

# Algoritmos e Lógica de Programação

80 horas // 4 h/semana

---

*Apresentação da  
Disciplina*

**Aula 01**  
Prof. Piva

# Conteúdo Programático

---

- Método para desenvolvimento de algoritmos; A lógica e os algoritmos; O raciocínio e as formas de resolução de problemas.
- Linearização de expressões matemáticas; Expressões lógico-matemáticas; Tipo de dados.
- Estrutura sequencial; Estrutura condicional simples e composta.
- Estrutura de repetição. Vetores, Matrizes e Strings.
- Modularização de algoritmos (procedimentos e funções)
- Testes de Software. Repositório de SW.

# Bibliografia...

---

## **BÁSICA:**

CORMEN, T. H. et al. Algoritmos. Rio de Janeiro: Campus, 2012.

MANZANO, J. A. N. G; OLIVEIRA, J. F. Algoritmos: Lógica para desenvolvimento de programação de computadores. São Paulo: Érica, 2009.

MEDINA, M., FERTIG, C. Algoritmos e Programação: Teoria e Prática. São Paulo: Novatec, 2006

## **COMPLEMENTAR:**

DEITEL, H; DEITEL, P. C: Como programar. 6 ed. São Paulo: Pearson, 2011.

BIANCHI, F. et al. Algoritmos e programação de computadores. Rio de Janeiro: Campus, 2012.

SOUZA, M. A. F. et al. Algoritmos e Lógica de Programação. São Paulo: Cengage Learning, 2019.

MENEZES, O. Introdução à Programação Com Python: Algoritmos e Lógica De Programação para iniciantes. 3 ed. São Paulo: Novatec. 2019.

SILVERMAN, R. E. Git: Guia prático. São Paulo: Novatec, 2019.

# Bibliografia...

## BÁSICA:

**CORMEN, T. H. et al. Algoritmos. Rio de Janeiro: Campus, 2012.**

MANZANO, J. A. N. G; OLIVEIRA, J. F. Fundamentos de programação para computadores. São Paulo: Pearson, 2012.  
MEDINA, M., FERTIG, C. Algoritmos. São Paulo: Pearson, 2006

## COMPLEMENTAR:

DEITEL, H. DEITEL, P. C. Como programar.



Thomas Cormen  
Algoritmos - Teoria e Prática



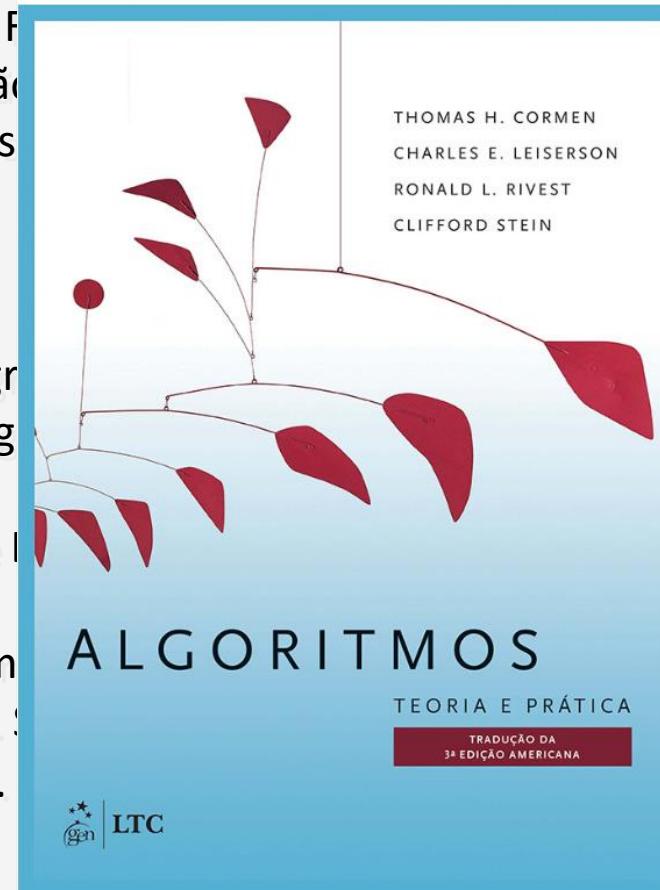
190 classificações

SOUZA, MENEZE Capa comum: R\$ 384,90

Programar Adicionar ao carrinho

SILVERMAN Vendido e enviado por Amazon.com.br.

Veja mais opções de compra



limento de  
São Paulo: Novatec,  
011.  
Janeiro: Campus,  
: Cengage Learning,  
ógica De

# Bibliografia...

## BÁSICA:

CORMEN, T. H. et al. Algoritmos. Rio de Janeiro: Campus, 2012.

MANZANO, J. A. N. G; OLIVEIRA, J. F. Algoritmos: Lógica para desenvolvimento de programação de computadores. São Paulo: Érica, 2009.

MEDINA, M., FERTIG, C. Algoritmos e Programação: Teoria e Prática. Rio de Janeiro: LTC, 2006

## COMPLEMENTAR:

DEITEL, H; DEITEL, P. C: Como Programar em C. São Paulo: Pearson, 2012.

BIANCHI, F. et al. Algoritmos e Estruturas de Dados. São Paulo: Pearson, 2012.

SOUZA, M. A. F. et al. Algoritmos e Estruturas de Dados. São Paulo: Pearson, 2019.

MENEZES, O. Introdução à Programação para iniciantes. São Paulo: Pearson, 2019.

SILVERMAN, R. E. Git: Guia prático. São Paulo: Pearson, 2019.



# Bibliografia...

## BÁSICA:

CORMEN, T. H. et al. Algoritmos. Rio de Janeiro: Campus, 2012.

MANZANO, J. A. N. G; OLIVEIRA, J. F. Algoritmos: Lógica para desenvolvimento de programação de computadores. São Paulo: Érica, 2009.

**MEDINA, M., FERTIG, C. Algoritmos e Programação: Teoria e Prática. São Paulo: Novatec, 2006 (esgotado)**

## COMPLEMENTAR:

DEITEL, H; DEITEL, P. C: Como programar. 6 ed. São Paulo: Pearson

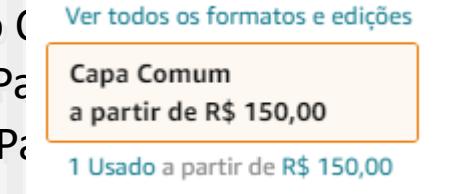
BIANCHI, F. et al. Algoritmos e programação de computadores. Rio de Janeiro: Elsevier, 2012.

SOUZA, M. A. F. et al. Algoritmos e Lógica de Programação. São Paulo: Pearson, 2019.

MENEZES, O. Introdução à Programação Com C. São Paulo: Pearson, 2019.

Programação para iniciantes. 3 ed. São Paulo: Pearson, 2019.

SILVERMAN, R. E. Git: Guia prático. São Paulo: Novatec, 2019.



# Bibliografia...

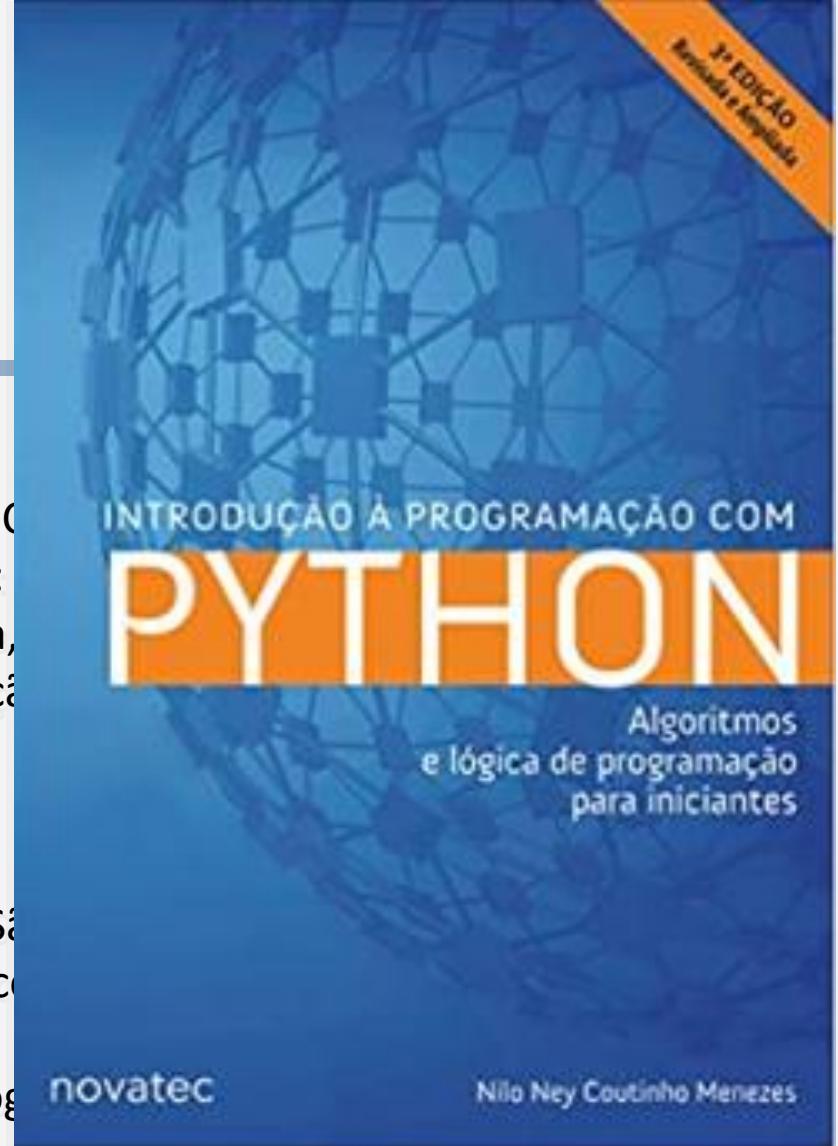
## BÁSICA:

CORMEN, T. H. et al. Algoritmos. Rio de Janeiro: Cengage Learning, 2012.  
 MANZANO, J. A. N. G; OLIVEIRA, J. F. Algoritmos: programação de computadores. São Paulo: Érica, 2006.  
 MEDINA, M., FERTIG, C. Algoritmos e Programação. São Paulo: Pearson, 2006.

## COMPLEMENTAR:

DEITEL, H; DEITEL, BIANCHI, F. et al. A Practical Approach to Computer Programming. Pearson, 2012.

SOUZA, M. A. F. et al. Git: Guia Prático. São Paulo: Novatec, 2019.

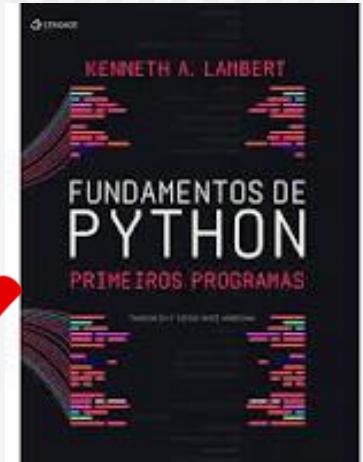


MENEZES, O. Introdução à Programação Com Python: Algoritmos e Lógica De Programação para iniciantes. 3 ed. São Paulo: Novatec. 2019.  
 SILVERMAN, R. E. Git: Guia prático. São Paulo: Novatec, 2019.

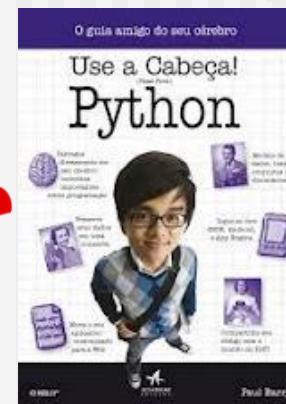
# Bibliografia em Python...

## Linguagem:

LAMBERT, K. A. Fundamentos de Python: primeiros programas. Editora CENGAGE, 2022



BARRY, P. Python. Use a Cabeça! Editora Altabooks. 2018



## Algoritmos:

BIANCHI, F. et al. Algoritmos e programação de computadores. 2<sup>a</sup>.Ed. RJ: Campus/Elsevier, 2012.

# Forma de Avaliação

<b>Instrumentos de Avaliação</b>	<b>Cronograma</b>	<b>Pesos</b>
Nota 1: Prova 1	<b>25/04</b>	35%
Nota 2: Prova 2	<b>13/06</b>	45%
Nota 3: Exercícios	<b>28/03 e 23/05</b>	20%
Nota 4: Avaliação Substitutiva	<b>27/06</b>	P1 ou P2

## Atenção:

A **avaliação substitutiva**, é aplicada aos estudantes que por ventura venham a perder alguma das duas avaliações presenciais (P1 ou P2). Existe a necessidade de comprovação formal do motivo da ausência.

# Algoritmos...

**Site / Blog da Disciplina:**

**http://www.piva.pro.br/**

## Algoritmos e Lógica de Programação

quarta-feira, 12 de julho de 2017

### Semana 00 - Orientações Gerais

#### Aulas

As aulas acontecerão nos dias e horários planejados no Sistema Acadêmico (SIGA). Por ser um curso presencial, exige a presença em pelo menos 75% dos 20 encontros (80 aulas) planejadas. Programação oficial da disciplina (igual o lançado no sistema SIGA) - <[aqui](#)>

#### Livros Textos

Qualquer livro de linguagem C pode ser utilizado/consultado. Essencialmente, utilizarmos como base, dois livros didáticos. São eles:

#### Fundamentos da Programação de Computadores: algoritmos, Pascal, C/C++ e Java.

Editora Logman.

Autoras: Ascencio, A.F.G. e Campos, E.A.V.

Quem sou eu  
 Prof. Dr. Dilermando Piva Jr.  
[Seguir](#) 337  
[Visualizar meu perfil completo](#)

Arquivo do blog  
 ▾ 2017 (21)  
 ▾ Julho (21)  
[Semana 00 - Orientações Gerais](#)  
[Semana 01](#)  
[Semana 02](#)  
[Semana 03](#)

Prof. Dr. Dilermando Piva Jr.

Principal Sobre Contato SisCoP

Sistemas M\_processados e M\_controlados  
 Disciplina no curso de Mecatrônica da Fatec Itu (4º Sem - Manhã e Noite). Foco em Microcontroladores.

Projetos de TI-1  
 Disciplina no curso de GTI da Fatec Itu (5º Sem - Manhã e Noite). Acompanhamento do desenvolvimento do TCC.

Métodos para Produção do Conhecimento  
 Disciplina no curso de GE da Fatec Itu (2º Sem - Manhã)

Info. Aplic. à Gestão da Qualidade  
 Disciplina no curso de Gestão da Qualidade na Fatec Sorocaba (3º sem - Manhã)

Algorítmos e Lógica de Programação  
 Disciplina no curso de ADS da Fatec Indaiatuba (1º Sem - Noite)

Projetos em Mecatrônica 1  
 Disciplina no curso de Mecatrônica da Fatec Itu (5º Sem - Manhã)

Sistemas Operacionais  
 Disciplina nos cursos de ADS e GTI da Fatec Itu

Ambientes Virtuais de Aprendizagem  
 Disciplina no curso de GE da Fatec Itu (5º Sem - Manhã)



# O que é um Algoritmo?



# O que é um Algoritmo?

---

Uma resposta de âmbito geral seria...

**“um conjunto de etapas para executar uma tarefa!”**

A vida é feita de algoritmos.

**Um algoritmo para escovar os dentes.**



# Algoritmo para escovar os dentes...

---

1. Abrir o tubo de pasta dental
2. Pegar a escova de dentes
3. Apertar o tubo de pasta dental sobre a escova e aplicar a qtd necessária de dentífricio
4. Fechar o tubo
5. Colocar a escova em um quadrante da boca
6. Movimentá-la para cima e para baixo durante alguns segundos etc.



# Outros exemplos de algoritmos em nossas vidas

---

- Se você pega ônibus ou metrô ou trem para ir trabalhar, terá um algoritmo.
- Se desejar fazer um sanduiche...
- Se tiver que trocar um pneu furado...
- Se for tomar banho...

E assim por diante!

# Algoritmos de Computadores

---

- **Você já utilizou um GPS para determinar uma rota de viagem?**
  - O aparelho executa um algoritmo denominado algoritmo de “caminho mínimo” para determinar essa rota!!
- **Você já fez ou faz compras na Internet?**
  - Você deve (ou deveria) ter utilizado um site seguro, que utiliza um algoritmo para criptografar e proteger os dados.
  - Tem também algoritmos que indicam possibilidades de produtos que você gostaria de comprar (algoritmos de recomendação)
  - Esses algoritmos são executados em todos os lugares, em laptops, tablets, celulares, servidores etc.

# Qual a diferença entre esses tipos de Algoritmos?

---

Algoritmo que você executa  
vs.

Algoritmo que um computador  
executa

?

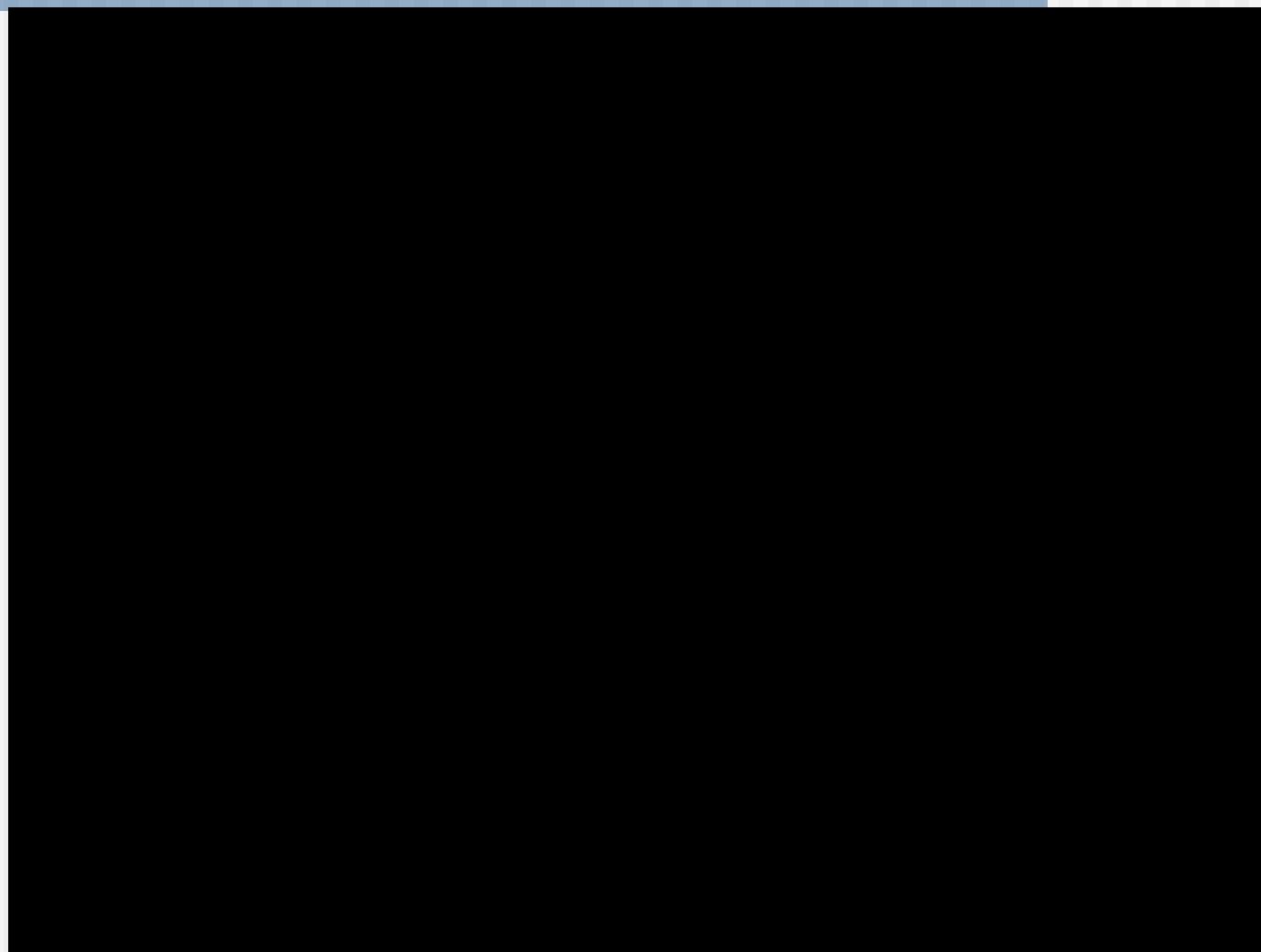


# Algoritmo Computacional

---

- Um algoritmo de computador é um **conjunto de etapas** para executar uma tarefa descrita com **precisão suficiente** para que o computador possa executá-la!!

# Exemplo de Linguagem...



# Linguagens de Programação

## PASCAL

```
program Hello;
var mensagem : string;
begin
  mensagem := 'Hello World!';
  write(mensagem);
End.
```

## JAVA

```
public class Main {
  public Main(){
    System.out.println("Hello World");
  }
  public static void main(String [] args){
    Main m =new Main();
  }
}
```

## C

```
#include <stdio.h>
#include <stdlib.h>

int main()
{
  printf("HELLO WORLD!!!!");
  return(0);
}
```

## COBOL

```
function Hello (){
  alert("Hello World!")
}
```

# Linguagens de Programação

## PASCAL

```
program Hello;
var mensagem : string;
begin
  mensagem := 'Hello World!';
  write(mensagem);
End.
```

## JAVA

```
public class Main {
    public Main(){
        System.out.println("Hello World");
    }
    public static void main(String [] args){
        Main m =new Main();
    }
}
```

## C

```
#include <stdio.h>
#include <stdlib.h>

int main()
{
    printf("HELLO WORLD!!!!");
    return(0);
}
```

## COBOL

```
function Hello (){
    alert("Hello World!")
}
```

## PYTHON

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

# Qual linguagem utilizar?

■ <https://www.tiobe.com/tiobe-index/>

Jan 2023	Jan 2022	Change	Programming Language	Ratings	Change
1	1		 Python	16.36%	+2.78%
2	2		 C	16.26%	+3.82%
3	4		 C++	12.91%	+4.62%
4	3		 Java	12.21%	+1.55%
5	5		 C#	5.73%	+0.05%
6	6		 Visual Basic	4.64%	-0.10%
7	7		 JavaScript	2.87%	+0.78%
8	9		 SQL	2.50%	+0.70%
9	8		 Assembly language	1.60%	-0.25%
10	11		 PHP	1.39%	-0.00%

# Qual linguagem utilizar?

■ <https://www.tiobe.com/tiobe-index/>

Jan 2023	Jan 2022	Change	Programming Language	Ratings	Change
1	1		 Python	16.36%	+2.78%
2	2		 C	16.26%	+3.82%
3	4		 C++	12.91%	+4.62%
4	3		 Java	12.21%	+1.55%
5	5		 C#	5.73%	+0.05%
6	6		 Visual Basic	4.64%	-0.10%
7	7		 JavaScript	2.87%	+0.78%
8	9		 SQL	2.50%	+0.70%
9	8		 Assembly language	1.60%	-0.25%
10	11		 PHP	1.39%	-0.00%

# Algoritmo Computacional

---

- Um algoritmo de computador é um conjunto de etapas para executar uma tarefa descrita com **precisão suficiente** para que o computador possa executá-la!!

# O que se pretende de um algoritmo de computador?

---

Algoritmos computacionais são feitos para resolver problemas...

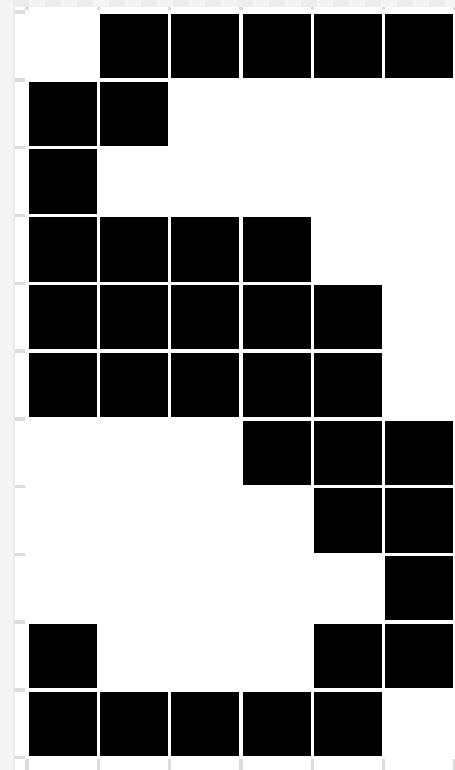
- O algoritmo deve fornecer uma solução **correta** para o problema
- O algoritmo deve usar recursos computacionais **eficientemente** ao resolver o problema.

# Solução correta...

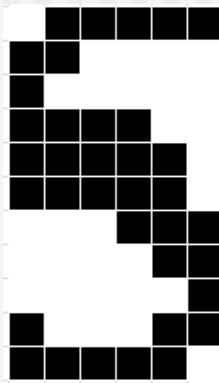
---

- O que significa produzir uma solução correta para um problema?
- **GPS**
  - Menor rota?
  - Mais rápido?
  - Mais barato (sem pagar pedágio)?

# Solução correta?



# Solução correta?



Um algoritmo de reconhecimento de caracteres ópticos, por exemplo.

**É um 5 ou um S?**

Podemos dizer que esse algoritmo produz um **resultado incorreto**?

# Sempre correta?

---

- Um algoritmo computacional sempre nos dará um resultado correto?
- Podemos aceitar um algoritmo que pode produzir uma resposta incorreta?

?

# Sempre correta?

---

- Um algoritmo computacional sempre nos dará um resultado correto?
- Podemos aceitar um algoritmo que pode produzir uma resposta incorreta?
- **DESDE QUE POSSAMOS CONTROLAR A FREQUÊNCIA COM QUE ISSO ACONTECE.... SIM!**

# Sempre correta?

- Um algoritmo computacional sempre nos dará um resultado correto?
- Podemos aceitar um algoritmo que pode produzir uma resposta incorreta?
- DESDE QUE POSSAMOS CONTROLAR A FREQUÊNCIA COM QUE ISSO ACONTECE.... SIM!

## Exemplo

### O Criptossistema RSA

- Dado um numero N (*grande*) determina se é PRIMO ou NÃO.
- 1 erro a cada  **$2^{50}$**  vezes
- Mais de um trilhão de vezes
- 1.125.899.906.842.620

# O que se pretende de um algoritmo de computador?

---

Algoritmos computacionais são feitos para resolver problemas...

- O algoritmo deve fornecer uma solução **correta** para o problema 
- O algoritmo deve usar recursos computacionais **eficientemente** ao resolver o problema.

# Eficientemente?

---

- O que significa um algoritmo usar recursos computacionais eficientemente?
- Pense novamente no GPS...
- Se ele demorar uma hora para determinar qual rota ele recomenda, você o usaria?

# Eficientemente?

---



## Tempo!!

É a medida principal de eficiência que usamos para avaliar um algoritmo.

# Eficientemente?

---

- Quantidade de memória (RAM limitada)
- Comunicação em rede (necessita de informações que estão em outro local)
- Operações em disco (HD, SSD...)

# Pergunta que não quer calar....

---

Por que eu tenho que me **preocupar** com ALGORITMOS de computador (ou computacionais)?



# Algoritmos de computador?



# Algoritmos de computador?

Muitas oportunidades!



[CODE.ORG](#)

Em 2030 haverá uma demanda de  
1.4 milhão de programadores

→ ritmo atual: 400 mil

1.000.000 de vagas



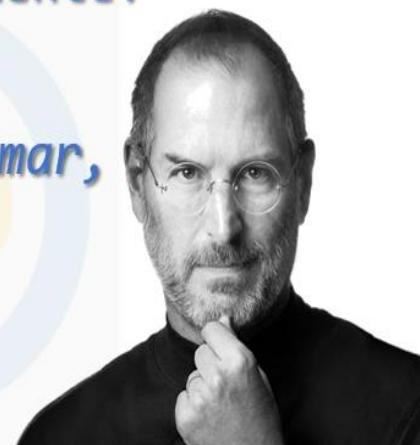
# Algoritmos de computador?

- Não só desktops ou notebooks
- Existe uma infinidade de dispositivos:
  - Celulares
  - Relógios (Smartwatch)
  - TVs Inteligentes
  - Carros inteligentes
  - Equipamentos usáveis
  - IoT (Internet das Coisas)
  - Servidores
  - Programação em nuvem...

# Algoritmos de computador?

- Aprender a programar faz bem para nossa vida!
- Nos ajuda a pensar melhor... Mais logicamente... Raciocínio mais rápido.
- Diferencial (social e profissional)
- Nos ajuda a resolver problemas mais rapidamente!

*“Todos nesse país deveriam aprender a programar,  
pois isso nos ensina a pensar.”* Steve Jobs



# Algoritmos de computador?

- VELHO ou JOVEM
- HOMEM ou MULHER

Não existe idade, raça, gênero ...

Existe sim...

- FORÇA DE VONTADE
- OBJETIVO
- DETERMINAÇÃO!

# Maiores Obstáculos?

---

- Falta de compromisso / objetivo
- Ter pressa
  - Aprender um novo idioma  
(não adianta ler o dicionário todo!!)
- Não praticar (fazer os exercícios/desafios)
- Dedicação do início ao fim
- Importância dos fundamentos (básico)
- Desafios
  - Eles te farão crescer

# O que é uma L.P.?



# O que é uma L.P.?



# O que é uma L.P.?



# O que é uma L.P.?

## *Compilada vs Interpretada*

---

### Linguagens Compiladas

- Do latim *compilare*
- Significa REUNIR, AJUNTAR
- Fonte traduzido diretamente para linguagem de máquina

# O que é uma L.P.?

## *Compilada vs Interpretada*

### Processo de Compilação

Programa (Alto nível)

Análise léxica

Análise sintática

Geração de código (Baixo nível)

Programa (Objeto executável)

# O que é uma L.P.?

## *Compilada vs Interpretada*

### Processo de Compilação

- Análise léxica

Reconhece as sequências de símbolos que representam uma unidade: o nome de uma variável, constante e palavras de instrução (while, for...)

pos = ini + val \* 60

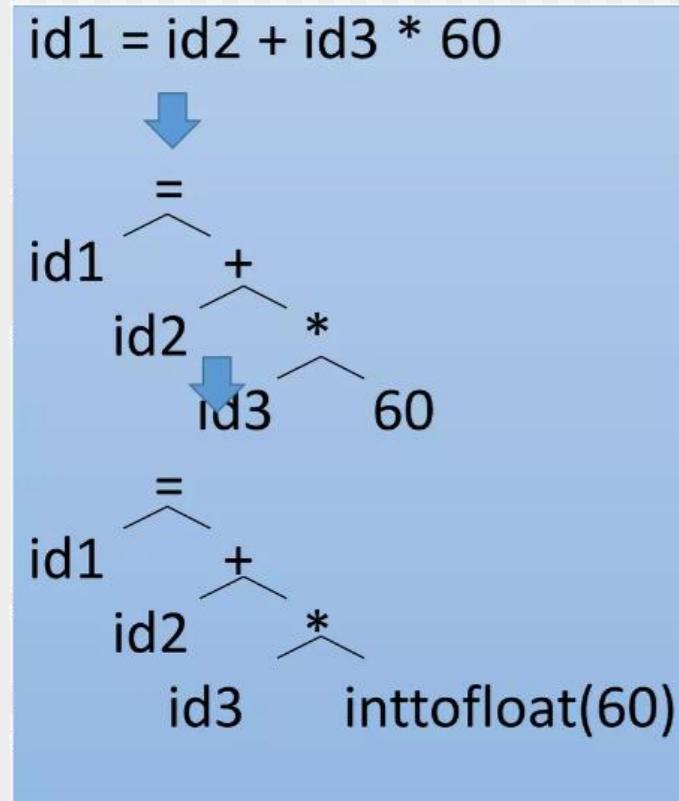
id1 = id2 + id3 \* 60

# O que é uma L.P.?

## *Compilada vs Interpretada*

### Processo de Compilação

- Análise sintática  
Identifica a estrutura gramatical do programa e o papel de cada componente.  
É construída uma *árvore sintática* (binária, estruturalmente) e uma tabela de símbolos que representam as variáveis.

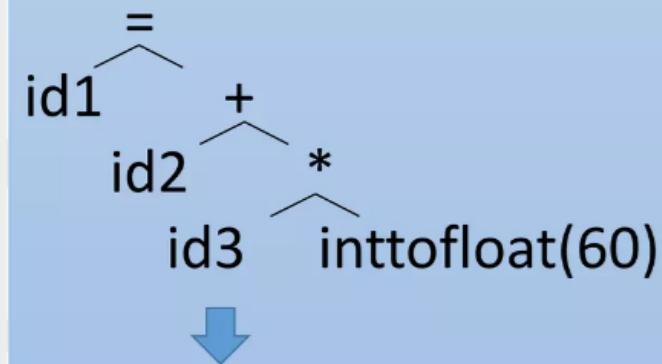


# O que é uma L.P.?

## *Compilada vs Interpretada*

### Processo de Compilação

- Geração de código  
Processo de construir instruções da linguagem de máquina (em assembly) que simulam as instruções reconhecidas na análise sintática.



```
temp1 = inttofloat(60)
temp2 = id3 * temp1
temp3 = id2 + temp2
id1 = temp3
```

↓ //otimização

```
temp1 = id3 * 60.0
id1 = id2 + temp1
```

# O que é uma L.P.?

## *Compilada vs Interpretada*

### Processo de Compilação

- Geração de código  
Processo de construir instruções da linguagem de máquina (em assembly) que simulam as instruções reconhecidas na análise sintática.

```
temp1 = id3 * 60.0
id1 = id2 + temp1
                ↓
load  id3  r2
mul   60.0 r2
load  id2  r1
add   r2   r1
store r1   id1
```

# O que é uma L.P.?

## *Compilada vs Interpretada*

---

### Linguagens Compiladas

- Vantagens:
  - Execução rápida, o código já está traduzido!
  - Executáveis são pequenos
  - VOCÊ CONTROLA o hardware
- Desvantagens:
  - Só roda na arquitetura em que for compilado
  - VOCÊ CONTROLA o hardware

# O que é uma L.P.?

## *Compilada vs Interpretada*

---

### Linguagens Interpretadas

- Do latim *interpretare*
- Significa EXPLICAR, TRADUZIR
- Fonte traduzido para uma linguagem intermediária (normalmente Bytecode) que mais tarde será novamente traduzido para linguagem de máquina

# O que é uma L.P.?

## *Compilada vs Interpretada*

---

### Processo de Interpretação

Programa (Alto nível)

Programa (Intermediário/Executável)

Máquina Virtual (Interpretadora)

# O que é uma L.P.?

## *Compilada vs Interpretada*

---

### Processo de Interpretação

Programa (Alto nível)

Programa (Intermediário/Executável)

Máquina Virtual (Interpretadora)

# O que é uma L.P.?

## *Compilada vs Interpretada*

---

### Processo de Interpretação

- Máquina virtual (VM)

Traduz os comandos da linguagem intermediária para linguagem de máquina em tempo de execução. A VM reconhece toda a arquitetura de hardware e traduz de acordo com o equipamento (culpa do Java).

# O que é uma L.P.?

## *Compilada vs Interpretada*

---

### Linguagens Interpretadas

- Vantagens
  - Independente de arquitetura (desde que suporte a máquina virtual)
  - A MÁQUINA VIRTUAL controla o hardware
- Desvantagens
  - Desempenho inferior à linguagem compilada por causa da tradução Just-in-time, mas nada alarmante
  - Consome bastante hardware (estou olhando pra você, Java)
  - A MÁQUINA VIRTUAL controla o hardware

# O que é uma L.P.?

## *Compilada vs Interpretada*

### Compilador

Lê todo o código, compila num arquivo binário e envia para a máquina executar

Exemplos de linguagens compiladas

C  
Rust  
Go  
C++

Exemplos de aplicações

- Motores para jogos
- Servidores com muitos dados/requisições

### Interpretador

Trechos de código são compilados dinamicamente por um interpretador que são enviados para o computador executar

Exemplos de linguagens interpretadas

Python  
Ruby  
PHP  
Js

Exemplos de aplicações

- Aplicações web
- Processo de desenvolvimento de software



**Uma mesma linguagem pode ter os dois!**

Just in time compilation  
(JIT Compilation)

Procura otimizações que podem ser feitas em tempo de execução. Caso uma função seja chamada uma única vez, ele interpreta, mas se uma mesma função for executada diversas vezes, ele compila essa função.

**Java**

Compila parcialmente o código, que é chamado de código intermediário (byte code ou .class), em seguida ele interpreta e executa.  
Máquina virtual Java

# VAMOS PARA A PRÁTICA ?!!!

---



# Por que Python?

- Simples o suficiente para um curso introdutório
- Muitos recursos
  - Orientação a Objetos
  - Escalável (módulos, classes, controle de exceções)
  - Biblioteca embutida extensa e grande número de módulos fornecidos por terceiros
- Grande variedade de aplicações
- Linguagem interpretada (script)
- Multi-plataforma
- Grátis!
- Comunidade bastante grande

# O que vamos precisar

- Uma implementação da linguagem
  - <http://www.python.org>
  - Implementação pronta para baixar (windows)
  - Linux normalmente já vem com python instalado
- Um editor de textos
  - Qualquer editor serve
  - Ambiente IDLE inclui um editor
    - Incluído na distribuição windows

# Python Interativo

- Rode o interpretador
- Digite comandos python
- Cada comando é executado imediatamente

```
[cancer]~> python
Python 2.4.1 (#1, May 16 2005, 15:19:29)
[GCC 4.0.0 20050512 (Red Hat 4.0.0-5)] on linux2
Type "help", "copyright", "credits" or "license" for more information.
>>> print("alo!")
alo!
>>>
```

# Executando um programa Python

- Escreva um programa python
- Invoque o interpretador para executá-lo

```
[cancer]~> cat prog.py
print("alo!")
[cancer]~> python prog.py
alo!
```

# Executando um programa Python

- Interfaces gráficas normalmente já associam os sufixos .py e .pyc com o interpretador

