



## Introdução à Python com RPA

Prof. Ms. Massaki de O. Igarashi

[massaki.igarashi@mackenzie.br](mailto:massaki.igarashi@mackenzie.br)



# Prof. Ms. Massaki de Oliveira Igarashi

[massaki.igarashi@mackenzie.br](mailto:massaki.igarashi@mackenzie.br)



Eng. Eletricista (Hab. Eletrônica), Mestre em Engenharia da Informação.

Professor de Linguagem de Programação e Tecnologia da Informação e Com. Eng no Centro de Ciências e Tecnologia/ CCT

Experiência c/ instrumentação analítica e desenvolvimento de equipamentos para análises químicas e petroquímicas.

<https://linktr.ee/rpapython>

Escaneie e acesse

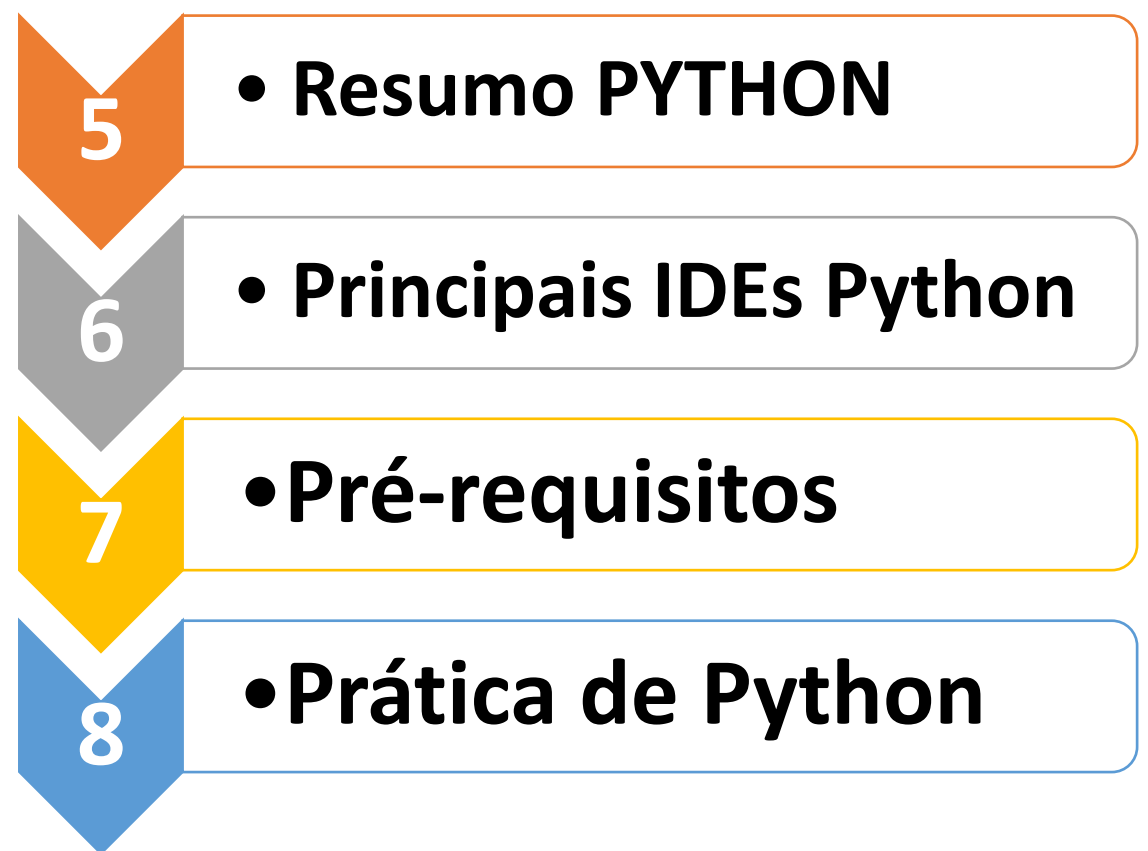


# AGENDA DA APRESENTAÇÃO

## O que é RPA?

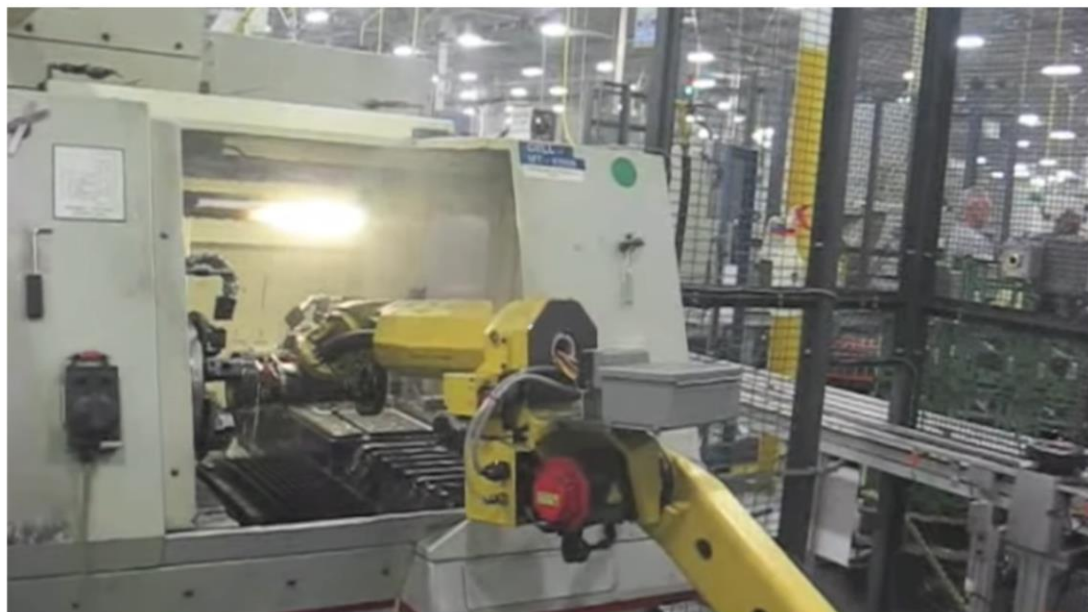


## Linguagem Python

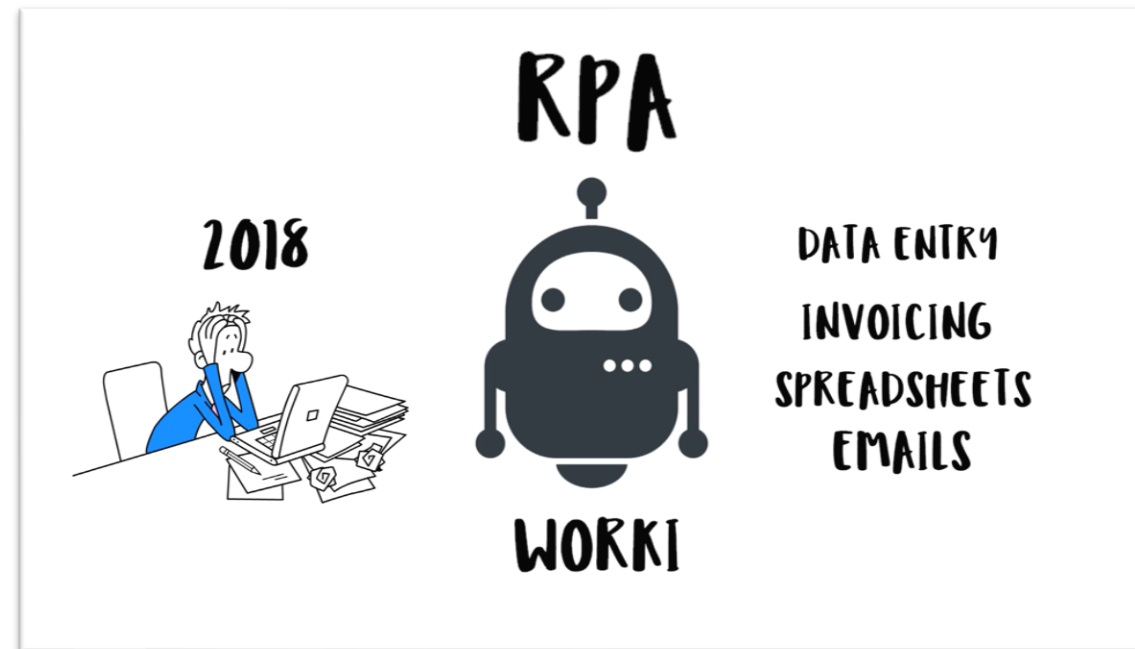




# DEFINIÇÃO



Robotic Automation for Industrial Processes



Robotic Process Automation (RPA)

*“Os robôs da indústria automatizam a produção de rotina e os robôs RPA automatizam o trabalho humano com dados e informações”.*

# DEFINIÇÃO

“

Automação Robótica de Processos (*Robotic Process Automation - RPA*) é um **termo genérico** para **ferramentas** que operam em interface do usuário de outros sistemas de computador **para imitar o comportamento humano em tarefas repetitivas**. O RPA visa **substituir as pessoas por automação** feita de maneira “de fora para dentro” (VAN DER AALST, 2018).

”

# DEFINIÇÃO

As ferramentas RPA executam **instruções [if, then, else]** em dados, normalmente usando uma **combinação de interface do usuário, interações ou conectando-se a APIs para direcionar servidores clientes, mainframes ou código HTML**. Elas mapeiam um processo descrito na linguagem da ferramenta RPA para o robô de software seguir, com o **tempo de execução alocado para executar o script** por um painel de controle (TORNBOHM, 2017; VAN DER AALST, 2018).



**As ferramentas de RPA reduzem a carga e simples tarefas nos funcionários**

# APLICAÇÕES

1. **Atendimento ao cliente**
  2. **Processamento de faturas**
  3. **Pedidos de vendas**
  4. **Folha de pagamento**
  5. **Comparação de preços**
  6. **Armazenamento de informações do cliente**
  7. **Processando informações de RH**
  8. **Processando reembolsos rápidos**
  9. **Recrutamento**
  10. **Extrair dados de diferentes formatos**
- ✓ **Preenchimento de formulários e/ou digitações** em sites ou sistemas com informações obtidas de diferentes fontes;
  - ✓ Extração de informações de outros sistemas, abastecendo o sistema interno (capturar informações contidas em **planilhas**, arquivos, textos ou PDF)
  - ✓ **Verificação e comparação de conteúdo** entre duas ou mais fontes distintas de documentos;
  - ✓ **Download/upload de arquivos**;
  - ✓ Integração entre sistemas s/ necessidade de desenvolv// ou customização
  - ✓ **Envio e recebimento de e-mails**
  - ✓ Gerenciamento de eventos
  - ✓ **Captura de documentos e transformação em dados estruturados**
  - ✓ Processos com **tarefas repetitivas em geral...**



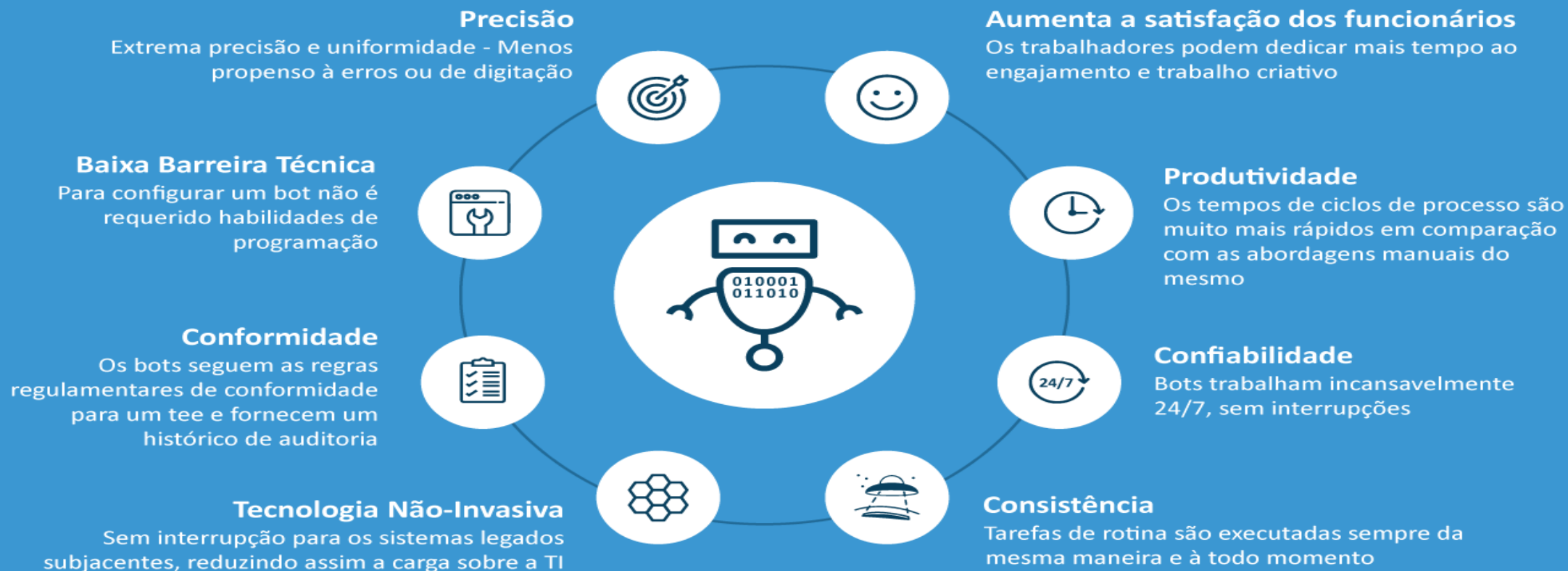
# BENEFÍCIOS DO RPA



- Agilidade;
- Consistência;
- Precisão;
- Uniformidade;
- Dispensa usuários de tarefas repetitivas;

# BENEFÍCIOS

## *Benefícios de Robotic Process Automation*



# BENEFÍCIOS

**Rastreabilidade  
e evidências  
para auditorias;**

**Maior agilidade  
para entender as  
mudanças de  
processos;**



**Mais tempo  
disponível  
para  
atividades de  
valor  
agregado.**









# Por que aprender Python?



- ✓ **Linguagem mais utilizada** hoje globalmente
- ✓ Fácil de aprender
- ✓ **Interoperabilidade** (comunica-se de forma transparente com outras linguagens:

Java, .NET e bibliotecas C/C ++);

- ✓ Permite **integração e desenvolvimento web**;
- ✓ Tem muitos recursos e **bibliotecas para visualização de dados**;
- ✓ **Interpreta scripts** (não requer compilação já que interpreta o código diretamente);

Rank	Language	Type	Score
1	Python	  	100.0
2	Java	  	95.4
3	C	  	94.7
4	C++	  	92.4
5	JavaScript		88.1
6	C#	   	82.4
7	R		81.7
8	Go	 	77.7
9	HTML		75.4
10	Swift	 	70.4

# Um resumo sobre Python

“

A linguagem **Python** foi **criada** pelo holandês **Guido van Rossum** por volta de **1990** e tem como principal filosofia, a simplicidade e legibilidade do código. Não obstante, é **utilizada amplamente por grandes empresas** como YouTube, Google, Yahoo e Microsoft.

Python é **uma das linguagens mais populares hoje**. Existem diversas bibliotecas para análise de dados e materiais para auxiliar o desenvolvimento de algoritmos

Além disso, é uma linguagem **poderosa ... e rápida; interage bem com outras, é amigável, fácil de aprender e é de código aberto**.



”



# Um resumo sobre Python

Na década de 1970, a BBC tinha um programa de TV popular do qual Van Rossum era um grande fã chamado Fly Circus de Monty Python, ou apenas Monty Python para os íntimos. Assim, quando desenvolveu a linguagem, ele pensou que precisava de um nome que fosse curto, único e um pouco misterioso, e por algum motivo que só ele conhecia, decidiu chamar o projeto de 'Python'.

A linguagem foi batizada em homenagem ao programa de humor britânico “*Monty Python*” (1970).

Para mais informação:

<http://www.montypython.com>



O grupo Monty Python foi muito famoso na Inglaterra, na década de 70 e recebeu muitas críticas de conservadores por causa do seu humor ácido e irreverente.

Você pode conhecer a licença do Python e fazer o download de sua última versão no seu site oficial [www.python.org](http://www.python.org).

# Um resumo sobre Python

**Python** é uma **linguagem de scripts** que permite executar e testar um código imediatamente depois de escrevê-lo, facilitando bastante as atualizações. Em outras palavras, linguagens de script são linguagens interpretadas. O **interpretador executa o programa apenas traduzindo comandos em uma série de uma ou mais sub-rotinas** que depois são traduzidas em outras linguagens.

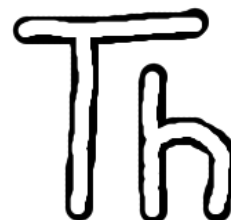
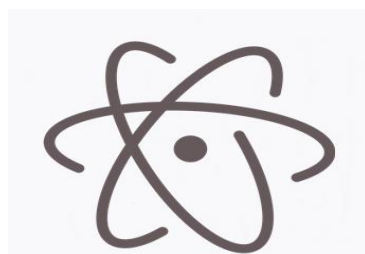
Um script é uma **coleção de comandos em um arquivo** projetada **para ser executada como um programa** e não pelo processador do computador, como acontece com linguagens compiladas. **O arquivo pode conter funções e módulos variáveis**, mas **a ideia central é que ele possa rodar e cumprir uma tarefa específica a partir de uma linha de comando**. Um exemplo clássico disso são as linguagens para prompts de comando, como no arquivo batch Windows.

Em geral, é mais rápido e fácil programar usando uma linguagem de script do que uma mais estruturada e compilada, como C ou C++.

# Principais IDEs PYTHON

O acrônimo **IDE** (**I**ntegrated **D**evelopment **E**nvironment) é usado para definir um software ou ambiente de desenvolvimento integrado que une ferramentas de desenvolvimento em uma única interface gráfica do usuário (GUI) para escrever e testar códigos escrito em diferentes linguagens de programação.

## Principais IDE's PYTHON:





# Pré-requisitos

1º) Fazer download e instalar o Anaconda Python:

<https://www.anaconda.com/products/distribution>

## Anaconda Distribution

[Download](#) 

For Windows

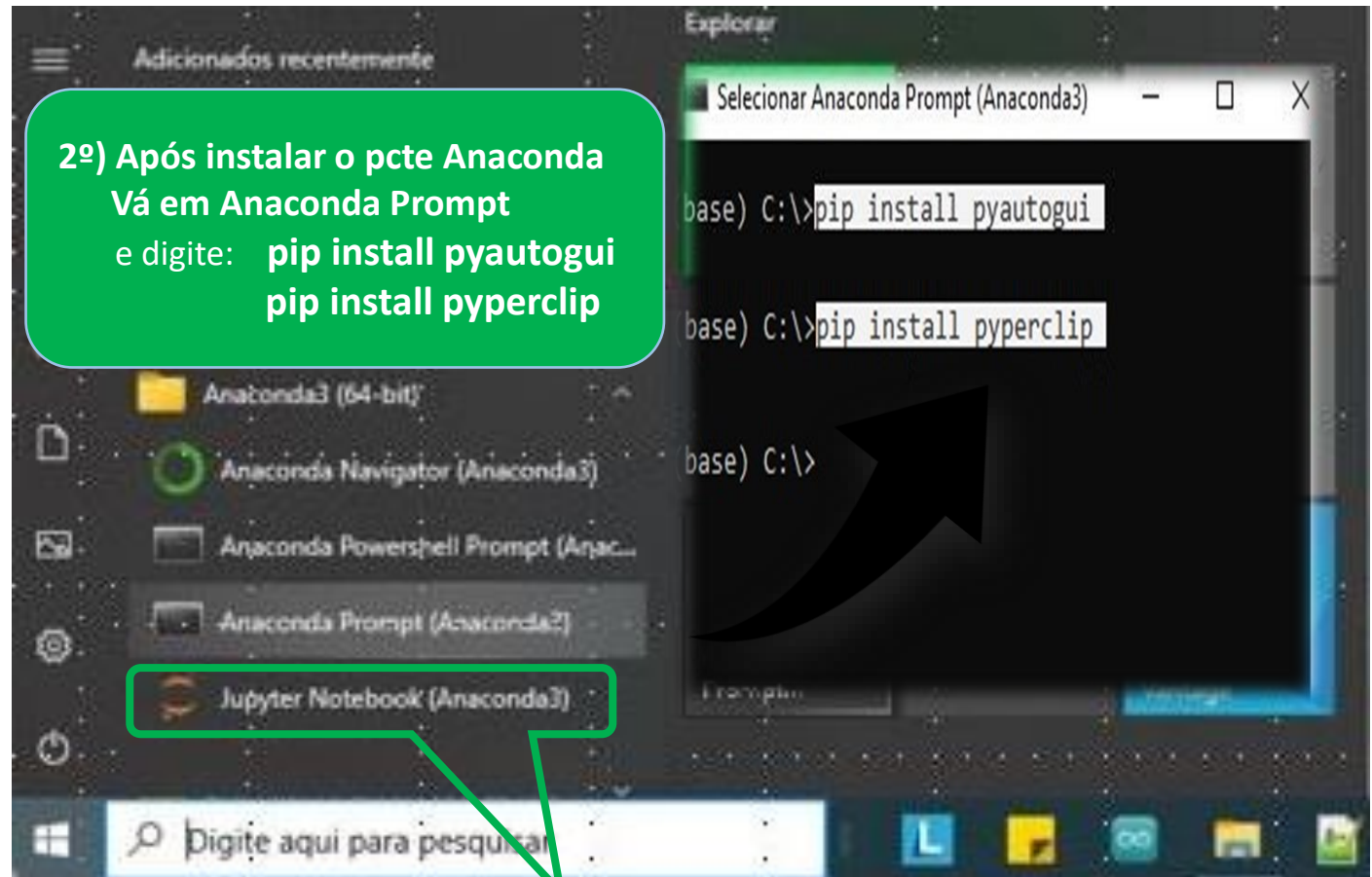
Python 3.9 • 64-Bit Graphical Installer • 594 MB

Get Additional Installers



Get Additional Installers

2º) Após instalar o pte Anaconda  
Vá em Anaconda Prompt  
e digite: **pip install pyautogui**  
**pip install pyperclip**



3º) Após instalar as Bibliotecas  
Você deverá clicar em **Jupyter Notebook**

# INTRODUÇÃO AO PYTHON

A capacidade de representar e abstrair conceitos é fundamental para o nosso pensamento, tornando possível a generalização de coisas e a construção de conceitos cada vez mais complexos. Por isso, na evolução das linguagens de programação, **a necessidade de se representar e manipular informações complexas** resultou no **conceito de classes e objetos**, onde **classes servem como abstrações (representação)** e **objetos seriam instâncias de classes** que mantêm e permitem a manipulação da informação.

Em **Python**, **toda informação** que usamos **é representada na forma de um objeto**. Assim, o número 6 é um objeto da classe int, o número 3.14 é um objeto da classe float, e assim por diante.

Em computação, criar um objeto significa criar uma instância de uma classe. Linguagens orientadas a objetos permitem a definição de novas classes.

**Uma classe é uma abstração de alguma “coisa”**, que possui um estado e comportamento. **Um estado é definido por um conjunto de variáveis chamadas de atributos**. Esses estados podem ser alterados por meio de **“ações” sobre o objeto, que definem seu comportamento**. **Essas ações são funções chamadas de “métodos”**.



# INTRODUÇÃO AO PYTHON

---

O primeiro passo para entender Python é entender que **PYTHON** é fundamentalmente ***Programação Orientada a Objetos***. Por isso precisamos entender os seguintes conceitos:

- Classes
- Objetos
- Propriedades/Atributos
- Métodos

# INTRODUÇÃO AO PYTHON

## Objetos

**São instâncias de uma classe.** Eu e você somos Pessoas, correto? Mas somos Pessoas diferentes. Eu sou eu, você é você. Cada objeto é específico. Isso que difere a classe do objeto.

## Propriedades

**São valores que um objeto possui.** Por exemplo, todas as **pessoas possuem uma altura**.

Logo, **altura é uma propriedade definida dentro da classe Pessoa**. Entretanto, sua altura é diferente da minha. A definição dos atributos é feita na classe, mas a valoração deles é feita no objeto. No código, uma propriedade é basicamente uma variável definida dentro de uma classe.

## Métodos

São similares às propriedades, com a diferença que ao invés de substantivos, são verbos. Por exemplo, **falar é um método da classe Pessoa**, pois pessoas, em grande maioria, falam. Apesar disso, cada pessoa pode falar de uma forma diferente (por exemplo eu posso falar em um idioma e você em outro). Outra analogia que a bibliografia relata é que **um método se comporta como uma função**, mas ele é chamado de uma instância específica. Por exemplo, com uma tartaruga chamada tesss, `tesss.right(90)` pede ao objeto tess para executar o seu método right e virar 90 graus. **Os métodos são acessados usando a notação de ponto.**

# Primeiros Passos

Além do nosso próprio código, existe o compartilhamento de código em forma de módulos e bibliotecas disponibilizados para auxiliar a programação. O uso de bibliotecas já validadas agiliza o desenvolvimento. Para utilizar estas bibliotecas, mais uma vez temos que utilizar o comando **“Import”**.

Para importar um módulo utilizamos o import.

**import package**

```
In [6]: import math  
print(math.sqrt(36))
```

6.0

O código acima importará todos os módulos de math, para importar apenas o necessário utilizamos from.

**from package import item**

O código abaixo importará o módulo sqrt do pacote math.

```
In [7]: from math import sqrt  
print(sqrt(36))
```

6.0

BORGES, Luiz Eduardo. **Python para desenvolvedores: aborda Python 3.3**. Novatec Editora, 2014.

VANDERPLAS, Jake. **Python data science handbook: Essential tools for working with data**. " O'Reilly Media, Inc.", 2016.

## Links úteis:

- ✓ <http://devfuria.com.br/python/imports/>
- ✓ <https://www.upgrad.com/blog/why-learn-python/>
- ✓ <https://spectrum.ieee.org/top-programming-languages-2021#toggle-gdpr>

## REFERÊNCIAS BIBLIOGRÁFICAS BÁSICAS

MONTGOMERY, D. C.; RUNGER, G. C. Estatística Aplicada e Probabilidade para Engenheiros. 6. ed. Rio de Janeiro: LTC, 2016 (ebook, disponível em: Minha biblioteca).

FACELI, K.; LOREBA, A. C.. Inteligência artificial: Uma abordagem de aprendizado de máquina. Brasil: LTC, 2011.

LESKOVEC, J. & others. Mining of massive Datasets. London: Cambridge University Press, 2014.

PAMBOUKIAN, S. V. D.; ZAMBONI, L. C.; BARROS, E. de A. R. Aplicações científicas em C++: da programação estruturada à programação orientada a objetos. 4. ed. São Paulo: Páginas & Letras, 2015. V2. 374 p.

PINOCHET, L.H.C. Tecnologia da Informação e Comunicação. Editora Campus, Rio de Janeiro, 2014.

## REFERÊNCIAS BIBLIOGRÁFICAS COMPLEMENTARES

AHLEMEYER-STUBBE, Andrea; COLEMAN, Shirley. **A practical guide to data mining for business and industry**. John Wiley & Sons, 2014.

GOLDSCHMIDT, Ronaldo Data mining : conceitos, técnicas, algoritmos, orientações e aplicações / Ronaldo Goldschmidt , Eduardo Bezerra. - 2. ed. - Rio de Janeiro : Elsevier, 2015. il. ; 24 cm.

<https://integrada.minhabiblioteca.com.br/#/books/9788595156395/epubcfi/6/2%5B%3Bvnd.vst.idref%3Dcover.html%5D!/4/2%5Bcover-image%5D/2%5Bvst-image-button-65196%5D%400:45.8>

REZENDE, P. A. D. A.; CARLOS RODRIGO DIAS . Regras de Associação Negativas em Mineração de Dados. 1. ed. Saarbrücken Alemanha: Novas Edições Acadêmicas, 2017. v. 1. 47p .

SHIKIDA, Claudio D.; MONASTERIO, Leonardo; NERY, Pedro Fernando. Guia brasileiro de análise de dados: armadilhas & soluções. 2021.



## REFERÊNCIAS BIBLIOGRÁFICAS COMPLEMENTARES

MARCONI, Marina de Andrade et al. **Técnicas de pesquisa**. São Paulo: Atlas, 2002.

MIGUEL, Paulo Augusto Cauchick. Estudo de caso na engenharia de produção: estruturação e recomendações para sua condução. *Production*, v. 17, n. 1, p. 216-229, 2007.

PASSOS, Rosemary; SANTOS, Gildenir Carolino. Como elaborar um relatório técnico científico. Campinas, SP: Biblioteca da Faculdade de Educação; UNICAMP, 2000, ISBN: 85-86091. Disponível em:  
<<https://www.fe.unicamp.br/biblioteca/como-elaborar-um-relatorio-tecnico-cientifico>>. Acesso em: 06 fev. 2020.