Faculdade Pitágoras de Uberlândia

Programação Orientada a Objetos 2017/2

https://goo.gl/F8dPBy

GitHub
https://github.com/mauro-hemerly/POO

Release	Year
JDK Beta	1994
JDK 1.0	1996
JDK 1.1	1997
J2SE 1.2	1998
J2SE 1.3	2000
J2SE 1.4	2002
J2SE 5.0	2005
Java SE 6	2006
Java SE 7	2011
Java SE 8	2014

pitágoras

Java 8

NetBeans

Mauro Hemerly (Hämmerli) Gazzani
maurog@kroton.com.br
mauro.hemerly@gmail.com

Faculdade Pitágoras de Uberlândia

Objetivos

- Compreender os conceitos fundamentais do Paradigma Orientado a Objetos
- O aluno ao final do curso deverá ser capaz de:
  - ✓ Entender os padrões da programação orientada a objetos
  - ✓ Utilizar e entender o conjunto de funções e comandos da linguagem de programação Java.

2

Faculdade Pitágoras de Uberlândia
Conteúdo

1. **INTRODUÇÃO**
   - 1.1. Paradigma de programação orientada a objetos
   - 1.2. Origens e Características da linguagem
   - 1.3. Ambiente de desenvolvimento e execução
   - 1.4. Expressões e comandos
2. **ABSTRAÇÃO E CLASSES**
   - 2.1. Conceito de abstração
   - 2.2. Classes e instâncias
   - 2.3. Encapsulamento
3. **CLASSES EM DETALHES**
   - 3.1. Relacionamentos entre Classes
   - 3.2. Construtores
   - 3.3. Sobrecarga
   - 3.4. Atributos e métodos de classe
   - 3.5. Auto-referência
   - 3.6. Modularização

3

Faculdade Pitágoras de Uberlândia
Conteúdo

4. **HERANÇA**
   - 4.1. Hierarquia de classes
   - 4.2. Classes abstratas
   - 4.3. Polimorfismo
5. **EXCEÇÕES EM DETALHES**
   - 5.1. Gerando exceções
   - 5.2. Criando exceções

4

Faculdade Pitágoras de Uberlândia
Bibliografia

- Barnes, D.J., **Programação Orientada a Objetos com Java**, Pearson Education, 2004.

5

Faculdade Pitágoras de Uberlândia
Bibliografia

- Deitel, H.M. and Deitel, P.J., **Java Como Programar**, Editora Bookman, 2005.

6

Faculdade Pitágoras de Uberlândia
Bibliografia

- Camarao, C., **Programaçao de Computadores em Java,** Editora LTC, 2001.

7

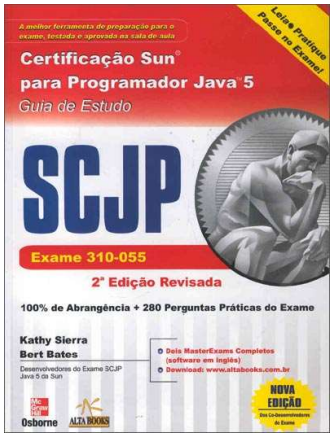
Faculdade Pitágoras de Uberlândia
Bibliografia

- Horstmann, Cay S., **Core Java 2**, Pearson Education, 2001.

8

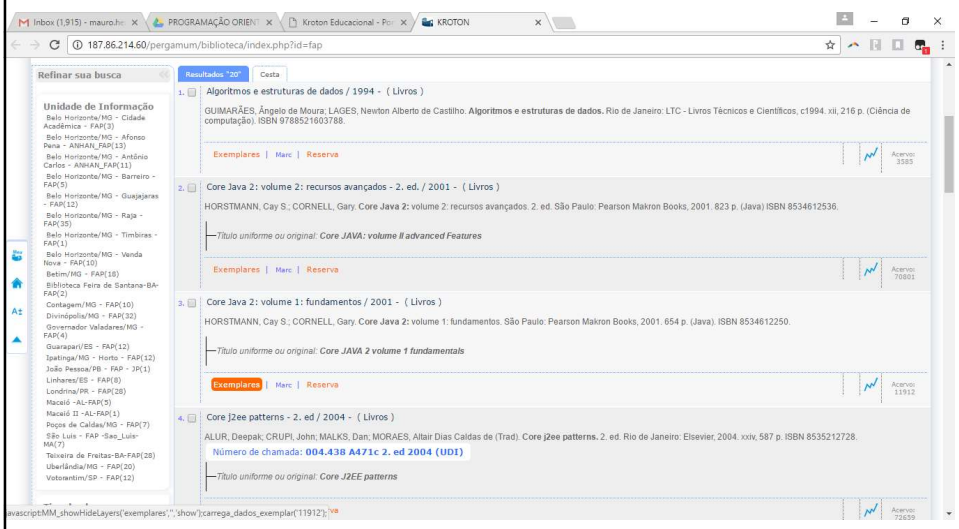
Faculdade Pitágoras de Uberlândia
Bibliografia

- Sierra, K., **Certificação Sun para Programador Java – Guia de Estudo**.



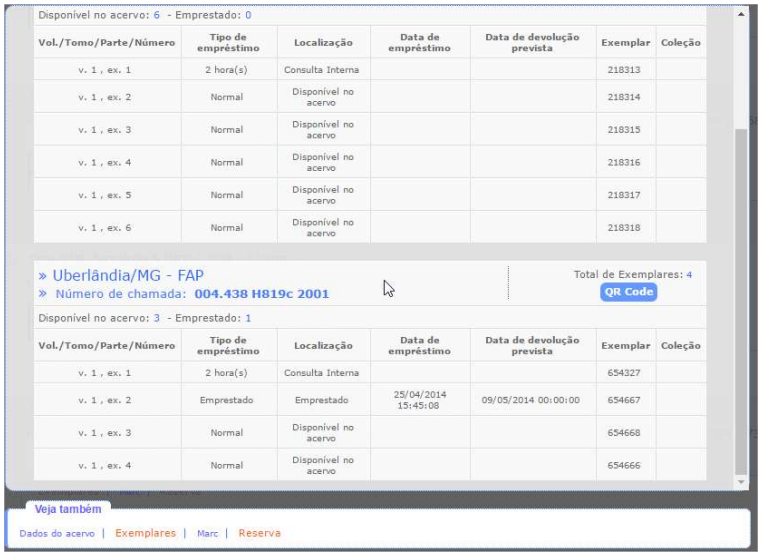
9

Faculdade Pitágoras de Uberlândia
Consulta ao Acervo da Biblioteca



10

Faculdade Pitágoras de Uberlândia
Consulta ao Acervo da Biblioteca



Faculdade Pitágoras de Uberlândia
Avaliação

- **2 Avaliações Bimestrais Individuais** (P1 e P2)
  - ✓ P1 = 10 P2 = 10
- **2 Avaliações Coletivas Bimestrais** (T1 e T2)
  - ✓ As avaliações T1 e T2 serão constituídas de várias atividades coletivas. T1 = 10 T2 = 5 + 5 de PI
- **1B = P1 * 0,7 + T1 * 0,3** (Primeira Bimestral)
- **2B = P2 * 0,7 + T2 * 0,3** (Segunda Bimestral)
- **Média Final** (MF) = 1B * 0,4 + 2B * 0,6
- Aprovação: **MF ≥ 6,0** Reprovação: **MF < 4,0**
- **Exame Final** (EF): 4,0 ≤ MF ≤ 5,9
- **Nota Final** (NF): (MF + EF) / 2 Aprovação: **NF ≥ 6,0**

Faculdade Pitágoras de Uberlândia

1. INTRODUÇÃO
   - 1.1. Paradigma de programação orientada a objetos
   - 1.2. Origens e Características da linguagem
   - 1.3. Ambiente de desenvolvimento e execução
   - 1.4. Expressões e comandos

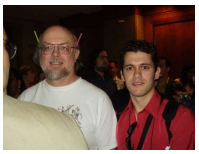
Faculdade Pitágoras de Uberlândia

Introdução

- Java é Linguagem e Plataforma de desenvolvimento de software
- http://www.oracle.com/technetwork/java/index.html (http://java.sun.com)

Faculdade Pitágoras de Uberlândia
Introdução

- 1992
  - ✓ A Sun criou uma equipe (conhecido como Green Team) para desenvolver inovações tecnológicas. Esta equipe foi liderada por James Gosling, considerado o pai do Java. Foi criada uma linguagem para essa tecnologia chamada OAK.
- 1995
  - ✓ Lançamento público Java 1.0
- 1997
  - ✓ JDK 1.1.4 (Sparkler)
- 1998
  - ✓ J2SE 1.2 (Playground)
- 2000
  - ✓ J2SE 1.3 (Kestrel)
- 2002
  - ✓ J2SE 1.4 (Merlin)
- 2004
  - ✓ J2SE 5.0 (Tiger)
- 2006
  - ✓ Java SE 6 (Mustang)
- 2011
  - ✓ Java SE 7 (Dolphin)
- 2014
  - ✓ Java SE 8
    - Update 121 (17/01/17)

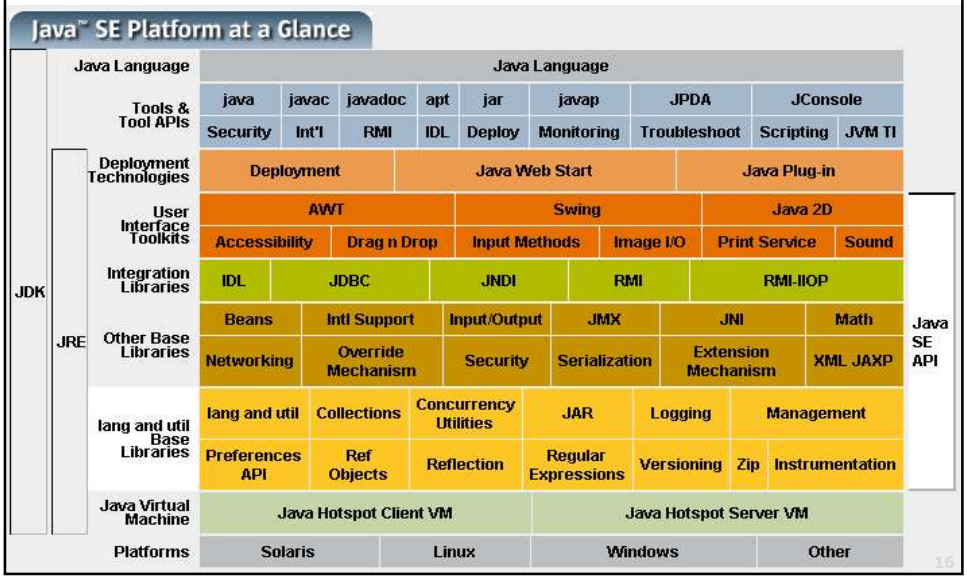


James Gosling e Glênio Damasceno*

James Gosling trabalhou desde 1984 na Sun Microsystems** até abril de 2010, quando se demitiu (02/04/2010). Atualmente, conforme postou em seu blog pessoal [1], Gosling está em um período de descanso, antes de buscar uma nova colocação no mercado.

*** Ex-aluno do curso de SI da Turma de 2001/1** **** Sun Microsystems foi adquirida pela Oracle Corporation em 2009** 15

Faculdade Pitágoras de Uberlândia
Plataforma Java SE

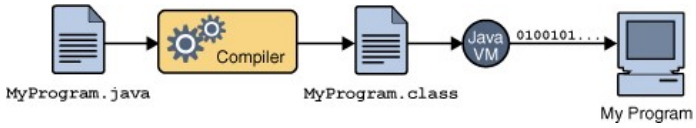


Faculdade Pitágoras de Uberlândia
Ambiente de Desenvolvimento

- Eclipse IDE (www.eclipse.org) – IBM (Maior projeto opensource do mundo)
- NetBeans (www.netbeans.org) – SUN
- Sun Studio Creator e Sun Studio Enterprise (www.java.sun.com) - SUN
- JDeveloper (www.oracle.com) – ORACLE
- IntelliJ (www.jetbrains.com)
- JBuilder (www.codegear.com)
- EditPlus
- NotePad++
- Etc

17

Faculdade Pitágoras de Uberlândia
Compilação



- Todo código Java é escrito em arquivo texto.
- Um compilador compila os fontes gerando arquivos de bytecodes (*.class)
- A execução do programa necessita de uma instância de uma JVM na plataforma (S.O e hardware) local que interpreta os bytecodes.
- O nome bytecode refere-se ao fato de que cada comando da JVM tem código de operação (OPCODE) de um byte
- Veja detalhes em **http://homepages.inf.ed.ac.uk/kwxm/JVM/codeByNo.html**

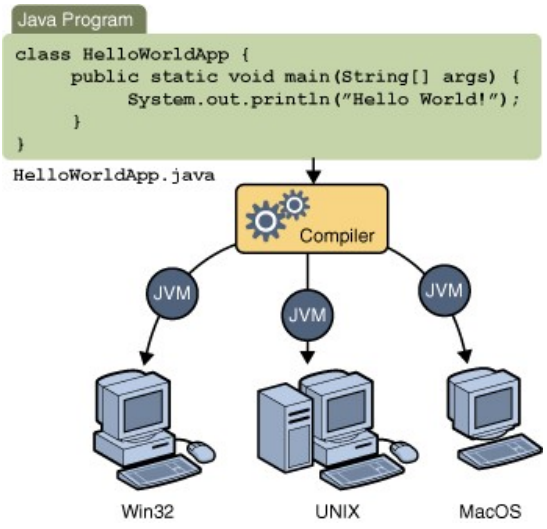
18

Faculdade Pitágoras de Uberlândia
A plataforma Java

- Programas **Java** são executados (interpretados) por outro programa chamado **Java VM**. O programa **Java** é interpretado pela **Java VM** para o S.O. nativo. Isto significa que qualquer computador com a **Java VM** instalada pode rodar programas **Java**, não importando o computador no qual a aplicação foi originalmente desenvolvida.
- Por exemplo, um programa **Java** desenvolvido em um PC com Windows NT rodará sem modificações em uma estação Sun Ultra workstation com S.O. Solaris, e vice-versa.

19

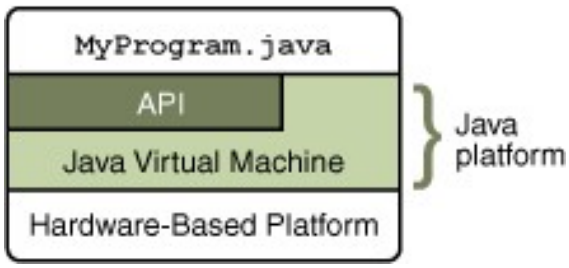
Faculdade Pitágoras de Uberlândia
Portabilidade



20

Faculdade Pitágoras de Uberlândia
A plataforma Java

- A plataforma Java consiste de interfaces para programação de aplicações Java (application programming interfaces – **API**) e a máquina virtual **Java** (Java virtual machine - **JVM**).
  - ✓ **Java APIs** são bibliotecas de código compilado que você pode usar em seus programas



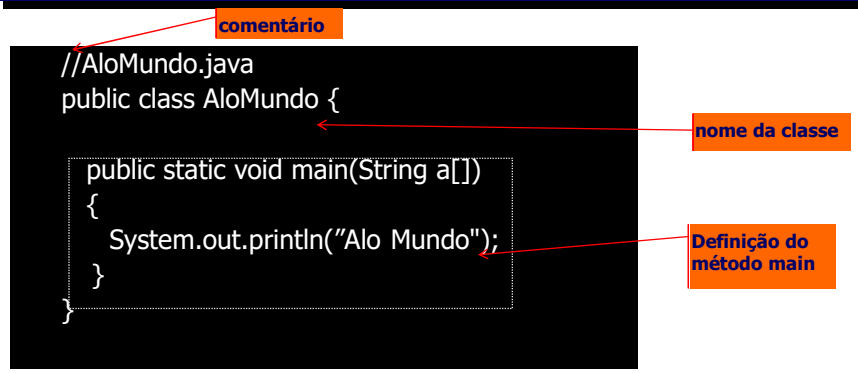
21

Faculdade Pitágoras de Uberlândia
Distribuições

- **JRE** (Java Runtime Environment)
  - ✓ JVM e APIs
- **JDK** (Java Development Kit)
  - ✓ JVM e APIs
  - ✓ Ferramentas de desenvolvimento (linha de comando) – compiladores, debugadores, etc
  - ✓ Fontes
- O download do instalador do **JDK** ou do **JRE** pode ser obtido em http://www.oracle.com/technetwork/java/index.html
- O instalador da **JDK** vem também com o instalador da **JRE**.

22

Faculdade Pitágoras de Uberlândia
Uma aplicação java simples



- Para compilar (no prompt de comandos):
  - ✓ c:/>meus_programas/javac AloMundo.java

23

Faculdade Pitágoras de Uberlândia
Uma aplicação java simples

- O compilador java gera o arquivo AloMundo.class, que é o programa compilado para bytecodes, a linguagem da máquina Java.
- Para executar o programa (interpretação) basta digitar no prompt de comandos:

```
c:/>meus_programas/java AloMundo
```

- E o resultado será:

```
c:/>meus_programas/java AloMundo
Alo Mundo
```

24

Faculdade Pitágoras de Uberlândia
Dissecando o código

- Comentários dentro do código
  - ✓ // resto da linha é comentário
    - Comentário é ignorado pelo compilador
    - Documenta código
  - ✓ /* múltiplas linhas */
  - ✓ /* comentário de muitas
  - ✓ linhas. */
- public class AloMundo
  - ✓ Começa definição da classe AloMundo
    - todo programa Java tem pelo menos uma classe definida pelo programador

25

Faculdade Pitágoras de Uberlândia
Dissecando o código

- Nome da classe é um identificador
  - Sequência de Caracteres consistindo de letras, dígitos, underscores (_) e dollar ($)
  - Não pode começar com um dígito, e não pode conter espaços
  - Case sensitive
    - a1 e A1 são diferentes
- Palavra reservada ***public***
  - modificador de acesso, torna a classe, método, variável ou objeto acessível para todos

26

Faculdade Pitágoras de Uberlândia
Dissecando o código

- Arquivo do código-fonte
  - ✓ Nome do arquivo é o nome da classe com extensão .java
  - ✓ AloMundo.java
- Chave esquerda e direita { ... }
  - ✓ Contém a definição da classe
- *public static void main(String a[])*
  - ✓ Toda aplicação começa a execução pelo método main
    - Parenteses indica que main é um método
    - Aplicações Java contém um ou mais métodos
    - Apenas um método pode ter o nome main

27

Faculdade Pitágoras de Uberlândia
Dissecando o código

```
{
    System.out.println(”Alo Mundo");
}
```

- System.out
  - ✓ Objeto de saída padrão
  - ✓ A saída é a janela do prompt de comandos
- Método System.out.println
  - ✓ Imprime texto
  - ✓ Toda instrução termina com ;
- A definição (ou corpo) do método fica entre { ... }

28

Faculdade Pitágoras de Uberlândia
Dissecando o código

- Caracteres de escape
  - ✓ barra invertida (\ )
    - \n - nova linha
    - \r - retorno de carro
    - \" - aspas duplas
    - \t - tabulação
    - \\ - barra invertida
- Uso
  - ✓ System.out.println(”Bem vindo\na\nJava!");
  - ✓ Saída

```
Bem vindo
a
Java!
```

29

Faculdade Pitágoras de Uberlândia

Objetos e Java

Definindo classes Java

Faculdade Pitágoras de Uberlândia
Objetos

- **Objeto** em software é uma maneira de representar as coisas do mundo real.
- Objeto é um modelo abstrato das **coisas** (reais ou virtuais) do mundo real.
- Coisas tais como um cliente ou agenda de telefones ou uma folha de pagamento ou uma tela com um formulário ou até um simples botão de uma interface gráfica com o usuário.

31

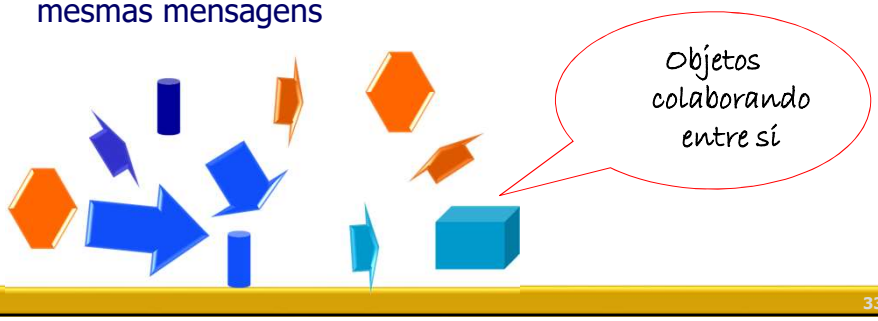
Faculdade Pitágoras de Uberlândia
Objetos

- Objetos possuem:
  - ✓ **Identidade**
    - Cada objeto tem sua própria existência, ou seja, eles "vivem" na memória do computador.
  - ✓ **Estado**
    - Conjunto de características com valores definidos
  - ✓ **Comportamento**
    - Como eles reagem ao mundo. Também dizemos que são as "mensagens" que eles recebem. Geralmente, um comportamento altera o estado do objeto.

32

Faculdade Pitágoras de Uberlândia
Programa Orientado a Objetos

- Características de um **programa** OO
  - ✓ Tudo é **objeto**
  - ✓ Um programa é uma **coleção** de objetos colaborando entre si através do envio de mensagens uns aos outros
  - ✓ Todo objeto possui um tipo (que descreve seus dados)
  - ✓ Objetos de um determinado tipo podem receber as mesmas mensagens



33

Faculdade Pitágoras de Uberlândia
Tipos primitivos e tipo de objeto

- Em **Java** tudo é objeto, exceto alguns valores "primitivos"
  - ✓ uma janela é objeto, um botão de uma interface gráfica com o usuário é um objeto, uma conexão com um banco de dados é um objeto, um programa é um objeto, uma palavra é um objeto, ou seja, quase tudo exceto os primitivos.
- Tipos primitivos
  - ✓ Inteiros: byte, short, int, long
  - ✓ Reais: float, double
  - ✓ Caracter: char
  - ✓ Lógico: boolean

Tipo	Tamanho (bits)	Valor default
byte	8	0
short	16	0
int	32	0
long	64	0
float	32	0.0
double	64	0.0
char	16	\u0000
boolean	-	false

34

Faculdade Pitágoras de Uberlândia
Exemplos de tipos primitivos e literais

- Literais de caracter
  - char c = 'a';
  - char z = '\u0041'; // em Unicode
- Literais inteiros
  - int i = 10; short s = 15; byte b = 1;
  - long hexa = 0x9af0L; int octal = 0633;
- Literais de ponto-flutuante
  - float f = 123.0f;
  - double d = 12.3;
  - double g = .1e-23;

35

Faculdade Pitágoras de Uberlândia
Exemplos de tipos primitivos e literais

- Literais booleanos
  - boolean v = true;
  - boolean f = false;
- Literais de string (não é tipo primitivo - s é uma referência)
  - String s = "abcde";
- Literais de vetor (não é tipo primitivo - v é uma referência)
  - int[] v = {5, 6};

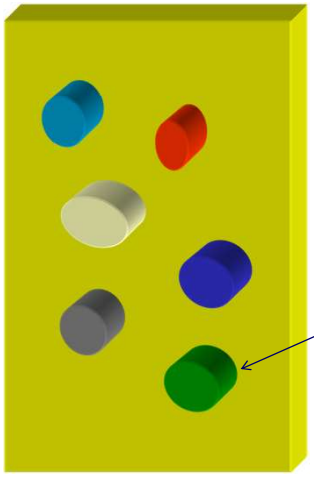
36

Faculdade Pitágoras de Uberlândia
Objetos

- Em **Java**, objetos são armazenados na memória e manipulados por meio de uma referência
- Os objetos possuem valores que definem suas características (estado), funções que definem seu comportamento (métodos) e identidade (referência)
- Um programa **Java** não manipula diretamente seus objetos. Deve-se usar referencias aos objetos para usá-los
- No livro **"Thinking in Java"** de **Bruce Eckel** ele faz uma analogia entre **objetos** e sua **referência** e uma TV e seu controle remoto (veja ilustração na próxima transparência)

37

Faculdade Pitágoras de Uberlândia
Objeto e sua referência

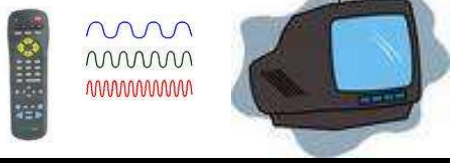


memória RAM do computador

As variáveis em programas Java podem armazenar dois tipos de valores:

1) Valores primitivos `x = 1;`

2) Referências a objetos `x = new objeto();`



•A tv é o objeto e o controle remoto é a referência.
•Acessamos a tv (enviando mensagens) através do controle remoto

38

Faculdade Pitágoras de Uberlândia
O que é classe?

- **Classe** é um documento que descreve um tipo de objeto
- Uma classe representa um tipo de dados complexo
- Classes descrevem
  - ✓ **Tipos** dos dados que compõem o objeto (o que podem armazenar)
  - ✓ **Procedimentos** que o objeto pode executar (o que podem fazer)

39

Faculdade Pitágoras de Uberlândia
Construção de Classes

- **Declaração**

```
[modificadores] class NomeClasse
                      [extends SuperClasse]
                      [implements Interface]
{
        atributos
        métodos
}
```

- **Modificadores**
  - ✓ **Classe pública** (***public***): a classe pode ser utilizada por objetos de fora do pacote. Por default, a classe só pode ser acessada no próprio pacote
  - ✓ **Classe Abstrata** (***abstract***): não pode ter objetos instanciados
  - ✓ **Classe final** (***final***): a classe não pode ter subclasses

40

Faculdade Pitágoras de Uberlândia
Atributos da Classe

- **Declaração de Atributos**
  - [modificador] [chaves] tipo nomeAtributo [= expressão];
- **Modificador**
  - ✓ ***public***: o mundo inteiro pode acessar
  - ✓ ***protected***: somente os métodos da classe e de suas subclasses podem acessar, ou ainda, estando na mesma package
  - ✓ ***private***: somente os métodos da classe podem acessar o atributo
- **Chaves**
  - ✓ ***static***: o atributo é da classe, não do objeto, logo, todos os objetos da classe compartilham o mesmo valor deste atributo
  - ✓ ***final***: o valor do atributo não pode ser alterado (constante)
  - ✓ ***transient***: o atributo não é serializado (não é persistente)

41

Faculdade Pitágoras de Uberlândia
Métodos da Classe

- **Declaração de Métodos**

```
[modificador] [chaves] tipoRetorno nomeMétodo (
  [parâmetros] ) [throws exceptions]
{
  corpo do método (lógica)
}
```

42

Faculdade Pitágoras de Uberlândia
Métodos da Classe

- chaves:
  - ✓ **static**: método da classe e não das instâncias
  - ✓ **abstract**: utilizado somente em classes abstratas, o método não tem corpo
  - ✓ **final**: o método não pode ser sobre-escrito
  - ✓ **synchronized**: declara o método como zona de exclusão mútua no caso de programas concorrentes
- a passagem de parâmetros em Java é sempre por valor.
- Um método é identificado pelo seu nome e pelos parâmetros **(assinatura do método)**

43

Faculdade Pitágoras de Uberlândia
Inicialização de Objetos

- **Construtor da Classe**
  - ✓ tem o mesmo nome que a classe
  - ✓ é chamado na criação do objeto **(new)**

```
class Teste {
  public Teste() { .... }
  public Teste(int i) { ..... }
  .....
}
```

44

Faculdade Pitágoras de Uberlândia
Exemplo

```
class Cachorro {

   // Atributos dos objetos da classe
   private String nome;
   private String cor;
   private int    peso;
   private float  energia;

   // Construtores
   Cachorro(String s) { nome = s; }
   Cachorro() { nome = "Sem nome"; }

   // Métodos (comportamentos dos objetos da classe)
   void setPeso(int v) { peso = v; }
   int getPeso() { return peso; }

   void corre() { ... }
   void late() { ... }
}
```

Cachorro
Nome : String Cor : String Peso : integer Energia : float
Cachorro (s : String) Cachorro () getPeso () : integer setPeso (v : integer) corre () late ()

45

Faculdade Pitágoras de Uberlândia
Exercícios

- 1. Construa as seguintes classes:
  - ✓ Uma Pessoa tem um nome (String)
  - ✓ Uma Porta tem um estado aberto, que pode ser true ou false, e pode ser aberta ou fechada
  - ✓ Uma Casa tem um proprietário Pessoa e um endereço
  - ✓ Um Ponto tem coordenadas x e y inteiras
  - ✓ Um Circulo tem um Ponto e um raio inteiro
  - ✓ Um Pixel é um tipo de Ponto que possui uma cor

46

Faculdade Pitágoras de Uberlândia

Exercícios

- 2. Escreva uma classe Ponto
  - ✓ contém x e y que podem ser definidos em construtor
  - ✓ métodos getX() e getY() que retornam x e y
  - ✓ métodos setX(int) e setY(int) que mudam x e y
- 3. Escreva uma classe Circulo, que contenha
  - ✓ raio inteiro e origem Ponto
  - ✓ construtor que define origem e raio
  - ✓ método que retorna a área
  - ✓ método que retorna a circunferência
  - ✓ use java.lang.Math.PI (Math.PI)
- 4. Crie um segundo construtor para Circulo que aceite
  - ✓ um raio do tipo int e coordenadas x e y

47