

# UM POUCO SOBRE O MASSAKI

## Professor Massaki de O. Igarashi

[massaki.igarashi@gmail.com](mailto:massaki.igarashi@gmail.com)



### Eng. Eletricista (Eletrônica)

Mestre em Engenharia da Informação;

Professor de Linguagem de Programação,

Análise de Dados, Inovação e desenv// de Produtos,

Tecnologia da Informação e Comunicação na Eng de Produção,

Metodologia Científica e Gestão de Inovação (Curso ADM)

Foi Pesquisador e Orientador / Trabalhos de Conclusão de Curso  
no Mackenzie Campinas – SP durante 10 anos (de 2014 até 2023)

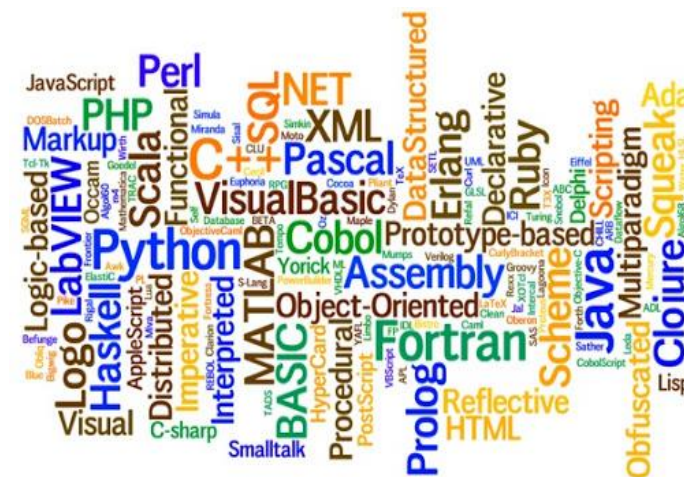
Experiência c/ instrumentação analítica e Pesquisa e Desenvolvimento  
Desenv// de equipamentos p/ análises químicas e petroquímicas.



# Lógica de Programação

Prof. Ms. Massaki de O. Igarashi

[massaki.igarashi@gmail.com](mailto:massaki.igarashi@gmail.com)



# Agenda desta apresentação!

---

01

- Ementa/Propósito desta Disciplina

02

- **Motivação** / Por que estudar Lógica de Programação?

03

- Compiladores x Interpretadores

04

- Definições e Conceitos Básicos de **Lógica e Narrativa**

05

- **Como aprender a programar?**

# EMENTA/Propósito

---

Estudar e entender conceitos de **algoritmos** e sua importância na computação; desenvolver algoritmos usando: **Narrativa, Pseudocódigo, Fluxogramas**; e codificar em Linguagem de programação (Python).

Desenvolver **Lógica** de Programação. Estudo dos Elementos básicos de programação: **variáveis e tipos de dados; entrada e saída de dados. Compreender e aplicar estruturas de controle de fluxo (condicionais e laços de repetição)**; além de introduzir **conceitos básicos de estruturas de dados como vetores e matrizes (manipulação de arranjos estáticos)**, resolver problemas utilizando a lógica de programação e incentivar o desenvolvimento do pensamento computacional.

# Motivação: Algumas frases para inspirar você a programar!

---

“Todos neste país deveriam aprender a programar um computador porque isto te ensina como pensar” (do inglês *“Everybody in this country should learn how to program a computer because it teaches you how to think”* - **Steve Jobs**).

“Se você deseja descobrir os segredos do universo ou apenas seguir uma carreira no século 21, a programação básica de computadores é uma habilidade essencial para aprender” (do inglês *“Whether you want to uncover the secrets of the universe, or you just want to pursue a career in the 21st century, basic computer programming is an essential skill to learn”* - **Stephen Hawking**).












# Justificativa: 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

# Introdução: 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++.

# Introdução: Interpretador x Compilador

Os **interpretadores** passam pelo programa **linha a linha** e **executam cada comando**. Enquanto os **Compiladores** **traduzem todo o programa** escrito para formato no qual o computador entenda!

**Tabela 1** – Vantagens e Desvantagens dos Interpretadores e Compiladores

	INTERPRETADORES	COMPIADORES
Vantagens	As linguagens interpretadas tendem a ser <b>mais flexíveis</b> ; já que oferecem recursos como <b>digitação dinâmica</b> e <b>tamanho reduzido de programa</b>	Os programas compilados tendem a ser <b>mais rápidos</b> . Pois o processo de traduzir o código em tempo de execução aumenta o tempo do processo.
Desvantagens	A desvantagem mais notável é a <b>menor velocidade</b> de execução em comparação com as linguagens compiladas.	<b>Tempo adicional</b> necessário para concluir toda a <b>etapa de compilação</b> antes dos testes; <b>dependência da plataforma</b> do código binário gerado.
	Ex: PHP, Ruby, Python, R, JS.	Ex: C, C++, C#, Visual Basic

O acrônimo **IDE** (***Integrated Development Environment***) significa software ou ambiente de desenvolvimento integrado que une ferramentas de desenvolvimento em uma única interface gráfica do usuário para escrever e testar códigos escritos em diferentes linguagens de programação.

Interpretadores  
Python:





# DEFINIÇÃO: Linguagem de Programação

Conforme o dicionário explica, **Linguagem** é qualquer meio sistemático (sistema de símbolos ou código) capaz de comunicar ideias ou sentimentos.

Por isso,

**Linguagem de Programação** tem o objetivo de **comunicar, dar instruções à um computador ou máquina** para atingir determinada finalidade.



**A linguagem de Programação é** um conjunto de **instruções lógicas e símbolos** escritas em um código fonte que permite a nós humanos traduzirmos nossos pensamentos em instruções **que os computadores possam entender** (já que a eletrônica é essencialmente binária). Este código pode ser compilado e transformado em um programa de computador, ou usado como script interpretado; que informará **instruções de processamento ao computador.**



# CONCEITOS BÁSICOS: Lógica

E o que são **instruções lógicas**?

Correspondem à **sequência de** passos (**ações**) necessários para que o programa cumpra seu objetivo. Nesse sentido, é importante um breve entendimento sobre **Lógica de Programação**.

## Lógica de Programação

Basicamente, **nossas ações costumam seguir uma sequência lógica**; mas sequer damos atenção a isso! Por exemplo, ao fazer uma análise do nosso cotidiano, perceberemos que **nossas ações são consequência de uma cadeia de outras ações menores** que nos levaram até uma atitude final.

**Exemplo:** O simples **ato de acordar até tomamos café da manhã**.

Então qual seria a **sequência de ações menores** que efetuamos ao

1. **Acordar;**
2. **Prepararmos o café com auxílio de uma cafeteira elétrica;**
3. **Colocarmos o café numa caneca e finalmente**
4. **tomamos o café?**

Ao detalhar este processo somos capazes de estipular uma sequência de ações menores que nos levaram ao ato final :

**Ao acordar:**

1. Me espreguiço e abro os olhos;
2. Levanto-me da cama;
3. Calço os chinelos;
4. Caminho até o corredor;
5. Sigo pelo corredor até a cozinha;

Na cozinha,

6. Pego os recipientes do pó de café e do açúcar no armário;
7. Coloco os recipientes ao lado da cafeteira;
8. Pego uma colher de sopa e de café na gaveta;
9. Com a colher de sopa, coloco o pó de café dentro da cafeteira;
10. Pego um copo com água
11. Coloco água no compartimento específico;

Após inserir todos os ingredientes na máquina:

12. Aperto o botão de ligar;

Quando o café está pronto:

13. Pego a caneca e pires;
14. Despejo o café dentro de uma caneca;
15. Adoço o café;
16. Coloco a caneca com café sobre o pires junto à colher de café;

**Bebo o café.**

# DEFINIÇÃO: Algoritmo

---

“ Um **algoritmo** corresponde à sequência de passos necessários para se atingir um objetivo; ou seja, é a **organização clara e precisa de uma sequência de instruções** voltadas à resolução de um problema específico. ”

**Assim,**

O primeiro passo para qualquer programador é criar o algoritmo e avaliar se o resultado obtido condiz com a solução esperada. Em seguida, define-se a linguagem de programação a ser utilizada na implementação (codificação) do algoritmo documentado.

Em outras palavras, entender e dominar a lógica de programação é a porta de entrada para tornar-se um(a) programador(a), seja em *front-end* ou em *back-end*.

# Introdução: Pensar de maneira lógica possibilita

---

- Desenvolver códigos mais eficientes.
- Melhor resolução de problemas do cotidiano; já que facilita dividir problema complexo em pequenas partes, e, portanto, auxilia a encontrar uma solução mais eficaz;;
- Ajuda na concentração; pois quanto mais claras e ordenadas as ações melhor será a concentração
- Entender como a tecnologia em geral funciona; já que todos os processos existentes em TI dependem de um código que os sustenta.

# Introdução: Como aprender a Programar?

1 - Compreenda

2 - Retenha

3 – Pratique

4 – Dissemine

5 - Crie

## DICAS IMPORTANTES:

- ❖ Não tenha medo de errar! Errar é normal e todos erram, principalmente no começo de um aprendizado ou de uma nova profissão; isso é também importante para sua evolução e crescimento.
- ❖ Nunca desista diante da primeira dificuldade; pergunte, peça auxílio! Como em qualquer jornada de aprendizado você enfrentará dificuldades; porém, não desista, o processo pode até ser longo e cansativo, mas saiba que superar uma dificuldade e alcançar um objetivo é algo que certamente contribuirá para o crescimento pessoal.
- ❖ Seja proativo(a); converse com colegas da área, participe de grupos e comunidades sobre o assunto; isto o ajudará a continuar evoluindo!
- ❖ Estude constantemente e revise os conhecimentos sempre que possível.
- ❖ Participe e desfrute o máximo desta aprendizagem de algo novo; você certamente vivenciará experiências únicas e provavelmente terá a chance de praticar e evoluir suas habilidades socioemocionais (muito valorizadas no mercado de trabalho atualmente). A comunicação e a facilidade de trabalhar em grupo são habilidades muito valorizadas hoje.



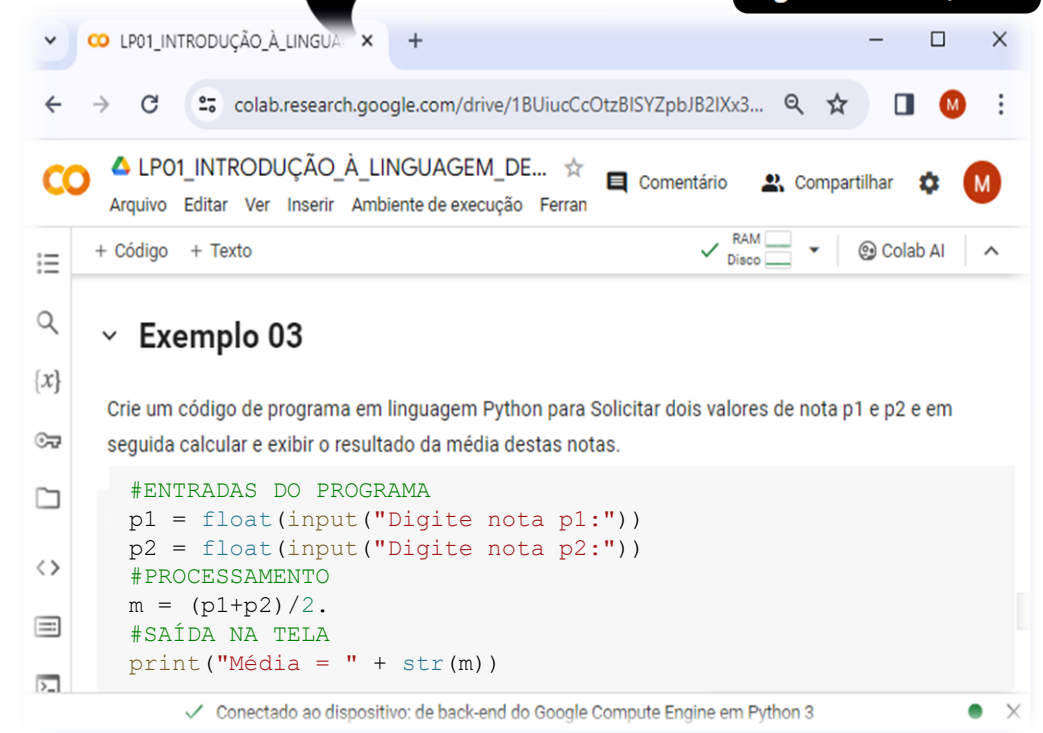
# PRÓXIMA AULA PRATICAREMOS!

Fazer a Narrativa, o Fluxograma, o Algoritmo e código de um programa em Linguagem de Programação Python!



Aponte seu celular e Digitalize este QRcode

NARRATIVA	FLUXOGRAMA	ALGORITMO	LINGUAGEM PYTHON
	INÍCIO	Início real: p1, p2, m	
Ler nota prova p1	Ler p1	Ler (p1)	<code>p1 = float(input("Digite p1:"))</code>
Ler nota prova p2	Ler p2	Ler (p2)	<code>p2 = float(input("Digite p2:"))</code>
Calcular a Média ( $m = (p1+p2)/2$ )	Calcular $m \leftarrow (p1+p2)/2$	$m \leftarrow (p1+p2)/2.$	<code>m = (p1 + p2)/2.0</code>
Exibir a média calculada	Exibir m	Escrever (m)	<code>print(m)</code>
	FIM	Fim	



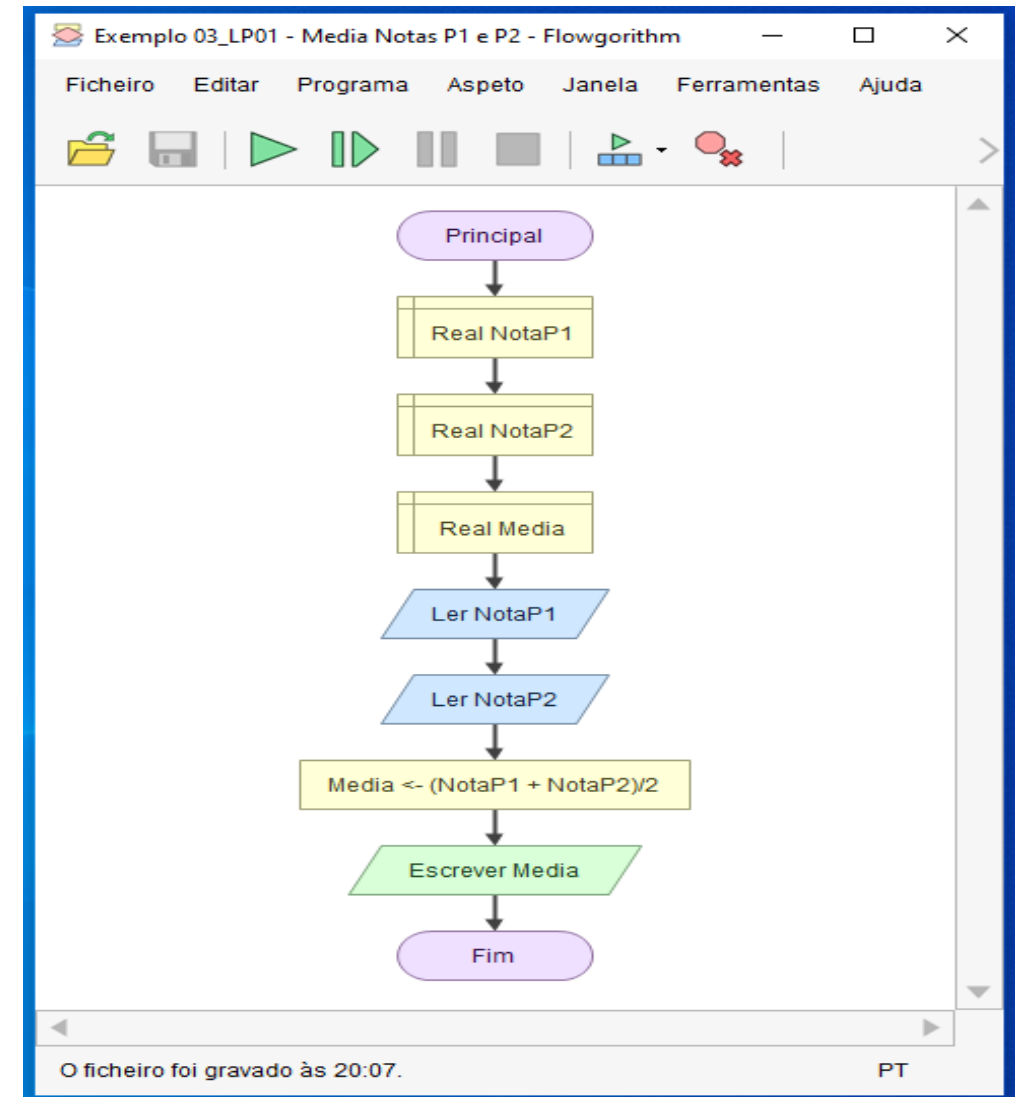
```
#ENTRADAS DO PROGRAMA
p1 = float(input("Digite nota p1:"))
p2 = float(input("Digite nota p2:"))
#PROCESSAMENTO
m = (p1+p2)/2.
#SAÍDA NA TELA
print("Média = " + str(m))
```

# Ferramenta Útil!



**Flowgorithm:** permite aos iniciantes programarem baseado em fluxogramas gráficos. Isto permite ao aluno se concentrar nos conceitos de programação, em vez de em todas as nuances de uma linguagem de programação típica. Os programas podem ser executados diretamente no Flowgorithm. Link para acesso: <http://flowgorithm.org/>

NARRATIVA	FLUXOGRAMA	ALGORITMO	LINGUAGEM PYTHON
	<b>INÍCIO</b>	Início real: p1, p2, m	
<b>Ler</b> nota prova p1	<b>Ler p1</b>	Ler (p1)	<b>p1</b> = float(input("Digite p1:"))
<b>Ler</b> nota prova p2	<b>Ler p2</b>	Ler (p2)	<b>p2</b> = float(input("Digite p2:"))
<b>Calcular</b> a Média ( $m = (p1+p2)/2$ )	<b>Calcular</b> $m \leftarrow (p1+p2)/2$	$m \leftarrow (p1+p2)/2$	<b>m</b> = (p1 + p2)/2.0
<b>Exibir</b> a média calculada	<b>Escrever media</b>	Escrever (m)	<b>print(m)</b>
	<b>FIM</b>	Fim	



# REFERÊNCIAS BIBLIOGRÁFICAS

---

## Básicas

ANDREW BRUCE, Peter Bruce, Estatística Prática para Cientistas de Dados. Rio de Janeiro. Alta Books, 2019.

BARRY, Paul. Use a cabeça! Python: um guia amigo do cérebro / Paul Barry; traduzido por Eveline Machado. Rio de Janeiro: Alta Books, 2019. 616 p.; il.; 17cm x 24 cm.

PAMBOUKIAN, S. V. D.; ZAMBONI, L. C.; BARROS, E. de A. R. Introdução à Linguagem Python. 1. ed. São Paulo: Editora Mackenzie, 2022. V1. 239 p.

TAVARES NETO, Roberto Fernandes. Introdução à Programação para Engenharia - Usando a Linguagem Python. Rio de Janeiro, LTC, 2022.

## Complementares

ALVES, William Pereira. Programação Python: aprenda de forma rápida. Rio de Janeiro: Editora Expressa, 2021. 150 p.; il.;

CHEN, Daniel Y. Análise de Dados com Python e Pandas. São Paulo: Novatec, 2019.

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

FERREIRA, Ronaldo Domingues. Linguagem de programação. Curitiba: Contentus, 2020. 56 p.: il.

FORBELLONE, André Luiz Villar; EBERSPACHER, Henri Frederico. Lógica de Programação: a construção de algoritmos e estruturas de dados com aplicações em Python – 4ª ed. Porto Alegre: Pearson, 2022. 331 p.

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

MCKINNEY, Wes. Python para Análise de Dados. São Paulo: Novatec, 2019.

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

MUELLER, John Paul . Começando a Programar em Python Para Leigos. Rio de Janeiro. Alta Books, 2020.

SOUZA, M. A. F.; GOMES, M. M.; SOARES, M. V.; CONCILIO, R. Algoritmos e Lógica de Programação. 2. Ed. São Paulo: Cengage Learning, 2012. 262 p.