



INTRODUÇÃO A PYTHON



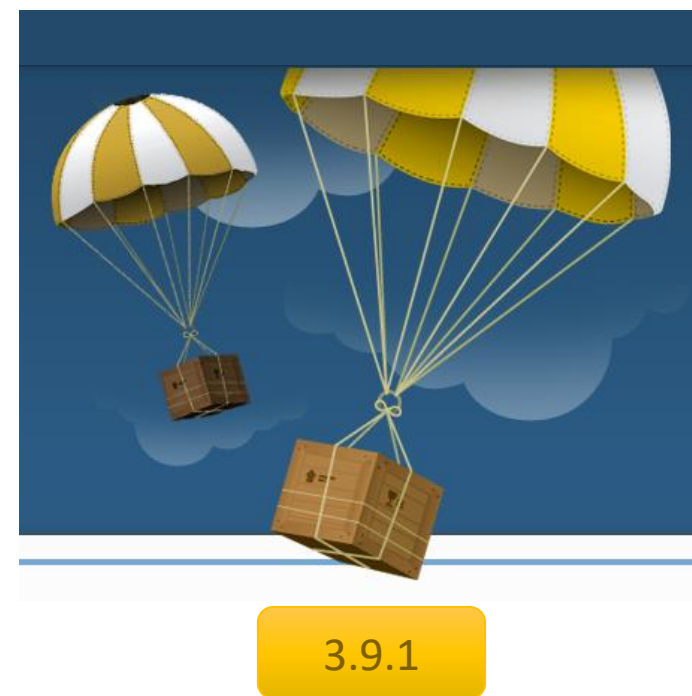
**INSTITUTO
FEDERAL**

Paraíba

Campus
João Pessoa

≡ Python ?!

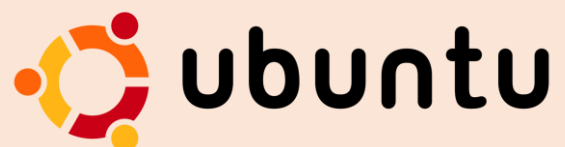
- **Linguagem eficiente e produtiva:**
 - Menor tempo de desenvolvimento;
 - Fácil interpretação;
 - Curva de aprendizagem reduzida.
- **Usada para administrar sistemas e desenvolver grandes projetos;**
- **Software livre:**
 - Mantida pela Python Foundation e inúmeros colaboradores.
- **Multiplataforma:**
 - Linux, Windows, Mac OS, ...
- **Produtividade:**
 - Frameworks!





Download

www.python.org/downloads



Já vem instalado !



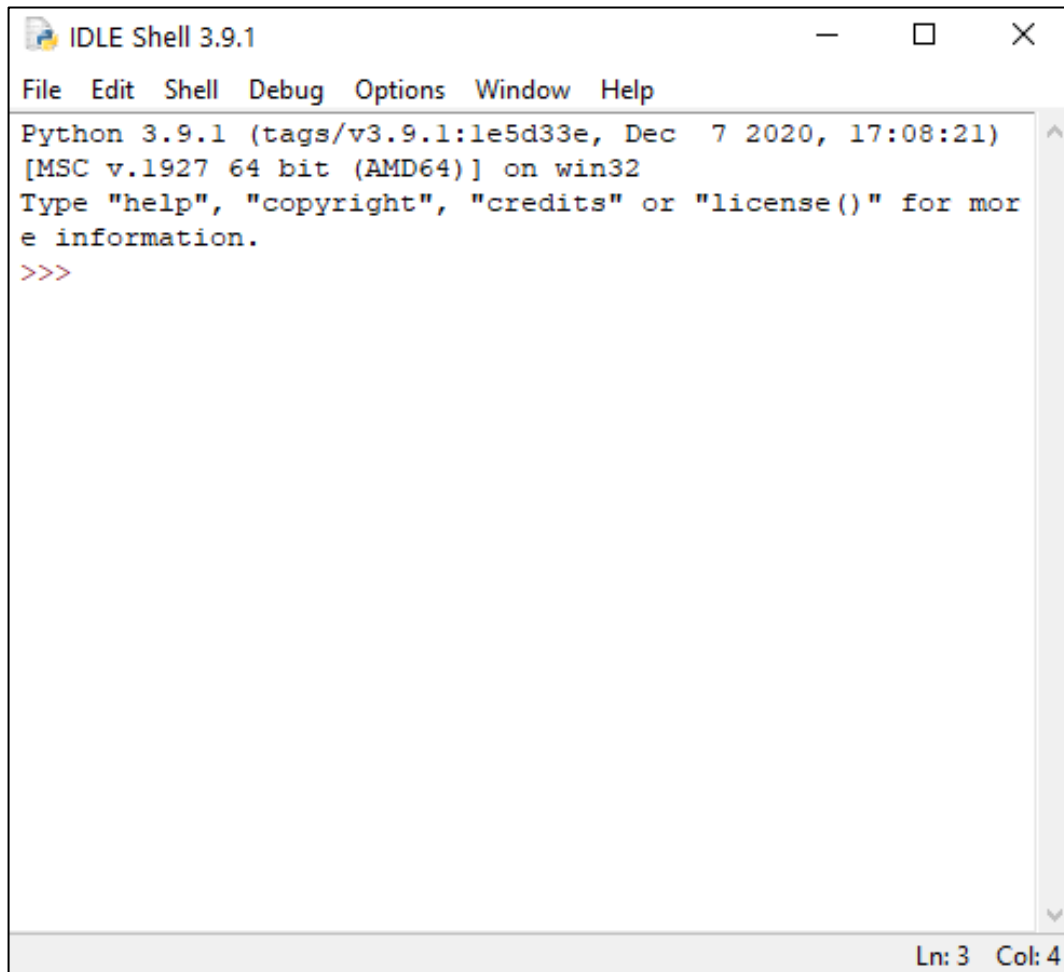
Já vem instalado !



Onde vou programar?

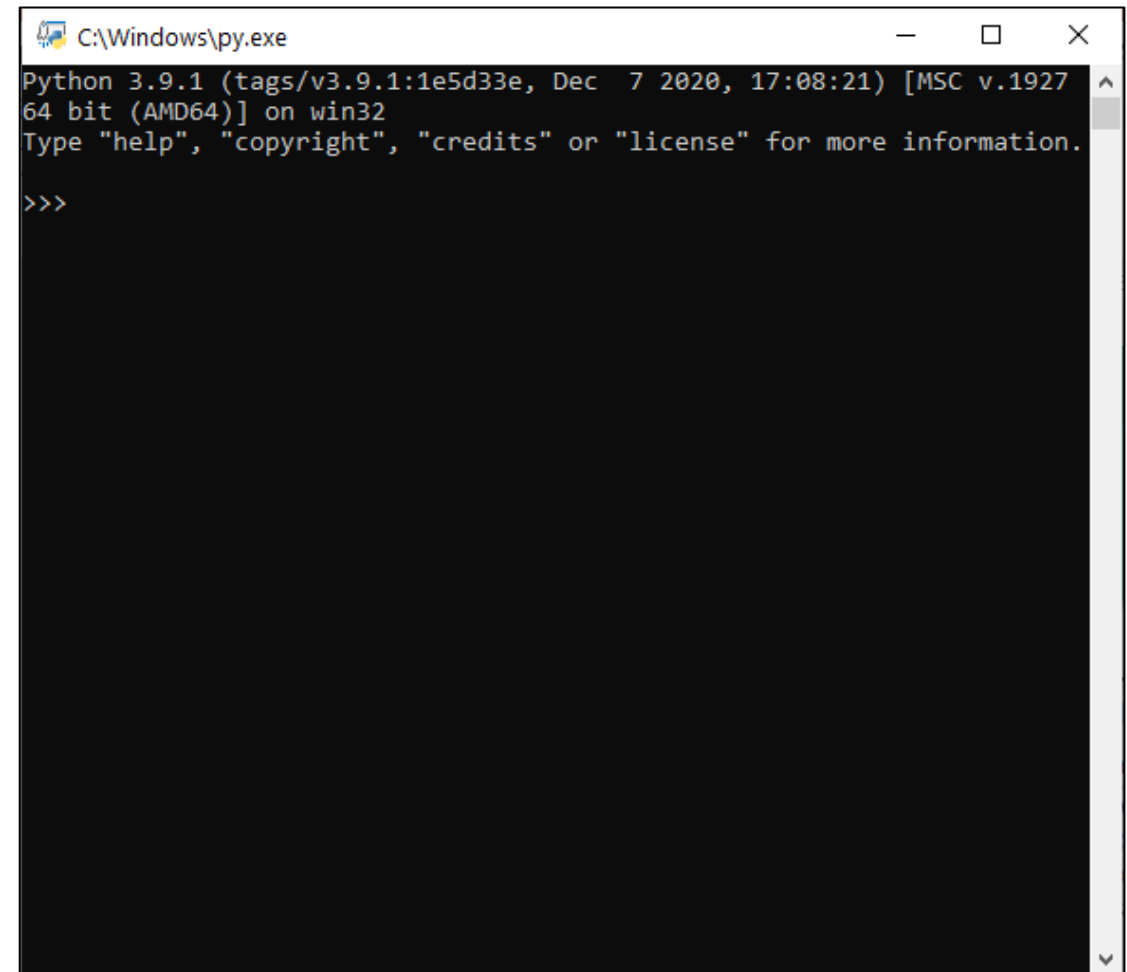


Windows



```
Python 3.9.1 (tags/v3.9.1:1e5d33e, Dec 7 2020, 17:08:21)
[MSC v.1927 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more
information.
>>>
```

IDLE (Python GUI)



```
Python 3.9.1 (tags/v3.9.1:1e5d33e, Dec 7 2020, 17:08:21) [MSC v.1927
64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>>
```

Linha de Comandos



≡ Elementos básicos da linguagem Python

- Variável
- Tipos de dados
- Operadores aritméticos
- Atribuição
- Entrada
- Saída

- Possui um identificador;
- Deve ser inicializada;
- Tipagem dinâmica:
 - Básicos: int, float, str e bool.

Declaração!

```
vi = 1  
vf = 1.1  
vs = "ifpb"  
vb = True
```


≡ Operadores Aritméticos

Operador	Descrição	Exemplo	Resultado
+	Soma	$2 + 2$	4
-	Subtração	$2 - 2$	0
*	Multiplicação	$2 * 2$	4
/	Divisão Real	$2 / 2$	1.0
//	Divisão Inteira	$5 // 2$	2
%	Resto da Divisão	$2 \% 2$	0
**	Potência	$2 ** 2$	4

≡ Atribuição

- Atribui um valor a uma variável
- Sintaxe:

<variável> = <valor>

Exemplos

```
>>> frase = 'Programando em Python'
>>> frase
'Programando em Python'
>>>
>>> n = 5
>>> n
5
>>>
>>> x = n + 2
>>> x
7
```

≡ Entrada: *input()*

- Sempre retorna um valor do tipo “str” (string);

Exemplo: leitura de uma frase

```
>>> frase = input ('Digite uma frase: ')
Digite uma frase: Programação em Python
>>> frase
'Programação em Python'
```

- Mas como Identificar e converter tipos em Python quando for preciso manipular valores numéricos?

Exemplo: somando dois números

```
>>> numero1 = input ('Digite um numero: ')
Digite um numero: 2
>>> numero2 = input ('Digite um numero: ')
Digite um numero: 3
>>> soma = numero1 + numero2
>>> soma
'23'
```

≡ Identificação e Conversão de Tipos

- Identificando um tipo:
 - `type (<valor ou variável>)`

Exemplo

```
>>> type(soma)
<class 'str'>
```

Em relação ao exemplo anterior, soma é do tipo string

- Funções de Conversão
 - `int()`
 - `float()`
 - `str()`

Exemplo 1

```
>>> soma = int(numero1) + int(numero2)
>>> soma
5
```

Exemplo 2

```
>>> numero1 = int(input('Digite um número: '))
Digite um número: 2
>>> type(numero1)
<class 'int'>
```

≡ Saída: *print()*

- `print(value, ..., sep=' ', end='\n', file=sys.stdout, flush=False)`

sep	O valor padrão de sep é um espaço em branco, quando dois ou mais argumentos são passados para a função print sep coloca entre eles um espaço em branco ou um outro valor que podemos passar para sep.
end	O valor padrão de end é uma nova linha “\n” e por isso que sempre a função print adiciona uma nova linha depois de imprimir tudo que lhe foi passado, mas, podemos definir outro valor com uma tabulação “\t” por exemplo.

Exemplo

```
nome=input('Informe um nome:')  
print('Nome =', nome)
```

≡ Apresentando dados em Python com *format()*

- Especificando a formatação de uma string:

```
>>> texto = '{} tem {} anos'
>>> print(texto.format('Paulo',20))
Paulo tem 20 anos
```



```
>>> print('{} tem {} anos'.format('Paulo',20))
Paulo tem 20 anos
```

- Numerando ou nomeando o espaço reservado para dados

```
>>> texto = '{0} tem {1} anos. {0} tem a mesma idade de {2}'
>>> print(texto.format('Paulo',20,'Luis'))
Paulo tem 20 anos. Paulo tem a mesma idade de Luis
```

```
>>> anoAtual=int(input('Digite o ano atual: '))
Digite o ano atual: 2019
>>> anoNasc=int(input('Digite o ano de nascimento: '))
Digite o ano de nascimento: 2000
>>> texto='{} tem {idade} anos'
>>> print(texto.format('Ana',idade=anoAtual-anoNasc))
Ana tem 19 anos
```

≡ Apresentando dados em Python com *format()*

- Especificando a formatação dos campos (outros exemplos)

```
>>> texto='Programando com Python'
>>>
>>> print('{:40}'.format(texto))
Programando com Python
>>>
>>> print('{:>40}'.format(texto))
                Programando com Python
>>>
>>> print('{:<40}'.format(texto))
Programando com Python
>>>
>>> print('{:^40}'.format(texto))
                Programando com Python
>>>
>>> print('{:@^40}'.format(texto))
@@@@@@@@@Programando com Python@@@@@@@@@@@@
```

≡ Apresentando dados em Python com *format()*

- Especificando a formatação dos campos (outros exemplos com *números*)

```
>>> print('inteiro: {0:d}  decimal: {1:5.2f}'.format(54, 54.789))
inteiro: 54  decimal: 54.79
>>>
>>> print('inteiro: {0:+d}  decimal: {1:5.2f}'.format(54, -54.789))
inteiro: +54  decimal: -54.79
>>>
>>> print('inteiro: {0:d}  decimal: {1:*^15.2f}'.format(54, 54.789))
inteiro: 54  decimal: *****54.79*****
```

- <https://docs.python.org/3/library/string.html>

≡ Apresentando dados em Python com *f-strings*

- Pode-se usar a partir da versão 3.6 do Python.
- As regras são as mesmas do método *format*, porém informamos o nome das variáveis na própria formatação, deixando, assim, o código mais legível.
- O nome *f-string* é devido ao prefixo *f* que deve existir junto à string.

```
>>> aluno = 'Pedro'
>>> curso = 'Sistemas para Internet'
>>> print(f'{aluno} está cursando {curso}')
Pedro está cursando Sistemas para Internet
```

```
>>> n = 3
>>> print(f'O dobro de {n} é {n*2}')
O dobro de 3 é 6
```