



Universidade  
**Cruzeiro do Sul**

# **Programação de Computadores**

## **Conceitos de programação**

Professores:  
Amilton Souza Martha  
Cristiane C. Hernandez  
Manuel F. Paradela Ledón

*As aulas disponibilizadas por meios digitais envolvem horas de estudo, pesquisa e organização dos(as) professores(as) responsável(eis) e destinam-se, única e exclusivamente, aos alunos regularmente matriculados em cursos da Universidade Cruzeiro do Sul, para fins acadêmicos*

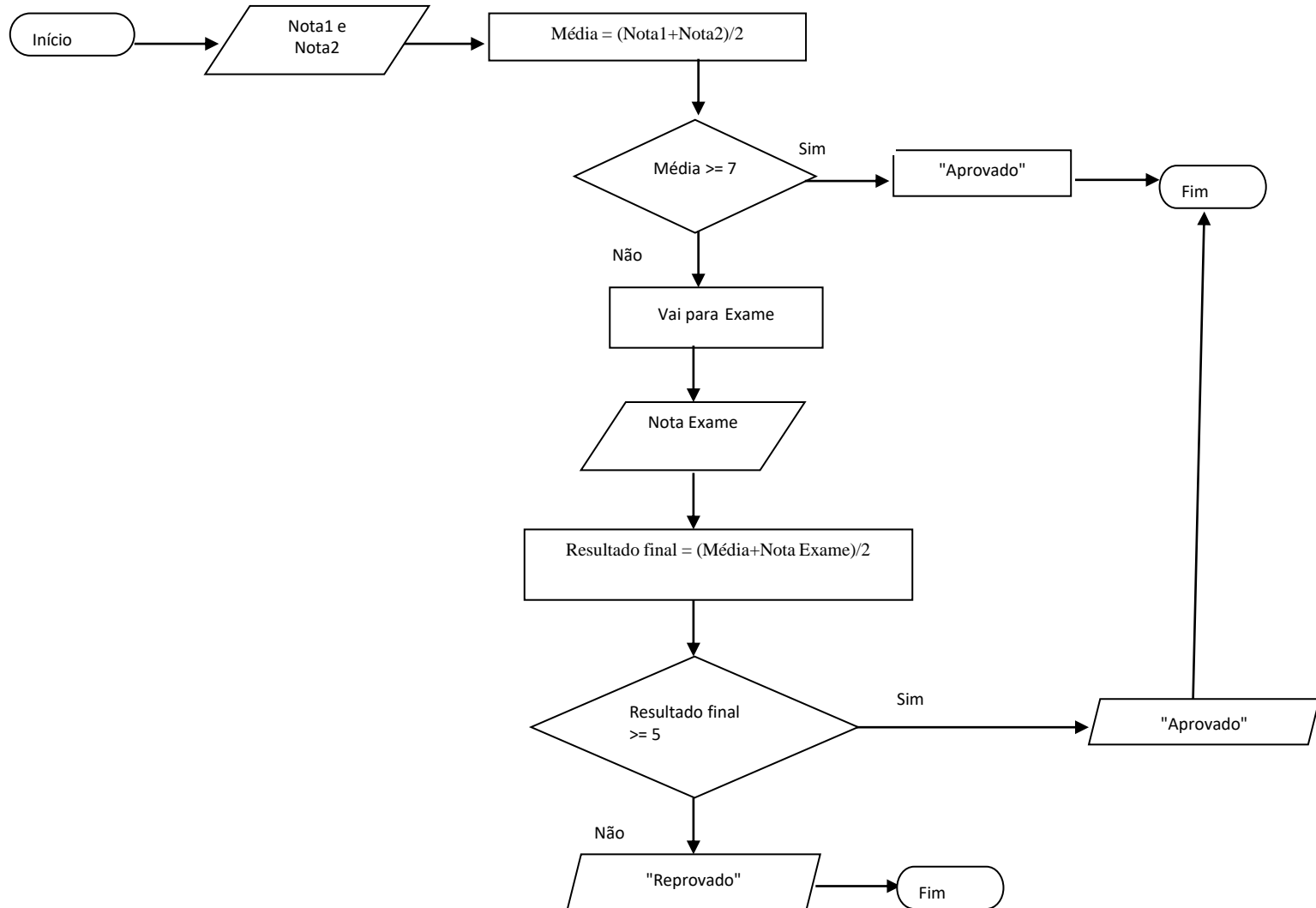
*A disponibilização/ divulgação dos links de acesso às aulas síncronas e/ ou registros das aulas a colegas não matriculados e a indivíduos estranhos à comunidade acadêmica da universidade implicará responsabilização civil e criminal conforme Artigo 46, inciso IV da Lei 9610/98, combinado com o artigo 184 do Código Penal, que estabelece direito autoral ao docente.*

*Assim, em atendimento à Lei e à valorização do trabalho docente, solicita-se que **não compartilhem as aulas com indivíduos que não estejam regularmente matriculados.***

# Algoritmo?

- Em computação: algoritmo é uma sequência bem definida de instruções ou operações básicas, sem ambiguidades, cuja execução, em tempo finito, resolve um problema computacional.
- O algoritmo pode ser representado graficamente (fluxogramas, diagramas), em pseudocódigo, na língua do desenvolvedor (português, inglês etc.), em outras formas, ou diretamente na **linguagem de programação** escolhida para desenvolvimento do programa.
- "Conjunto de regras e operações bem definidas e ordenadas, destinadas à solução de um problema, ou de uma classe de problemas, em um número finito de etapas."

# Algoritmo – representação gráfica (fluxograma)



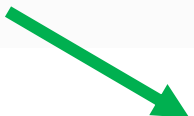
# Algoritmo – representação em pseudocódigo (Portugol)

```
início
    real media, nota1, nota2, exame, final;
    escreva "Digite a 1ª nota: ";
    leia nota1;
    escreva "Digite a 2ª nota: ";
    leia nota2;
    media  $\leftarrow$  (nota1 + nota2)/2;
    se (media  $\geq$  7)
        escreva "Aprovado";
    senão
        escreva "Digite a nota de exame";
        leia exame;
        final  $\leftarrow$  (media + exame)/2;
        se (final  $\geq$  5)
            escreva "Aprovado";
        senão
            escreva "Reprovado";
        fim se
    fim se
fim
```

# Um exemplo de algoritmo do 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.

## Por que o algoritmo é importante ?

- Um algoritmo prepara uma lógica adequada, correta, para resolver um determinado problema ou grupo de problemas.
- A partir do **algoritmo** será construído um **programa**, que estará escrito em alguma **linguagem de programação** para que possa ser executado em um computador.
- Como o algoritmo descreve uma lógica geral de solução, poderá 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.

## Programa de computador?

- Um **programa** é um grupo de comandos ou instruções que descrevem a lógica de solução de um problema a ser executada por um computador, escrito em alguma **linguagem de programação**.
- "Programa de computador. *Inform.:* conjunto de instruções numa sequência lógica, que, por meio de linguagem de programação, capacita a máquina para determinado fim."

Fonte: Michaelis, em <http://michaelis.uol.com.br/>



## **Exemplo de algoritmo muito simples**

Neste algoritmo, o usuário entrará com o seu nome e o programa visualizará na tela "Boa noite " e o nome da pessoa.

### **Pseucocódigo (portugol)**

```
início  
    string nome  
    escreva "Digite o seu nome: "  
    leia nome  
    escreva "Boa noite " + nome  
fim
```

# O algoritmo anterior programado na linguagem Java

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

Obs: também poderíamos usar os métodos da classe Scanner.

## Linguagem C

```
#include<stdio.h>

void main()
{
    char nome[30];
    printf("Digite o seu nome: ");
    gets(nome);
    printf("Boa Noite %s", nome);
}
```

## Linguagem C++

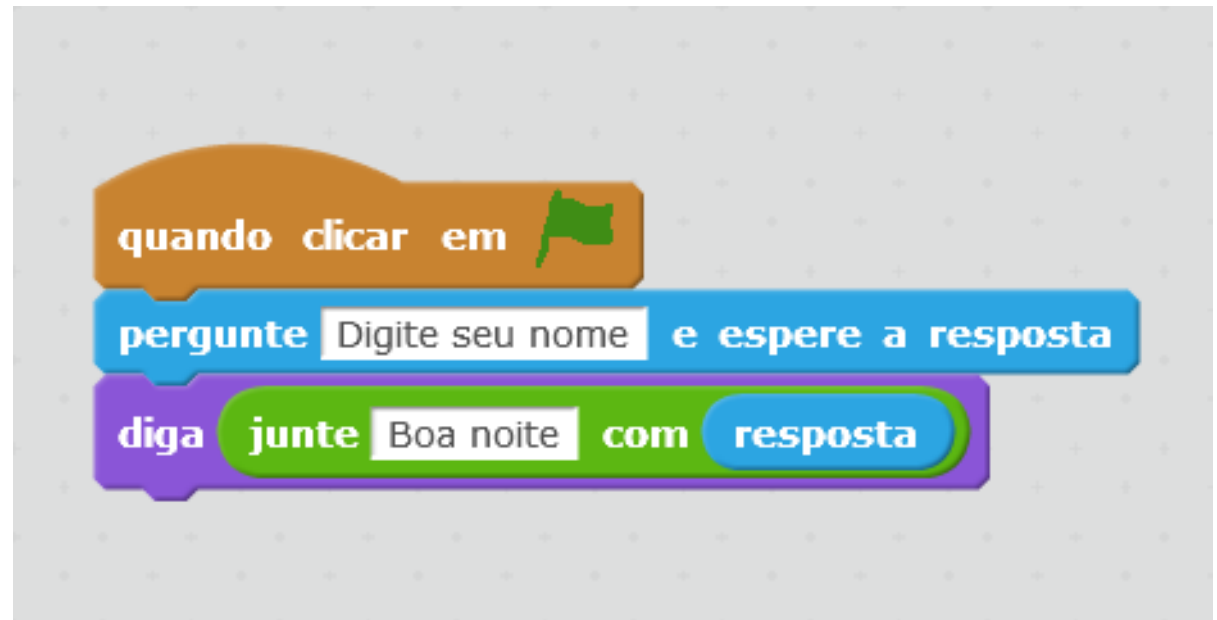
```
#include<iostream>
using namespace std;

void main()
{
    char nome[30];
    cout << "Digite o seu nome: ";
    gets(nome);
    cout << "Boa Noite " << nome;
}
```

# Linguagem JavaScript

```
<script language="Javascript">  
  var nome;  
  nome = prompt("Digite o seu nome");  
  alert("Boa noite " + nome);  
</script>
```

## Comandos do Scratch



# Linguagem Python

```
nome = input("Digite o seu nome: ")  
print("Boa noite " + nome + "!")
```

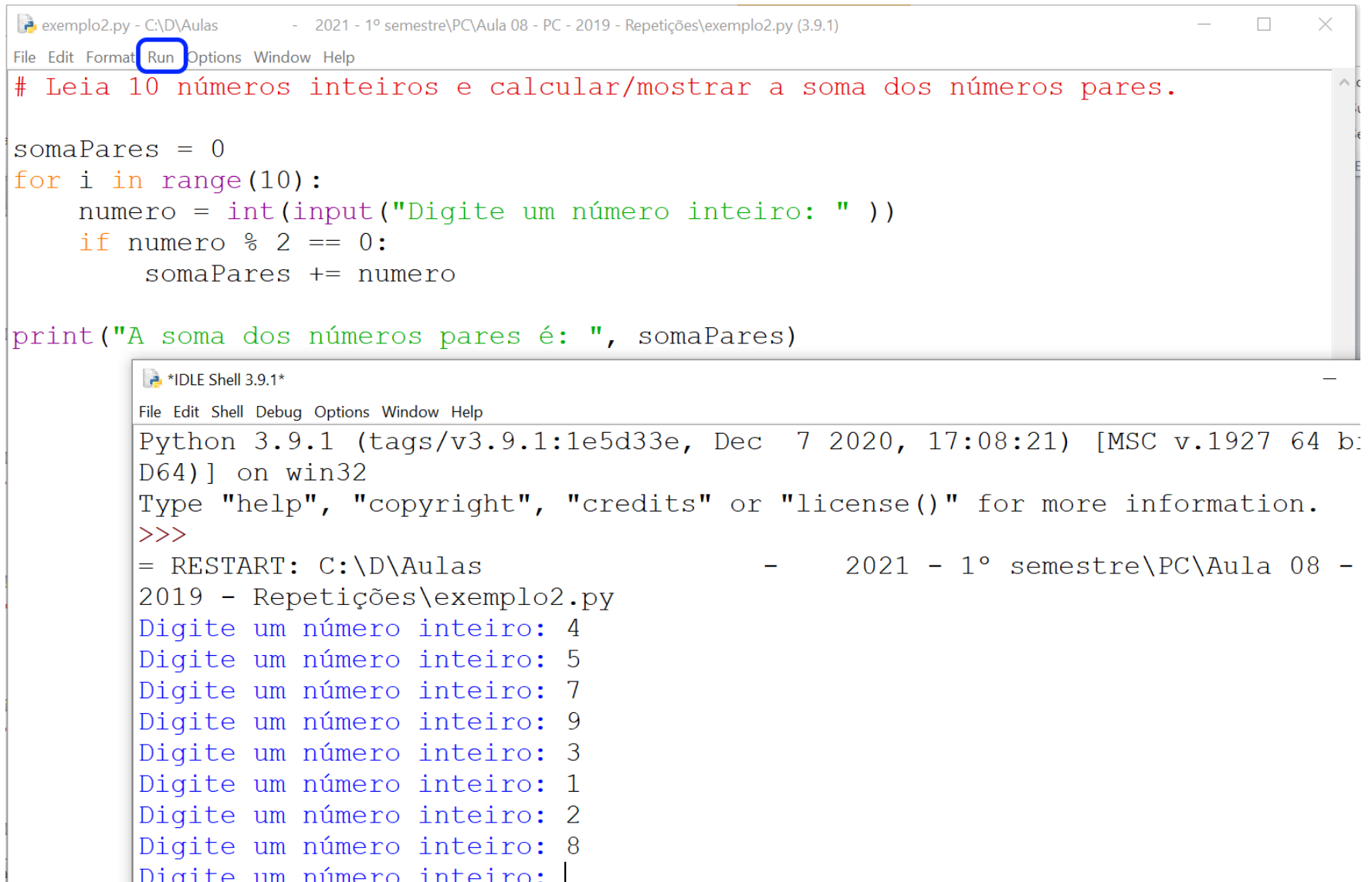
# Processos que sofre um programa...

- 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. Em forma resumida:
  - algoritmo
  - programação
  - edição
  - compilação
  - enlace (link)
  - execução
  - depuração e testes

# Edição, compilação, execução, enlace, depuração

- Os programas são editados, utilizando normalmente um editor de textos especial do próprio ambiente de desenvolvimento (IDE).
- Os programas deverão ser compilados, isto é, traduzidos para código de máquina executável, que o computador consiga finalmente executar.
- Na compilação, o compilador detecta erros sintáticos do programa (se houver) e o enlaça com códigos de bibliotecas necessitadas pelo programa, para gerar, finalmente, o pacote ou arquivo que será carregado/executado.
- O programador faz testes, trabalha na depuração dos erros, otimiza o programa, até a obtenção de uma versão final satisfatória do programa.

# Edição e execução de um programa no Python IDLE



The screenshot displays the Python IDLE environment. The top window, titled 'exemplo2.py - C:\D\Aulas', contains a Python script that calculates the sum of even numbers from 0 to 9. The 'Run' button in the menu bar is highlighted. Below it, the 'IDLE Shell 3.9.1\*' window shows the execution of the script, including the prompt for input and the resulting output.

```
exemplo2.py - C:\D\Aulas - 2021 - 1º semestre\PC\Aula 08 - PC - 2019 - Repetições\exemplo2.py (3.9.1)
File Edit Format Run Options Window Help

# Leia 10 números inteiros e calcular/mostrar a soma dos números pares.

somaPares = 0
for i in range(10):
    numero = int(input("Digite um número inteiro: "))
    if numero % 2 == 0:
        somaPares += numero

print("A soma dos números pares é: ", somaPares)
```

```
*IDLE Shell 3.9.1*
File Edit Shell Debug Options Window Help
Python 3.9.1 (tags/v3.9.1:1e5d33e, Dec 7 2020, 17:08:21) [MSC v.1927 64 b:
D64] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
= RESTART: C:\D\Aulas - 2021 - 1º semestre\PC\Aula 08 -
2019 - Repetições\exemplo2.py
Digite um número inteiro: 4
Digite um número inteiro: 5
Digite um número inteiro: 7
Digite um número inteiro: 9
Digite um número inteiro: 3
Digite um número inteiro: 1
Digite um número inteiro: 2
Digite um número inteiro: 8
Digite um número inteiro: |
```



# Ambientes integrados de desenvolvimento (IDE)

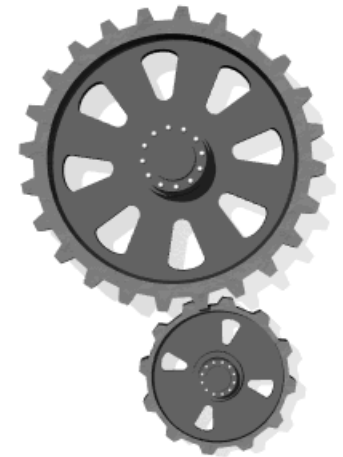
- O código de um programa poderia ser escrito até no Bloco de Notas, mas, dependendo da linguagem de programação, existem ambientes chamados IDEs (*Integrated Development Environment* ou Ambiente Integrado de Desenvolvimento) que facilitam o desenvolvimento do programa, devido à interface amigável e aos recursos que o compõem.
- Alguns exemplos:
  - **Para a linguagem Java:**
    - JBuilder, JCreator, Visual J++, Eclipse, NetBeans, Android Studio.
  - **Para C++:**
    - Dev C++, Turbo C++, C++ Builder, Visual C++.
  - **Para JavaScript, HTML5, CSS3:**
    - Dreamweaver, Brackets, Expression Web, Visual Studio Code.
  - **Para C#, VB, C++** Microsoft Visual Studio .NET.
  - **Para Python** IDLE e outros

# Linguagem compilada X interpretada

- Uma programa pode ser convertido ou traduzido em código de máquina por **compilação** ou **interpretação**, processos que podem ser chamados em forma geral de **tradução**.
- Vamos analisar as diferenças nos próximos slides.

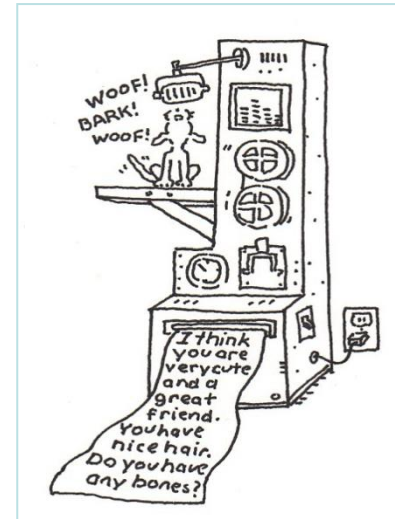
# Linguagem Compilada

- Um compilador converte o código fonte do programa em código de máquina somente depois de não haver mais erros de sintaxe;
- Permite que o programa seja executado quantas vezes for necessário, sem necessidade de recompilação;
- Algumas linguagens compiladas são: C, C++, Pascal, COBOL etc.



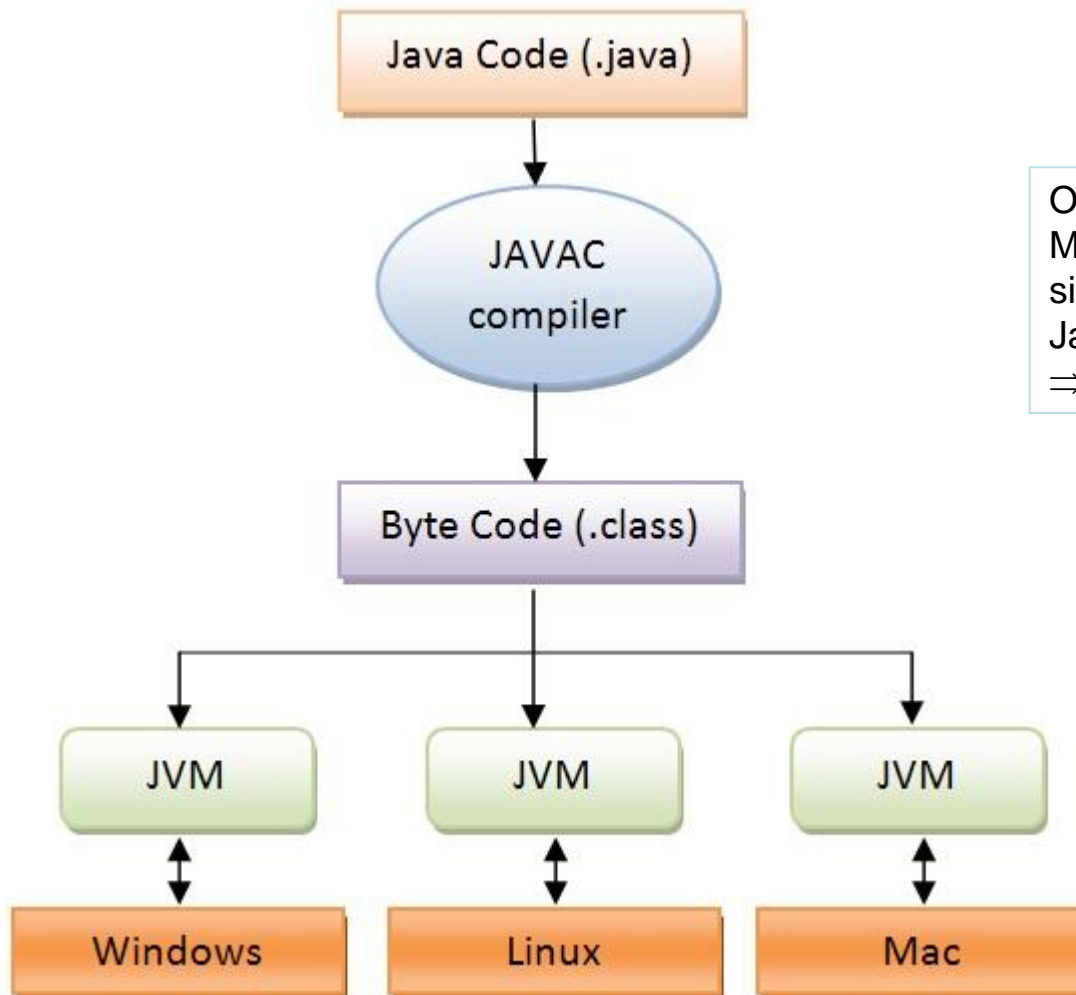
# Linguagem Interpretada

- O código fonte é executado na medida que vai sendo traduzido;
- Sempre que é executado novamente, o código fonte passa pelo interpretador, verificando erros de sintaxe;
- Uma linguagem interpretada, geralmente, é mais lenta que programas compilados;
- São também conhecidas como linguagens de script;
- Algumas linguagens interpretadas são: JavaScript, Python, Perl, PHP, Prolog.



# Linguagens compiladas / interpretadas

- Há linguagens de programação onde os programas são compilados para um código de máquina virtual (seria um **código intermediário**) e então esse código precisará ser verificado e executado por uma **máquina virtual**.
- Uma vez obtido esse código de máquina virtual, ele poderá ser carregado/verificado/executado diretamente. Este segundo processo normalmente é rápido.
- Uma alteração no código fonte do programa necessitará de uma nova compilação para obter o novo código de máquina virtual.
  - Java (veja o próximo slide)
  - C#
  - C++ .NET



Obs: existe uma JVM (Java Virtual Machine) específica para cada sistema operacional ⇒ código Java independente de plataforma ⇒ portabilidade da aplicação.

# Java

## Exemplo1 em Python

# Exemplo1: programa Python que solicita a digitação de  
# dois valores para calcular e mostrar a soma dos mesmos

```
x = float(input("Entre com o 1º valor: "))  
y = float(input("Entre com o 2º valor: "))  
print("A soma dos dois valores digitados é ", (x + y) )
```



## Exemplo2 em Python

# Exemplo2: programa Python que solicita a digitação de  
# dois valores para calcular e mostrar a soma dos mesmos

```
x = float(input("Entre com o 1º valor: "))  
y = float(input("Entre com o 2º valor: "))  
print("A soma dos valores", x, "e", y, "é", x + y)
```





## Exemplo3 em Python

# Exemplo3: programa Python que solicita a digitação de  
# dois valores para calcular e mostrar a soma dos mesmos.  
# Observe a notação de chaves {} e a letra f inicial.

```
x = float(input("Entre com o 1º valor: "))  
y = float(input("Entre com o 2º valor: "))  
print(f"A soma dos valores {x} e {y} é {x + y} ")
```



## Exemplo4 em Python

# Exemplo4: programa Python que solicita a digitação de  
# um valor, para calcular e mostrar seu quadrado.

```
x = float(input("Digite um valor: "))  
print("O quadrado de", x, "é", (x*x))
```



## Exemplo5 em Python

```
# Exemplo5: uma forma de formatar o resultado (saída) para mostrar
# uma quantidade determinada de casas decimais.
soma = 36500 #soma dos salários no ano
media = soma/12.0
# desta forma a média será mostrada com muitas casas decimais:
print("A média dos salários é ", media)
# então, melhor vamos formatar com só duas (2) casas decimais:
print(f"A média dos salários é {media:.{2}f}")
# em lugar de uma constante 2, poderíamos usar uma variável
```



## Bibliografia

BIBLIOGRAFIA BÁSICA	BIBLIOGRAFIA COMPLEMENTAR
FORBELLONE, A. L. V.; EBERSPACHER, H. F. Logica de Programacao: A Construcão de Algoritmos e Estrutura de Dados. 3. ed. Sao Paulo: Makron Books do Brasil, 2005.	ASCENCIO, A. F. G.; CAMPOS, E. A. V. Fundamentos da programação de computadores. São Paulo: Pearson, 2012 (e-book)
MANZANO, J. A. N. G. Algoritmos: Logica Para Desenvolvimento de Programacao. 20. ed. Sao Paulo: Erica, 2007.	DASGUPTA, SANJOY; PAPADIMITRIOU, CHRISTOS; VAZIRANI, UMESH Algoritmos Porto Alegre: Grupo A, 2011 (e-book)
VILARIM, G. O. Algoritmos: Programacao Para Iniciantes. Rio de Janeiro: Ciencia Moderna, 2004.	Python Software Foundation. The Python Language Reference. <a href="https://docs.python.org/3/reference/index.html">https://docs.python.org/3/reference/index.html</a>
	Python Software Foundation. The Python Standard Library. <a href="https://docs.python.org/3/library/index.html">https://docs.python.org/3/library/index.html</a>
	Python Software Foundation. The Python Tutorial. <a href="https://docs.python.org/3/tutorial/index.html">https://docs.python.org/3/tutorial/index.html</a>

Python Software Foundation. **The Python Tutorial.**

<https://docs.python.org/3/tutorial/index.html>

Python Software Foundation. **The Python Standard Library.**

<https://docs.python.org/3/library/index.html>

Python Software Foundation. **The Python Language Reference.**

<https://docs.python.org/3/reference/index.html>