

# Laboratório de Desenvolvimento de Algoritmos



- ✓ Conceitos iniciais
- ✓ IDE e linguagem utilizada
- ✓ Meu primeiro programa

# Motivação

- ✔ A programação de computadores é uma atividade que leva à representação dos passos necessários à resolução de um problema em linguagem de programação.
- ✔ Para dar início ao aprendizado dessa atividade, é importante compreender seu contexto, seu propósito, os conceitos básicos subjacentes, bem como tomar contato com as ferramentas necessárias para sua realização.



# O que é um Programa de Computador?

- ✔ **Programa** é uma coleção de instruções que descrevem uma tarefa a ser realizada por um computador.
- ✔ Um programa é a formalização de um algoritmo em qualquer **linguagem de programação**, capaz de ser transformado em instruções que serão executadas por um computador, gerando os resultados esperados.

Fonte: Wikipédia [http://pt.wikipedia.org/wiki/Programa\\_de\\_computador](http://pt.wikipedia.org/wiki/Programa_de_computador)

- ✔ Sequência completa de instruções a serem executadas por computador.

Fonte: Miniaurêlio Eletrônico versão 5.12 (2004)



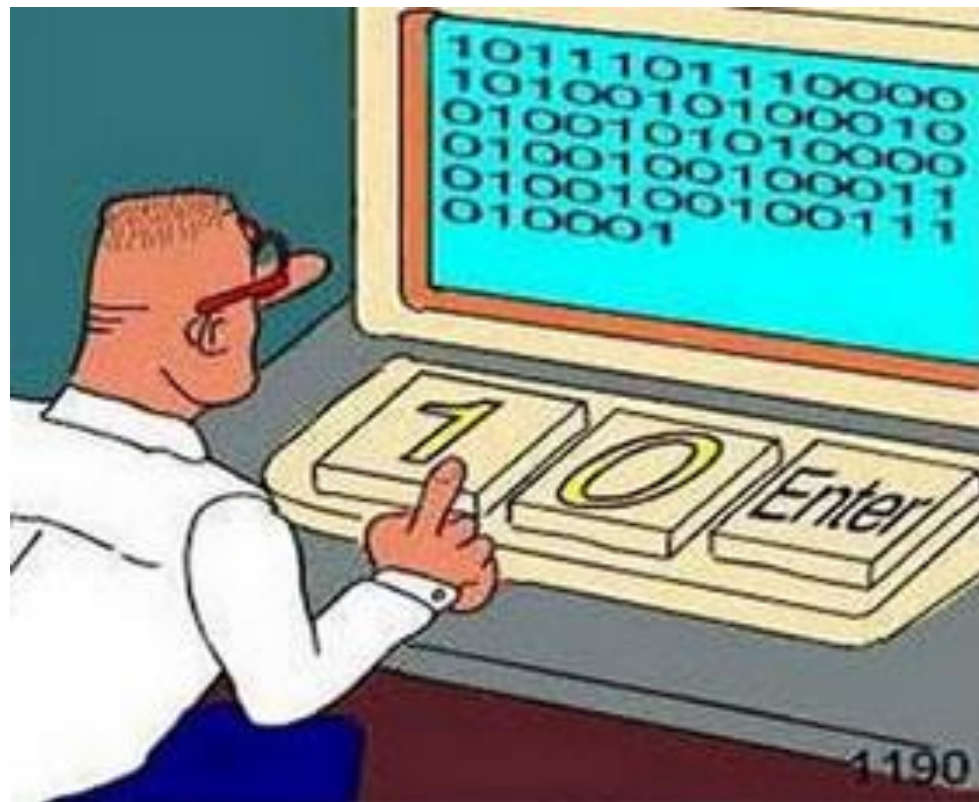
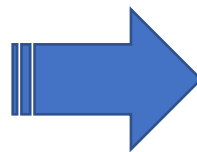
# O que é um Programa de Computador?

As instruções estão escritas em uma linguagem que permite a comunicação entre o programador e o computador (0's e 1's) → Linguagem de programação

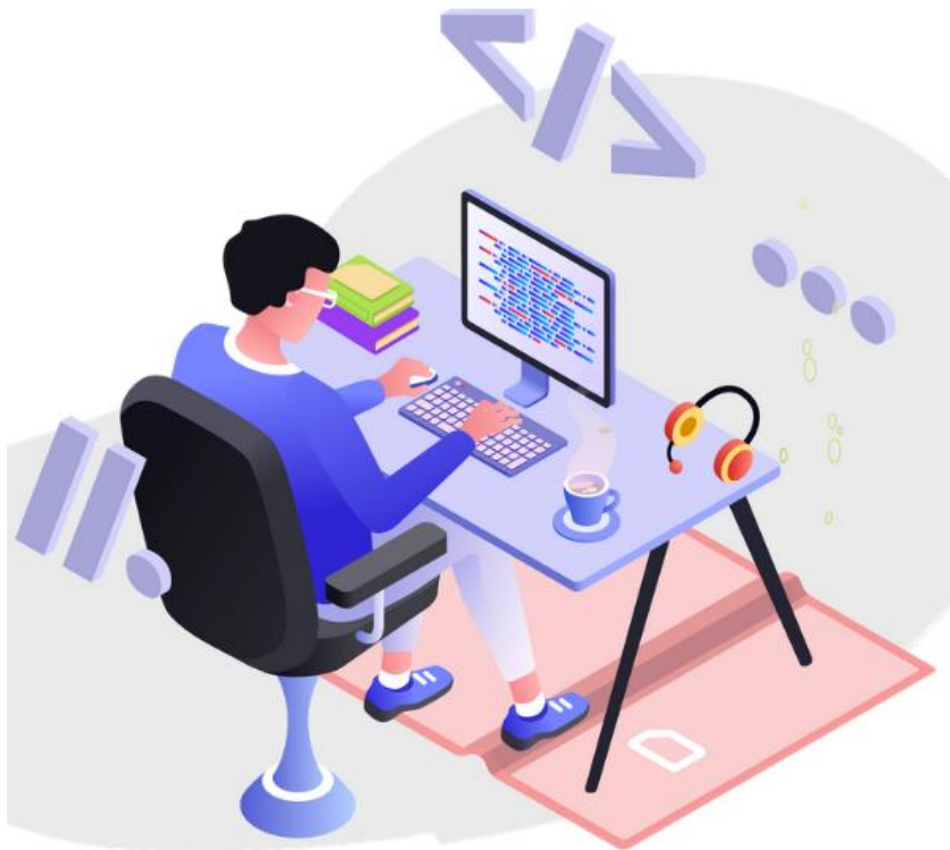
```
000000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000001000001000
0000000101011000111100011000000011
11111000110101001111001111001111
00011010000110001111010101111000
11100000111111000011111101010111
11100000111111000011111101010111
```

Código de máquina

Um arquivo contendo instruções em linguagem de máquina é chamado de **executável**.



# Como é construído?



De forma bem genérica podemos dizer que construir um programa envolve as seguintes etapas:

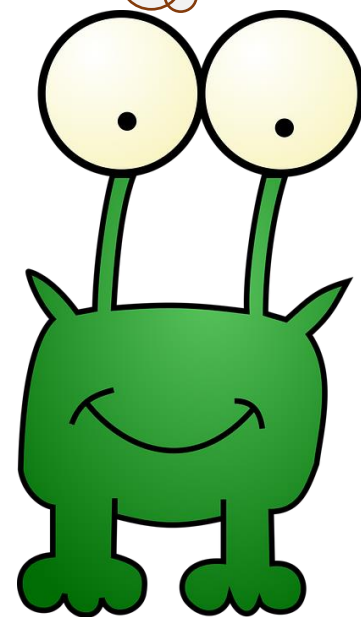
- ✓ Analisar o problema apresentado em busca de uma solução;
- ✓ Escrever esta solução na forma de um algoritmo;
- ✓ Implementar este algoritmo na linguagem de programação escolhida;
- ✓ Realizar testes;
- ✓ Corrigir erros, tanto sintáticos como lógicos;
- ✓ Gerar pacote de instalação.



# Algoritmo?

- ✓ Em computação pode ser definido como uma sequência de instruções ou operações básicas, cuja execução, em tempo finito resolve um problema computacional.
- ✓ Ele pode ser representado graficamente, em pseudocódigo ou diretamente na linguagem escolhida para desenvolvimento do programa.
- ✓ A partir do **algoritmo** será construído um **programa**, que estará escrito em alguma **linguagem de programação** para que possa ser executado em um computador.

Série de passos,  
regras ou  
procedimentos para  
tentar solucionar  
um problema



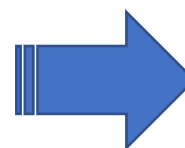
# Algoritmo no dia-a-dia

## **ALGORITMO 1.1** Troca de lâmpada

- pegar uma escada;
- posicionar a escada embaixo da lâmpada;
- buscar uma lâmpada nova;
- subir na escada;
- retirar a lâmpada velha;
- colocar a lâmpada nova.



- descer da escada
- guardar a escada



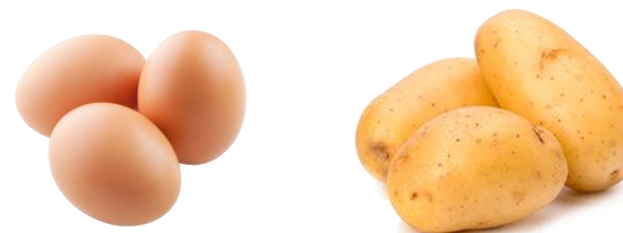
muitas vezes  
podemos  
aprimorar um  
algoritmo...

FORBELLONE, A. L. V.; EBERSPACHER, H. F. **Lógica de Programação: A Construção de Algoritmos e Estrutura de Dados**. 3. ed. São Paulo: Pearson Prentice Hall, 2008.

# Algoritmo no dia-a-dia

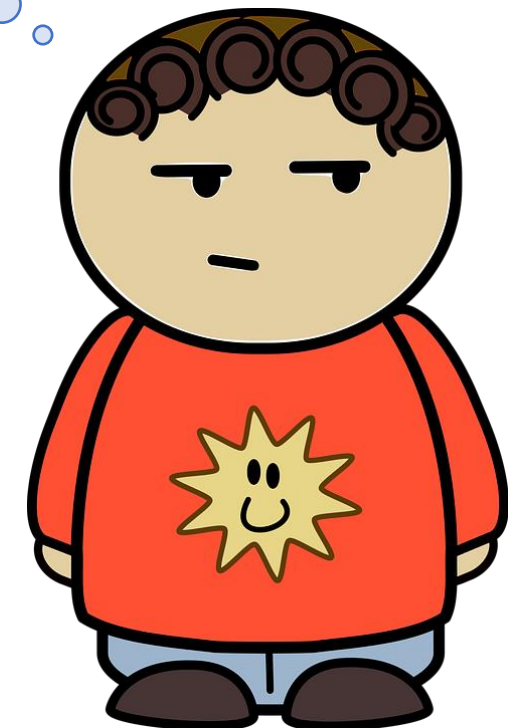
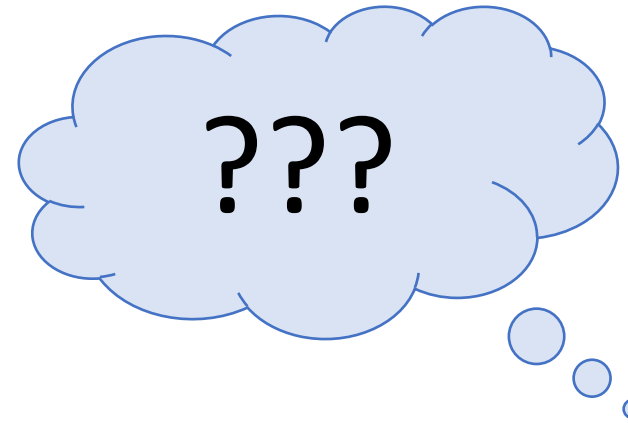


?





# Algoritmo no dia-a-dia



# Algoritmos – Como eles te controlam

Você sabe dizer como um algoritmo influencia sua vida?

A estrutura de um algoritmo em português coloquial

Algoritmo "Trabalhar pela manhã"

1. Acordar
2. Tomar banho
3. Vestir-se
4. Tomar café
5. Tirar o carro da garagem
6. Ir para o trabalho

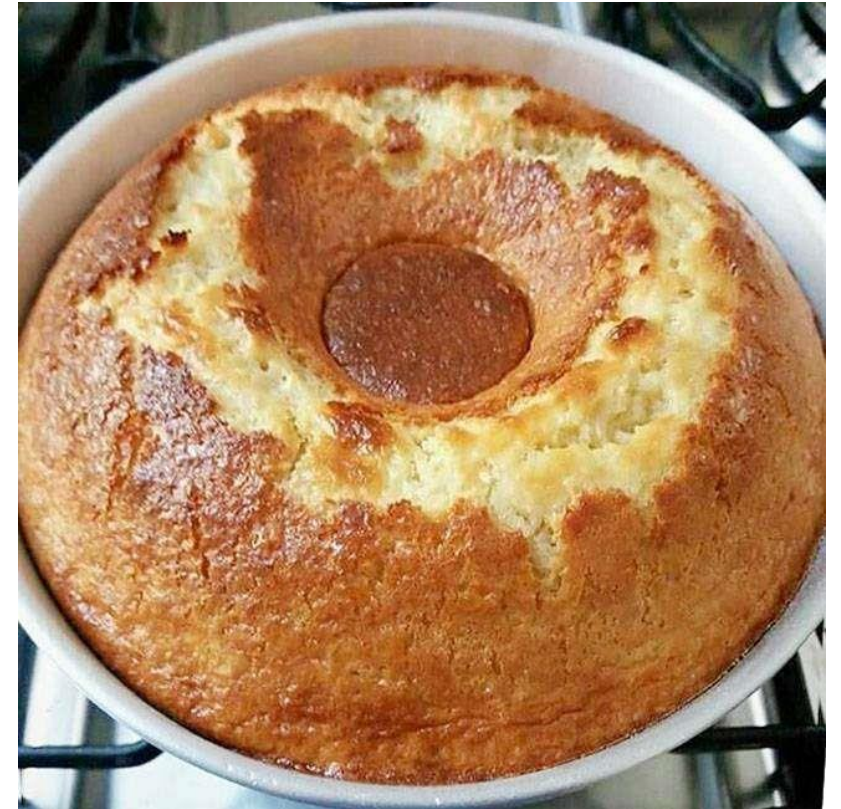


# A estrutura de um algoritmo em português coloquial

Você sabe dizer como um algoritmo influencia sua vida?

## **Algoritmo(receita de bolo):**

- 1) Bater duas claras em castelo;
- 2) Adicionar duas gemas;
- 3) Adicionar um xícara de açúcar;
- 4) Adicionar duas colheres de manteiga;
- 5) Adicionar uma xícara de leite de coco;
- 6) Adicionar farinha e fermento;
- 7) Colocar numa forma e levar ao forno em lume brando.



# Por que o algoritmo é importante ?

- ✔ Porque prepara uma lógica adequada, correta, para resolver um determinado problema.
- ✔ A partir dele será construído um programa, que estará escrito em alguma linguagem de programação para que possa ser executado em um computador.
- ✔ Pode ser programado em diferentes linguagens de programação.
- ✔ É imprescindível considerar todas as operações ou passos necessários de um algoritmo e a ordem em que deverão ser executadas estas operações pelo computador.



# Como podemos elaborar um algoritmo?

---

- ✓ Utilizando português coloquial.
- ✓ Utilizando pseudocódigo, com algumas palavras chaves diretamente relacionadas com a posterior programação do algoritmo.
- ✓ Utilizando diagramas de blocos ou fluxogramas.
- ✓ Utilizando diagramas de Chapin.
- ✓ Entre outros.





# Como podemos elaborar um algoritmo?

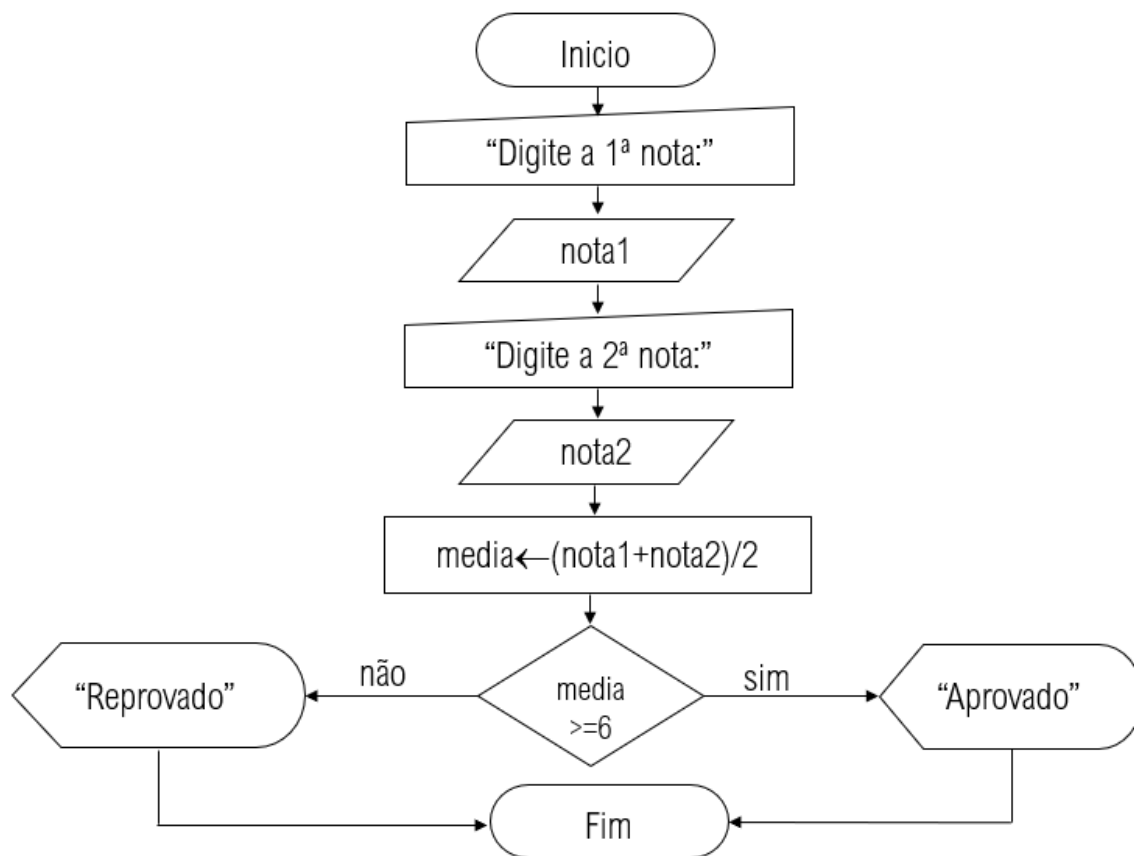
```
algoritmo exemplo
    real media, nota1, nota2
inicio
    escreva ("Digite a 1ª nota: ")
    leia (nota1)
    escreva ("Digite a 2ª nota: ")
    leia (nota2)
    media = (nota1 + nota2)/2
    se (media >= 6){
        escreva ("Aprovado")
    } senão {
        escreva ("Reprovado")
    }
fim
```

Representação Pseudocódigo (Portugol)

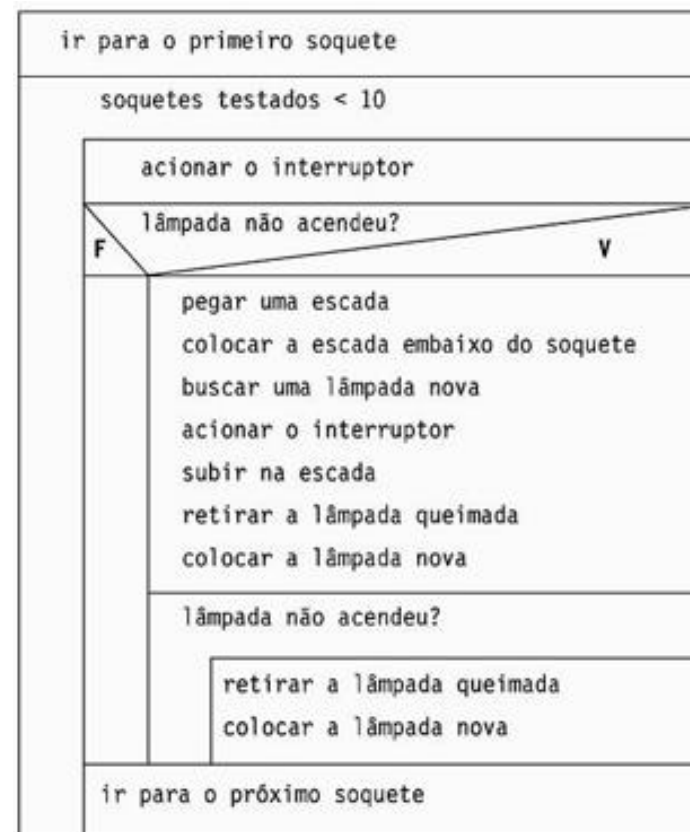


# Como podemos elaborar um algoritmo?

## Representação Gráfica (Fluxograma)

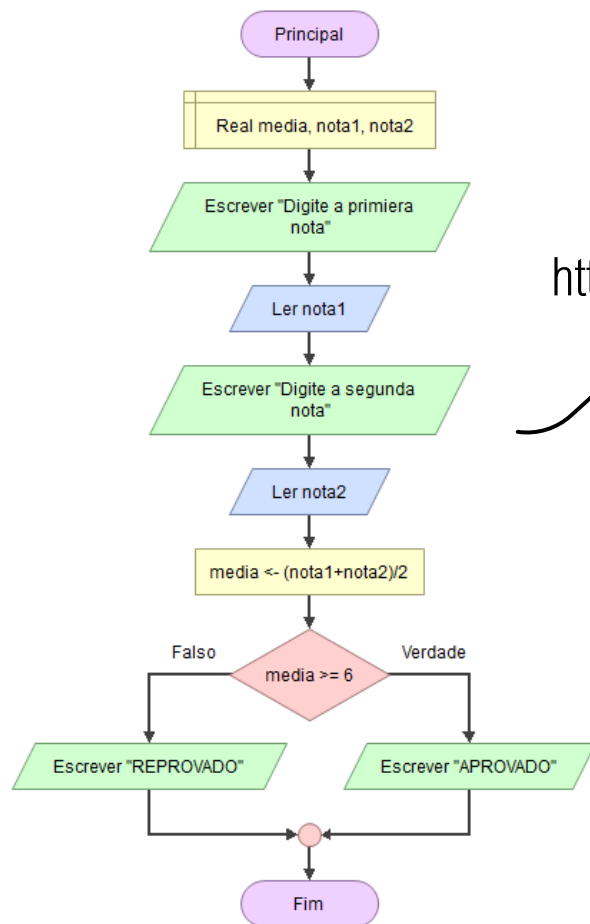


## Diagrama de Chapin



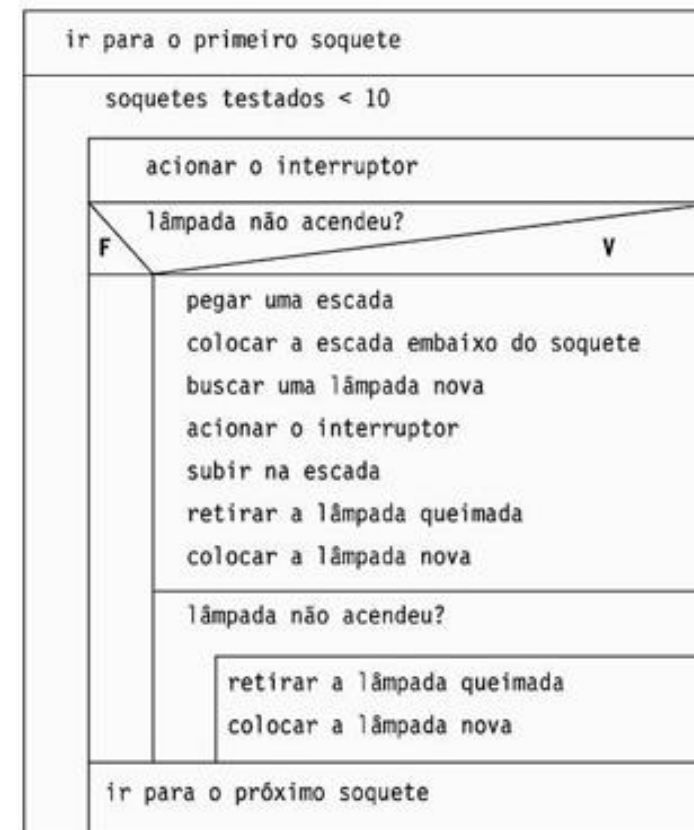
# Como podemos elaborar um algoritmo?

## Representação Gráfica (Fluxograma)



<http://www.flowgorithm.org/>

## Diagrama de Chapin

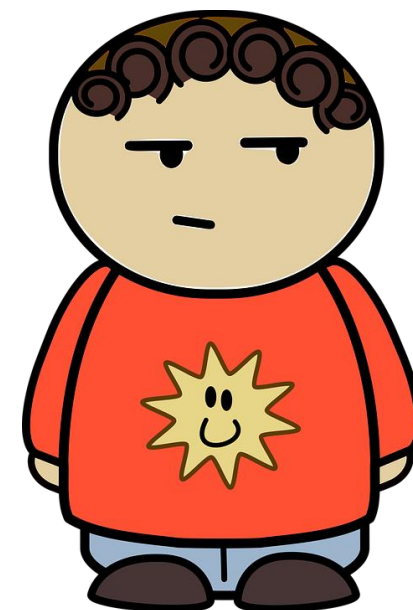


FORBELLONE, A. L. V.; EBERSPACHER, H. F. **Lógica de Programação: A Construção de Algoritmos e Estrutura de Dados**. 3. ed. São Paulo: Pearson Prentice Hall, 2008.

# Processos que o algoritmo sofre...

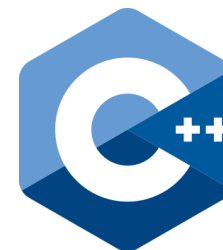
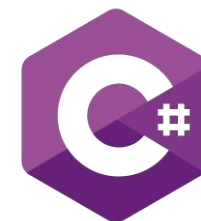
---

- ✓ Um computador não está preparado para executar algoritmos diretamente.
- ✓ Será necessário escrever o algoritmo em alguma linguagem de programação.
- ✓ O programa obtido ou criado a partir do algoritmo será editado, compilado (traduzido) e executado, utilizando um computador.
- ✓ Nesse processo é necessário levar em consideração algumas etapas:
  - programação
  - edição
  - compilação
  - execução
  - depuração e testes



# Linguagens de programação

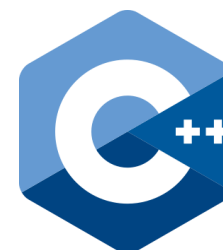
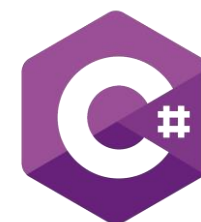
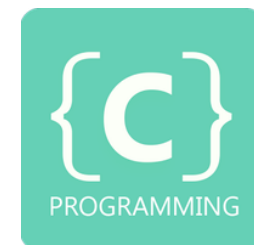
- ✓ Como mencionado, será necessário escrever o algoritmo em alguma linguagem de programação. A partir do algoritmo obteremos um **programa**.
- ✓ Uma linguagem natural ou idioma (português, inglês, espanhol etc.) estabelece um vocabulário ou dicionário válido e um grupo de regras sintáticas e semânticas para que as pessoas se expressem adequadamente nessa linguagem e se comuniquem.
- ✓ Isto acontece em forma semelhante com as linguagens de programação. Uma linguagem de programação estabelece um grupo de regras, um vocabulário ou dicionário, para escrever programas válidos para essa linguagem.



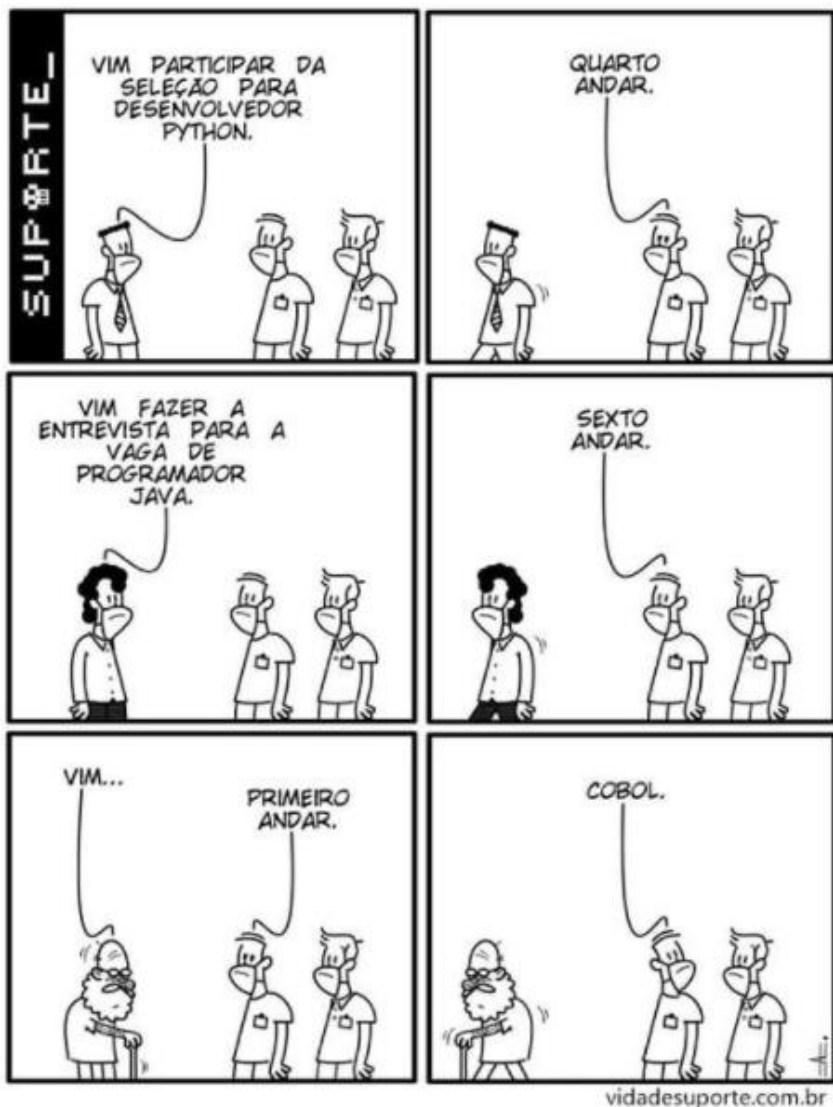


# Linguagens de programação

- ✔ Para "programar" o algoritmo (elaboração de um "programa") faremos uso de alguma linguagem de programação, como, por exemplo:
  - Pascal, Ada, COBOL, Fortran, C, C++.
  - Java, Kotlin, Dart, Objective-C, Swift.
  - C#.NET, VB.NET, C++.NET, todas da plataforma .NET da Microsoft.
  - PHP, Perl, Python e Java Script, que são alguns exemplos específicos de linguagens de programação para Web.
  - AutoLISP, ActionScript e VBScript, que são exemplos de linguagens de programação para sistemas específicos.



# Linguagens de programação



# Exemplo Java

```
1 import javax.swing.*;
2 public class Teste {
3     public static void main(String args[ ]) {
4         String nome;
5         nome = JOptionPane.showInputDialog("Digite o seu nome");
6         JOptionPane.showMessageDialog(null, "Boa noite " + nome);
7     }
8 }
```



# Exemplo C

main.c

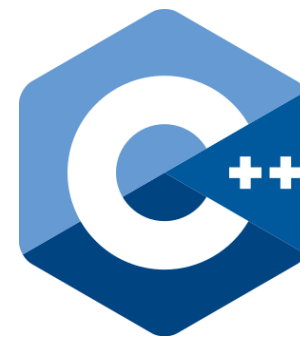
```
1  #include <stdio.h>
2
3  int main(void) {
4      char nome[30];
5      printf("Digite o seu nome: ");
6      gets(nome);
7      printf("Boa noite %s", nome);
8  }
```



# Exemplo C++

main.cpp

```
1  #include <iostream>
2  using namespace std;
3
4  int main() {
5      char nome[30];
6      std::cout << "Digite o seu nome: ";
7      cin.getline(nome, 30);
8
9      cout<< "Boa noite " << nome;
10 }
```





# Exemplo JavaScript

---

```
1 <script>
2     let nome;
3     nome = prompt("Digite o seu nome");
4     alert("Boa noite " + nome);
5 </script>
```



# Exemplo Python

---

```
1 nome = input("Digite seu nome: ")  
2 print("Boa noite ", nome)
```



# Como a máquina entende os códigos?

Para que o computador “entenda” um programa escrito em uma linguagem (de alto nível) é necessário um meio de tradução entre a linguagem de alto nível utilizada no programa e a linguagem de máquina.

Para essa tarefa temos basicamente dois métodos:

- ✓ Compilador
- ✓ Interpretador



# Interpretador

- ✓ Traduz e faz a checagem da sintaxe e envia para execução, instrução por instrução.
- ✓ Precisa estar presente todas as vezes que vamos executar o programa e o processo acima é repetido.



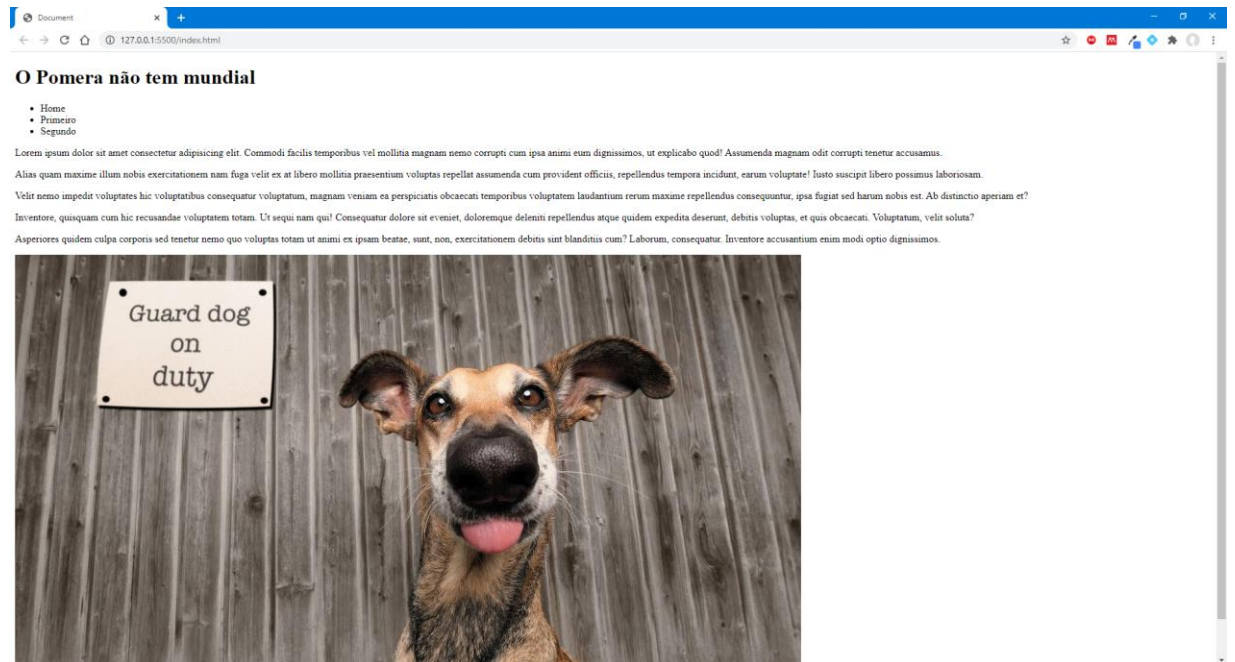
**Vantagem:** consome menos memória

**Desvantagem:** execução mais lenta

# Interpretador

- ✓ Traduz e faz a checagem da sintaxe e envia para execução, instrução por instrução.
- ✓ Precisa estar presente todas as vezes que vamos executar o programa e o processo acima é repetido.

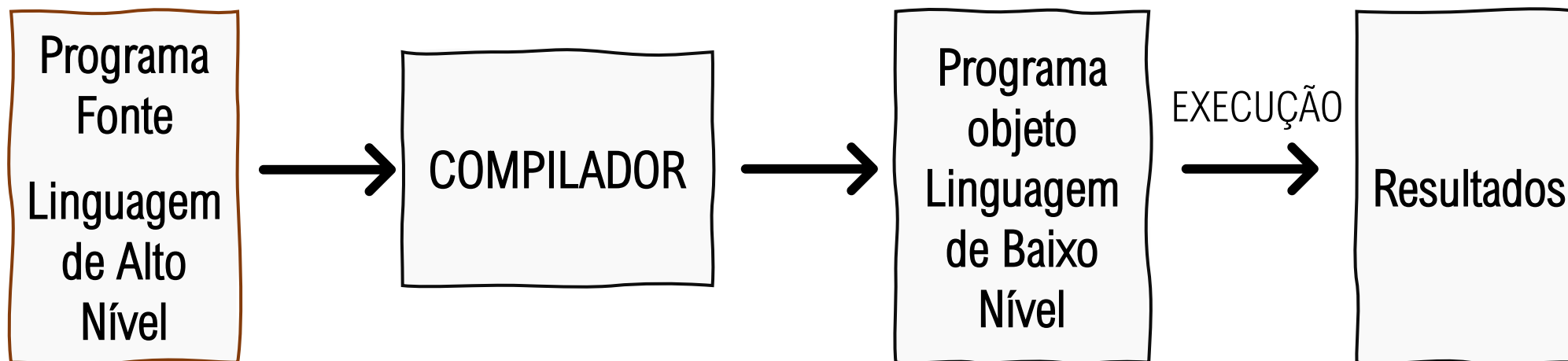
**Exemplo:** Uma página HTML é interpretada pelo Navegador.





# Compilador

Traduz o programa escrito em uma linguagem de programação para um programa equivalente escrito em linguagem de máquina (programa-objeto).



## Vantagens:

- ✓ Velocidade de execução
- ✓ Oculta o código fonte

## Desvantagem:

- ✓ A cada alteração no programa fonte é necessário gerar novamente o programa-objeto

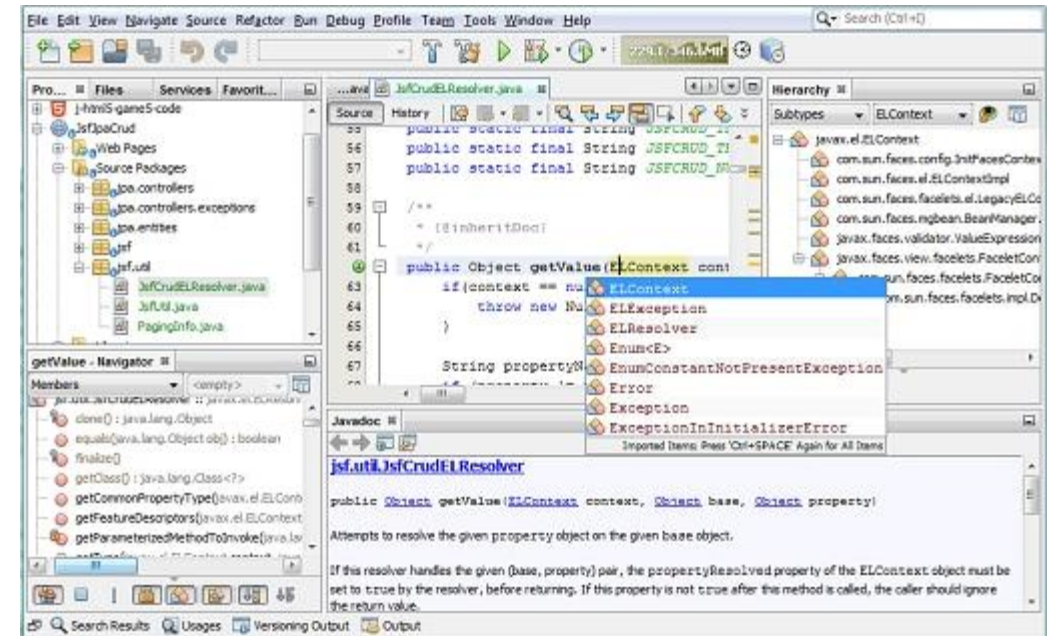
# Ambientes para desenvolvimento de programas (IDE)

---

- ✓ Os programadores utilizam ambientes integrados para desenvolvimento de programas (IDE, das siglas em inglês de Integrated Development Environment).
- ✓ Estes IDEs fornecem recursos para edição, compilação e execução de programas e outros recursos importantes.
- ✓ Alguns dos IDEs mais utilizados são:
  - **NetBeans**, **Android Studio** e **Eclipse**: para desenvolvimento na plataforma Java.
  - **Jcreator**: IDE simples para Java que não permite projeto visual, nem depuração. É bastante limitado.
  - **Microsoft Visual Studio** .NET: para desenvolvimento na plataforma .NET, permitindo, entre outras, as linguagens VB.NET, C#.NET e C++.NET.

# NetBeans

- ✔ O NetBeans IDE é um ambiente de desenvolvimento integrado de código aberto e gratuito para desenvolvimento de aplicativos nos sistemas operacionais Windows, Mac, Linux e Solaris.
- ✔ O IDE simplifica o desenvolvimento de aplicativos para Web, corporativos, de desktop e móveis que usam as plataformas Java e HTML5, oferecendo suporte também para o desenvolvimento de aplicativos PHP e C/C ++.



Para download acesse: <https://netbeans.apache.org/download/index.html>

# Eclipse

---

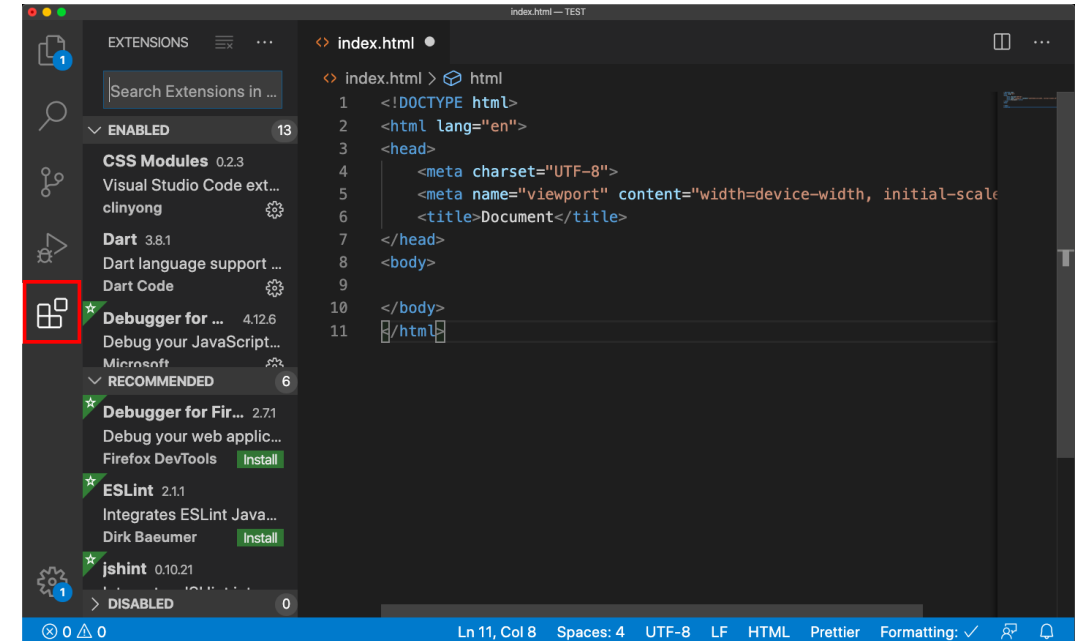
- ✓ O Eclipse IDE é um ambiente de desenvolvimento integrado de código aberto e gratuito para desenvolvimento de aplicativos Java, porém suporta várias outras linguagens a partir de plugins como C/C++, PHP, Python entre outras.
- ✓ Sua principal característica é o uso da SWT e não do Swing como biblioteca gráfica, forte orientação ao desenvolvimento baseado em plug-ins, que procuram atender as diferentes necessidades de diferentes programadores.



Para download acesse: <https://www.eclipse.org/downloads/>

# Visual Studio Code

- ✓ O Visual Studio Code é um editor de código-fonte desenvolvido pela Microsoft para Windows, Linux e macOS, que possui suporte para depuração, controle de versionamento GIT incorporado, realce de sintaxe e complementação de código inteligente.
- ✓ O Visual Studio Code suporta um número de linguagens de programação, que podem ser adicionadas por meio da instalação de plugins disponíveis em um repositório central.



Para download acesse: <https://code.visualstudio.com/Download>

# A linguagem Java

---

- ✓ É uma linguagem de programação orientada a objetos desenvolvida na década de 90 por uma equipe de programadores chefiada por James Gosling, na empresa Sun Microsystems.
- ✓ Diferente das linguagens de programação modernas, que são compiladas para código nativo, a linguagem Java é compilada para um bytecode que é interpretado por uma máquina virtual (Java Virtual Machine, mais conhecida pela sua abreviação JVM).



# A linguagem Java - Característica

---

A linguagem Java foi projetada tendo em vista os seguintes objetivos:

- ✓ Orientação a objetos.
- ✓ Portabilidade - Independência de plataforma - "escreva uma vez, execute em qualquer lugar" ("write once, run anywhere").
- ✓ Recursos de Rede - Possui extensa biblioteca de rotinas que facilitam a cooperação com protocolos TCP/IP, como HTTP e FTP.
- ✓ Segurança - Pode executar programas via rede com restrições de execução.





# A linguagem Java - Versões

---

As principais APIs são distribuídas juntamente com os produtos para desenvolvimento de aplicações

- ✓ **Java Standard Edition (Java SE):** ferramentas e APIs essenciais para qualquer aplicação Java (inclusive GUI - Graphical User Interface)
- ✓ **Java Enterprise Edition (Java EE):** ferramentas e APIs para o desenvolvimento de aplicações corporativas
- ✓ **Java Micro Edition (Java ME):** ferramentas e APIs para o desenvolvimento de aplicações para aparelhos portáteis



# A linguagem Java - Característica

---

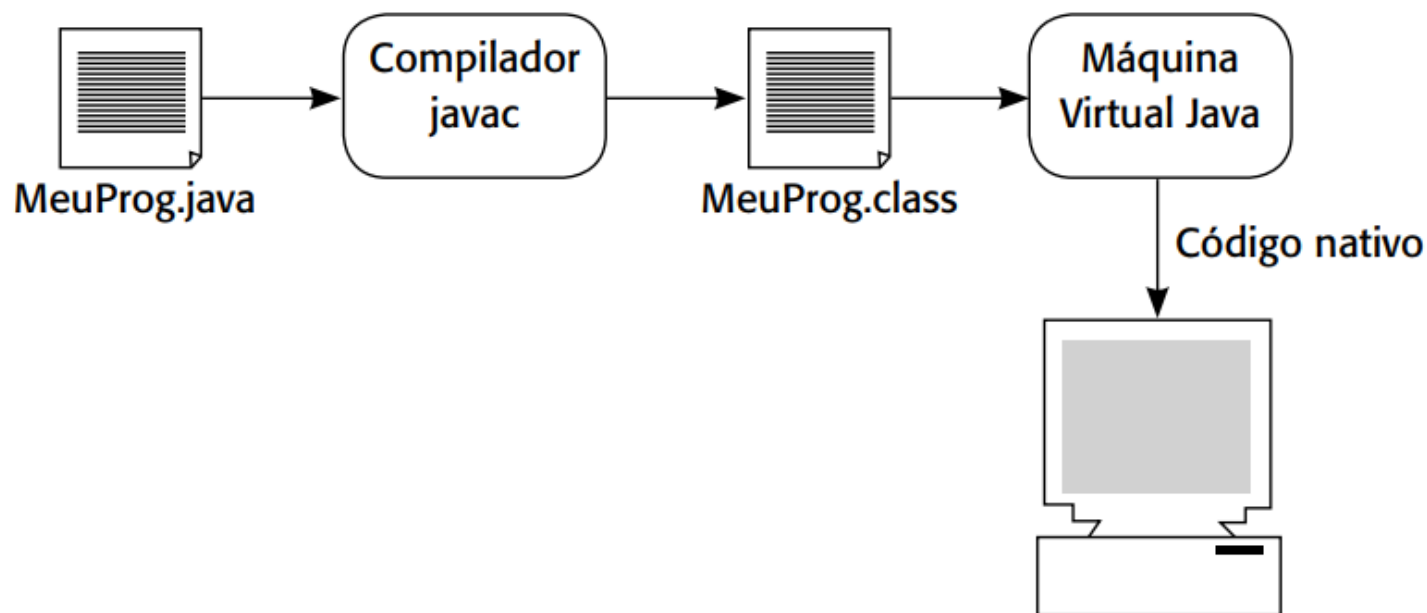
Além disso, podem-se destacar outras vantagens apresentadas pela linguagem:

- ✓ Sintaxe similar a C/C++
- ✓ Simplicidade na especificação, tanto da linguagem como do "ambiente" de execução (JVM).
- ✓ É distribuída com um vasto conjunto de bibliotecas (ou APIs).
- ✓ Possui facilidades para criação de programas distribuídos e multitarefa (múltiplas linhas de execução num mesmo programa).



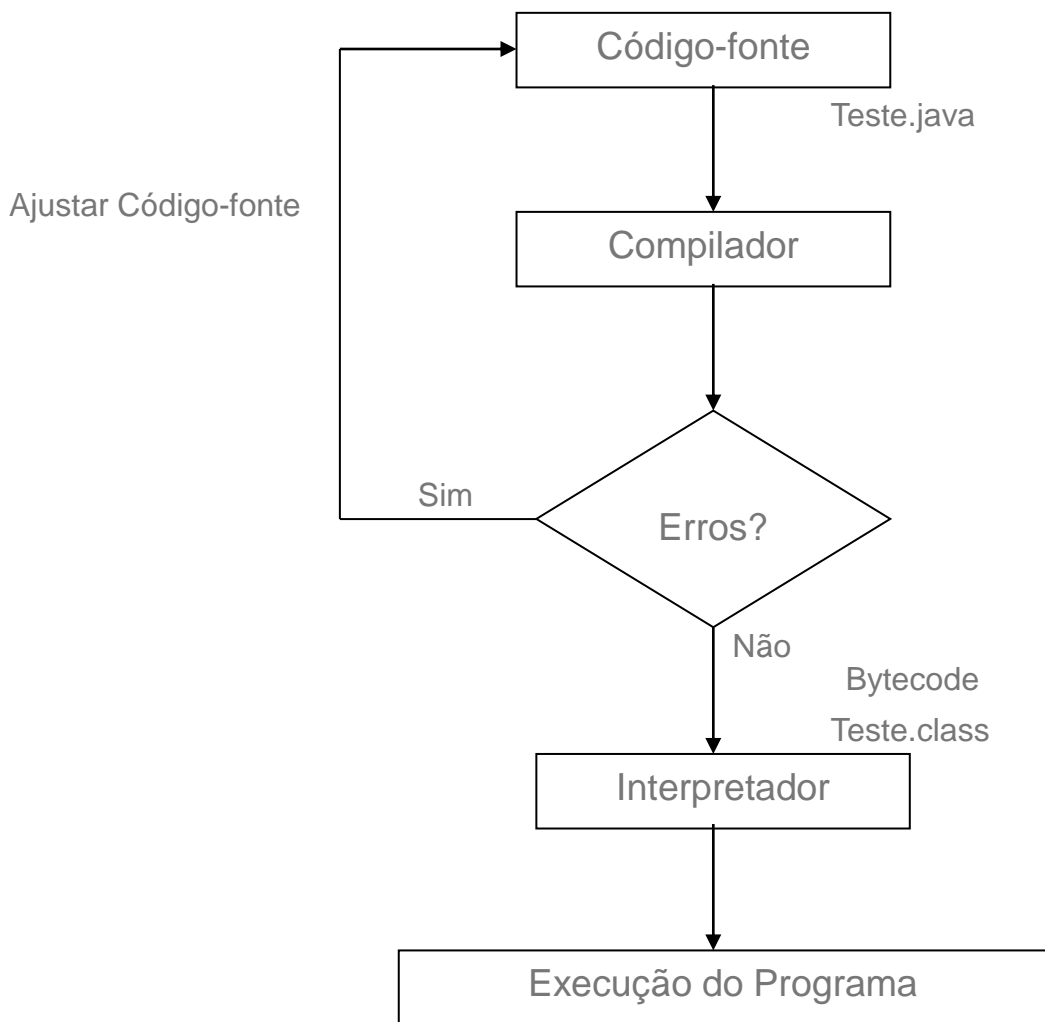
# Execução de programas em Java

- ✓ Os programas são escritos em arquivos texto com a extensão .java
- ✓ Ao serem compilados com o compilador javac, são gerados os arquivos .class
- ✓ Um arquivo .class é constituído por bytecodes, código interpretado pela Máquina Virtual Java (Java Virtual Machine).



ASCENCIO, A. F. G.; CAMPOS, E.A.V. Fundamentos da Programação de Computadores: algoritmos, PASCAL, C/C++ (padrão ANSI) e JAVA. São Paulo: Pearson, 2012.

# Execução de programas em Java



Na fase de execução é necessário que haja a Máquina Virtual Java (MVJ também conhecida como JVM)

A MVJ interpreta os bytecodes gerados pelo compilador.

O objetivo da MVJ é permitir que qualquer sistema operacional possa executar uma aplicação Java

# Estrutura de uma classe executável em Java

**class** é a palavra reservada que marca o início da declaração de uma classe.

Nome da Classe  
**Deve ter o mesmo nome do arquivo**

```
public class [nome]
{
    public static void main (String args[ ])
    {
        ...
    }
}
```

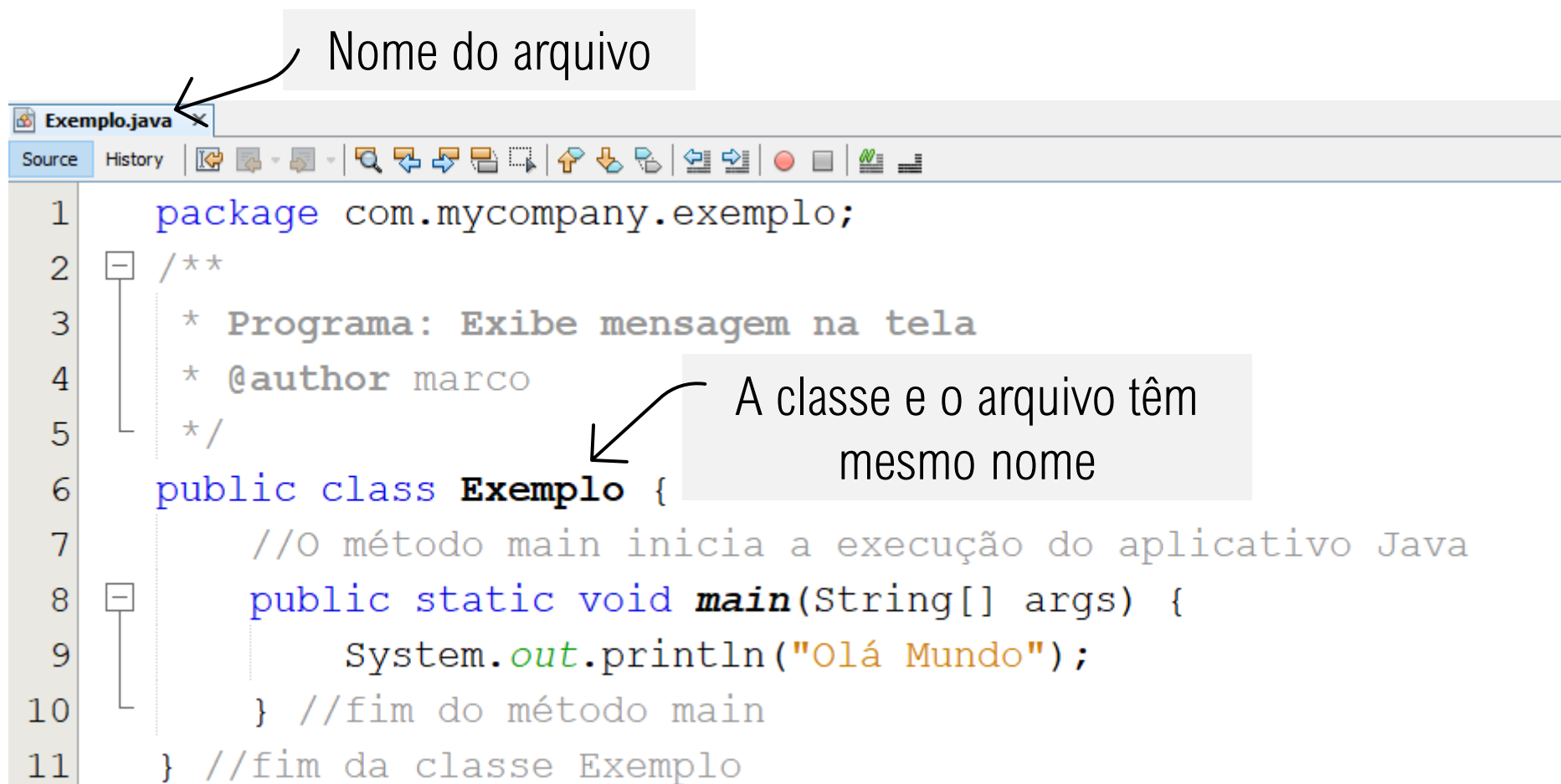
Instruções

Método **main**: onde inicia a execução

**Essa estrutura estará em todos os programa desenvolvidos em Java**



# Estrutura de uma classe executável em Java





# Estrutura de uma classe executável em Java

Nome do arquivo

```
Exemplo.java
Source History
1 package com.mycompany.exemplo;
2 /**
3  * Programa: Exibe mensagem na tela
4  * @author marco
5  */
6 public class Exemplo {
7     //O método main inicia a execução do aplicativo Java
8     public static void main(String[] args) {
9         System.out.println("Olá Mundo");
10    } //fim do método main
11 } //fim da classe Exemplo
/* Esse é um
comentário
com várias linhas*/
```

A classe e o arquivo têm mesmo nome

Inicia-se com o delimitador `/*` e termina com `*/`

Os comentários são ignorados pelo compilador



No comentário de uma única linha usa-se o delimitador `//`

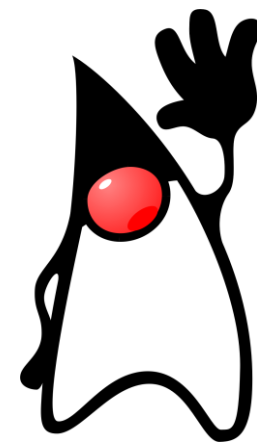
# Primeiro programa em Java



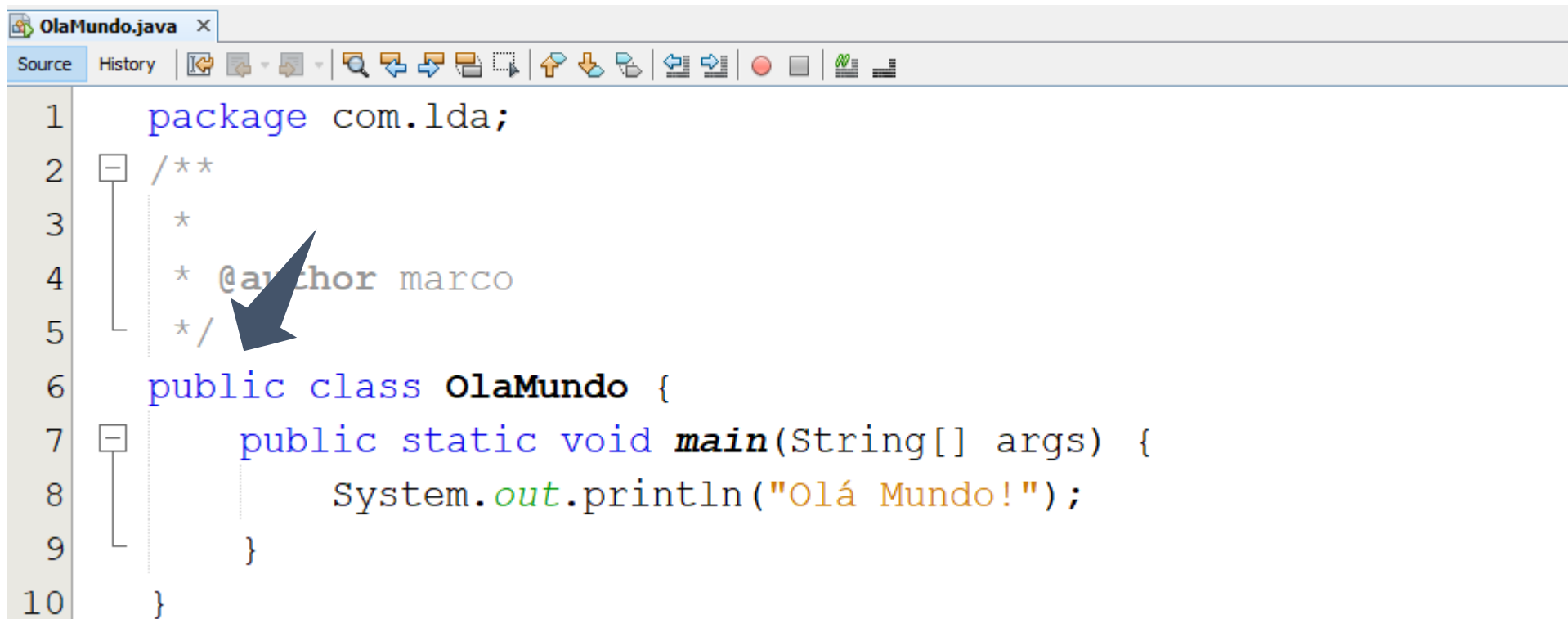
```
OlaMundo.java x
Source History
1 package com.lda;
2 /**
3  *
4  * @author marco
5  */
6 public class OlaMundo {
7     public static void main(String[] args) {
8         System.out.println("Olá Mundo!");
9     }
10 }
```

A screenshot of a Java IDE window titled 'OlaMundo.java'. The window has a menu bar with 'Source' and 'History' tabs. Below the menu bar is a toolbar with various icons for editing and running code. The code is displayed in a text area with line numbers 1 through 10 on the left. The code defines a package 'com.lda', a class 'OlaMundo' with a Javadoc comment, and a 'main' method that prints 'Olá Mundo!'. A blue arrow points to the 'main' method signature on line 7.

- ✓ O método **main** é onde o programa inicia.
- ✓ Os parâmetros de linha de comando são enviados para o array de Strings chamado *args*.

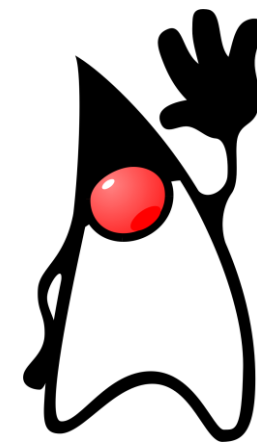


# Primeiro programa em Java

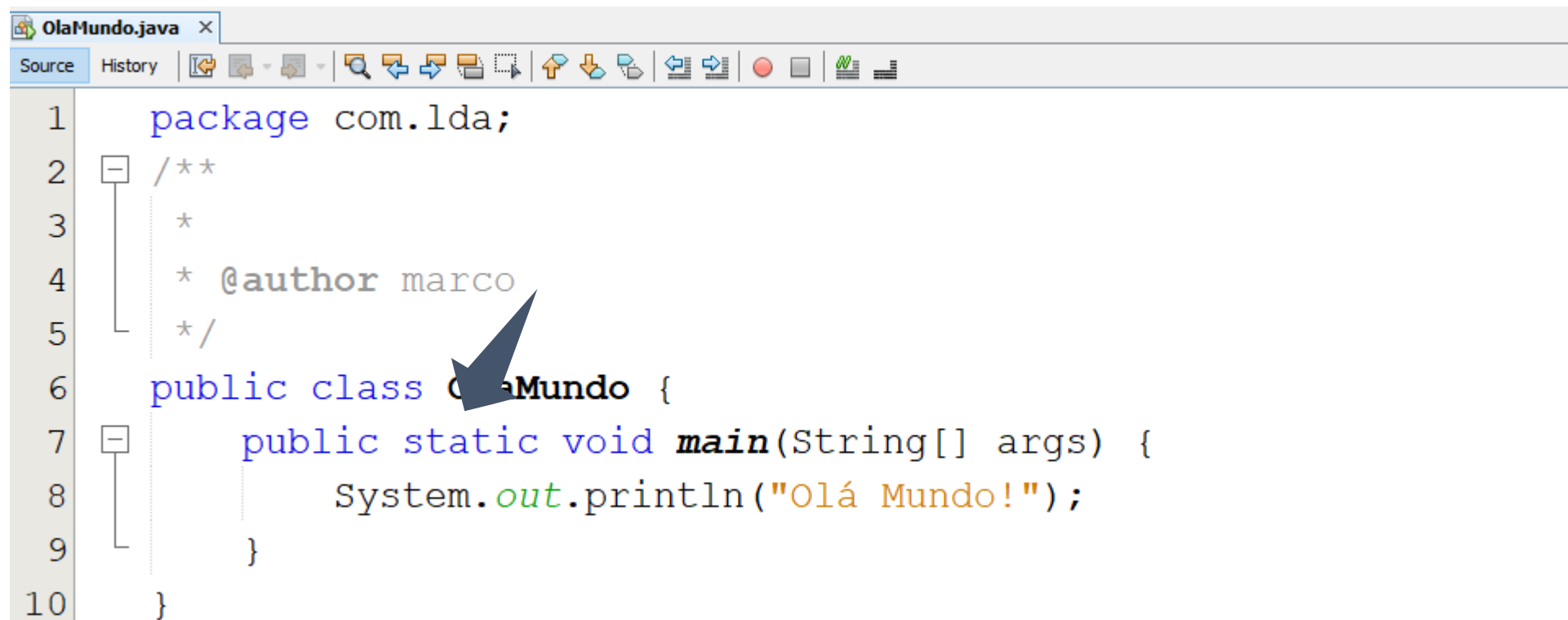


```
1 package com.lda;
2 /**
3  *
4  * @author marco
5  */
6 public class OlaMundo {
7     public static void main(String[] args) {
8         System.out.println("Olá Mundo!");
9     }
10 }
```

- ✔ **public** É o modificador de acesso que permite a visibilidade em qualquer lugar da aplicação.

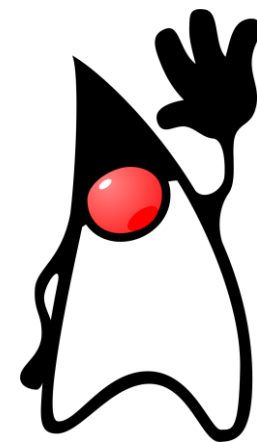


# Primeiro programa em Java



```
1 package com.lida;
2 /**
3  *
4  * @author marco
5  */
6 public class OlaMundo {
7     public static void main(String[] args) {
8         System.out.println("Olá Mundo!");
9     }
10 }
```

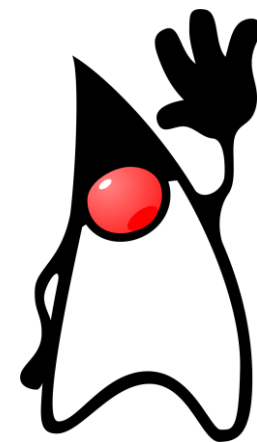
- ✓ **static** é iniciado automaticamente pela JVM, sem precisar de uma instância.



# Primeiro programa em Java

```
OlaMundo.java x
Source History
1 package com.lda;
2 /**
3  *
4  * @author marco
5  */
6 public class OlaMundo {
7     public static void main(String[] args) {
8         System.out.println("Olá Mundo!");
9     }
10 }
```

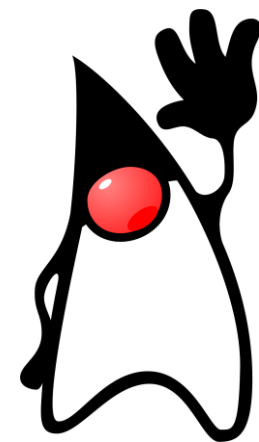
- ✓ **void** método sem retorno (vazio)



# Primeiro programa em Java

```
OlaMundo.java x
Source History
1 package com.lda;
2 /**
3  *
4  * @author marco
5  */
6 public class OlaMundo {
7     public static void main(String[] args) {
8         System.out.println("Olá Mundo!");
9     }
10 }
```

- ✓ **main** é o nome do método principal da aplicação.
- ✓ A aplicação só pode ter um método denominado **main**, que recebe como parâmetro um array de **String**.





# Alguma dúvida????

---



# Exercícios de aplicação

---





# Observações sobre exercícios

---

- ✓ Todos os exercícios devem ser resolvidos em algoritmo e Java.
- ✓ Os códigos podem ser feito no NetBeans, Eclipse, VSCode ou no Repl.it.
- ✓ Após finalizar todos os exercícios da aula, compacte os arquivos e pastas e envie no Blackboard.



# Exercícios de aplicação

- 1- Aplicações desenvolvidas em Java só podem rodar no mesmo tipo de plataforma em que foram desenvolvidas. Isso está correto? Justifique.
- 2- A partir do exemplo visto na aula, desenvolva um programa que coloque na tela três mensagens em linhas separadas. Pode ser seu nome, na primeira linha, “Programa em Java”, na segunda e “Até o próximo programa” na última.



# Créditos

Esta aula foi elaborada com base no material produzido e cedido gentilmente pelos **Professores Alcides, Lédon, Ana e Cristiane.**







*That's all Folks!*