ORIENTAÇÃO A OBJETOS

Professora Lucélia Oliveira

CONCEITOS

- Abstração
- Classes
- Objetos
- Atributos
- Métodos
- Método Construtor
- Herança
- Polimorfismo
- Sobrecarga
- Encapsulamento

ORIENTAÇÃO A OBJETOS

- Abstração
- Classes
- Objetos
- Atributos
- Métodos
- Método Construtor
- Herança
- Polimorfismo
- Sobrecarga
- Encapsulamento

ABSTRAÇÃO - CONCEITOS

- "Extrair tudo o que for essencial e nada mais"
  (Aaron Walsh)

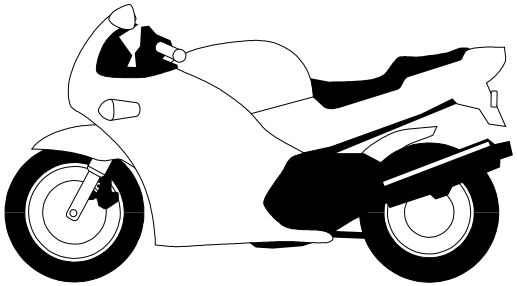
- "A abstração é o processo de filtragem de detalhes sem importância do objeto, para que apenas as características apropriadas que o descrevem permaneçam"
  (Peter Van Der Linden)

- Conceito aplicado a criação de software baseado em objetos, partindo do princípio que devemos considerar a essência de cada objeto e não pensar em todos os detalhes de implementação;

ABSTRAÇÃO

Visão do mundo real:

Estamos acostumados a sempre abstrair de objetos aquilo que nos interessa.



- placa
- cor
- númeroChassi

- aplicarMulta()

- cor
- cilindrada
- velocidadeMax

- acelerar()

ORIENTAÇÃO A OBJETOS

- Abstração
- Classes
- Objetos
- Atributos
- Métodos
- Método Construtor
- Herança
- Polimorfismo
- Sobrecarga
- Encapsulamento

CLASSES

- ✓ As crianças aprendem conceitos simples como pessoa, carro e casa, por exemplo, e ao fazerem isso definem classes, ou seja:

- ✓ É um grupo de objetos, sendo que cada objeto é um exemplo de um determinado grupo.

ORIENTAÇÃO A OBJETOS

- Abstração
- Classes
- Objetos
- Atributos
- Métodos
- Método Construtor
- Herança
- Polimorfismo
- Sobrecarga
- Encapsulamento

OBJETOS

- ✓ É um exemplo de um determinado grupo.

Exemplo: pessoa1

ORIENTAÇÃO A OBJETOS

- Abstração
- Classes
- Objetos
- Atributos
- Métodos
- Método Construtor
- Herança
- Polimorfismo
- Sobrecarga
- Encapsulamento

ATRIBUTOS

- ✓ São as características do objeto.

Exemplos:

pessoa1.nome = “José”;
pessoa1.idade = 21;
pessoa1.endereco = “QSA 10”;

ORIENTAÇÃO A OBJETOS

- Abstração
- Classes
- Objetos
- Atributos
- Métodos
- Método Construtor
- Herança
- Polimorfismo
- Sobrecarga
- Encapsulamento

MÉTODOS

- São subprogramas com finalidades específicas, ou seja, são as operações ou ações realizadas dentro das classes.
- Para o nome do método, é padrão começar com letra minúscula.

Exemplos:

incluirPessoa();

PesquisarPeloNome(nome:String);

ORIENTAÇÃO A OBJETOS

- Abstração
- Classes
- Objetos
- Atributos
- Métodos
- Método Construtor
- Herança
- Polimorfismo
- Sobrecarga
- Encapsulamento

MÉTODO CONSTRUTOR OU CONSTRUTOR

- É um método especial, cuja finalidade é permitir a instanciação de um objeto a partir de uma classe.
- Possui assinatura diferente dos demais métodos.
- Possui o mesmo nome da classe.

Exemplo:

Pessoa(){…};

ORIENTAÇÃO A OBJETOS

- Abstração
- Classes
- Objetos
- Atributos
- Métodos
- Método Construtor
- Herança
- Polimorfismo
- Sobrecarga
- Encapsulamento

HERANÇA - CONCEITOS

- Herança significa ser capaz de incorporar os atributos e métodos de uma outra classe previamente definida.

- Quando necessário, pode-se especializar métodos da classe ancestral e especificar novas operações e dados, para refinar, especializar, substituir ou estender a funcionalidade da classe progenitora.

HERANÇA - CONCEITOS

- Você pode fazer sempre com que um objeto mais geral armazene um objeto mais especializado, mas o contrário não é verdadeiro sem uma conversão explícita de tipos.

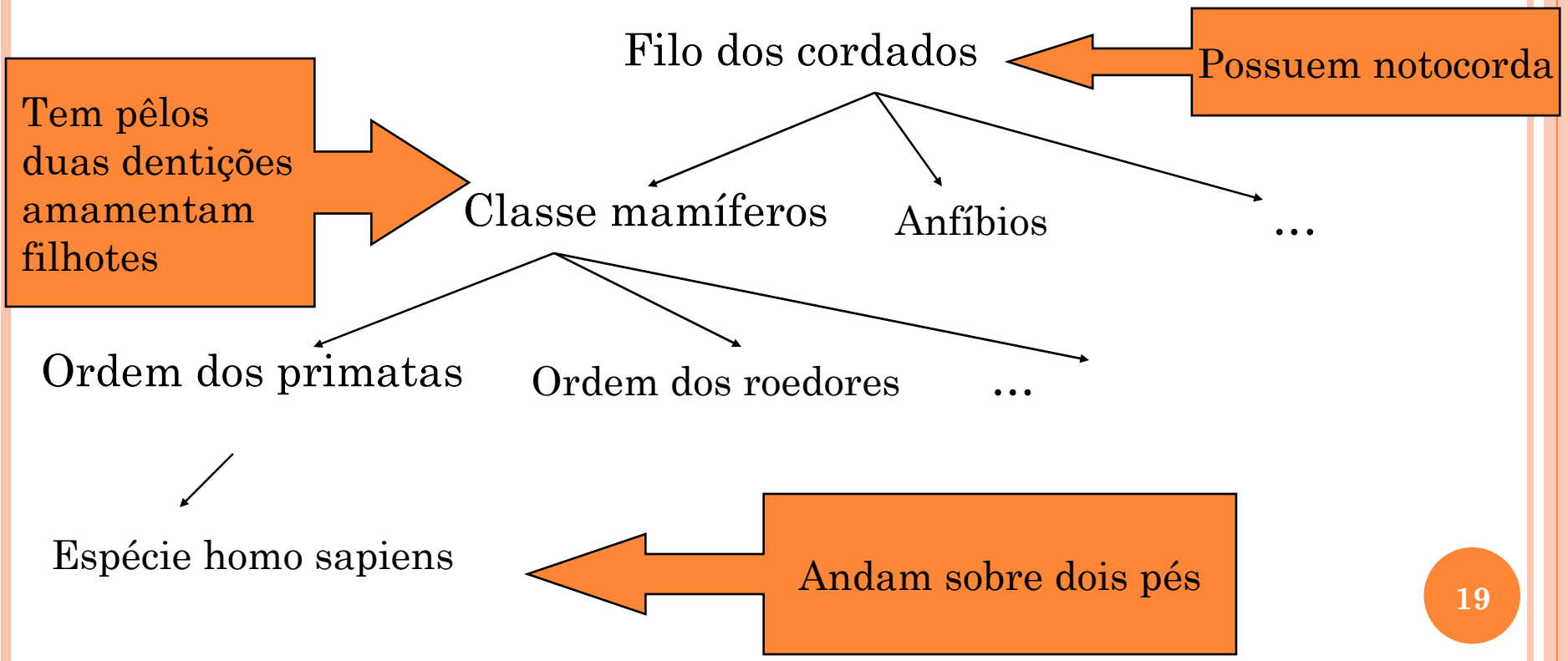
Exemplo:

Todos os cítricos são frutas, mas nem todas as frutas são cítricos!

HERANÇA

Visão do mundo real:

✓Estamos acostumados a lidar com classes agrupando-as em estruturas hierárquicas;



HERANÇA -TERMINOLOGIA

- **Estender**

  Criar uma nova classe que herda todo o conteúdo da classe existente
- **Superclasse**

  Uma classe progenitora ou “base”.
- **Subclasse**

  Uma classe filha que herda, ou estende, uma superclasse

ORIENTAÇÃO A OBJETOS

- Abstração
- Classes
- Objetos
- Atributos
- Métodos
- Método Construtor
- Herança
- Polimorfismo
- Sobrecarga
- Encapsulamento

POLIMORFISMO

- Ocorre quando uma classe possui um método com o mesmo nome e assinatura (número, tipo e ordem de parâmetros) de um método na sua superclasse, mas, diferencia-se pela lógica de implementação.

Exemplo:

EfetuarSaque em conta comum e
EfetuarSaque em conta especial.

ORIENTAÇÃO A OBJETOS

- Abstração
- Classes
- Objetos
- Atributos
- Métodos
- Método Construtor
- Herança
- Polimorfismo
- Sobrecarga
- Encapsulamento

SOBRECARGA

✓É semelhante ao polimorfismo, diferenciando-se pelo fato de possuir assinatura diferente, ao contrário do que ocorre no polimorfismo.

Exemplo:

calcularSalario(double salBruto, double impostos);

calcularSalario(double salBruto, double impostos, double gratificacao);

ORIENTAÇÃO A OBJETOS

- Abstração
- Classes
- Objetos
- Atributos
- Métodos
- Método Construtor
- Herança
- Polimorfismo
- Sobrecarga
- Encapsulamento

ENCAPSULAMENTO - CONCEITOS

- Mecanismo utilizado visando obter segurança, modularidade e autonomia para objetos;

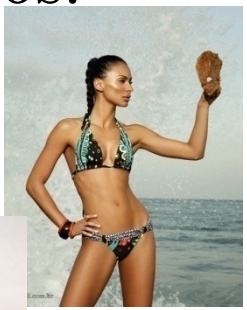
- Conseguido através da definição de visibilidade privada dos atributos, ganhando-se assim autonomia para definir o que o mundo externo ao objeto poderá visualizar e acessar, normalmente através de métodos públicos.

ENCAPSULAMENTO - CONCEITOS

- Dica: sempre defina os atributos de uma classe como privados, a não ser que tenha uma boa justificativa para não serem.

ENCAPSULAMENTO

- ✓Sempre escondemos do mundo externo alguns atributos pessoais;
- ✓Definimos, através de ações se iremos ou não externar estes atributos.



A namoradinha

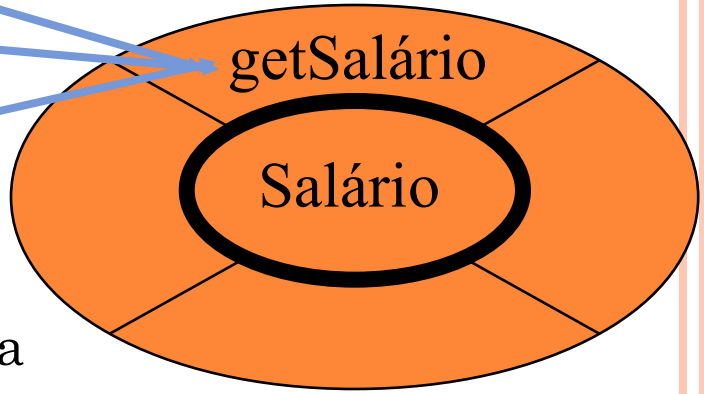
Salário * 2

O Cunhado

salário/10

Salário*0.3

O Leão do Imposto de Renda



ENCAPSULAMENTO - EXEMPLO

```
public class Pessoa
{
   private double salario;
   public double getSalario(Object quemPergunta)
   {
      if (quemPergunta instanceof Namorada)
      {
         return (salario * 2);
      }
      if (quemPergunta instanceof Cunhado)
      {
         return (salario / 10);
      }
      if (quemPergunta instanceof LeaoImpostoDeRenda)
      {
         return (salario * 0.3);
      }
      return (0.0); // esconde o salario do resto !!!
   }
}
```

ENCAPSULAMENTO - MODIFICADORES DE VISIBILIDADE

✓**Public =>** Os atributos e métodos são sempre acessíveis em todos os métodos de todas as classes. Este é o nível menos rígido de encapsulamento, equivale a não encapsular.

✓**Protected =>** Os atributos e métodos são acessíveis no pacote, nos métodos da própria classe e suas subclasses.

✓**Default =>** Os atributos e métodos são acessíveis nas classes que pertencem ao mesmo pacote.

✓**Private =>** Os atributos e métodos são acessíveis somente nos métodos (todos) da própria classe. Este é o nível mais rígido de encapsulamento.

ATRIBUTOS E MÉTODOS CONTROLE DE ACESSO

Visibilidade dos membros de uma classe

Especificador	Classe	Subclasse	Package	Mundo
privado	✓			
<branco>	✓		✓	
protegido	✓	✓	✓	
publico	✓	✓	✓	✓

OBRIGADA!

ENTÃO...
QUE TAL O UNIVERSO ORIENTADO A OBJETOS???