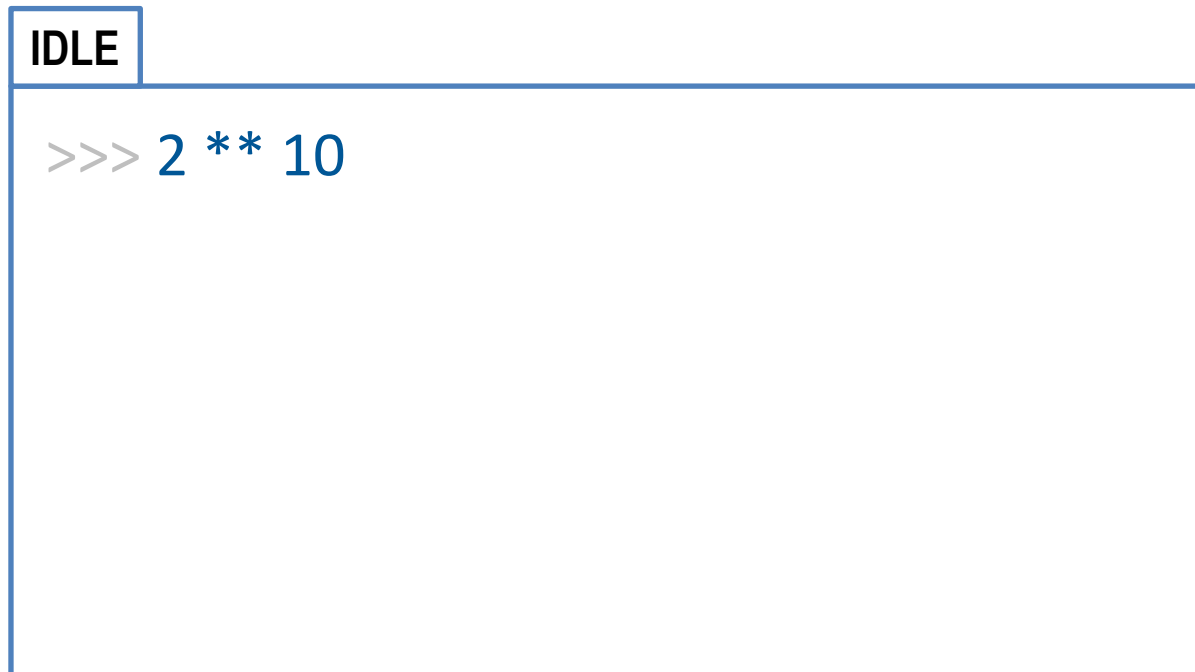
- Permite digitar e testar comandos diretamente
- Tente agora:

IDLE

```
>>> 175//100
```

# Usando Python com o IDLE

- Permite digitar e testar comandos diretamente
- Tente agora:

A screenshot of the Python IDLE (Integrated Development and Learning Environment) shell. The window title is 'IDLE'. The prompt '>>>' is followed by the expression '2 \*\* 10'.

```
IDLE
>>> 2 ** 10
```

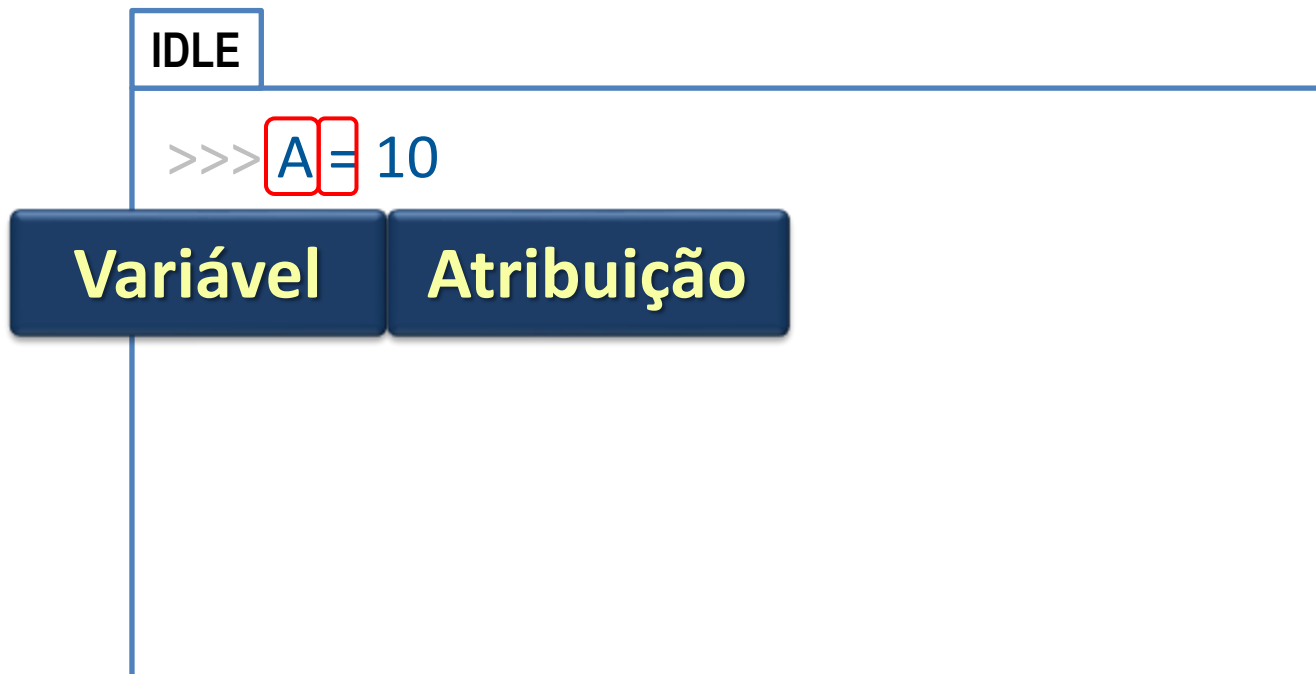
# Usando Python com o IDLE

- Permite digitar e testar comandos diretamente
- Tente agora:

```
IDLE
>>> 2 ** 10000
```

# Usando Python com o IDLE

- Permite digitar e testar comandos diretamente
- Guardando um valor em uma variável



# Usando Python com o IDLE

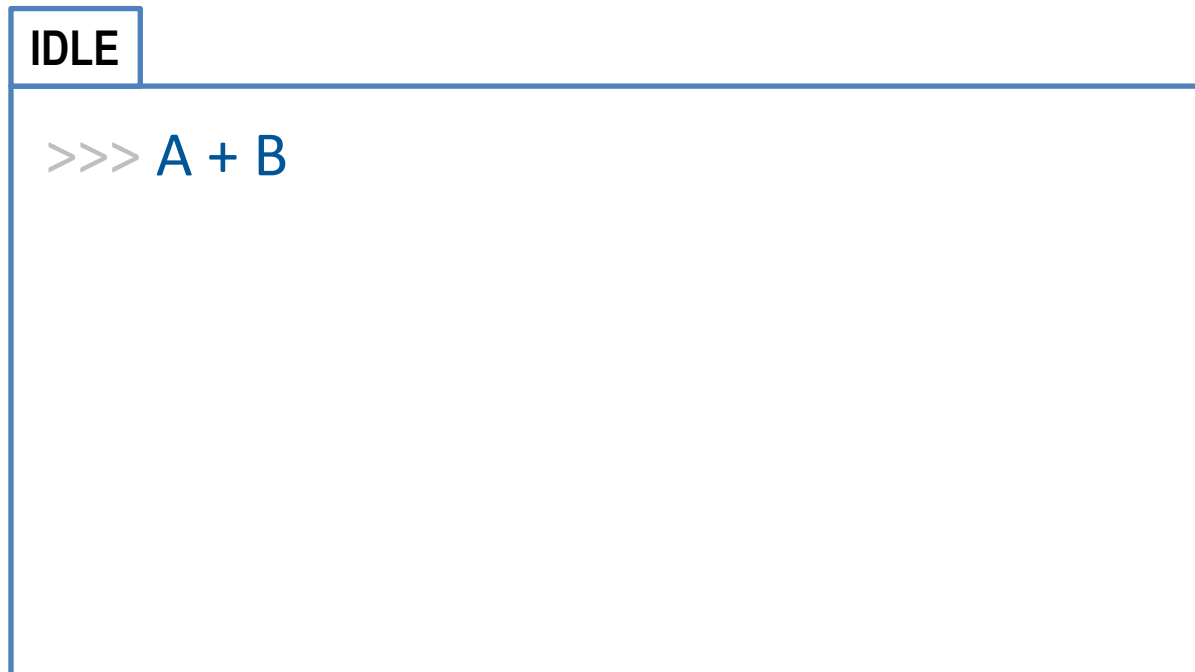
- Permite digitar e testar comandos diretamente
- Guardando outro valor em uma variável

IDLE

```
>>> B = 2.5
```

# Usando Python com o IDLE

- Permite digitar e testar comandos diretamente
- Fazendo contas com variáveis

A screenshot of the Python IDLE (Integrated Development and Learning Environment) shell. The window title is 'IDLE'. The prompt '>>>' is followed by the expression 'A + B'.

```
IDLE
>>> A + B
```

# Usando Python com o IDLE

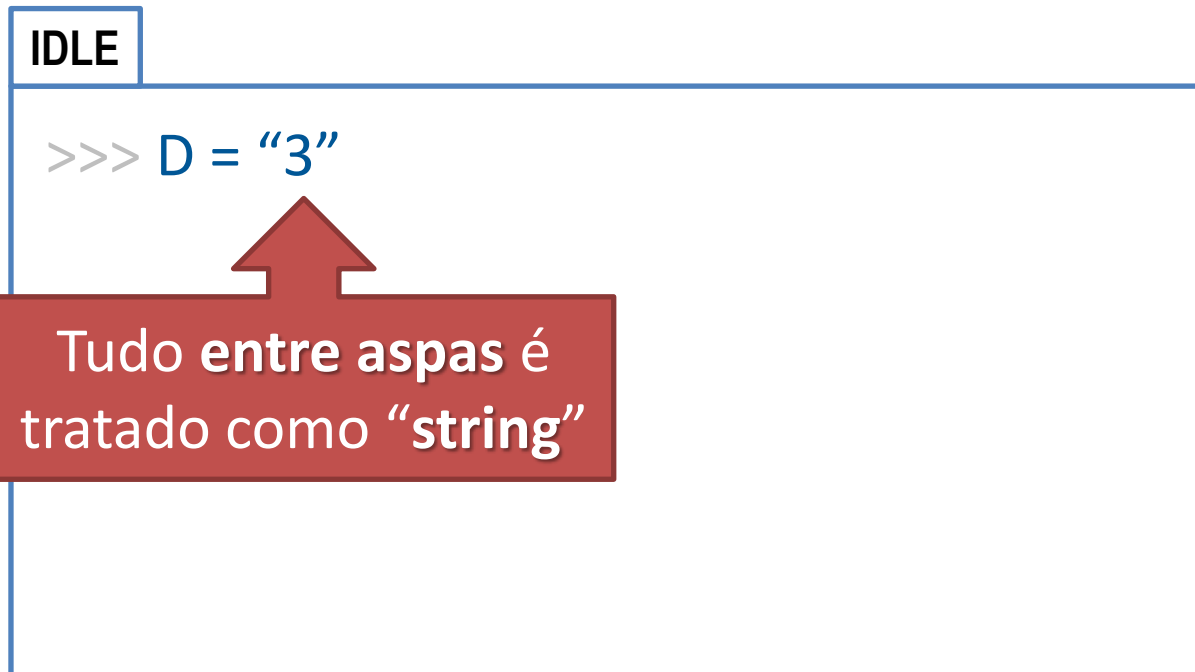
- Permite digitar e testar comandos diretamente
- Guardando um texto em uma variável

IDLE

```
>>> C = "Professor"
```

# Usando Python com o IDLE

- Permite digitar e testar comandos diretamente
- Guardando um texto em uma variável



The image shows a screenshot of the Python IDLE (Integrated Development and Learning Environment) shell. The window title is "IDLE". The prompt is ">>>". The code entered is `D = "3"`. A red callout box with a white border and a red arrow pointing to the string "3" contains the text: "Tudo entre aspas é tratado como 'string'".

```
IDLE
>>> D = "3"
```

Tudo entre aspas é tratado como "string"



# Usando Python com o IDLE

- Permite digitar e testar comandos diretamente
- Tente essa, agora...

A screenshot of the Python IDLE (Integrated Development and Learning Environment) shell. The window title is 'IDLE'. The prompt '>>>' is followed by the command 'C + D'. The text is in a blue monospace font on a white background.

```
IDLE
>>> C + D
```

Ao somar duas strings, o Python **concatena** seus conteúdos

# Usando Python com o IDLE

- Permite digitar e testar comandos diretamente
- E essa?

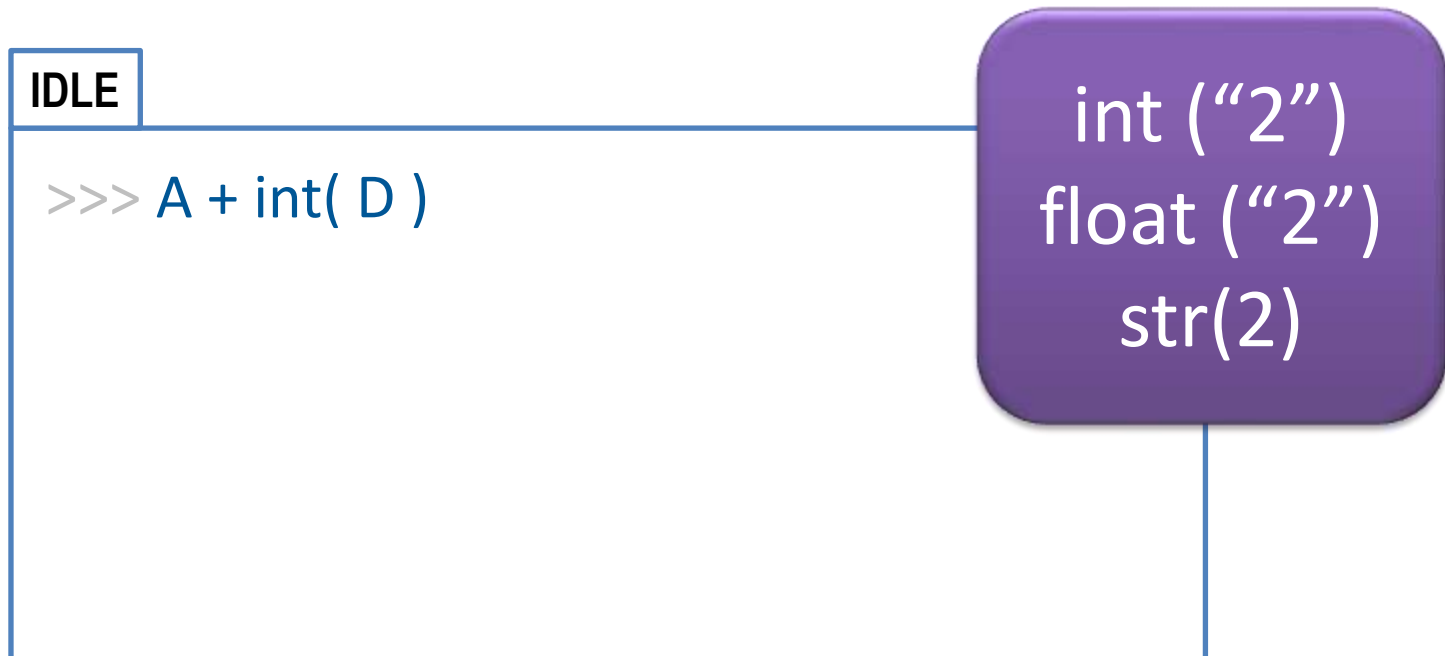


```
IDLE
>>> A + D
```

**Python não permite somar números com strings diretamente**

# Usando Python com o IDLE

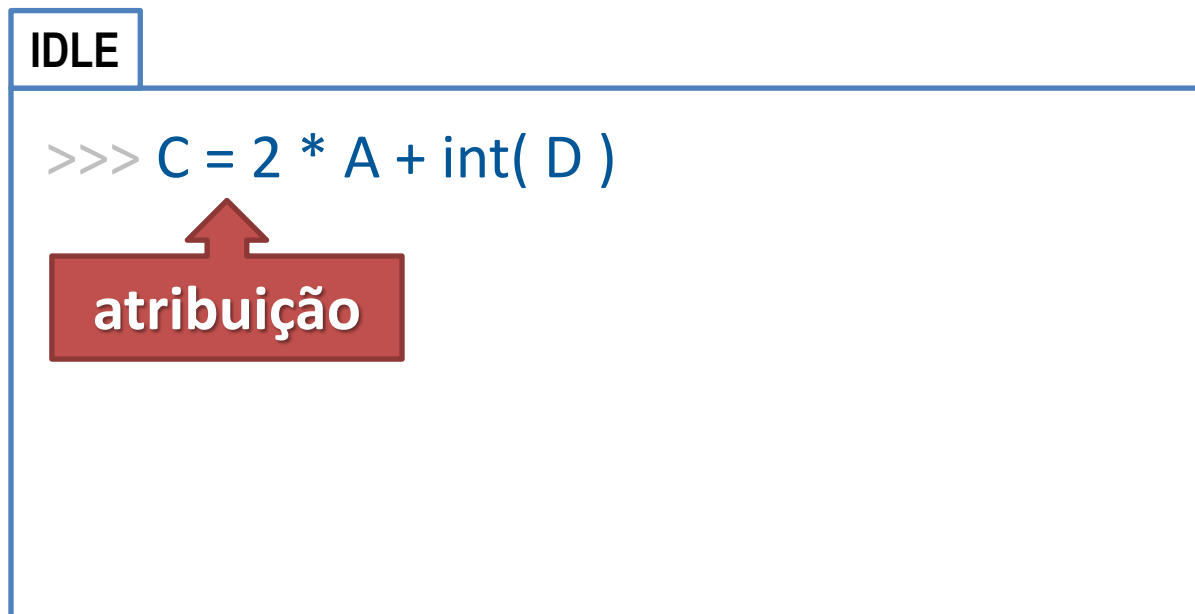
- Permite digitar e testar comandos diretamente
- Vamos aprender a transformar as coisas....



**`int( "texto" )` converte a string "texto" para um número inteiro**

# Usando Python com o IDLE

- Permite digitar e testar comandos diretamente
- Vamos guardar um resultado...



The image shows a screenshot of the Python IDLE (Integrated Development and Learning Environment) shell. The window title is "IDLE". The prompt is ">>>". The code entered is "C = 2 \* A + int( D )". A red arrow points from a red box containing the word "atribuição" (assignment) to the equals sign in the code.

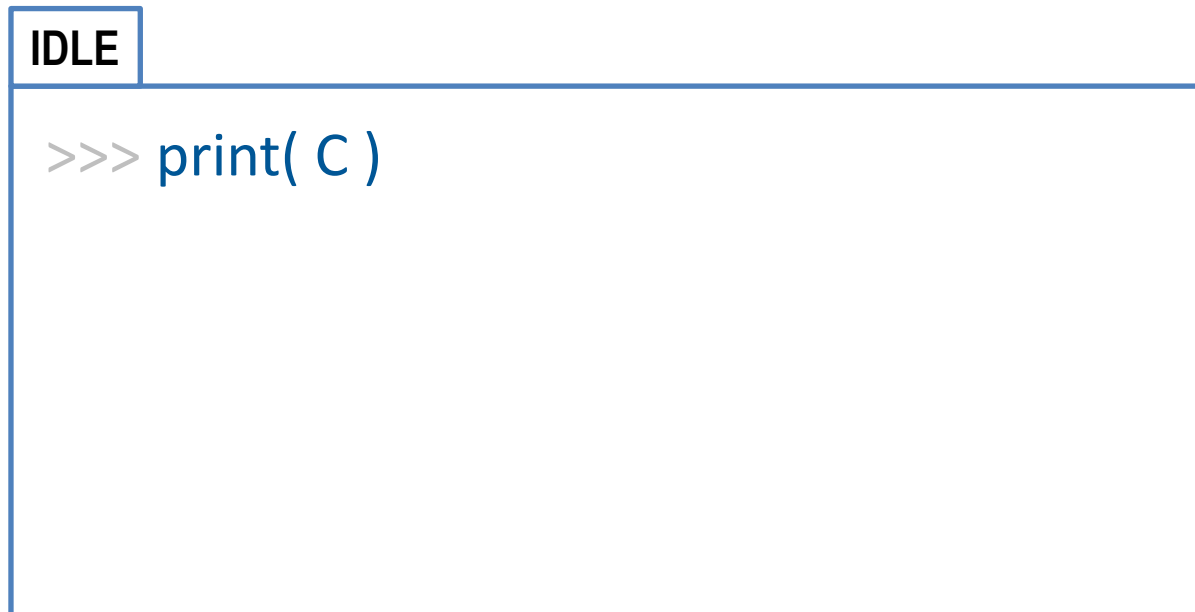
```
IDLE
>>> C = 2 * A + int( D )
```

atribuição

Posso armazenar um **resultado** para uso posterior

# Usando Python com o IDLE

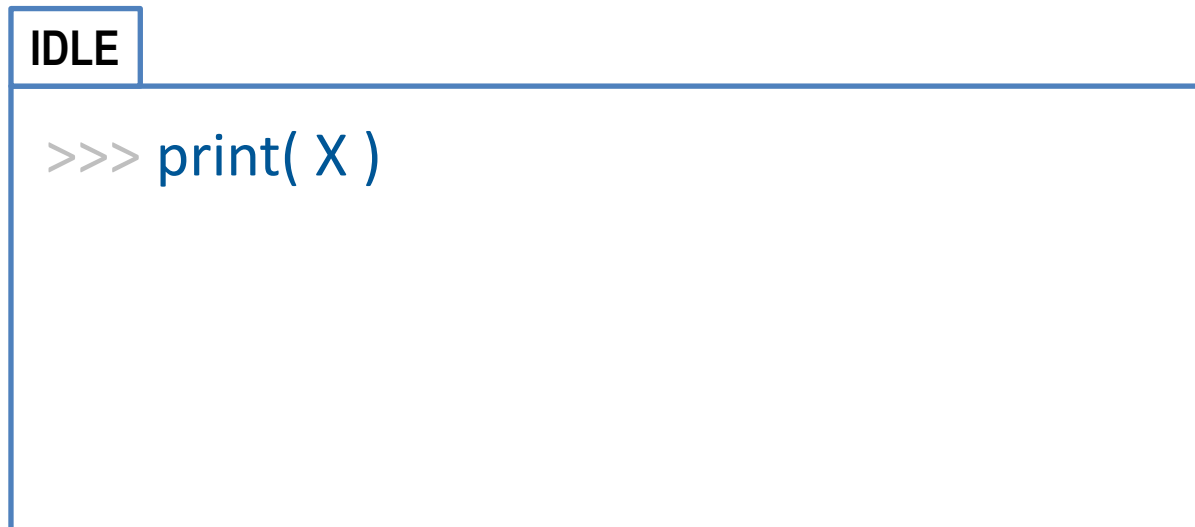
- Permite digitar e testar comandos diretamente
- Mostrando o resultado anterior

A screenshot of the Python IDLE (Integrated Development and Learning Environment) shell window. The window title bar at the top left says "IDLE". The main area of the window contains the text ">>> print( C )" in a blue monospace font, representing a command entered in the Python interpreter.

**print** mostra valores na tela

# Usando Python com o IDLE

- Permite digitar e testar comandos diretamente
- É possível mostrar uma variável sem valor?



The image shows a screenshot of the Python IDLE (Integrated Development and Learning Environment) shell. The window title is 'IDLE'. The prompt '>>>' is followed by the command 'print( X )'. This command is syntactically correct but will result in a NameError because the variable 'X' has not been defined.

```
IDLE
>>> print( X )
```

**Em Python, Só podemos usar valores de variáveis que foram declaradas!**

# Usando Python com o IDLE

- Permite digitar e testar comandos diretamente
- É possível mostrar uma variável sem valor?

IDLE

```
>>> X = 12  
>>> print( X )
```

O primeiro valor que guardamos em uma **variável** é a “**declaração**” da variável

Aperte a tecla **Enter** ao fim de cada linha

# Usando Python com o IDLE

- Permite digitar e testar comandos diretamente
- É possível mostrar uma variável sem valor?

IDLE

```
>>> print( abacaxi )
```

No caso, o Python entende **abacaxi** como uma **variável não declarada**



# Usando Python com o IDLE

- Permite digitar e testar comandos diretamente
- É possível mostrar uma variável sem valor?

IDLE

```
>>> print( "abacaxi" )
```

No caso, o Python entende que deve **reproduzir a string "abacaxi"** na saída

# Usando Python com o IDLE

- Permite digitar e testar comandos diretamente
- Dá pra mostrar várias coisas ao mesmo tempo?

IDLE

```
>>> print( "3*9 vale: ", 3*9 )
```

**Usamos vírgulas para imprimir vários valores om um único `print`**

# Usando Python com o IDLE

- Permite digitar e testar comandos diretamente
- Combinando outros recursos com o **print**

IDLE

```
>>> print( "3*", C, "vale:", 3*C )
```

**O print é um comando bastante completo!**

# Usando Python com o IDLE

- Permite digitar e testar comandos diretamente
- Como pedir para o usuário digitar um texto?

IDLE

```
>>> input( "Digite seu nome: ")
```

**O `input` faz a pergunta, mas não guarda o valor automaticamente!**

# Usando Python com o IDLE

- Permite digitar e testar comandos diretamente
- Guardando o texto digitado pelo usuário

IDLE

```
>>> nome = input( "Digite seu nome: ")
```

# Usando Python com o IDLE

- Permite digitar e testar comandos diretamente
- Guardando um **valor** digitado pelo usuário

IDLE

```
>>> idade = input( "Digite sua idade: ")
```

# Usando Python com o IDLE

- Permite digitar e testar comandos diretamente
- Fazendo contas com o valor digitado

IDLE

```
>>> idade = idade + 1
```

Como resolver  
esse problema?

# Usando Python com o IDLE

- Permite digitar e testar comandos diretamente
- Guardando um **número** digitado pelo usuário

IDLE

```
>>> idade = int( input( "Digite sua idade: ") )
```



# Usando Python com o IDLE

- Permite digitar e testar comandos diretamente
- Fazendo contas com o valor digitado

IDLE

```
>>> idade = idade + 1
```

# Usando Python com o IDLE

- Permite digitar e testar comandos diretamente
- Fazendo contas com o valor digitado

IDLE

```
>>> idade = idade + 1  
>>> print (idade)
```

# Usando Python com o IDLE

- Instruções/Comandos que vimos:
  - Operações aritméticas e atribuições
  - Mostrar dados: `print`
  - Receber dados: `input`
  - Converter dados: `int`, `float`, `str`
- Você teve dificuldade com algum deles?
  - Qual?



Mentimeter

<https://www.menti.com/>



# **ESCREVENDO UM PROGRAMA EM PYTHON**

# Programas em Python

- Console do IDLE: executar comandos simples
- Como definir a sequência lógica?
  - Criando um arquivo de programa

A screenshot of the Python IDLE editor window. The title bar reads 'Teste.py - C:/Users/MarioRaul/Desktop/Teste.py (3.5.1)'. The menu bar includes 'File', 'Edit', 'Format', 'Run', 'Options', 'Window', and 'Help'. The code editor contains the following Python code:

```
from math import sqrt

a = 10
b = 4

c = sqrt(a*b)

print('Valor de a: ' + str(a))
print('Valor de b: ' + str(b))
print('Raiz de a*b: ' + str(c))
```

# Digitando o Primeiro Programa

- O arquivo tem um nome com extensão **.py**
  - aula04ex01.py
  - **File > Save As...**

aula04ex01.py

```
# Imprime Olá  
print("Olá mundo!")
```

**Comentário:** essa linha será ignorada!

**Run > Run Module**  
**F5**

# Criando outro Programa

- aula04ex02.py
  - File > Save As...

Aula04ex02.py

```
# Lê o nome
PNOME = input("Nome?")
SNOME = input("Sobrenome?")
NOME = PNOME + SNOME
print("Nome:", NOME)
```

**Como arrumar?**

# Criando outro Programa

- aula04ex02a.py
  - **File > Save As...**

Aula04ex02a.py

```
# Lê o nome  
PNOME = input("Nome?")  
SNOME = input("Sobrenome?")  
NOME = PNOME + " " + SNOME  
print("Nome:", NOME)
```

**Observe a mudança!**



# Criando outro Programa

- aula04ex03.py
  - **File > Save As...**

Aula04ex03.py

```
# Soma 2 Números
```

```
N1 = int( input("Digite um No:") )
```

```
N2 = int( input("Digite outro No:") )
```

```
S = N1 + N2;
```

```
print("Soma: ", S)
```

# Criando outro Programa

- aula04ex04.py
  - **File > Save As...**

Aula04ex04.py

```
# Calcula IMC
```

```
P = float( input("Digite peso (Kg):"))
```

```
A = float( input("Digite altura (m):"))
```

```
IMC = P / A**2
```

```
print("IMC: ", IMC)
```



# ATIVIDADE

# Atividade 1

- Individual, em Python – 10 minutos
- Faça um Programa que peça um número e então mostre a mensagem *O número informado foi [número]*.

# Atividade 2

- Individual, em Python – 10 minutos
- Faça um Programa que peça três números (A, B e C) e imprima a soma de A e B multiplicada por C.

# Atividade 3

- Individual, em Python – 5 minutos
- Faça um Programa que calcule a área de um quadrado, em seguida mostre o dobro desta área para o usuário.

# Atividade 4

- Individual, em Python – 5 minutos
- Faça um Programa que peça a temperatura em graus Fahrenheit, transforme e mostre a temperatura em graus Celsius.

$$C = 5 * ((F-32) / 9)$$



# ENCERRAMENTO



# Resumo e Próximos Passos

- O que são variáveis em Python
  - Bases de funcionamento da linguagem
  - Uso do console e criação de programas
  - Capacitação para desenvolvimento de software
  - **Pós Aula:** Aprenda Mais, Pós Aula e Desafio!
    - No padlet: <https://padlet.com/djcaetano/paradigmas>
- 
- Conceito de escopo em Python
    - O que é e para que serve o “escopo”?



# PERGUNTAS?