

# **Python - Introdução**

Projeto Interdisciplinar para Sistemas  
de Informação 2 (PISI2)

# Conteúdo

- Características
- Instalando o Python e a IDE
- Transição Blocos -> Python
- Comandos Básicos
- Exercícios

# Características

- **Interpretada**
  - usa máquina virtual (PVM – Python Virtual Machine), facilita portabilidade.
- **Interativa**
  - pode-se programar interativamente, os comandos são executados enquanto são digitados. Facilita testes, desenvolvimento rápido e outros. Facilitadores estão presentes [help(obj)].
- **Orientada a Objetos**
  - tudo (ou quase tudo) é objeto: números, strings, funções, classes, instâncias, métodos, ...
- **Tipagem Dinâmica**
  - A definição do tipo de um objeto é feita em tempo de execução. Um objeto tem tipo, uma variável, não.

# Para que serve?

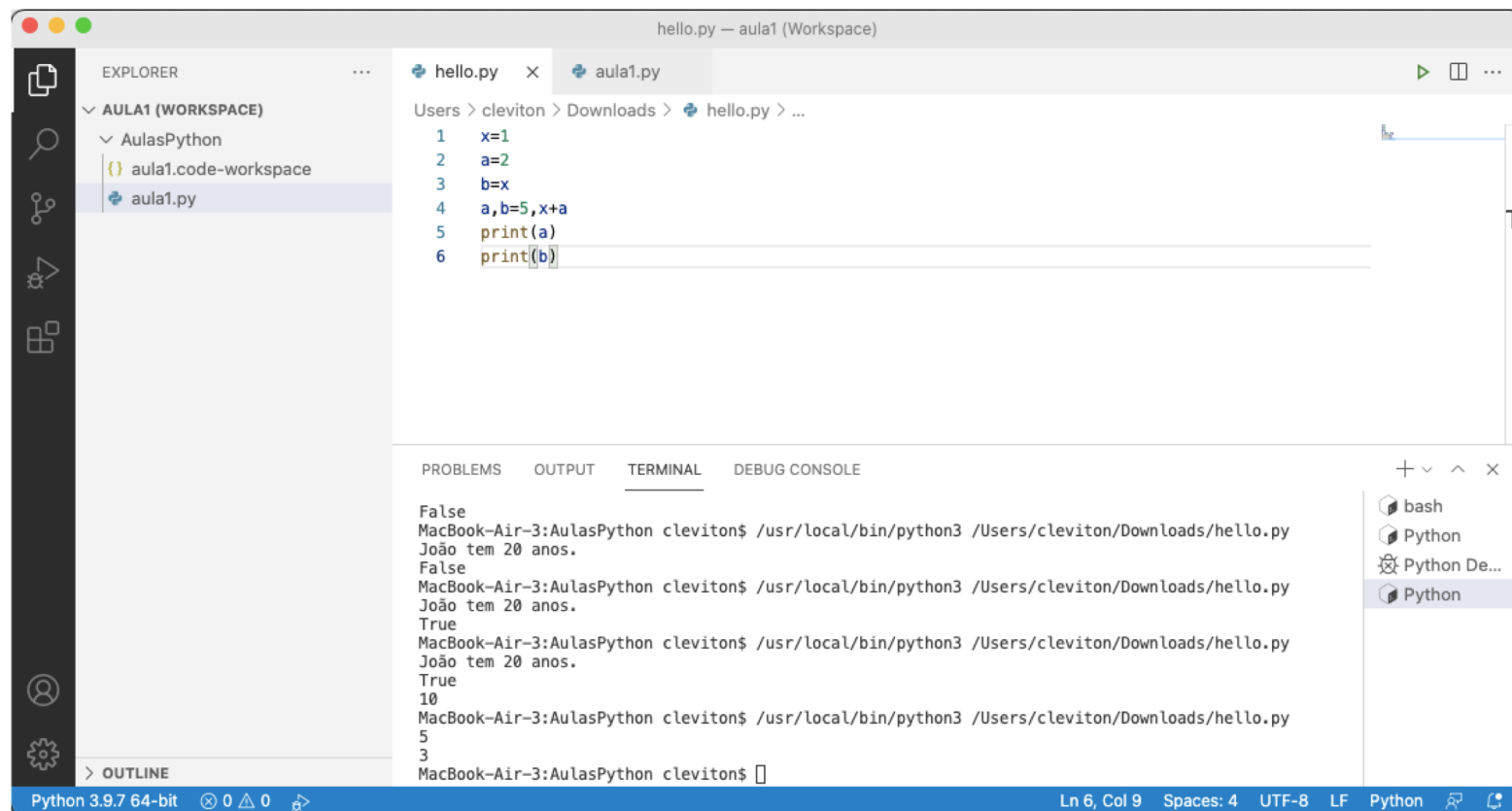
- **Prototipação** rápida
- Desenvolvimento **Web**
- Acesso a **Banco de Dados**
- Manipulação de **String**
- Computação **numérica** e **científica**
- **Jogos**
- Aplicações **3D**
- Modelagem de **Hardware**

# Quem usa Python?



# Instalação Python

- Sugestão: Programar no Visual Studio Code
- Tutorial de instalação no Classroom

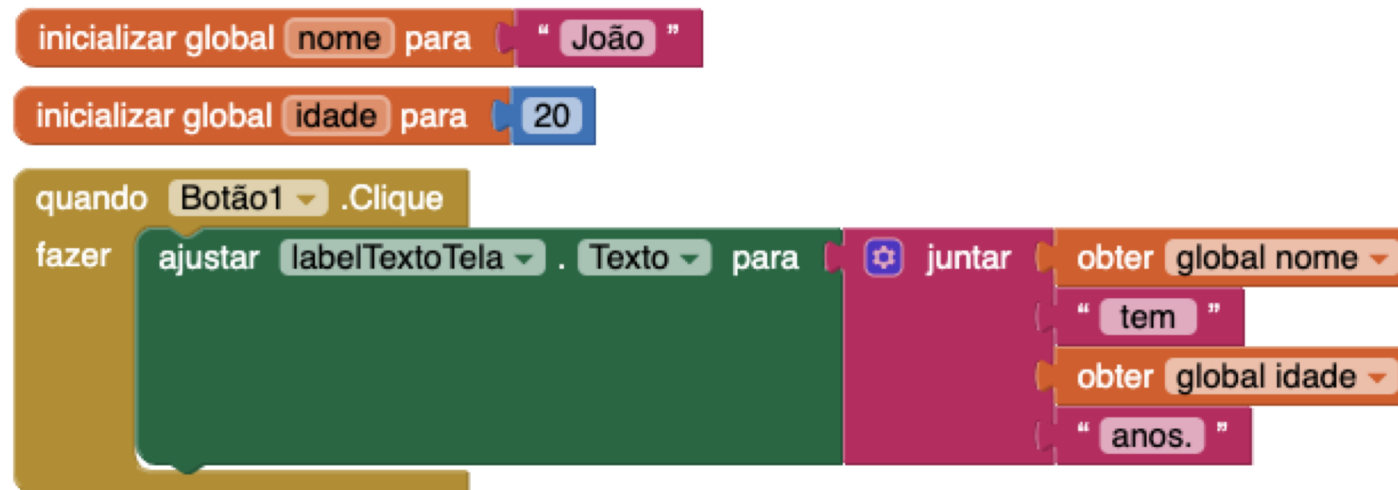


The screenshot shows the Visual Studio Code interface with a workspace named 'aula1'. The Explorer panel on the left shows the file structure: 'AULA1 (WORKSPACE)' containing 'AulasPython' and 'aula1.py'. The main editor displays the contents of 'aula1.py', which is a Python script. The script defines variables 'x' and 'a', calculates 'b' as 'x+a', and prints both 'a' and 'b'. The output panel at the bottom shows the results of running the script three times, demonstrating the state of the variables after each execution.

```
hello.py — aula1 (Workspace)
EXPLORER
  AULA1 (WORKSPACE)
    AulasPython
      aula1.code-workspace
      aula1.py
  hello.py x aula1.py
  Users > cleviton > Downloads > hello.py > ...
  1 x=1
  2 a=2
  3 b=x
  4 a,b=5,x+a
  5 print(a)
  6 print(b)
  PROBLEMS OUTPUT TERMINAL DEBUG CONSOLE
  False
  MacBook-Air-3:AulasPython cleviton$ /usr/local/bin/python3 /Users/cleviton/Downloads/hello.py
  João tem 20 anos.
  False
  MacBook-Air-3:AulasPython cleviton$ /usr/local/bin/python3 /Users/cleviton/Downloads/hello.py
  João tem 20 anos.
  True
  MacBook-Air-3:AulasPython cleviton$ /usr/local/bin/python3 /Users/cleviton/Downloads/hello.py
  João tem 20 anos.
  True
  10
  MacBook-Air-3:AulasPython cleviton$ /usr/local/bin/python3 /Users/cleviton/Downloads/hello.py
  5
  3
  MacBook-Air-3:AulasPython cleviton$
  Python 3.9.7 64-bit 0 0 0 Ln 6, Col 9 Spaces: 4 UTF-8 LF Python
```

# Transição dos blocos para Python

## Blocos

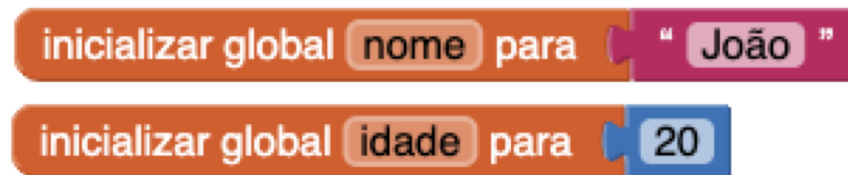


## Python

```
nome = "João"
idade = 20
print(nome + " tem " + str(idade) + " anos.")
```

# Características do Python

- Comentários são feitos usando `#`
- Não possui declaração de tipos
  - Blocos



– Java

```
String nome = "João";  
int idade = 20;
```

– Python

```
nome = "João"  
idade = 20
```




# Dados e Operações

| Operação                     | Resultado  |
|------------------------------|--|
| $x + y$                      | Soma dos valores x e y                               |
| $x - y$                      | Subtração de x por y                                 |
| $x * y$                      | Multiplicação de x por y                             |
| $x / y$                      | Divisão de x por y                                   |
| $x // y$                     | Divisão de x por y, obs.: Pegando o piso.            |
| $x \% y$                     | Resto da divisão de x por y                          |
| $+x$                         | Não altera nada                                      |
| $-x$                         | Inverte o sinal de x                                 |
| <code>abs(x)</code>          | Valor absoluto de x                                  |
| <code>int(x)</code>          | x convertido em inteiro                              |
| <code>long(x)</code>         | x convertido em long                                 |
| <code>float(x)</code>        | x convertido em float                                |
| <code>complex(re, im)</code> | Um número complexo com parte real re e imaginária im |
| $x ** y$                     | x elevado a y  |
| <code>pow(x, y)</code>       | x elevado a y  |

# Operações

Equivalentes



|            |              |
|------------|--------------|
| $a += b$   | $a = a + b$  |
| $a -= b$   | $a = a - b$  |
| $a *= b$   | $a = a * b$  |
| $a /= b$   | $a = a / b$  |
| $a **= b$  | $a = a ** b$ |
| $a \% = b$ | $a = a \% b$ |

# Dados e Operações

| Símbolo  | ação comparativa             |
|----------|------------------------------|
| "<"      | Menor que                    |
| "<="     | menor ou igual               |
| ">"      | maior que                    |
| ">="     | maior ou igual               |
| "=="     | igual (objeto -> referência) |
| "!="     | diferente                    |
| "<>"     | diferente                    |
| "is"     | igualdade de objetos         |
| "is not" | diferença de objetos         |

# Expressões Booleanas

- Também chamadas expressões lógicas
- Resultam em verdadeiro (True) ou falso (False)
- Usadas em comandos *condicionais* e de *repetição*
- Analisar o estado de uma computação e escolher o **próximo passo**

# Expressões Booleanas

- Operadores
  - Relacionais:  $>$  ,  $<$  ,  $==$  ,  $!=$  ,  $>=$  ,  $<=$
  - Booleanos: **and** , **or** , **not**
- Expressão avaliada da esquerda para a direita

# Expressões Booleanas

```
1==1      #True  
1==2      #False  
1==1 or 1==2  #True  
1==1 and 1==2 #False
```

```
1<2 and 2<3  #True  
not 1<2      #False  
not 1<2 or 2<3  #True  
not (1<2 or 2<3) #False
```

# Comandos Básicos

```
print('Hello World!')
```

Exibe: Hello World!

```
print("Hello World!")
```

Exibe: Hello World!

# Atribuição

```
x=1  
print(x)      #Exibe: 1
```

```
a=2  
b=x  
print(a)      #Exibe: 2  
print(b)      #Exibe: 1
```

```
a,b=5,x+a  
print(a)      #Exibe: 5  
print(b)      #Exibe: 3
```



# Entrada de Dados

- Função **input()** : lê um valor do dispositivo de entrada padrão

```
nome=input("Digite seu nome: ")
Exibe: Digite seu nome: 'Ana Paula'
idade=input('Digite sua idade: ')
Exibe: Digite sua idade: 13
print(nome)
Exibe: Ana Paula
print(idade)
Exibe: 13
```

# Entrada de Dados

- Função **input()** – Lê como String
  - Defina o tipo de dado lido (cast)

```
num=input("Digite um número inteiro: ")
```

```
Exibe: Digite um número inteiro: 3
```

```
print(num)
```

```
Exibe: 3
```

```
num = num * 2
```

```
print(num)
```

```
Exibe: 33 <- (provavelmente não é o que você queria ☹)
```

```
num = int(num) * 2 (transforme o num em inteiro)
```

```
print(num)
```

```
Exibe: 66 <- (agora sim! Usou um cast para inteiro ☺)
```

# Saída de Dados

- Função **print()**

```
print("Hello World!")
```

Exibe: Hello World!

```
print("Escreve no console.")
```

Exibe: Escreve no console.

# Saída de Dados

- Formatação com a função `print()`

```
nome = "Ana Paula"
idade = 13
print("Nome: %s, Idade: %d" %(nome,idade))
Exibe: Nome: Ana Paula, Idade: 13
print(nome, idade)
Exibe: Ana Paula 13
print(nome,idade, sep=",")
Exibe: Ana Paula,13
```

# Tipos

- *Inteiro*

```
Print(type (idade))
```

```
Exibe: <class'int'>
```

- **Float**

```
a = int(3 / 2)
```

```
print (a)
```

```
Exibe: 1
```

```
b = 3.0 / 2
```

```
print(b)
```

```
Exibe: 1.5
```

# Tipos

- **String:** limitadas por aspas simples ou duplas

```
print('Alo "Mundo"!')  
Exibe: Alo "Mundo"!  
print("Alo 'Mundo'!")  
Exibe: Alo 'Mundo'!  
print('''')  
Exibe: SyntaxError: EOL while  
scanning single-quoted string  
print('"\''')  
Exibe: "'
```

# Cálculos

```
print(2*2)
```

Exibe: 4

```
print(2/4)
```

Exibe: 0

```
print(2.0/4)
```

Exibe: 0.5

```
print(2-3)
```

Exibe: -1

```
base=10
```

```
altura=20
```

```
area=base*altura
```

```
print(area)
```

Exibe: 200

# Exercícios

1. Ler um número inteiro e exibir seu dobro.
2. Exibir a multiplicação de dois números reais informados pelo usuário.
3. Calcular a média aritmética de três notas fornecidas pelo usuário.
4. A imobiliária XYZ vende apenas terrenos retangulares. Faça um programa para ler as dimensões de um terreno e exibir a área do mesmo.



# Exercícios

5. Faça um programa para ler o salário de um funcionário e aumentá-lo em 20%. Imprima seu salário final.
6. Ler o valor de um cheque e escrever o quanto vai ser recolhido de CPMF. Considere que imposto recolhe uma taxa de 0,3%. Imprimir o valor do imposto.
7. Escreva uma seqüência de comandos para solicitar o nome e a matrícula do aluno. Em seguida exibir as informações no seguinte formato:
  - Nome do Aluno: “XXXXXXXX”, Matrícula: “ZZZZ”

# Exercícios

8. Faça um programa para uma loja de tintas. O programa deverá pedir o tamanho em metros quadrados da área a ser pintada. Considere que a cobertura da tinta é de 1 litro para cada 3 metros quadrados e que a tinta é vendida em latas de 18 litros, que custam R\$ 80,00. Informe ao usuário a quantidades de latas de tinta a serem compradas e o preço total.

# Bibliografia

- Python Tutorial -  
<http://www.python.org/doc/current/tut/tut.html>
- Dive into Python  
<http://www.diveintopython.org/>
- Python Brasil -  
<http://www.pythonbrasil.com.br/moin.cgi/DocumentacaoPython#head5a7ba2746c5191e7703830e02d0f5328346bcaac>
- Slides de Python: Rodrigo José Sarmento Peixoto e Flávio Dias