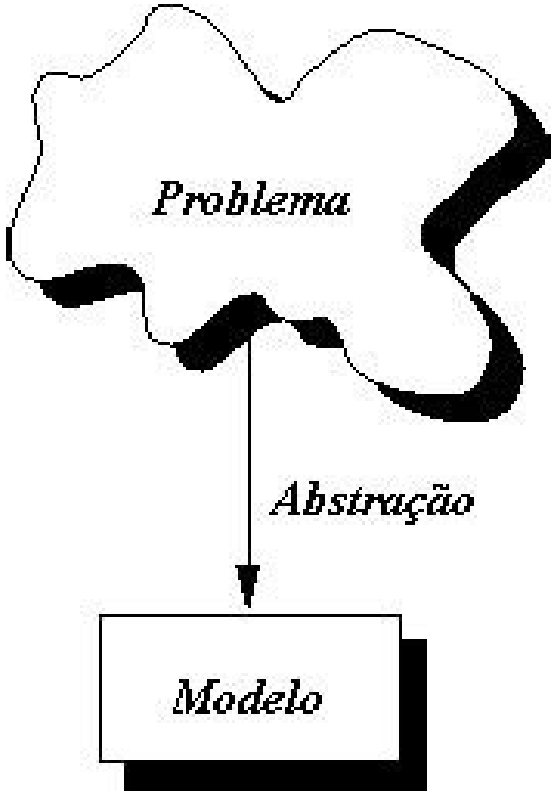
Orientação a Objeto

ABSTRAÇÃO

Abstrair consiste no processo de retirar do domínio do problema os detalhes relevantes e representá-los não mais na linguagem do domínio, e sim na linguagem da solução.



ORIENTAÇÃO A OBJETOS

- Todas as linguagens de programação fornecem abstrações.
- A programação orientada a objetos tenta trazer o espaço da solução para o espaço do problema: ambos são representados como objetos!!
- O programa permite se adaptar ao problema, adicionando novos tipos ao espaço da solução que mapeiam os tipos existentes no espaço do problema, ou seja, descreve-se o problema em termos do problema e não em termos da solução!!

ORIENTAÇÃO A OBJETOS

O que são objetos?

Um objeto é uma variável ... Ele armazena dados. Cabe, então, uma pergunta:

Uma struct é um objeto?

E a resposta é:

Uma struct é um objeto, mas um objeto pode ser mais que uma struct: você pode pedir que determinadas operações sejam feitas sobre os objetos.

ORIENTAÇÃO A OBJETOS

Um objeto possui, então, atributos (dados) e comportamentos (métodos, procedimentos, funções, que atuam sobre ele).

Exemplos de objetos: cachorros, carros,

funcionários, ...

Um programa é um conjunto de objetos dizendo uns para os outros o que fazer através do envio de mensagens.

ORIENTAÇÃO A OBJETOS

Concretamente, pode-se pensar nas mensagens como sendo chamadas a funções que pertencem a um objeto em particular.

Cada objeto tem a sua própria região de memória, que pode ser composta por outros objetos, também. Exemplo: o objeto carro

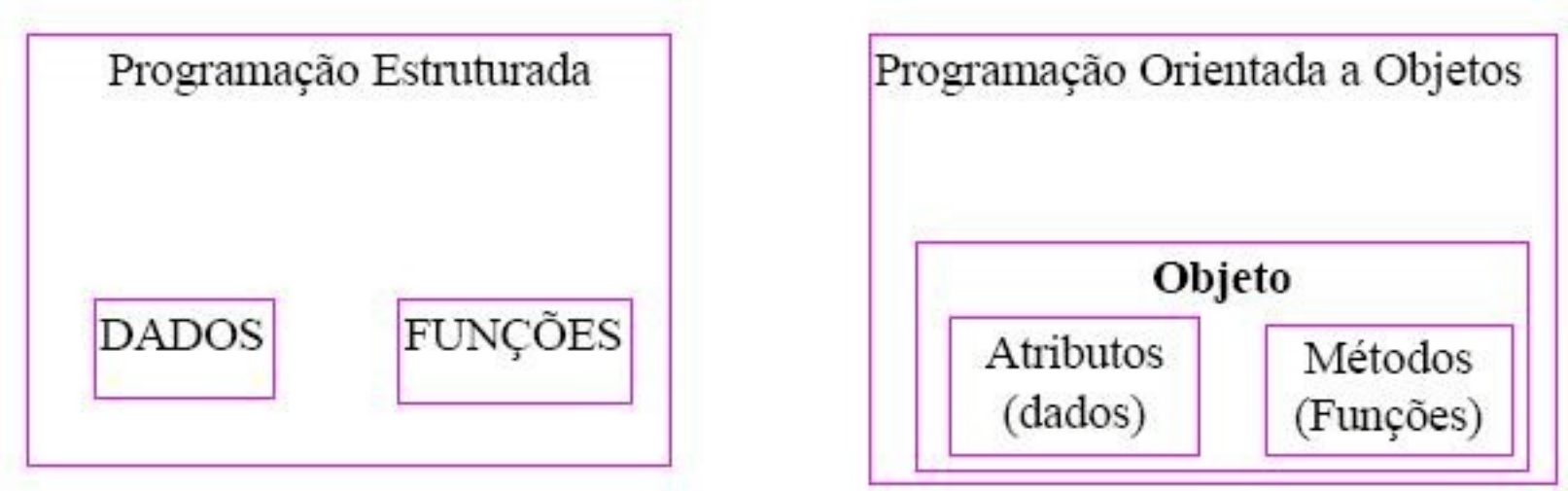
pode ser composto pelos objetos lataria, rodas, motor, etc.

ORIENTAÇÃO A OBJETOS

Programação estruturada x Orientado a Objetos:

- **Estruturado**: Ênfase nos procedimentos, implementados em blocos estruturados, com comunicação entre procedimentos por passagem de dados;
- **OO**: Dados e procedimentos fazem parte de um só elemento básico (objeto). Os elementos básicos comunicam-se entre si, caracterizando a execução do programa à Dados e procedimentos ENCAPSULADOS em um só elemento.

ORIENTAÇÃO A OBJETOS



ORIENTAÇÃO A OBJETOS

Em um objeto estão *encapsulados* os **dados** (atributos) e os **procedimentos** (serviços ou métodos) exclusivos dele. Os procedimentos são aplicáveis aos dados residentes no objeto.

Exemplo 1: Objeto funcionário “Zé das couves”.

Atributos: nome, endereço, data de admissão, identidade, CPF ...

Procedimentos: Consulta endereço, aumenta salário, paga comissão, ...

ORIENTAÇÃO A OBJETOS

Exemplo 2: Objeto Matriz A

Atributos: Linhas, colunas, elementos da matriz.

Procedimentos: inicialização, leitura, multiplicação por um escalar, inversão, multiplicação por um vetor, etc ...

ORIENTAÇÃO A OBJETOS

Um objeto possui estado, comportamento e identidade:

- **Estado**: valores de atributos;
- **Comportamento**: Definido pelos métodos (como o objeto age e reage);
- **Identidade**: Aquilo que diferencia um objeto de outro.

ORIENTAÇÃO A OBJETOS

Classes:

Descrevem um conjunto de objetos semelhantes.

Exemplo:

Zé das Couves, pertence à classe dos funcionários. Portanto todos funcionários possuem nome, CPF, ..., todos têm comportamento definido pelos métodos que compartilham, etc.

ORIENTAÇÃO A OBJETOS

Qual a diferença entre classes e objetos?

Objeto é uma entidade concreta e a classe é uma *abstração.*

Exemplo de classes: Números inteiros, matrizes, números complexos, automóveis, árvores, casas, cidades, países, etc ....

Os objetos são também denominados *instâncias* de uma classe;

Exemplo: Um Uno-Mille é uma instância da classe automóvel.

ORIENTAÇÃO A OBJETOS

Uma classe possui dados que definem suas propriedades (atributos) e os procedimentos (serviços) que devem ser executados sobre estes dados. A eles dá-se o nome de *membros* de uma classe.

Os atributos são os dados ou informações do objeto ou da classe. Existem atributos de objetos e atributos de classe.

ORIENTAÇÃO A OBJETOS

- Atributos de classe: São menos freqüentes que os atributos de objetos. Exemplo:

  Funcionário: número de funcionários da loja.

- Exemplos de atributos de objetos:

  Funcionário: nome, endereço, telefone, ...;

  Carro: nome, marca, ano, cor, ...;

  Livro: autor,editora, ano, ...

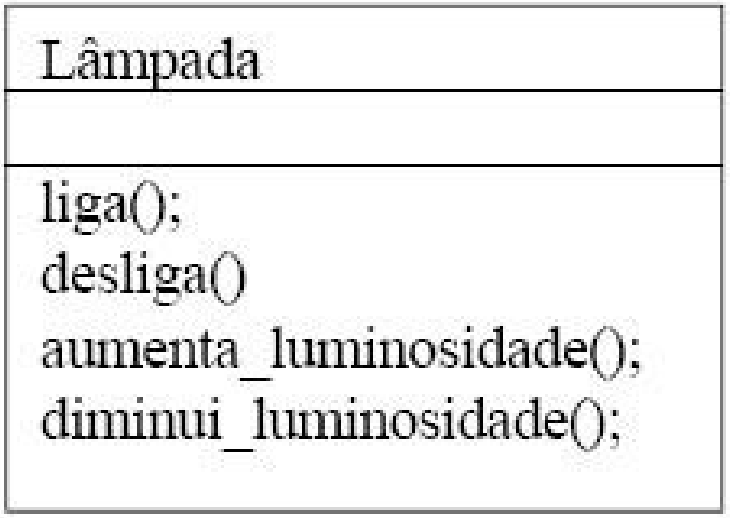
ORIENTAÇÃO A OBJETOS

Serviços: Implantados como funções colocados no nível do objeto ao qual se relacionam.

Mensagens: São os pedidos enviados a um objeto, afim de que ele desempenhe algum serviço. A comunicação entre objetos se faz através destas mensagens.

ORIENTAÇÃO A OBJETOS

Protocolo: é o conjunto de mensagens que um objeto suporta. Define a maneira como um objeto interage com o mundo. Define a **interface** do objeto. Exemplo: Lâmpada.



ORIENTAÇÃO A OBJETOS

Encapsulamento

A idéia por trás do Encapsulamento é a de que a utilização dos objetos não deve depender de sua implementação interna, e sim de sua interface.

A abstração define a *Interface* dos objetos, ou seja, a forma como eles se relacionarão com as demais entidades do sistema.

ORIENTAÇÃO A OBJETOS

- A *Implementação* definirá como a abstração será representada (o código). Este não deve ser visível às entidades que usarão a classe de objetos, por ser exclusivamente do interesse da própria classe.
- A interface não apresenta necessariamente todos os métodos de um objeto, mas somente aqueles que podem ser acessados pelo público em geral, os chamados métodos públicos. Existem métodos internos aos objetos: os métodos privados.

ORIENTAÇÃO A OBJETOS

Como exemplo, temos um Vídeocassete:

ORIENTAÇÃO A OBJETOS

Reutilizando por Composição

Uma vez que uma classe foi criada e testada, ela pode ser utilizada por outras classes para auxiliar sua implementação. Esta é uma das principais vantagens da Programação Orientada a Objetos.

ORIENTAÇÃO A OBJETOS

A forma mais simples de reutilização é usar um objeto daquela classe diretamente. Exemplo: janela no Windows; Matriz em um programa de cálculo, etc ...

Porém, pode-se utilizar um objeto de uma classe dentro de uma nova classe: sua classe pode ser composta pelo número e tipo de objetos que se fizerem necessários, o que é chamado de Composição.

ORIENTAÇÃO A OBJETOS

Exemplos:

- Classes roda, motor, lataria, ... e a classe carro;

- Classes motor, controle do motor, cabeça de leitura, cabeça de gravação, ... e a classe videocassete.

ORIENTAÇÃO A OBJETOS

Identifica-se a possibilidade de composição através dos seguintes verbos típicos: **conter, possuir**. Ex: Um carro contém 5 rodas, motor, lataria, ...

Outra possibilidade de reutilização é a utilização de herança.

ORIENTAÇÃO A OBJETOS

Herança

Permite modelar uma hierarquia entre classes: classes mais especializadas (classes filhas) herdam propriedades da classe mais geral (classe pai);

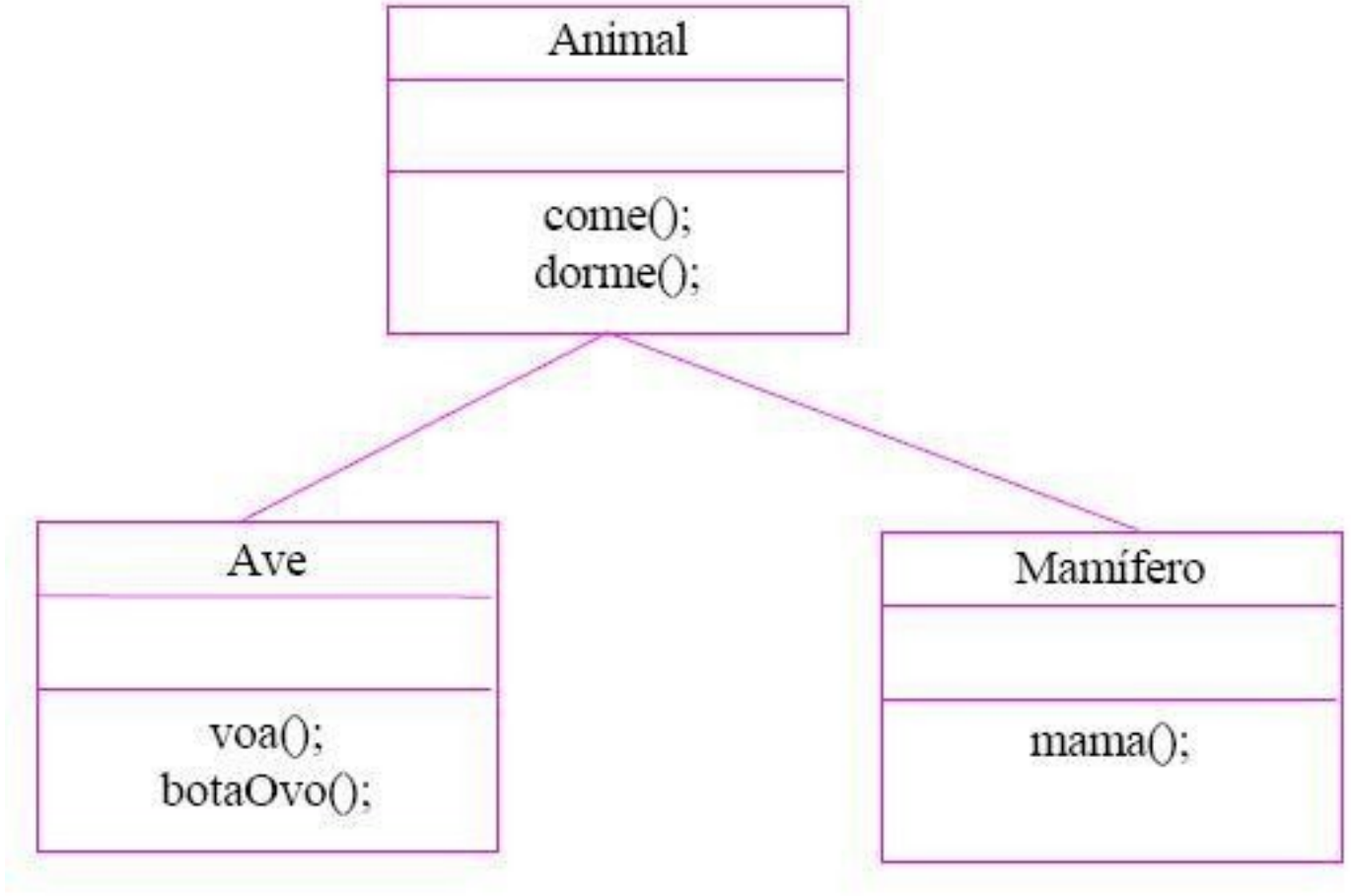
Pode-se compartilhar automaticamente métodos e dados entre diferentes classes, subclasses e objetos. Criar uma nova classe programando somente as diferenças desta para a classe pai.

ORIENTAÇÃO A OBJETOS

A classe filha (Especialização) herda a interface da classe pai (Generalização), podendo substituí-la quando se espera um objeto da classe pai.

Identifica-se a possibilidade de herança através da seguinte expressão típica: **é um tipo de.**

ORIENTAÇÃO A OBJETOS



ORIENTAÇÃO A OBJETOS

- Ave é um tipo de Animal; Mamífero é um tipo de Animal;

- Ave come, dorme, voa e botaOvo; Mamífero come, dorme e mama.

- Uma Ave (ou um Mamífero) podem substituir Animal pois é um tipo de Animal (tem a mesma interface!).

ORIENTAÇÃO A OBJETOS

Herança múltipla: quando a classe derivada possui características herdadas de duas ou mais classes base.

Exemplo: sistema com interface gráfica associada à manipulação de uma base de dados. Biblioteca gráfica + biblioteca de manipulação de bancos de dados, baseadas em classes: classes do nosso programa construídas herdando as características de interesse das classes de ambas bibliotecas.

ORIENTAÇÃO A OBJETOS

A herança é uma das responsáveis pela facilidade de reaproveitamento de código da POO. Precisamos fazer uma implementação semelhante a uma anterior: derivamos uma classe, programando as diferenças e reaproveitando o código útil da classe base.

ORIENTAÇÃO A OBJETOS

Polimorfismo

"O que possui várias formas". Propriedade de se usar o mesmo nome para métodos diferentes, implementados em diferentes níveis de uma hierarquia de classes. Para cada classe, tem-se um comportamento específico para o método.

Exemplo: Método desenhar() para hierarquia de objetos gráficos.

ORIENTAÇÃO A OBJETOS

O polimorfismo é um dos responsáveis pela facilidade de extensão de um programa orientado a objetos: para se efetivar uma extensão de um programa que utiliza polimorfismo, basta derivar novas subclasses de uma classe herdada, programando as modificações na nova classe.

Não se preocupem demais com o conceito, no momento ...

ORIENTAÇÃO A OBJETOS

Modularidade

Programas grandes e complexos: separação em conjuntos de módulos, cada um contendo classes com independência de funcionamento, podendo ser compilados independentemente.

Modularizar consiste em decidir onde empacotar fisicamente classes e objetos presentes na estrutura lógica do programa.

ORIENTAÇÃO A OBJETOS

Persistência

Objetos persistentes são aqueles que permanecem existindo mesmo após o término da execução do programa. Associados à persistência estão o gerenciamento dinâmico da memória e o armazenamento de objetos em bases de dados.

ORIENTAÇÃO A OBJETOS

Tipificação

Característica desejável em um modelo OO, é a capacidade de distinguir os diferentes tipos de classe. Uso de um objeto de uma classe onde previa-se o uso de objeto de outra, só permitido em condições controladas (conversões explícitas).