

# Fundamentos de algoritmos e introdução à programação em python

Prof. Everson Otoni

# OBMEP - Mentores (Apresentação)

## Quem são vocês?

- De onde são?
- Em qual ano escolar estão?
- Pretende fazer uma graduação ou licenciatura?
- Por que se inscreveram no curso?
- O que esperam deste curso?

## Título do curso

Fundamentos de algoritmos e iniciação à programação em *Python*

## Breve descrição do curso

Neste curso introdutório de fundamentos de algoritmos e iniciação à programação em *Python*, iremos abordar inicialmente os conceitos básicos de software. Avançaremos introduzindo a concepção de variáveis e instruções, bem como técnicas de controle de fluxo, ferramentas essenciais para direcionamento de algoritmos. Também exploraremos estruturas condicionais, que influenciam na tomada de decisões com base em certas condições, sempre partindo do paradigma procedural, que orienta a construção algorítmica dividindo as etapas computacionais em módulos ou funções.

Com esses conceitos fundamentais, os estudantes serão capazes de criar algoritmos simples para resolver uma variedade de problemas do mundo real. A linguagem de programação *Python* será utilizada como veículo principal para a implementação prática dos conceitos que serão abordados e discutidos.

Para participar deste curso, não é necessário ter experiência prévia em programação. É recomendável o compromisso com a experimentação e prática regular, uma vez que a programação é uma habilidade que se aprimora com a prática constante. Toda produção prática que compõe o curso poderá ser realizada utilizando um computador (principalmente) ou um *smartphone*.

**A ideia do curso é oferecer um primeiro contato com a programação.**

**Tornar a programação não um fim, mas sim um meio.**

**E principalmente ajudar vocês a pensar como um cientista da computação.**

# Mãos à obra

## Cientista da computação

- Assim como os matemáticos, os cientistas da computação usam linguagens formais para denotar ideias (especificamente operações de computação).
- Como engenheiros, cientistas da computação projetam, reunindo componentes em sistemas e avaliando as melhores opções de retorno.
- A habilidade mais importante de um cientista da computação é a resolução de problemas. Isso significa uma capacidade desenvolvida de formular problemas, pensar criativamente em soluções e expressá-las de forma clara e precisa.

# O que é um programa?

Um programa é uma sequência de instruções que executa uma operação de computador, especificamente.

Um operação de computador pode ser algo matemático, como encontrar a solução de um sistema de equações ou encontrar as raízes de um polinômio. Mas também pode ser uma operação computacional simbólica, como busca e substituição de textos em um documento; ou algo de cunho gráfico, como o processamento de uma imagem ou vídeo.

Esses slides mesmos, foram gerados computacionalmente por meio de uma linguagem de marcação e de um programa que converte o texto simples em slides.

# A forma básica

Os detalhes mudam em linguagem diferentes, mas algumas instruções básicas aparecem em quase todas:

*entrada:*

Receber dados do teclado, de um arquivo, da rede ou de algum outro dispositivo.

*saída:*

Exibir dados na tela, salvá-los em um arquivo, enviá-los pela rede etc.

# A forma básica

*matemática:*

| Executar operações matemáticas básicas.

*execução condicional:*

| Executar o código adequado, diante da existência de certas condições.

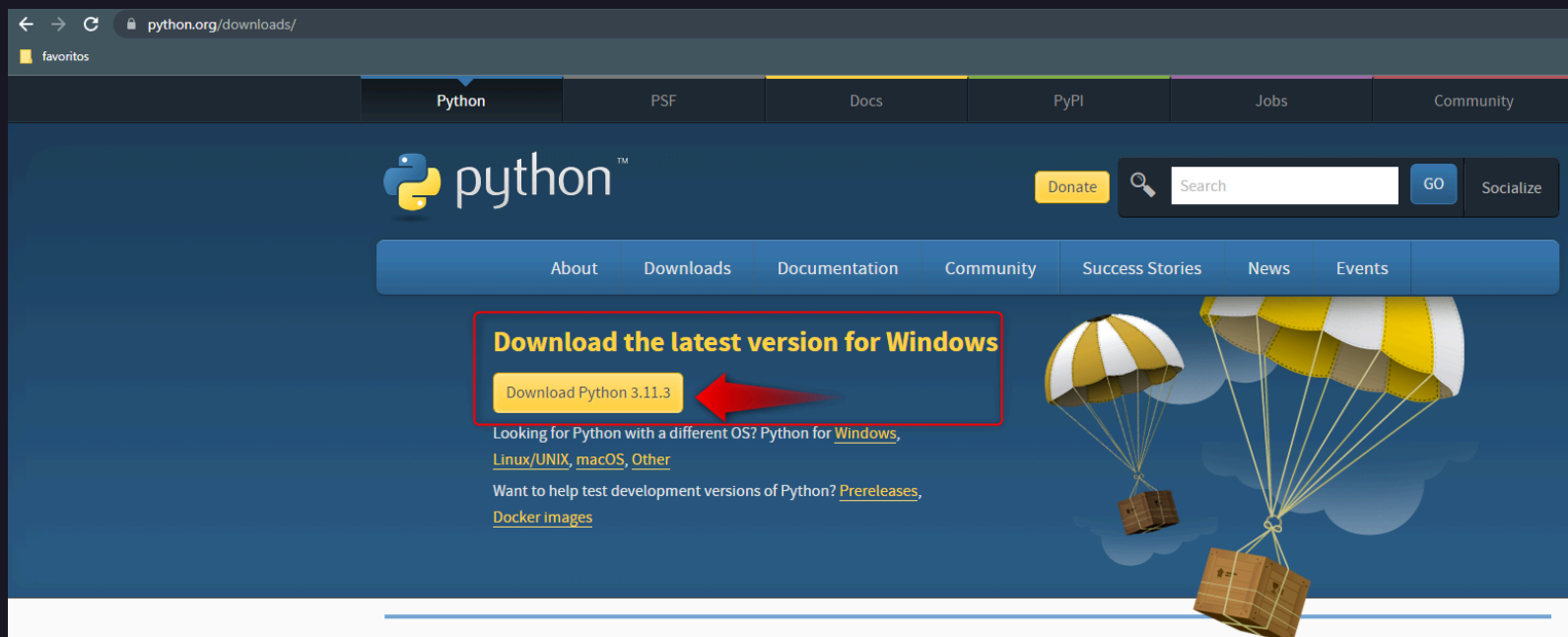
*repetição:*

| Executar várias vezes alguma ação, normalmente com variações.

# Instalando o Python 3 no Windows

Para instalar o Python no seu sistema operacional Windows, você precisará baixar o instalador.

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

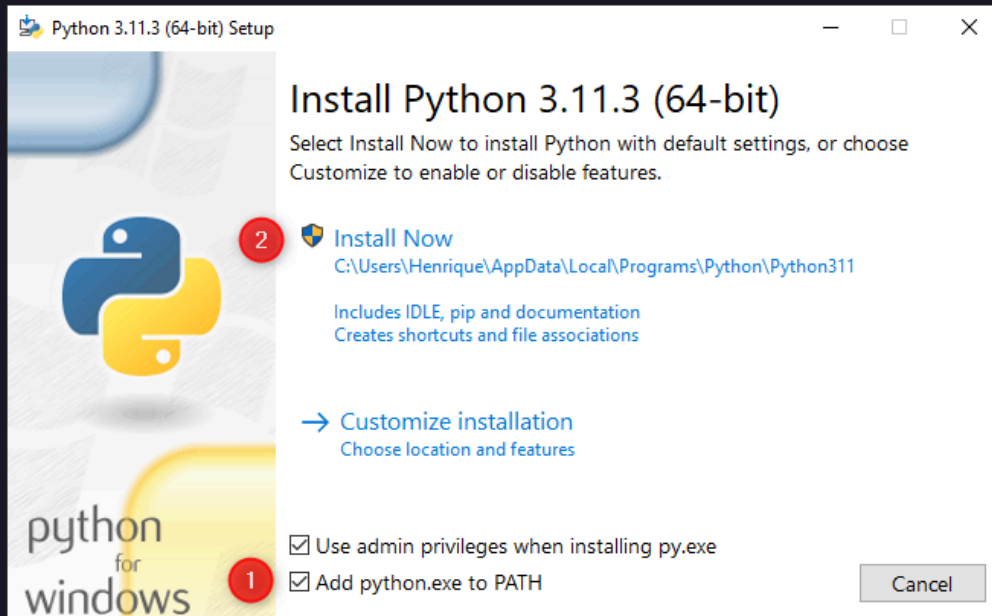


Isso fará o download do Python 3 para sistemas 64 bits



# Instalando o Python 3 no Windows

Faça o download do instalador executável do Windows (32 ou 64 bits) e clique duas vezes nele para iniciar o assistente de instalação do python, como mostrado abaixo.

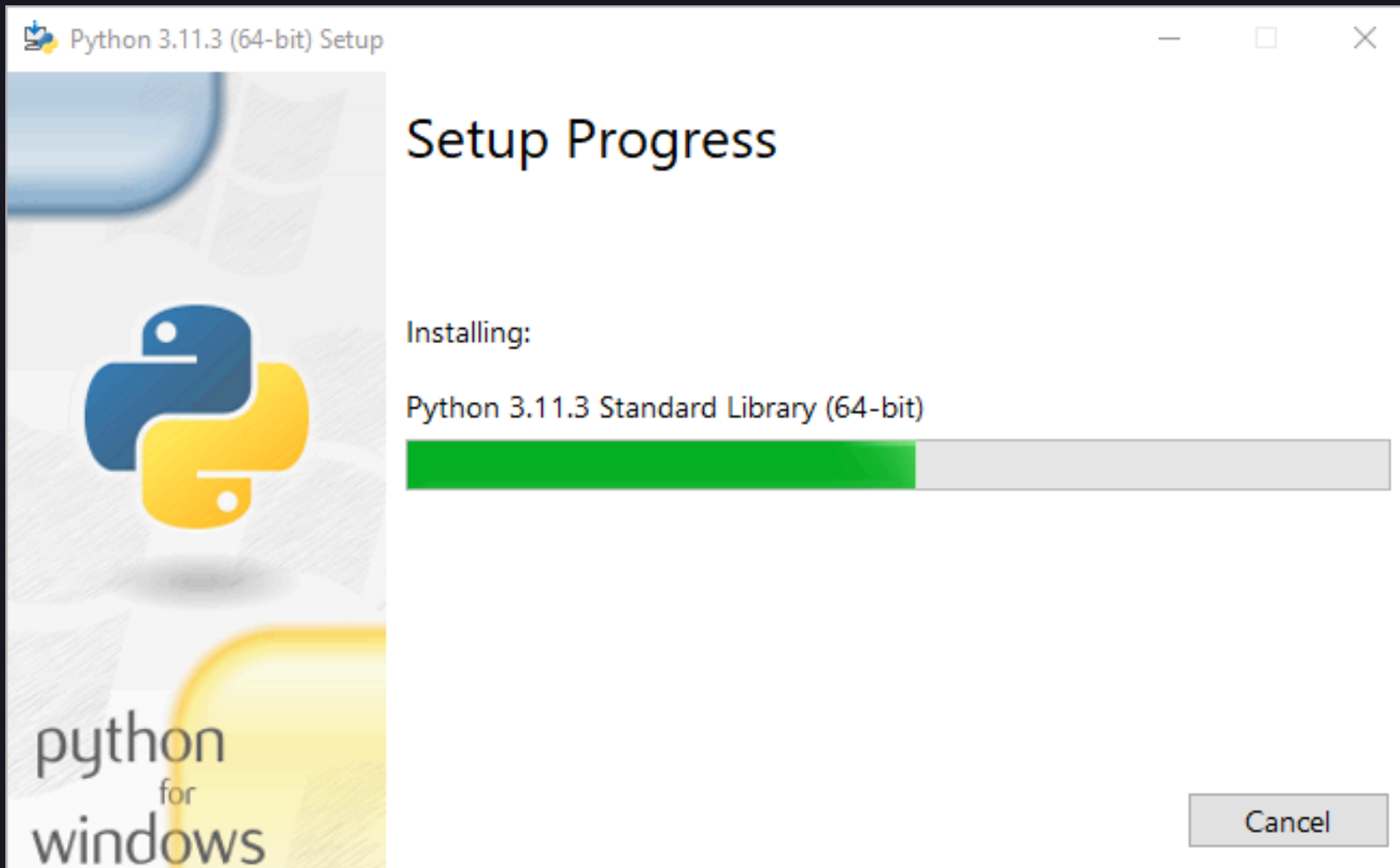


O processo de instalação é bem simples.

1. Marque a opção "Add Python to PATH"
2. Clique em "Install Now"

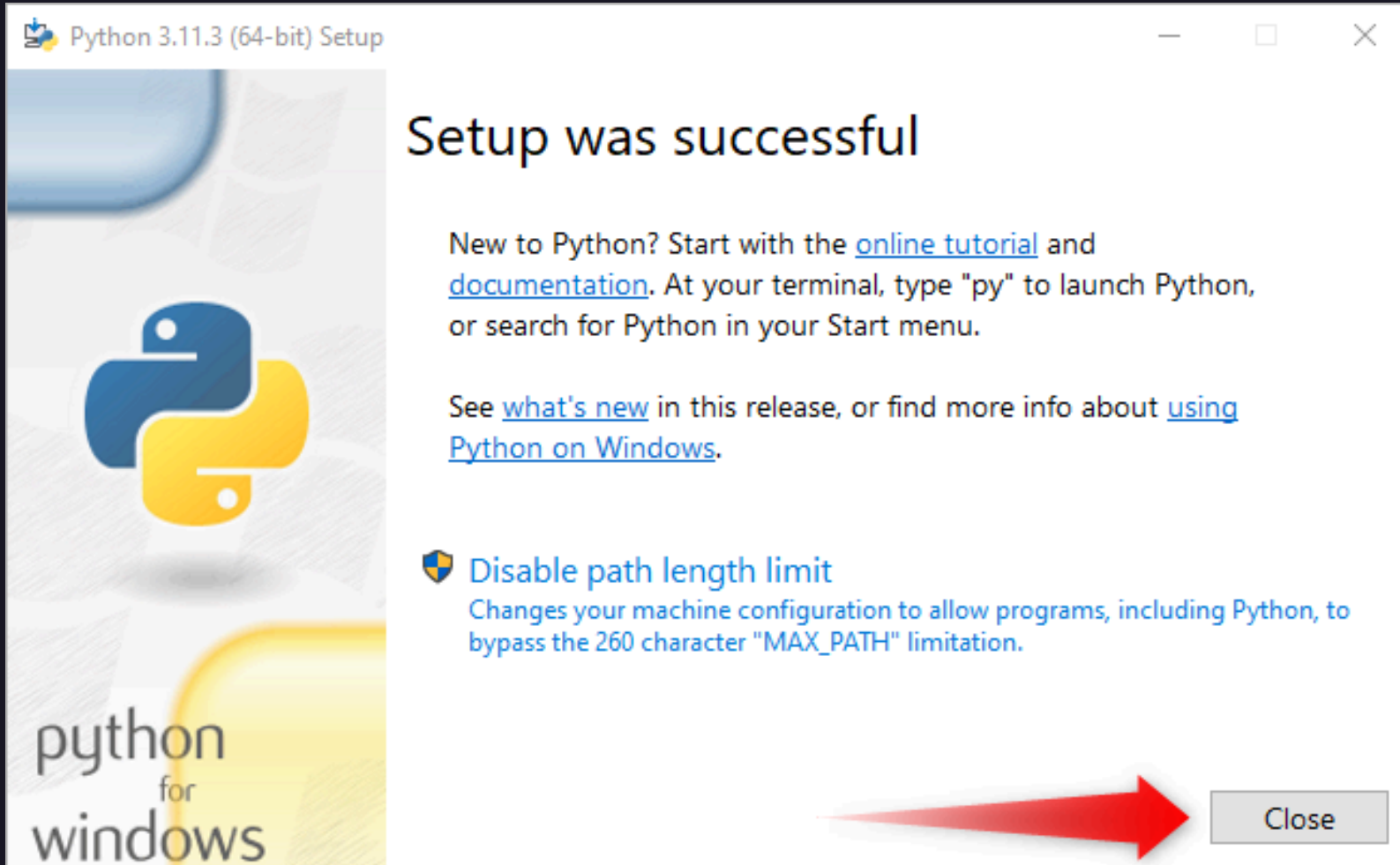
# Instalando o Python 3 no Windows

A tela abaixo será mostrada. Aguarde enquanto o instalador completa o processo de instalação.



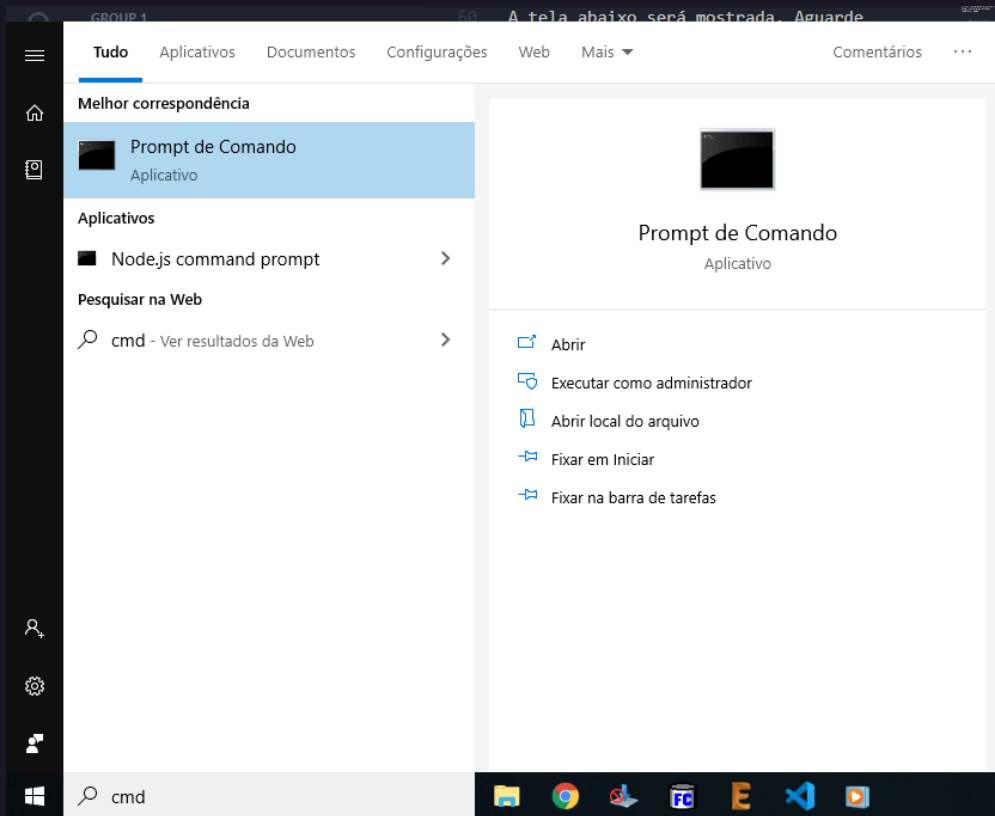
# Instalando o Python 3 no Windows

Se tudo ocorrer bem, a próxima tela será mostrada. Clique em "Close".



# Instalando o Python 3 no Windows

Para verificar se a instalação do Python foi bem-sucedida, pesquise no menu iniciar por "cmd" e clique duas vezes para abri-lo.



# Instalando o Python 3 no Windows

**Digite o seguinte comando**

```
$ python --version  
Python 3.12.3
```

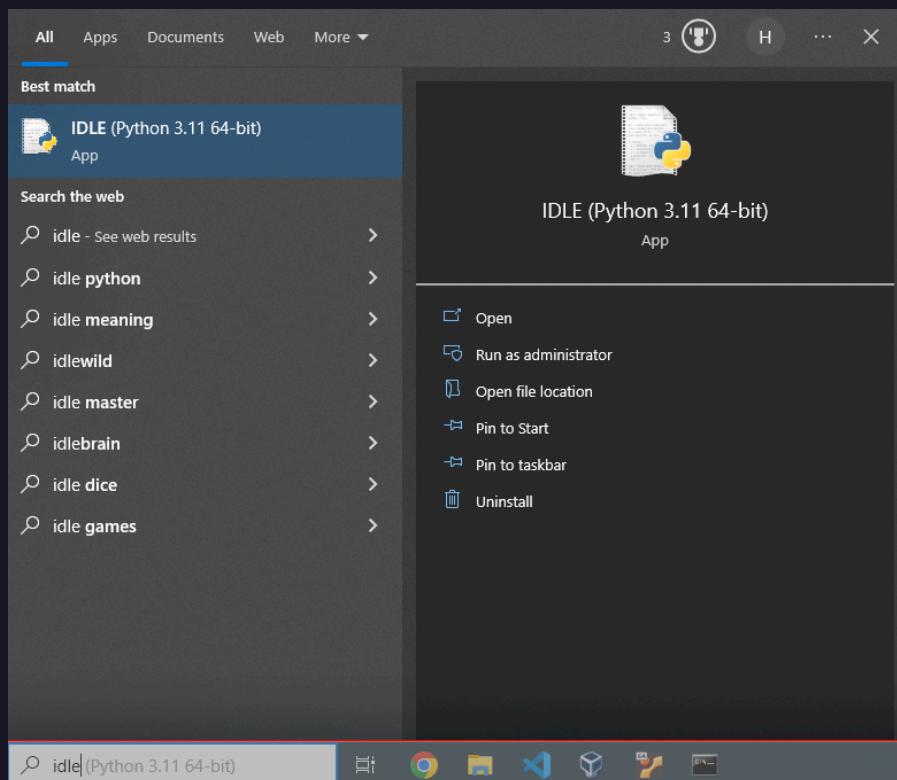
**Este comando retornará a versão do python que está instalada em seu computador.**

# IDLE

O instalador do Python para Windows contém o módulo IDLE por padrão.

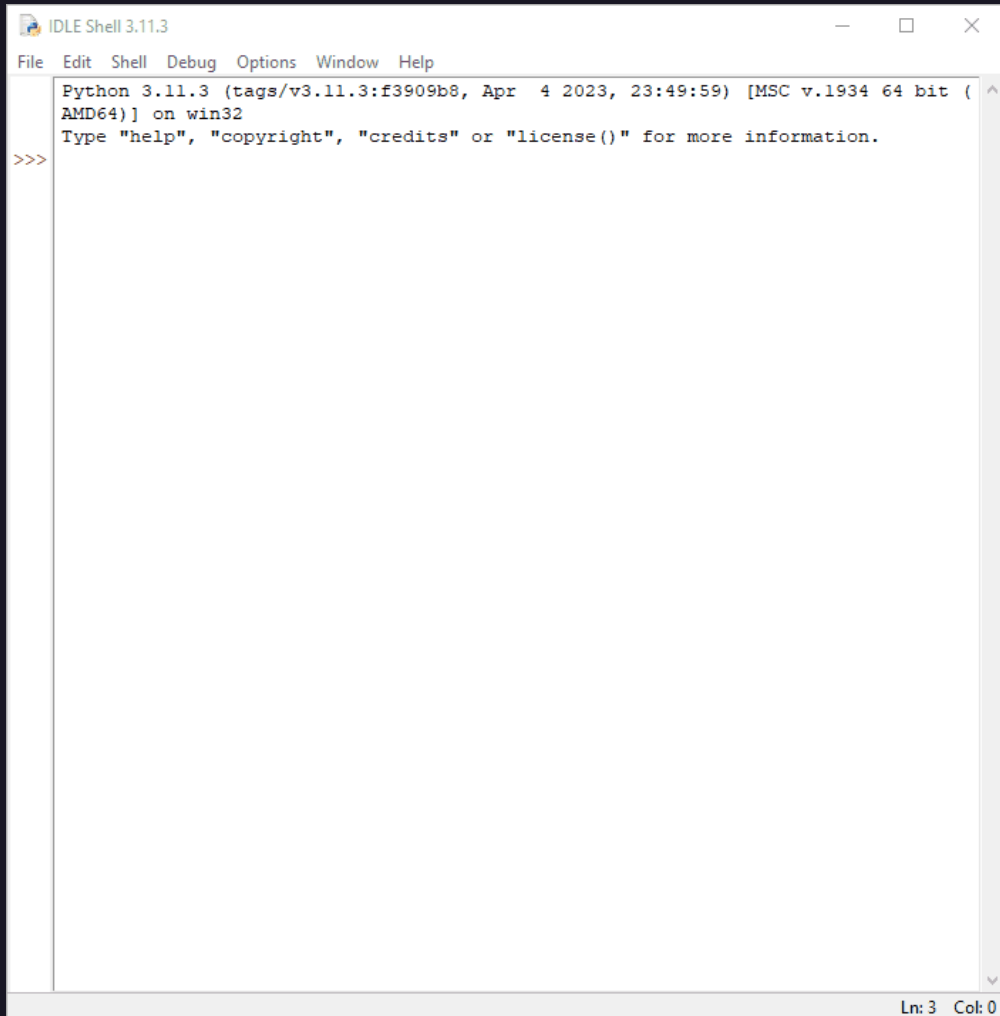
IDLE é um Ambiente de Desenvolvimento e Aprendizagem Integrado.

Para iniciar o shell interativo IDLE, procure o ícone IDLE no menu Iniciar e clique duas vezes nele.



# IDLE

Onde efetivamente, o *interpretador* Python, um programa. Lê e executa o código Python.



```
Python 3.11.3 (tags/v3.11.3:f3909b8, Apr  4 2023, 23:49:59) [MSC v.1934 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
```

The screenshot shows the IDLE Shell 3.11.3 window. The title bar reads "IDLE Shell 3.11.3". The menu bar includes "File", "Edit", "Shell", "Debug", "Options", "Window", and "Help". The main text area displays the Python 3.11.3 startup message and the interactive prompt. The status bar at the bottom right indicates "Ln: 3 Col: 0".

# Finalmente! O interpretador python

```
--  
Python 3.11.8 (main, Feb 12 2024, 14:50:05) [GCC 13.2.1 20230801] on linux  
Type "help", "copyright", "credits" or "license" for more information.  
>>> █
```

Essas primeiras linhas contêm informações sobre o interpretador e o sistema operacional em que está sendo executado.

Podemos perceber que há uma tríade de número após o nome "Python", esse número indica qual é versão que está sendo utilizada.

No nosso caso a versão utilizada é **3.11.8**.

A última linha que inicia com **>>>** é um prompt. São caracteres expostos pelo interpretador para indicar que está tudo pronto para receber entradas do usuário.



# Nosso olá mundo

Se você digitar uma linha de código e pressionar **Enter**, o interpretador exibe o resultado.

Por exemplo:

```
>>> 1 + 1
```

Exibe o resultado:

```
2
```

A partir de agora iremos começar, já que agora você sabe iniciar o interpretador do Python e executar um código.

# O primeiro programa

Existe uma tradição em ciência da computação, em que o primeiro programa que se escreve em uma nova linguagem chama-se "Hello, Word", ou para nós "Olá, Mundo"), porque tudo o que faz é exibir as essas palavras na tela.:

```
>>> print('Olá, Mundo!')
```

Este é um exemplo de um *instrução print* (instrução de impressão), porém a impressão ocorre na tela. Nesse caso, o resultado são as palavras:

```
Olá, Mundo!
```

As aspas apenas marcam o começo e o fim do texto a ser exibido, elas não aparecem no resultado.

# Operadores aritméticos

O python possui *operadores*, que são símbolos especiais representando operações de computação, como adição e multiplicação.

Os operadores `+, - e *` executam a adição, a subtração e a multiplicação.

```
>>> 40 + 2  
42
```

```
>>> 43 + 1  
44
```

```
>>> 6 * 7  
42
```

# O operador /

```
>>> 84 / 2
```

```
42.0
```

Perceba que em do resultado 42, o resultado exibido foi 42.0

# O operador \*\*

```
>>> 6 ** 2 + 6
```

```
42
```

O operador `**` executa a exponenciação, eleva um número a uma potência.

Em algumas linguagens, o `^` é utilizado para exponenciação.

# Valores e tipos

Um *valor* é uma das coisas mais básicas com as quais o programa trabalha.

Até agora vimos como valores:

```
2
```

```
42.0
```

```
Olá, Mundo!
```

Esses valores pertencem a tipos diferentes: `2` é um *número inteiro*, `42.0` é um *número de ponto flutuante* (conhecidos usualmente como número decimais) e `Olá, Mundo!` é uma *string*, chamado assim porque as letras que contém estão em uma sequência em cadeia (como uma corda ou fio).

# A função type

O Python possui um função nativa chamada de `type`.

Se executarmos o código:

```
>>> type(2)
```

Irá ser exibido para nós:

```
<class 'int'>
```

A palavra `class` é usada no sentido de *categoria*; Nesse caso, um tipo é uma categoria de valores.

# A função `type`

Teste com os outros valores e verifique o resultado exibido.

```
>>> type(42.0)
```

```
>>> type('Olá, Mundo!')
```

Número inteiros pertencem ao tipo *int*, strings pertencem ao tipo *str* e os números de ponto flutuante pertencem ao tipo *float* (flutuador).

# A função type

O que acontece se executarmos:

```
>>> type('2')
```

```
>>> type('42.0')
```

**Lembre-se as aspas marcam o começo e o fim do texto a ser exibido.**



# A função type

Parecem número, mas estão entre aspas como se fossem strings:

```
>>> type('2')  
<class 'str'>
```

```
>>> type('42.0')  
<class 'str'>
```

Então são strings.

# Linguagens

## As linguagens naturais

São idiomas que as pessoas falam, não foram criados pelas pessoas, desenvolveram-se naturalmente.

## As linguagens formais

São linguagens criadas pelas pessoas para aplicações específicas.

Por exemplo, notações que os matemáticos usam é uma linguagem formal utilizada, especialmente, para denotar relações entre números e símbolos.

# Linguagens

**As linguagens de programação são idiomas formais criados para expressar operações de computação.**

# Linguagem - Regras de sintaxe

As linguagens formais geralmente possuem *regras de sintaxe* estritas que governam a estrutura de declarações.

$3 + 3 = 6$ , tem uma sintaxe correta, diferente de  $3+ = 3\$6$

Assim, como na matemática a química também possui uma linguagem formal com regras de sintaxe.

$H_2O$ , é uma fórmula sintaticamente correta, diferente de  $_2Z_Z\$$ .

# Linguagem - Regras de sintaxe

Possuem duas categorias

## Símbolos

Elementos básicos da linguagem, como palavras, números, e elementos químicos.

${}_2Z_Z$  não é legítimo porque não há nenhum elemento com a abreviatura  $Z_Z$

## Estrutura

A estrutura refere-se a maneira como os elementos são combinados.

Por exemplo, apesar de que  $+$  e  $=$  serem símbolos legítimos, não se pode ter um na sequência do outro como em  $3+ = 3$ .

# Linguagem

**As linguagens formais são mais densas que as naturais, então exigem mais tempo para a leitura.**

**A estrutura é importante, então nem sempre é melhor ler de cima para baixo e da esquerda para a direita. Em vez disso, aprenda a analisar o programa primeiro, identificando os símbolos e interpretando a estrutura.**

**Os detalhes fazem diferença.**

**Pequenos erros de ortografia e pontuação, que podem não importar tanto nas linguagens naturais, podem fazer uma grande diferença em uma linguagem formal.**

# Depuração

Programadores erram.

Erros de programação são chamados de *bugs* (insetos). - Longa história.

O processo de rastreá-los chama-se *depuração* (debugging).

Programar, e especialmente fazer a depuração, às vezes traz emoções fortes. Prepare-se para essas reações.

Aprender a depurar erros pode ser frustrante, mas é uma habilidade valiosa, útil para muitas atividades além da programação.

