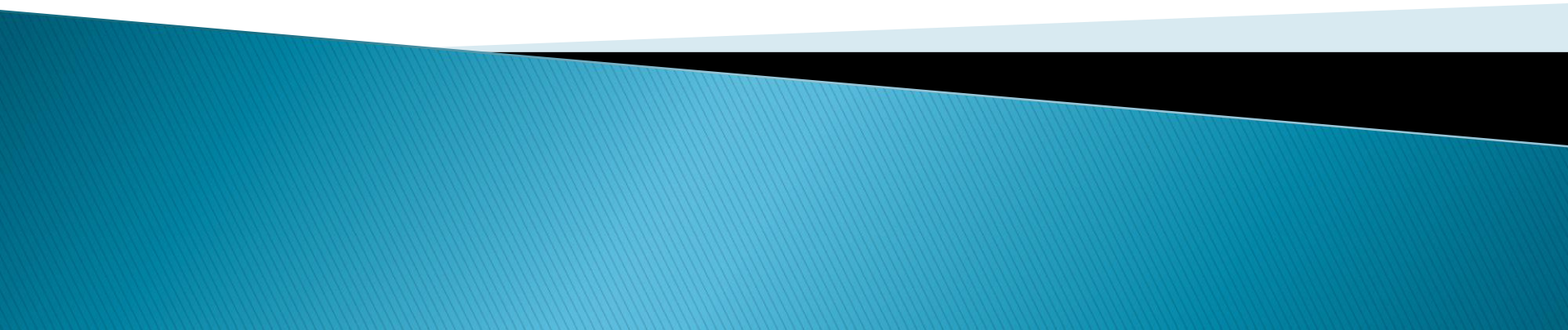


Programação Orientada a Objetos I

Prof. Me. Anderson Costa



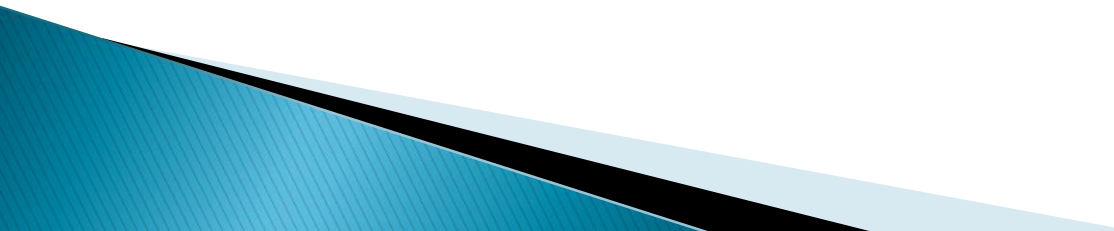
Plano de Ensino

Prof. Me. Anderson Costa

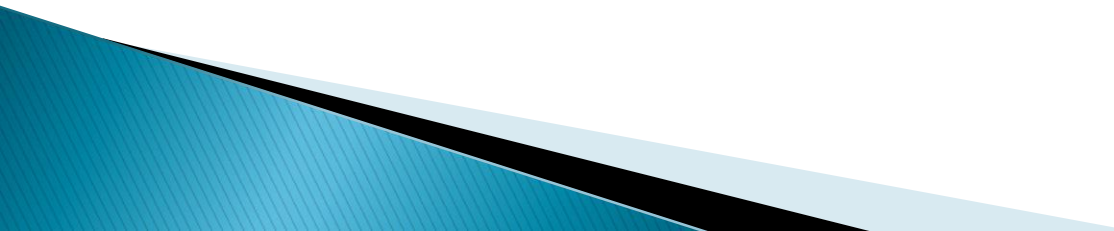
Objetivo

- ▶ Desenvolver programas de computador utilizando o paradigma de Orientação a Objetos


Metodologia

- ▶ Aulas Expositivas
 - ▶ Aulas Práticas
 - ▶ Seminário
 - ▶ Aplicação de Exercícios
 - ▶ Trabalho Prático
- 

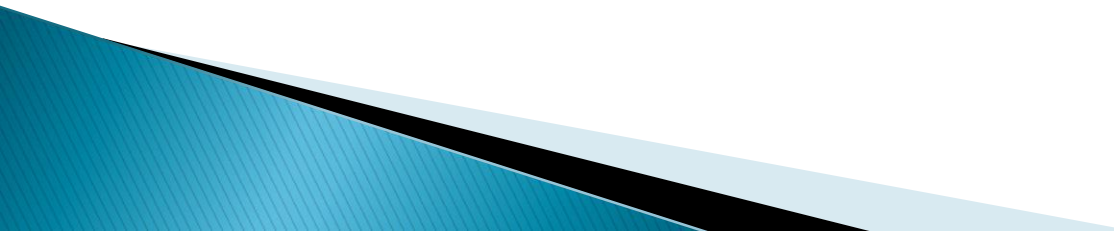
Avaliação

- ▶ Prova Escrita para primeira avaliação
 - ▶ Participação por meio de exercícios
 - ▶ Trabalho Prático
 - ▶ Seminário
- 

Avaliação

- ▶ Prova Escrita para a primeira avaliação equivale a 90% da primeira avaliação
 - ▶ Seminário equivale a 40% da segunda Avaliação
 - ▶ Trabalho Prático equivale a 40% da segunda Avaliação
 - ▶ Participação equivale a 10% para a primeira e para e 20% para a segunda Avaliação
- 

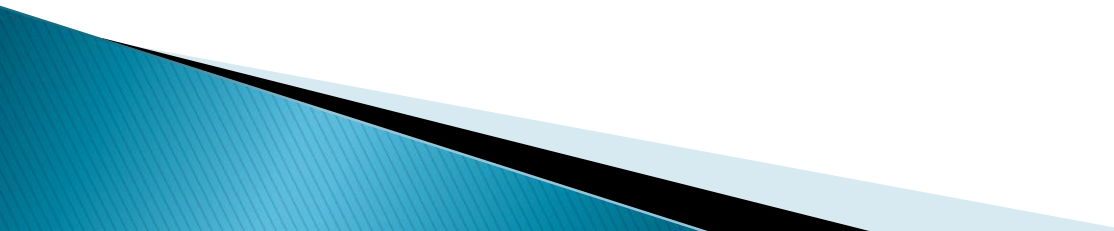
Seminário

- ▶ Seminário:
 - Grupo 1: C++ para WEB
 - Grupo 2: Conexão Java com MySQL
 - Grupo 3: Conexão PHP com MySQL
 - Grupo 4: Java para WEB
 - ▶ A equipe deve definir os tópicos a serem apresentados com foco na Orientação a Objetos
 - ▶ Apresentar em forma de tutorial
- 

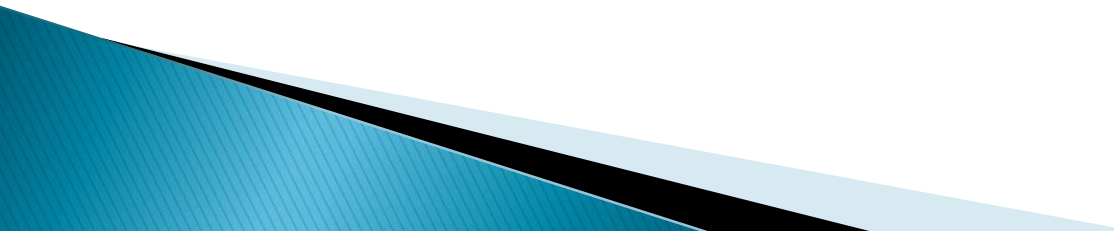
Trabalho Prático

- ▶ Deve-se dividir a turma em quatro equipes
- ▶ Deve-se escolher um sistema a ser desenvolvido
- ▶ O trabalho deve ser realizado em sala

Sumário

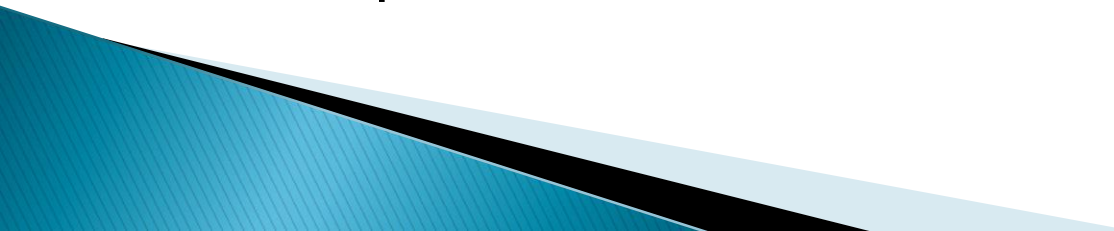
- ▶ Aula 1 – Introdução à orientação a objetos
 - ▶ Aula 2 – Utilização a IDE Eclipse, Tipos de dados, Expressões, fluxo de controle, condições e métodos em Java
 - ▶ Aula 3 – Aplicando POO
 - ▶ Aula 4 – Herança, polimorfismo, classes abstratas, interfaces, exceções
 - ▶ Aula 5 – Aplicações GUI (Graphic User Interface) com Netbeans
 - ▶ Aula 6 – Revisão
 - ▶ Aula 7 – Prova Primeira Avaliação
- 

Sumário

- ▶ Aula 8 – Aula Prática 1 – Identificação do Sistema, planejamento
 - ▶ Aula 9 – Aula Prática 2 – Levantamento e Análise de Requisitos
 - ▶ Aula 10 – Aula Prática 3 – Casos de Uso e Classes
 - ▶ Aula 11 – Aula Prática 4 – Arquitetura e Implementação
 - ▶ Aula 12 – Aula Prática 5 – Continuação da Implementação
 - ▶ Aula 13 – Aula Prática 6 – Continuação da Implementação e Teste
 - ▶ Aula 14 – Apresentação do Seminário
 - ▶ Aula 15 – Entrega e apresentação do Trabalho Prático
 - ▶ Aula 16 – Entrega do Resultado e Prova Final
- 

Bibliografia

► Bibliografia Básica:

- SARAIVA Jr. O. Introdução à Orientação a Objetos com C++ e Python. Uma Abordagem Prática. 1. Ed. 2017. Novatec.
 - HARVEY M. DEITEL E PAUL J. DEITEL. Java – Como Programar. 10. Ed. 2016. Prentice–Hall.
 - SILVA FILHO, Antônio Mendes da. Introdução à Programação Orientada a Objetos com C++. 2010. Campus.
- 

Bibliografia

- ▶ Bibliografia Complementar:
 - NOONAN, Robert; TUCKER, Allen. Linguagem de Programação: princípios e paradigmas. 2. ed. McGraw Hill, 2009.
 - NOONAN, Robert; TUCKER, Allen. Linguagem de Programação: princípios e paradigmas. 2009. 2. ed. McGraw Hill.

Programação Orientado a Objetos

Histórico de OO

- ▶ A OO surgiu no final da década de 60
 - Quando dois cientistas dinamarqueses criaram a linguagem Simula (Simulation Language)
 - 1967 – Linguagem de Programação Simula-67 – conceitos de classe e herança
- ▶ O termo Programação Orientada a Objetos (POO) é introduzido com a linguagem Smalltalk (1983)
- ▶ FINS DOS ANOS 80 ⇒ Paradigma de Orientação a Objetos
 - abordagem poderosa e prática para o desenvolvimento de software

Histórico de OO

- ▶ Surgiram linguagens híbridas:
 - C++ (1986), Object-Pascal (1986)
- ▶ Surgiram diversos Métodos de Análise e Projeto OO
 - **CRC** (Class Responsibility Collaborator, Beecke e Cunningham, 1989)
 - **OOA** (Object Oriented Analysis, Coad e Yourdon, 1990)
 - **Booch** (1991)
 - **OMT** (Object Modeling Technique, Rumbaugh, 1991)
 - **Objectory** (Jacobson, 1992)
 - **Fusion** (Coleman, 1994)
 - **UML** (Unified Modeling Language, 1997)

Orientação a Objetos

- O ser humano se relaciona com o mundo através do conceito de objetos
- Estamos sempre identificando qualquer objeto ao nosso redor
- Para isso lhe damos nomes, e lhes classificamos em grupos
 - Ou seja, classes

PROGRAMAÇÃO ORIENTADA A OBJETOS

- ▶ O que são objetos?
 - Conjunto de variáveis e procedimentos utilizados para modelar objetos do mundo real (e imaginário)
 - Um objeto é caracterizado pelo seu **estado** e pelo seu **comportamento**
- ▶ Exemplo – Lâmpada
 - Estados – ligada/desligada
 - Comportamento – mudar de ligada para desligada
mudar de desligada para ligada

Orientação a Objetos

Aluno
Nome Matrícula Nota Média

Classe

João 193.31.098-7 7,6

Maria 195.31.022-5 8,7

Orientação a Objetos

- Objetos do mundo real possuem **estado** e **comportamento**
- Exemplos:
 - cachorros → **estado**: nome, cor, raça
comportamento: latir, correr
 - Bicicletas → **estado**: marcha atual, velocidade atual
comportamento: trocar marcha, aplicar freios

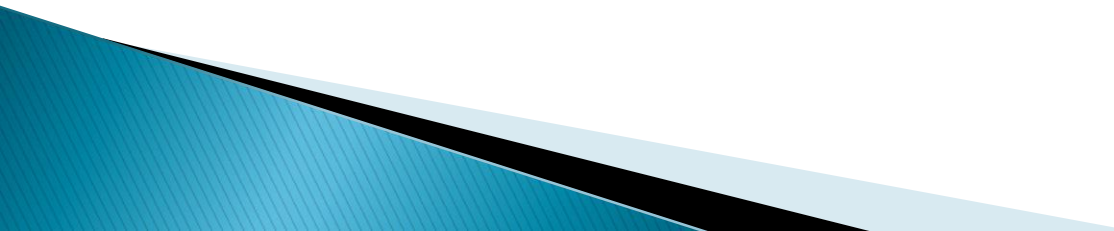
Orientação a Objetos

- Identificar o **estado** e o **comportamento** de objetos do mundo real é o primeiro passo para começar a pensar em programação OO
- Observe um objeto e pergunte:
 - Quais os possíveis estados que esse objeto pode estar?
 - Quais os possíveis comportamentos que ele pode executar?

Orientação a Objetos

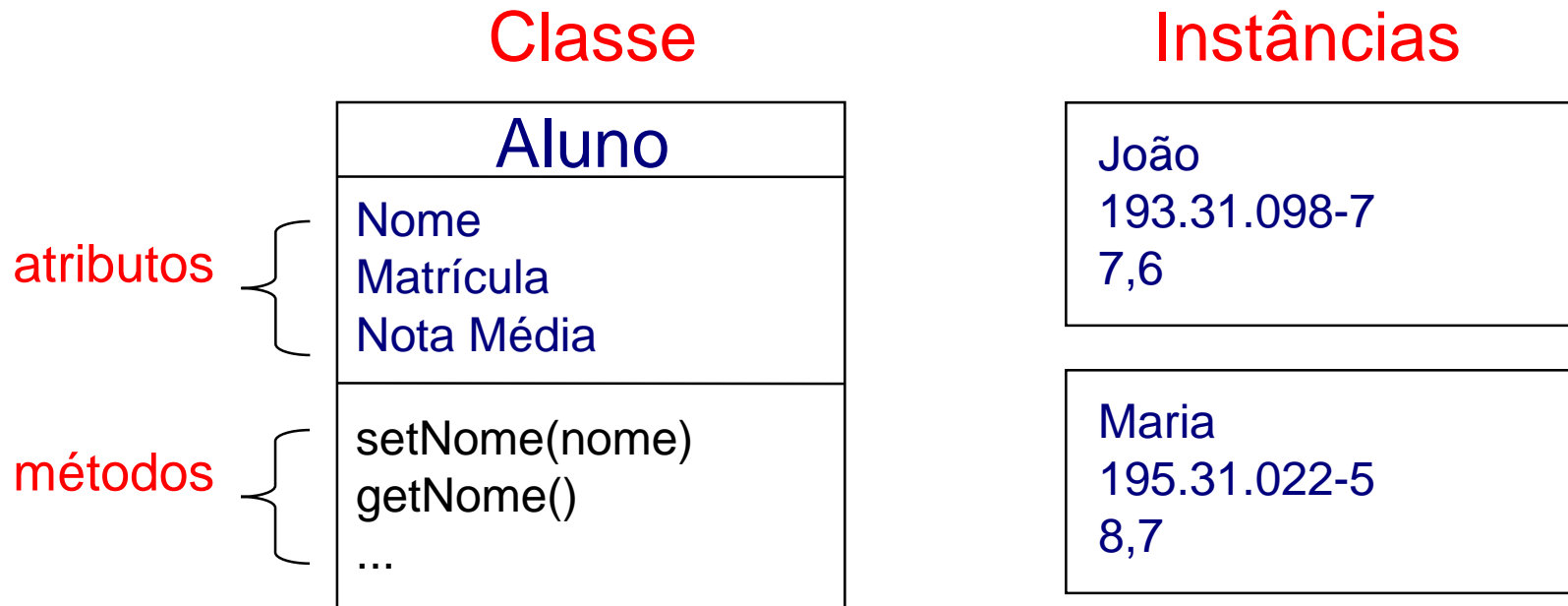
- A *unidade fundamental* de programação em orientação a objetos (POO) é a *classe*
- Classes contém:
 - *Atributos*: determinam o *estado* do objeto
 - *Métodos*: semelhantes a procedimentos em linguagens convencionais
 - São utilizados para *manipular os atributos*

PROGRAMAÇÃO ORIENTADA A OBJETOS

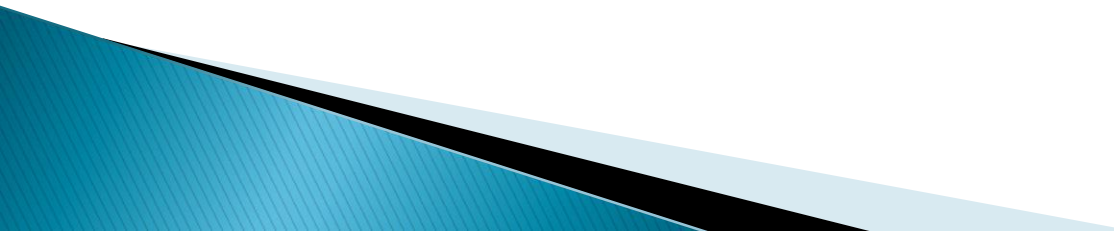
- ▶ Mensagens ou métodos:
 - Mensagens são utilizadas para fazer a comunicação entre os objetos
 - Cada objeto entende um conjunto de mensagens que podem mudar seu estado atual
 - Quando há a necessidade de maiores informações, as mensagens podem conter parâmetros
- 

Orientação a Objetos

- As classes provêm a *estrutura para a construção de objetos* - estes são ditos *instâncias* das classes



Vantagens de OO

- ▶ Abstração de dados: os detalhes referentes às representações das classes serão visíveis apenas a seus atributos
 - ▶ Compatibilidade: a construção das classes e suas interfaces levam a componentes de software que são fáceis de se combinar
 - ▶ Flexibilidade: as classes delimitam-se em unidades naturais para a alocação de tarefas de desenvolvimento de software
- 

Vantagens de OO

- ▶ Reutilização: o encapsulamento dos métodos facilitam o desenvolvimento de software reutilizável
 - Aumenta a produtividade de sistemas
- ▶ Estensibilidade: facilidade de estender o software devido a duas razões:
 - herança: novas classes são construídas a partir das que já existem
 - as classes formam uma estrutura fracamente acoplada o que facilita alterações
- ▶ Manutenibilidade: a modularização natural em classes facilita a realização de alterações no software

Vantagens de OO

- ▶ Redução da quantidade de erros no sistema
 - Diminuindo o tempo nas etapas de codificação e teste
- ▶ Maior dedicação à fase de análise
 - Preocupando-se com a essência do sistema;
- ▶ Mesma notação é utilizada desde a fase de análise até a implementação

Linguagens OO

- ▶ Existem diversas linguagens OO, tais como:
 - Smalltalk (1972)
 - Ada (1983)
 - Eiffel (~1985)
 - Object Pascal (1986)
 - Common Lisp (1986)
 - C++ (~1989)
 - PHP
 - Java

Conceitos Básicos

- ▶ Orientação a Objetos (OO) é uma abordagem de programação que procura explorar nosso lado intuitivo
- ▶ Os objetos trocam mensagens que resultam na ativação de métodos
 - Os quais realizam as ações necessárias

Conceitos Básicos

- ▶ Objeto é algo DINÂMICO: é criado em algum método, tem uma vida e morre
- ▶ Assim, durante a execução do sistema, os objetos podem:
 - Ser construídos
 - Executar ações
 - Ser destruídos

Conceitos Básicos

► Objetos e Classes

Palio	JWO-4567
--------------	-----------------

Parati	KLJ-0978
---------------	-----------------

Celta	JDK-6543
--------------	-----------------



OBJETOS

(Instâncias da classe Automóvel)

Automóvel
Marca
Placa



CLASSE

Conceitos Básicos

▶ Objetos:

- Objeto, no mundo físico, é tipicamente um produtor e consumidor de itens de informação
- Definição (mundo do software)
 - “Qualquer coisa, real ou abstrata, a respeito da qual armazenamos dados e métodos que os manipulam”
Martin, Odell (1995)
- Abstração de uma entidade do mundo real

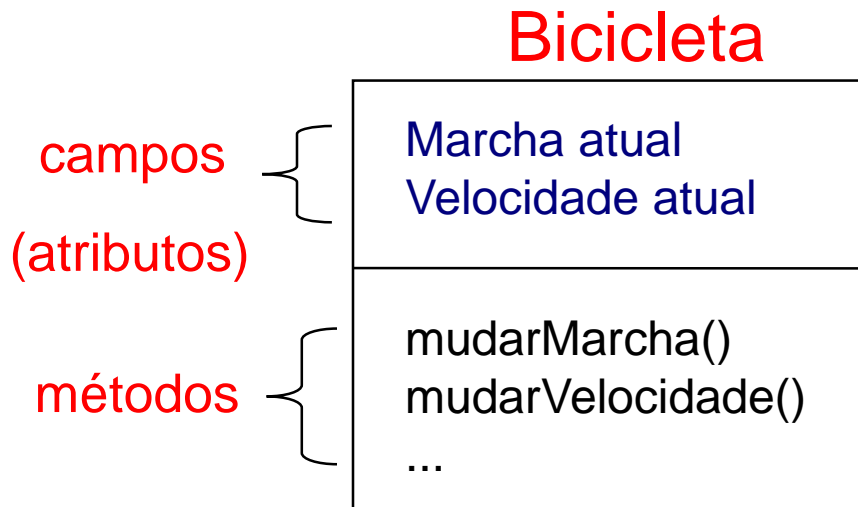
Orientação a Objetos

- Objetos
 - São instâncias da classe
 - Sob o ponto de vista da programação orientada a objetos
 - Um objeto não é muito diferente de uma variável
- Um programa orientado a objetos é composto por um conjunto de objetos que interagem entre si

Orientação a Objetos

- Objetos
 - Objetos de software são conceitualmente similares a objetos do mundo real:
 - Consistem do **estado** e do **comportamento** relacionado
 - Um objeto armazena seu estado em **campos** (variáveis) e expõe seu comportamento através de **métodos** (funções)

Orientação a Objetos



Instâncias

Bibicleta A

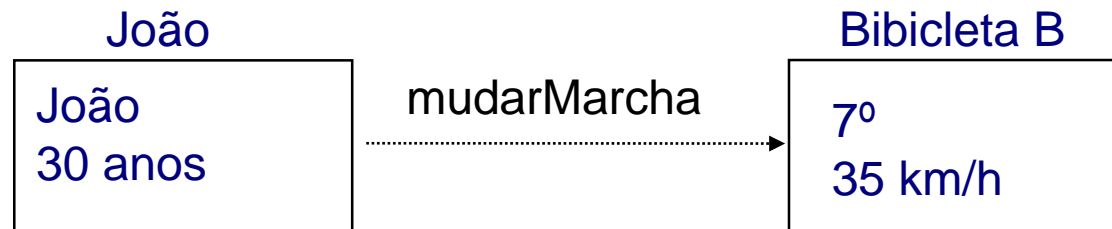
3ª
20 km/h

Bibicleta B

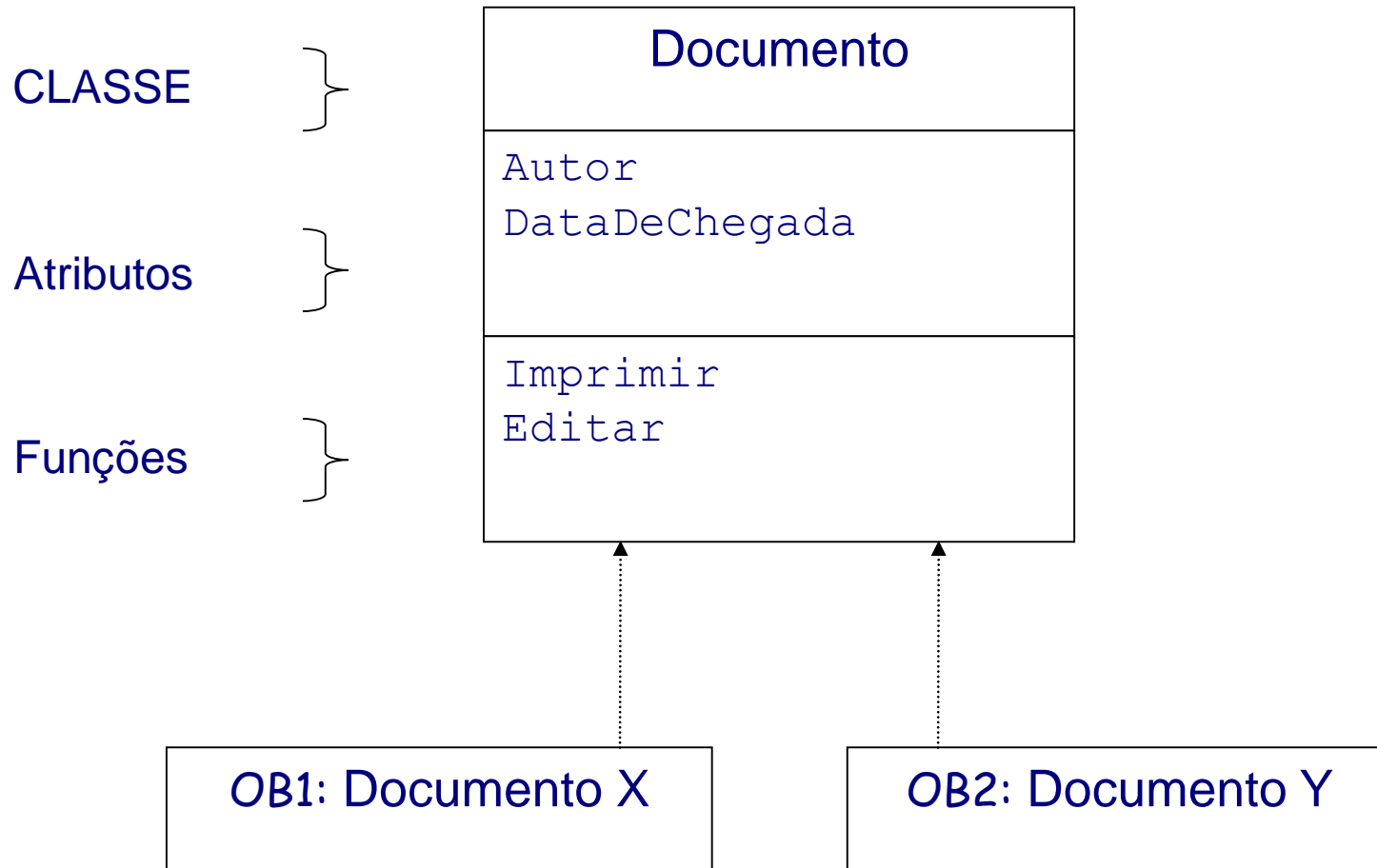
7ª
35 km/h

Orientação a Objetos

Métodos operam no estado interno de um objeto e servem como mecanismo de comunicação entre objetos



Orientação a Objetos – Classes x Objetos



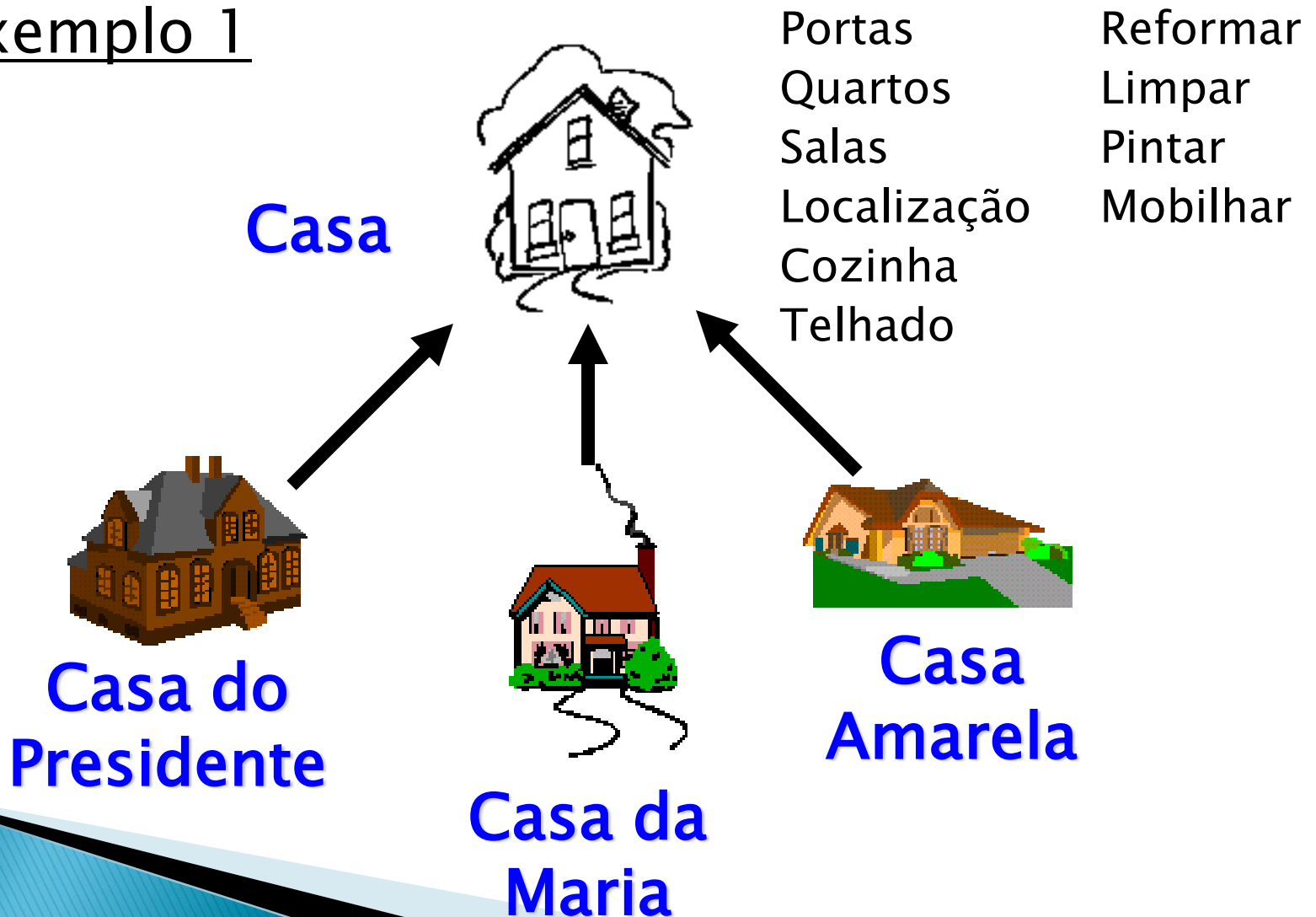
Conceitos Básicos

▶ Classes:

- Agrupamento de objetos similares
- Todo objeto é uma instância de uma Classe
- Os objetos representados por determinada classe diferenciam-se entre si pelos valores de seus atributos
- Conjunto de objetos que possuem:
 - Propriedades semelhantes (ATRIBUTOS)
 - O mesmo comportamento (MÉTODOS)
 - Os mesmos relacionamentos com outros objetos
 - A mesma semântica.

Conceitos Básicos

► Exemplo 1



Conceitos Básicos

▶ Atributos e Métodos: Exemplo 2

Automóvel
Proprietário Marca Placa Ano
Registrar Transferir_Proprietário Mudar_Placa



ATRIBUTOS



MÉTODOS

Conceitos Básicos

▶ Atributos e Métodos: Exemplo 3

Figura
Largura Altura Posicao_X Posicao_Y Cor_preenchimento
Mover Redimensionar




ATRIBUTOS



MÉTODOS

Conceitos Básicos

▶ Atributos:

- Representam um conjunto de informações, ou seja, elementos de dados que caracterizam um objeto
 - Descrevem as informações que ficam escondidas em um objeto para serem exclusivamente manipulados pelas operações daquele objeto
 - São variáveis que definem o estado de um objeto, ou seja, são entidades que caracterizam os objetos
 - Cada objeto possui seu próprio conjunto de atributos
- 


Conceitos Básicos

▶ Métodos:

- Os processos que podem mudar a estrutura de dados dos objetos são denominados Operações ou Métodos
- Métodos são invocados por Mensagens
- Cada objeto possui seu próprio conjunto de métodos

Conceitos Básicos

▶ Encapsulamento:

- Objetos encapsulam seus atributos
 - Propriedade segundo a qual os atributos de uma classe são acessíveis apenas pelos métodos da própria classe
 - Outras classes só podem acessar os atributos de uma classe invocando os métodos públicos
 - Restringe a visibilidade do objeto mas facilita a manutenção
- 

Conceitos Básicos

▶ Herança:

- É o mecanismo pelo qual:
 - Uma subclasse herda todas as propriedades da superclasse
 - Além de acrescentar suas próprias características
- As propriedades da superclasse não precisam ser repetidas em cada subclasse
- Por exemplo:
 - JanelaRolante e JanelaFixa são subclasses de Janela
 - Elas herdam as propriedades de Janela
 - JanelaRolante acrescenta uma barra para afastamento

Conceitos Básicos

► Herança



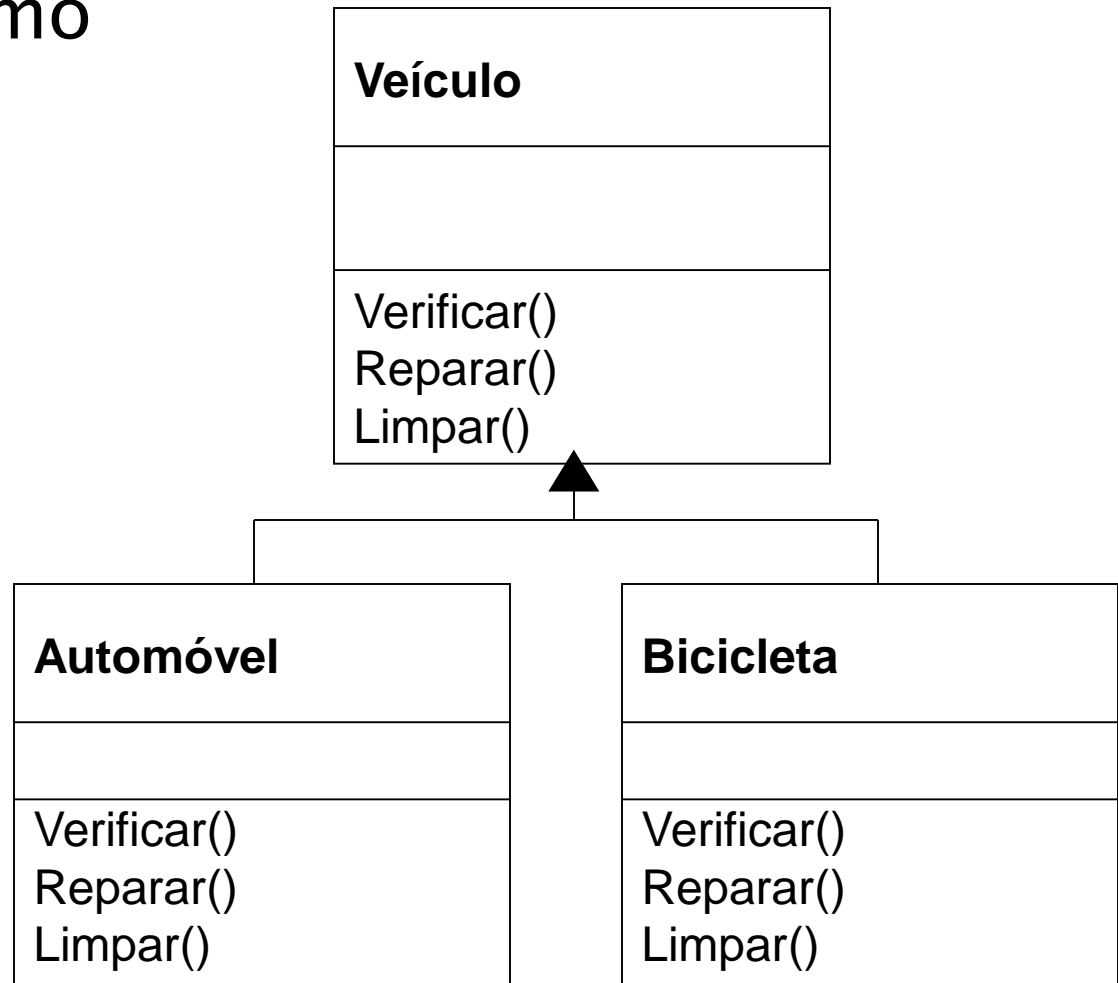
Conceitos Básicos

► Polimorfismo:

- Geralmente representa a qualidade ou estado de um objeto ser capaz de assumir diferentes formas
- Por exemplo, vários métodos podem existir com o mesmo nome
- Permite a criação de várias classes com interfaces idênticas, porém objetos e implementações diferentes
- Exemplos:
 - O operador “+” pode ser usado com inteiros, pontos flutuantes ou strings
 - A operação mover pode atuar diferentemente nas classes Janela e PeçaDeXadrez

Conceitos Básicos

► Polimorfismo



Conceitos Básicos

- ▶ Analogia dos conceitos principais no paradigma orientado a objeto e no paradigma tradicional de programação

Linguagens Orientadas a Objetos

Objeto

Classe

Mensagem

Método

Interface

Linguagens Tradicionais

Valor

Tipo

Chamada de Procedimento

Procedimento ou Função

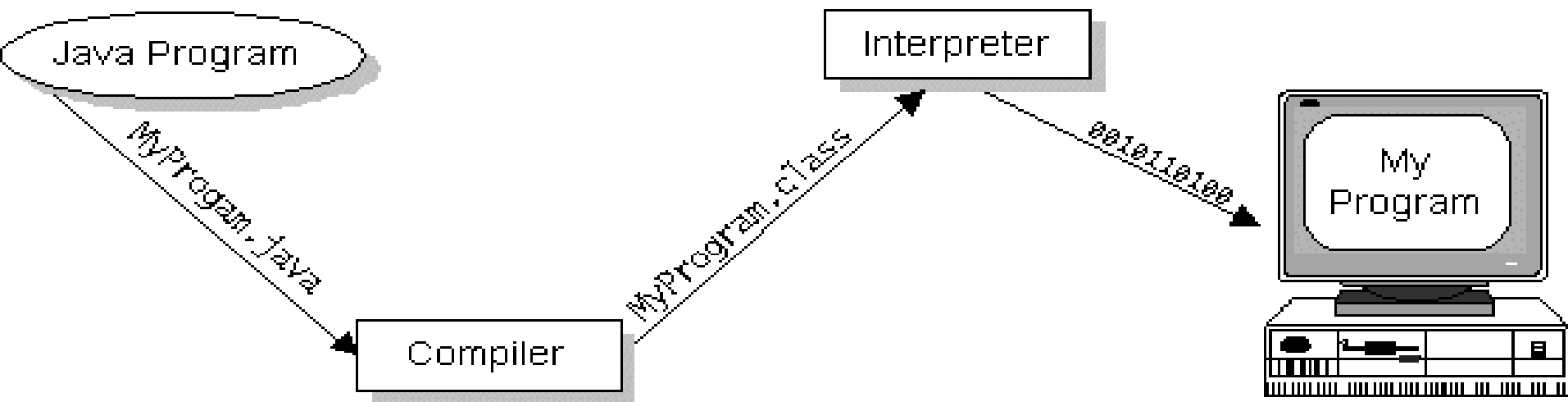
Conjunto de nomes e funções
para um fim específico

O QUE É JAVA ?

- ▶ Linguagem de Programação
 - Simples
 - Orientada a Objetos
 - Distribuída
 - Portável
 - Robusta
 - Interpretada
 - Independente de arquitetura
 - Alta Performance
 - Dinâmica

O QUE É JAVA ?

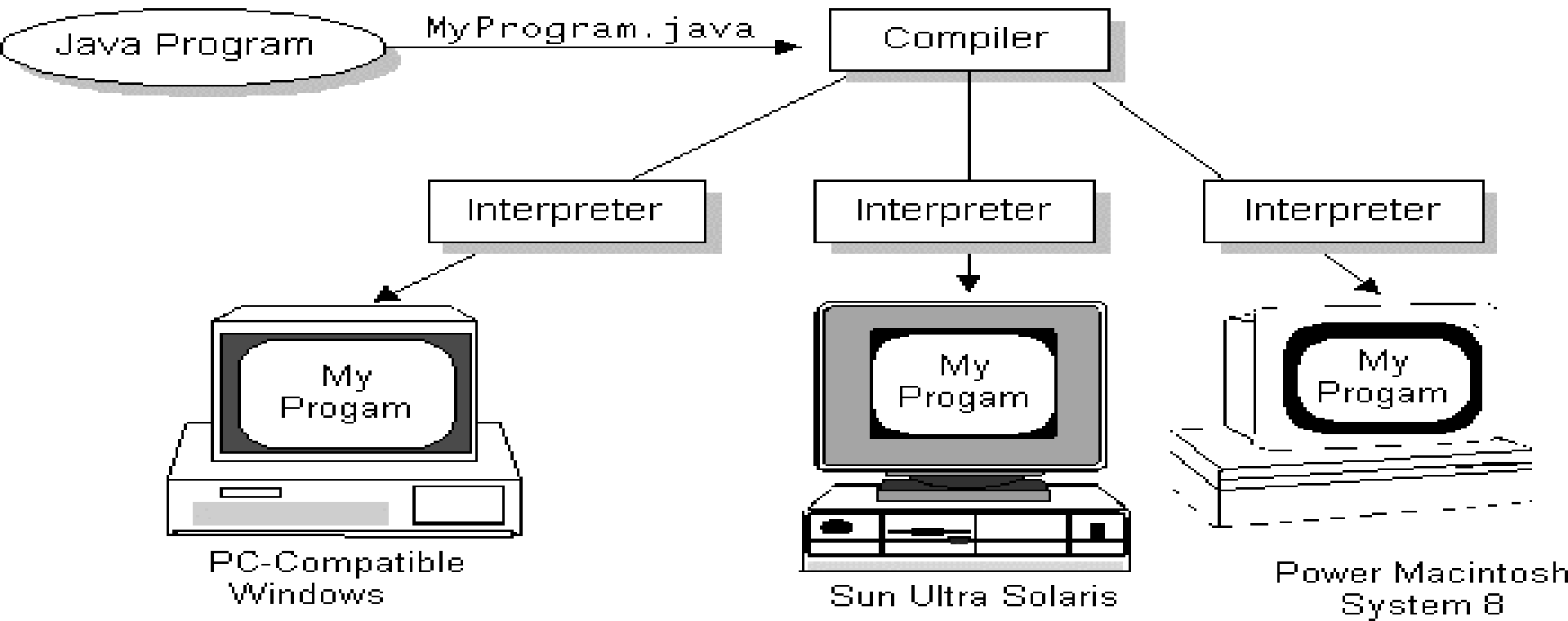
- ▶ Linguagem de Programação
 - Todos os programas Java são compilados e interpretados
 - O compilador transforma o programa em *bytecodes* independentes de plataforma
 - O interpretador testa e executa os *bytecodes*



O QUE É JAVA ?

▶ Linguagem de Programação

- Cada interpretador é uma implementação da JVM – *Java Virtual Machine* (ferramenta, browser, hardware)
- “*Write Once, Run Anywhere*”



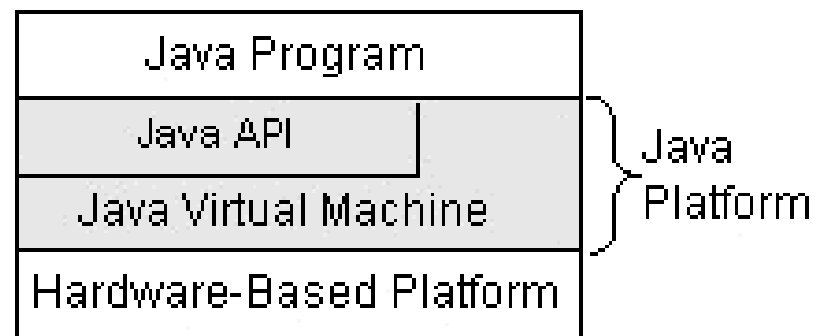
O QUE É JAVA ?

▶ Plataforma Java

- Uma plataforma é o ambiente de hardware e software onde um programa é executado
- A plataforma Java é um ambiente somente de software
- Componentes:

Java Virtual Machine (Java VM)

Java Application Programming Interface (Java API)



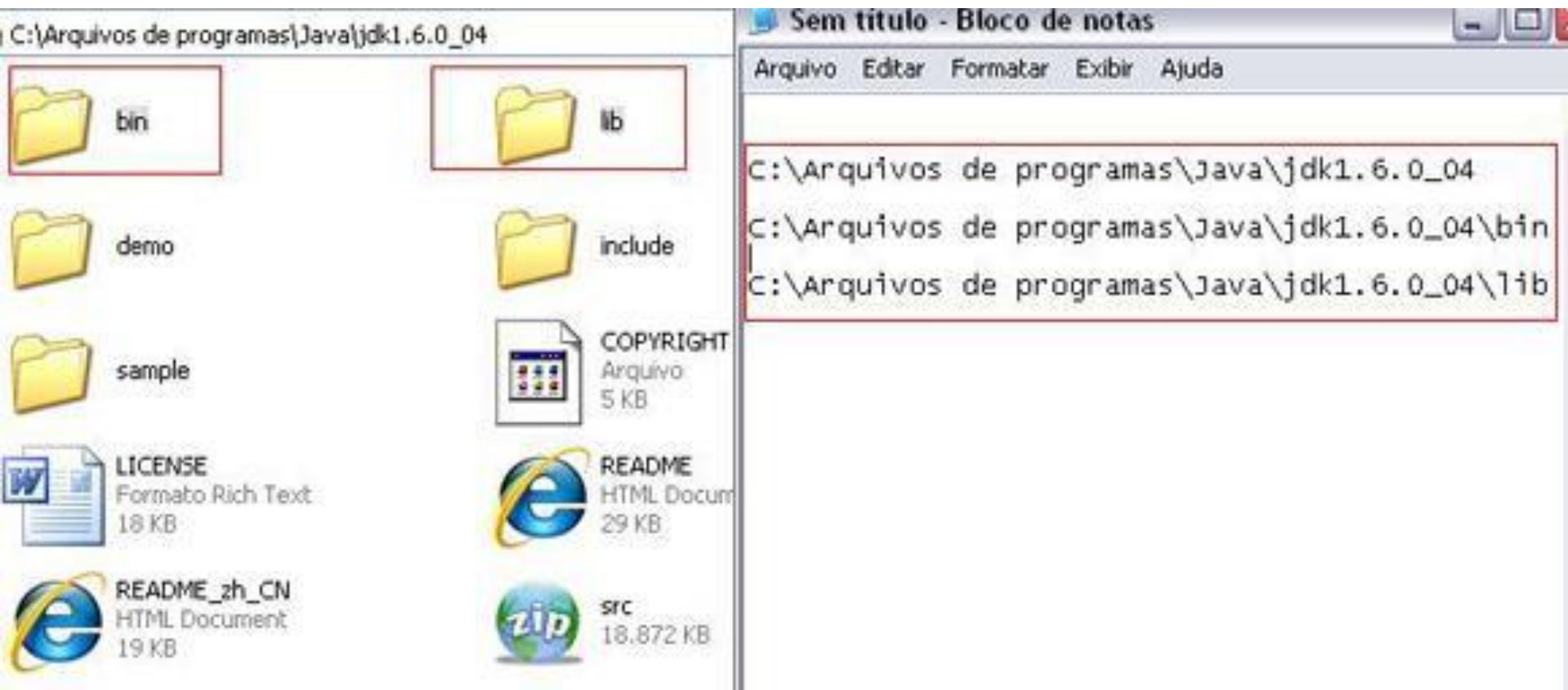
Instalando e Configurando o Java

Instalando e Configurando o Java

- ▶ Deve-se instalar o JDK do Java
 - Diferente do JRE (Java Runtime Environment) que só possui o Interpretador Java
 - O JDK (Java Development Kit) possui o compilador java
- ▶ Após a instalação anote o caminho onde instalou o java JDK
- ▶ Por padrão seria **C:\Arquivos de programas\Java**

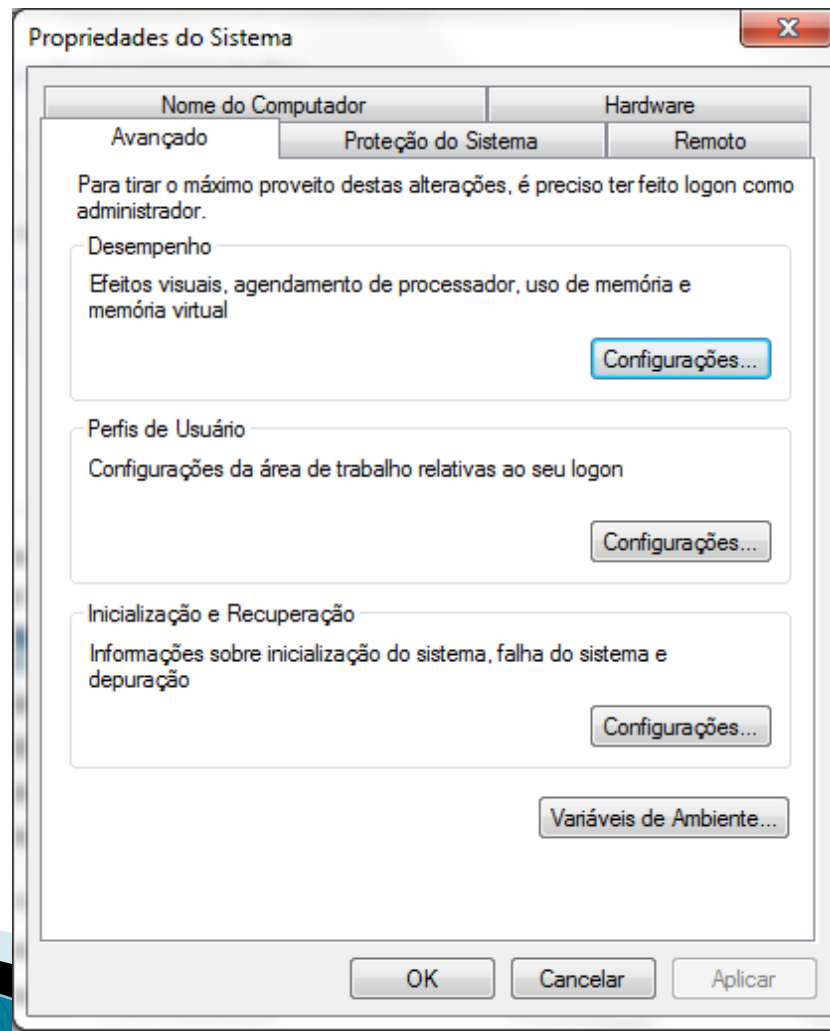
Instalando e Configurando o Java

- ▶ Copie o caminho até esta pasta jdk
 - C:\Arquivos de programas\Java\jdk1.6.0_04\



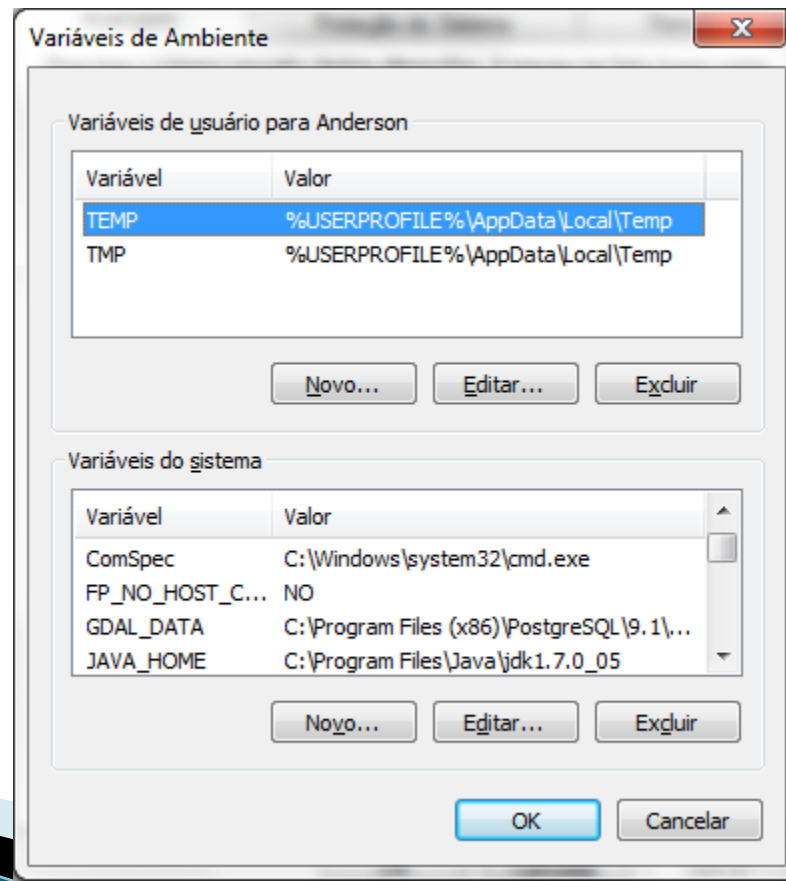
Instalando e Configurando o Java

- ▶ Acesse a opção de Propriedades do Sistema Windows e selecione as **Variáveis de Ambiente**



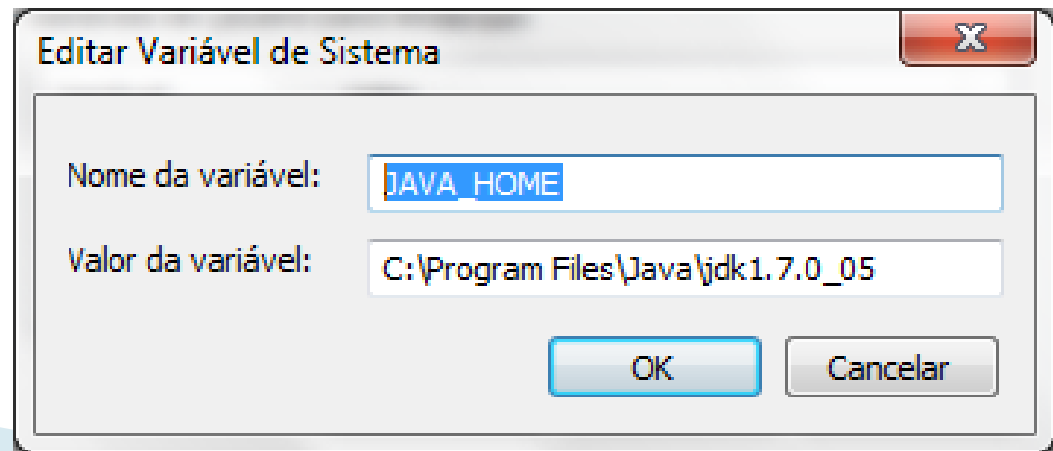
Instalando e Configurando o Java

- ▶ Em **Variáveis do Sistema** deve criar ou editar (caso já exista a opção JAVA_HOME)



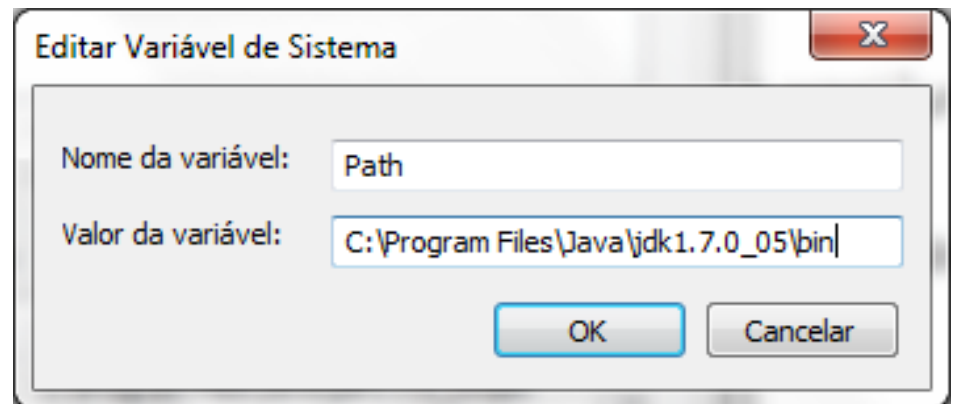
Instalando e Configurando o Java

- ▶ Após isso abrirá esta janela:
- ▶ No 1º campo deverá ser colocado o Nome da Variável (no caso JAVA_HOME)
- ▶ No 2º campo deverá ser colocado o caminho em que se encontra o jdk
- ▶ Confirme



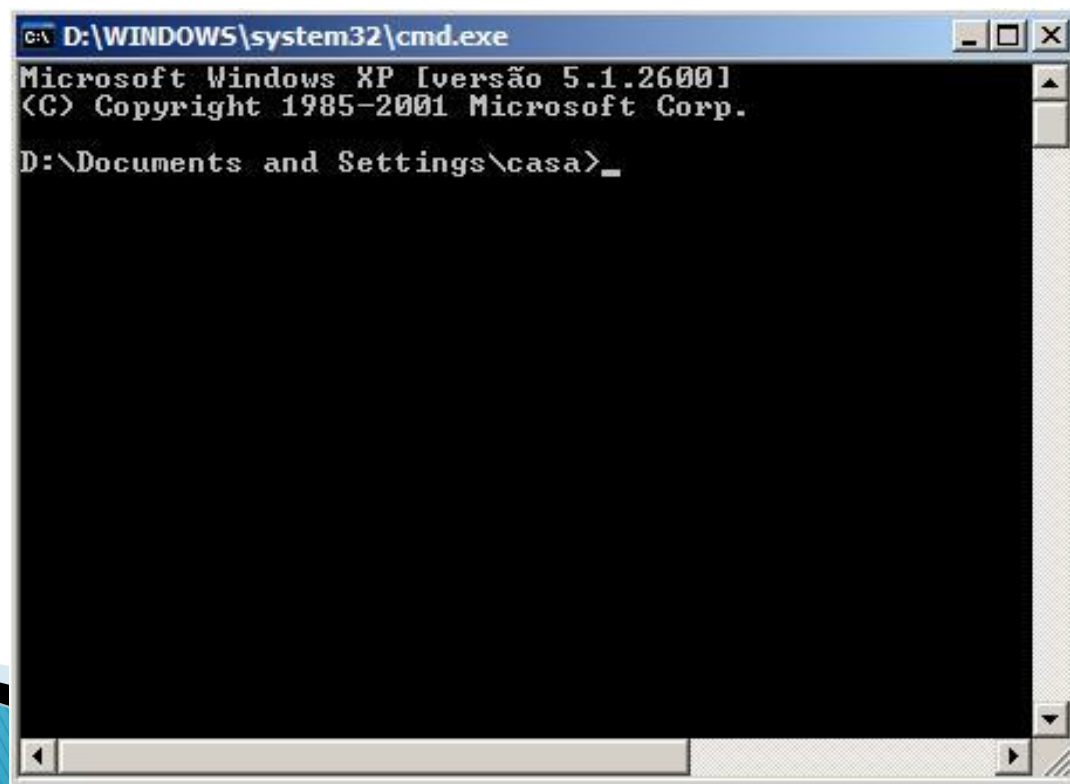
Instalando e Configurando o Java

- ▶ Deve-se criar ou editar a variável Path:
- ▶ No 1º campo deverá ser colocado o Nome da Variável (no caso Path)
- ▶ E no 2º campo deverá ser colocado o caminho do jdk adicionando o \bin ou colocar %JAVA_HOME%\bin



Instalando e Configurando o Java

- ▶ Abra o prompt de comando:
 - No Iniciar > Executar digite `cmd` e click em OK o prompt aparecerá na sua tela:
 - Escreva `Java` e aperte o Enter.



Instalando e Configurando o Java

- ▶ Abra o notepad

Digite:

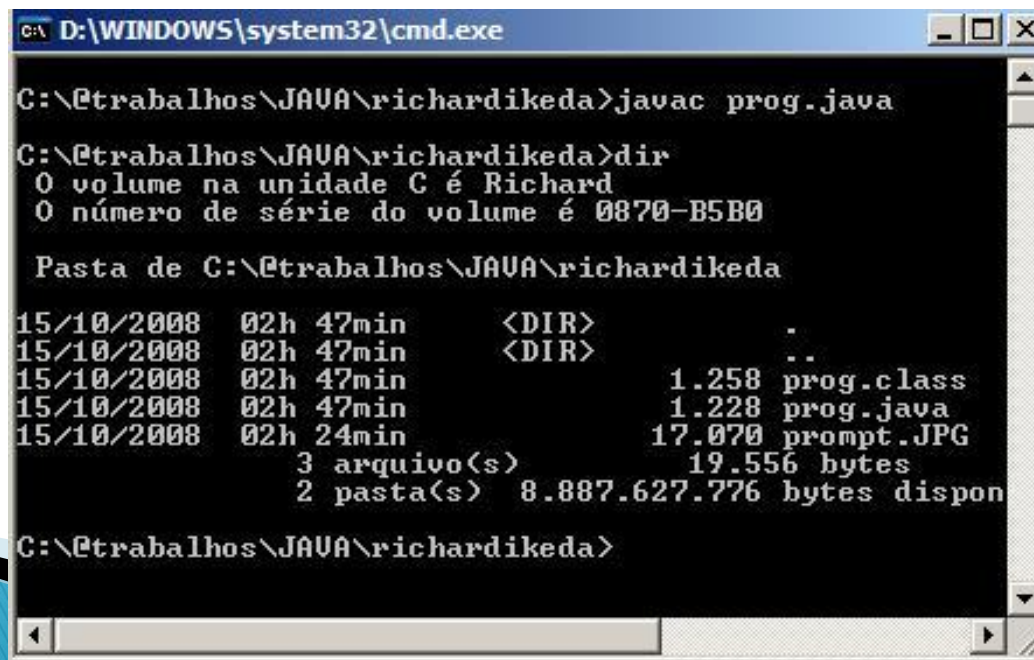
```
class Prog {  
    public static void main( String[] args ) {  
        System.out.println("Hello World!");  
    }  
}
```

Salve com a extensão prog.java



Instalando e Configurando o Java

- ▶ Para compilar
 - Use o comando `javac Prog.java`
 - Na mesma pasta onde está o arquivo foi gerado outro chamado `Prog.class`



The screenshot shows a Windows command prompt window titled "D:\WINDOWS\system32\cmd.exe". The user is in the directory "C:\@trabalhos\JAVA\richardikeda". They have executed the command "javac prog.java". The output shows the directory listing for "C:\@trabalhos\JAVA\richardikeda", which includes a subdirectory ".." and files "prog.class" (1.258 bytes), "prog.java" (1.228 bytes), and "prompt.JPG" (17.070 bytes). The total size of the files is 19.556 bytes, and the total size of the subdirectories is 8.887.627.776 bytes.

```
C:\@trabalhos\JAVA\richardikeda>javac prog.java
C:\@trabalhos\JAVA\richardikeda>dir
O volume na unidade C é Richard
O número de série do volume é 0870-B5B0

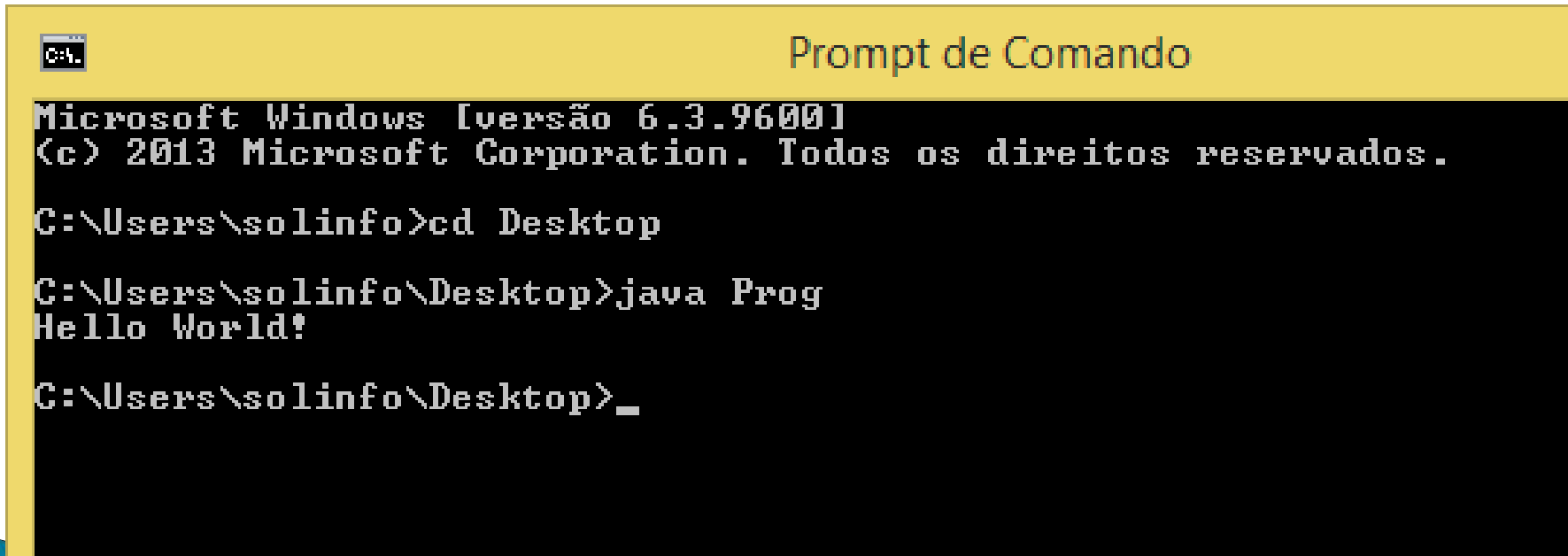
Pasta de C:\@trabalhos\JAVA\richardikeda

15/10/2008  02h 47min    <DIR>          .
15/10/2008  02h 47min    <DIR>          ..
15/10/2008  02h 47min                1.258 prog.class
15/10/2008  02h 47min                1.228 prog.java
15/10/2008  02h 24min               17.070 prompt.JPG
                3 arquivo(s)          19.556 bytes
                2 pasta(s)  8.887.627.776 bytes dispon

C:\@trabalhos\JAVA\richardikeda>
```

Instalando e Configurando o Java

- ▶ Para executar o programa digite o comando **java Prog** (*sem o .java*)
 - O prog.class será executado:



```
C:\>
Microsoft Windows [versão 6.3.9600]
(c) 2013 Microsoft Corporation. Todos os direitos reservados.

C:\Users\solinfo>cd Desktop

C:\Users\solinfo\Desktop>java Prog
Hello World!

C:\Users\solinfo\Desktop>_
```

Orientação a Objetos

Escreva a classe abaixo e faça o mesmo do anterior:

```
public class Lampada
{
    // Atributos
    boolean acesa;
    // Métodos
    public void ligar(){
        acesa = true;
    }
    public void desligar(){
        acesa = false;
    }
}
```

Sem o Método Main não
é possível executar o
código

Orientação a Objetos

```
public class Lampada{  
    boolean acesa = true;  
    public void ligar(){  
        acesa = true;  
    }  
    public void desligar(){  
        acesa = false;  
    }  
    public static void main(String[] args){  
        Lampada l = new Lampada();  
        System.out.println(l.acesa);  
        l.desligar();  
        System.out.println(l.acesa);  
    }  
}
```

A LINGUAGEM JAVA

► O Método *main*

- ```
class HelloWorld {
 public static void main(String[] args) {
 System.out.println("Hello World!");
 }
}
```
- Quando o interpretador Java executa uma aplicação, ele começa chamando o método `main`
- O método `main` então chama todos os outros métodos necessários para executar a aplicação

# Exercício

1. Descreva o que você entendeu sobre Orientação a Objetos
  2. Diferencie Classe de Objeto
  3. O que são métodos?
  4. O que são atributos?
  5. Quais são as vantagens da Orientação a Objetos?
  6. O que é Java?
  7. Pesquise exemplos de códigos em Java e execute via linha de comando
- 