

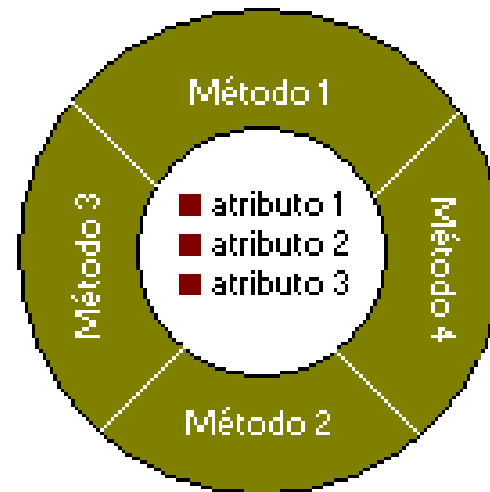
Orientação a objetos



- ❑ Objetos ou Instâncias
- ❑ Métodos ou Mensagens
- ❑ Encapsulamento
- ❑ Classes
- ❑ Variáveis da Classe X Variáveis da Instância
- ❑ Métodos da Classe X Métodos da Instância
- ❑ Relacionamentos
- ❑ Identificando Objetos
- ❑ Classes Abstratas
- ❑ Polimorfismo

Objetos ou Instâncias I

- ❑ Objetos do mundo real: os objetos possuem estados e possuem comportamento
- ❑ Objeto de software: mantém seus estados em variáveis e implementam seus comportamentos com métodos
- ❑ Tudo que um objeto de software sabe (estado) e pode fazer (comportamento) é expresso por variáveis e métodos do objeto



Objetos ou Instâncias II

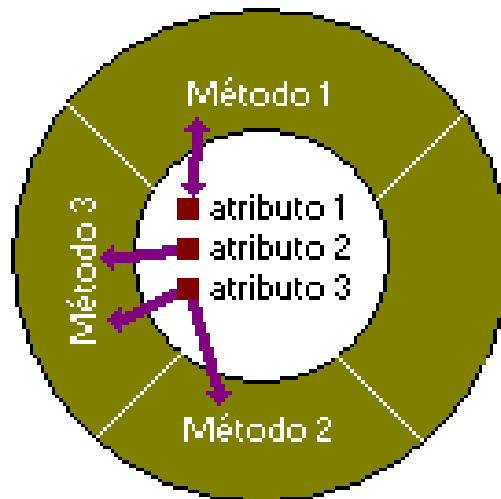
□ Exemplo: bicicleta



- O objeto tem total controle sobre o acesso a seus métodos e suas variáveis
- O objeto pode disponibilizar algumas variáveis e métodos e esconder outras variáveis e métodos

Métodos ou Mensagens I

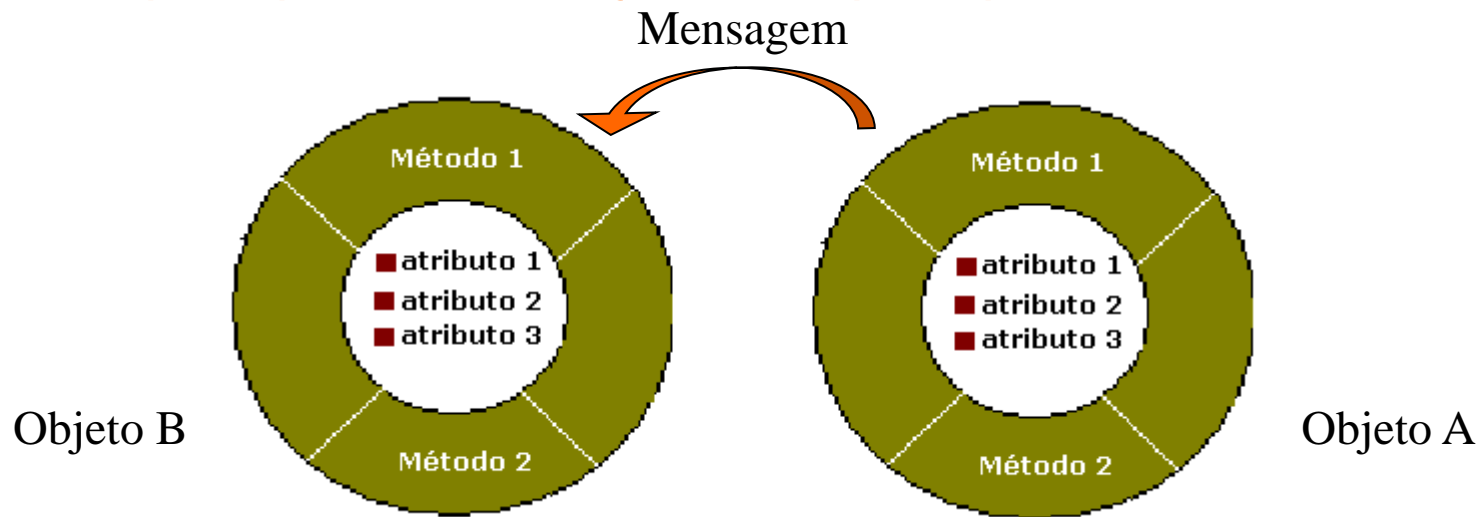
- ❑ Função(retorna um valor) ou procedimento(não retorna) para alterar ou verificar o estado (atributos) de um objeto
- ❑ Execução de um método é a reação à recepção de uma mensagem
- ❑ Deve operar somente com os atributos do próprio objeto e atributos recebidos como parâmetro



Métodos ou Mensagens II

- Objetos de software interagem e se comunicam uns com os outros através do envio de mensagem
- Quando um objeto A quer que o objeto B faça uma ação, o objeto A envia uma mensagem ao objeto B

"A pede para B fazer Ação" = "A pede para B executar Método X"

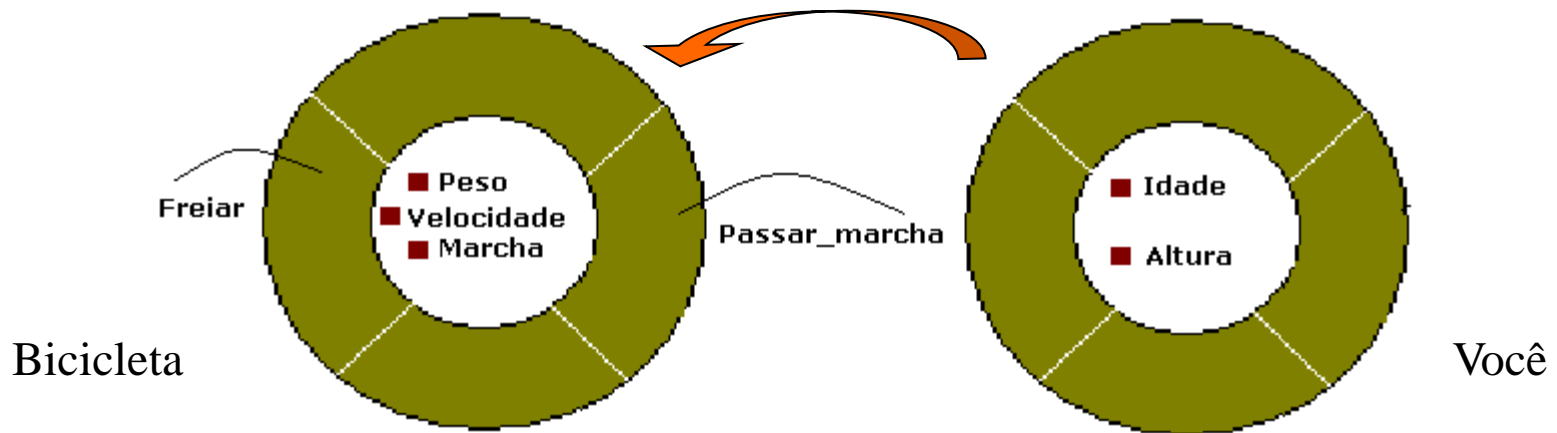


Métodos ou Mensagens III

- Três componentes compreendem uma mensagem:
 - o objeto para o qual a mensagem é enviada
 - o nome do método a ser executado
 - parâmetros necessários para execução do método

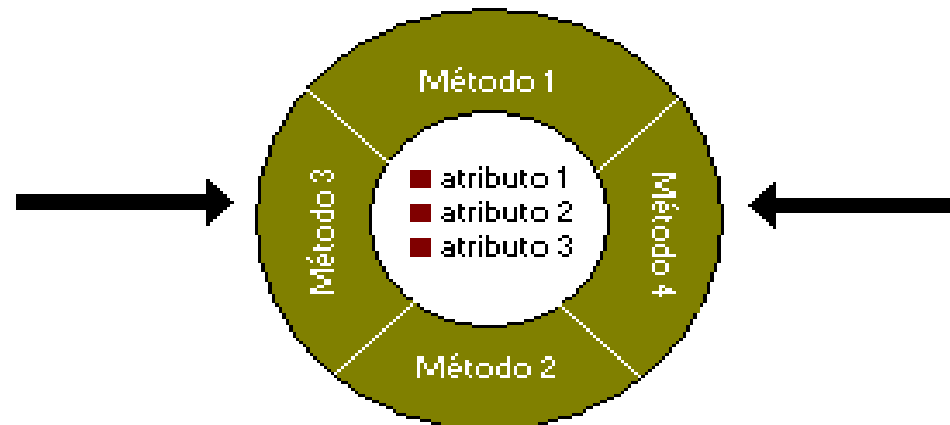
"A pede para B fazer Ação" =

"Você pede para a Bicicleta Passar_Marcha(marchaAnterior)"
Passar_marcha (marchaAnterior)



Encapsulamento

- ❑ Usado para esconder detalhes de implementação
- ❑ Não é necessário saber como a classe está implementada para chamar o método necessário.
- ❑ Basta saber a interface do método
- ❑ Os atributos de um objeto só devem ser manipulados pelos métodos do próprio objeto (orientação a objeto pura).



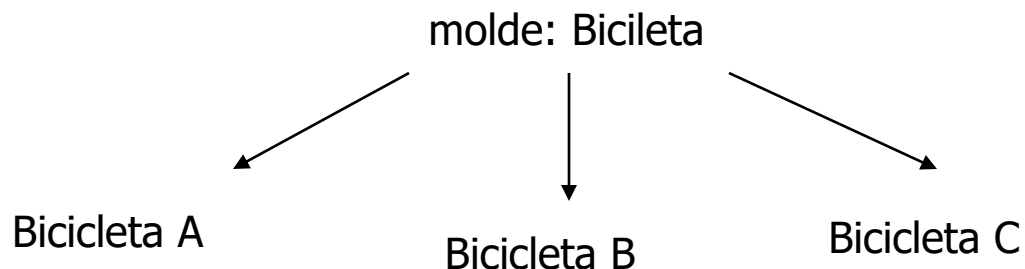
Restrição de acesso (métodos e atributos)



- *Public*: sem restrição. Qualquer objeto pode acessar tal método/atributo
- *Private*: apenas o objeto que possui o método/atributo pode acessá-lo
- *Protect*: apenas os objetos das classes do mesmo pacote podem acessar o método/atributo
- *Default*:
 - atributo -> C++: *private* / JAVA: *private*
 - método -> C++: *private* / JAVA: *public*

Classe

- No mundo real sua bicicleta é apenas uma das existentes no mundo
- As bicicletas possuem estado e comportamento em comum
- “Molde” para construção da bicicleta: classe
- Terminologia OO: sua bicicleta é uma instância da classe *Bicicleta*.



Classe X Instância I



- ❑ Para criar o objeto bicicleta no mundo OO é necessário:
 - ❑ criar uma classe de bicicletas (molde)
 - ❑ instanciar a classe gerando então o objeto bicicleta (instância)
- ❑ Com um molde você gera vários objetos parecidos. Todos os objetos possuem os mesmos atributos e os mesmos métodos (implementação dos métodos é a mesma para todos os objetos)
- ❑ Os valores dos atributos podem ser modificados por cada objeto
- ❑ Quando uma instância da classe é gerada, um objeto é criado e o sistema *aloca* memória para suas variáveis

Exemplo

Molde: Classe Empregado

atributos:

nome

endereço

salário = R\$ 300,00

métodos:

atualizarSalário(salárioNovo)

fornecerInformações()

Objeto Empregado A

atributos:

nome = João

endereço = R. Maria 12

salário= R\$ 300,00

métodos:

atualizarSalário(salárioNovo)

fornecerInformações()

Objeto Empregado B

atributos:

nome = Ana

endereço = R. D. José 30

salário= R\$ 300,00

métodos:

atualizarSalário(salárioNovo)

fornecerInformações()

Variáveis da Instância X da Classe



Variáveis da classe

- ❑ Só existe uma cópia de cada variável
- ❑ Todas as instâncias acessam a mesma cópia das variáveis
- ❑ São criadas apenas uma vez assim que o sistema reconhece a classe

Variáveis da Instância

- ❑ Existe uma cópia para cada instância criada
- ❑ Cada instância acessa a sua cópia
- ❑ São criadas sempre quando uma nova instância é gerada

Métodos de Instância X de Classe



Métodos de Classe

- Podem acessar apenas variáveis da classe
- Podem ser acessados pela classe ou pela instância

Métodos de Instância

- Podem acessar variáveis da classe e da instância
- Só podem ser acessados por instâncias

Relacionamentos entre Classes



- Para que objetos se comuniquem eles precisam se relacionar
- Tipos de Relacionamentos:
 - Associação
 - Agregação
 - Composição
 - Dependência
 - Generalização / Herança

Relacionamentos (continuação)



- Associação:
 - descreve uma relação entre duas classes
 - Usuário possui bicicleta

- Agregação:
 - descreve o relacionamento entre um todo e sua parte
 - são indicadas por frases do tipo "tem um", "é parte de"
 - Uma teclado é parte de um *notebook*

Relacionamentos (continuação)

□ Generalização / Herança

- descreve o relacionamento entre classes definidas a partir de outras classes.
- toda a subclasse herda os estados e os comportamentos definidos na superclasse
- as subclasses não estão limitadas a estes estados e comportamentos.

□ Uma *mountain bike* é uma bicicleta

subclasse superclasse

Identificando Objetos

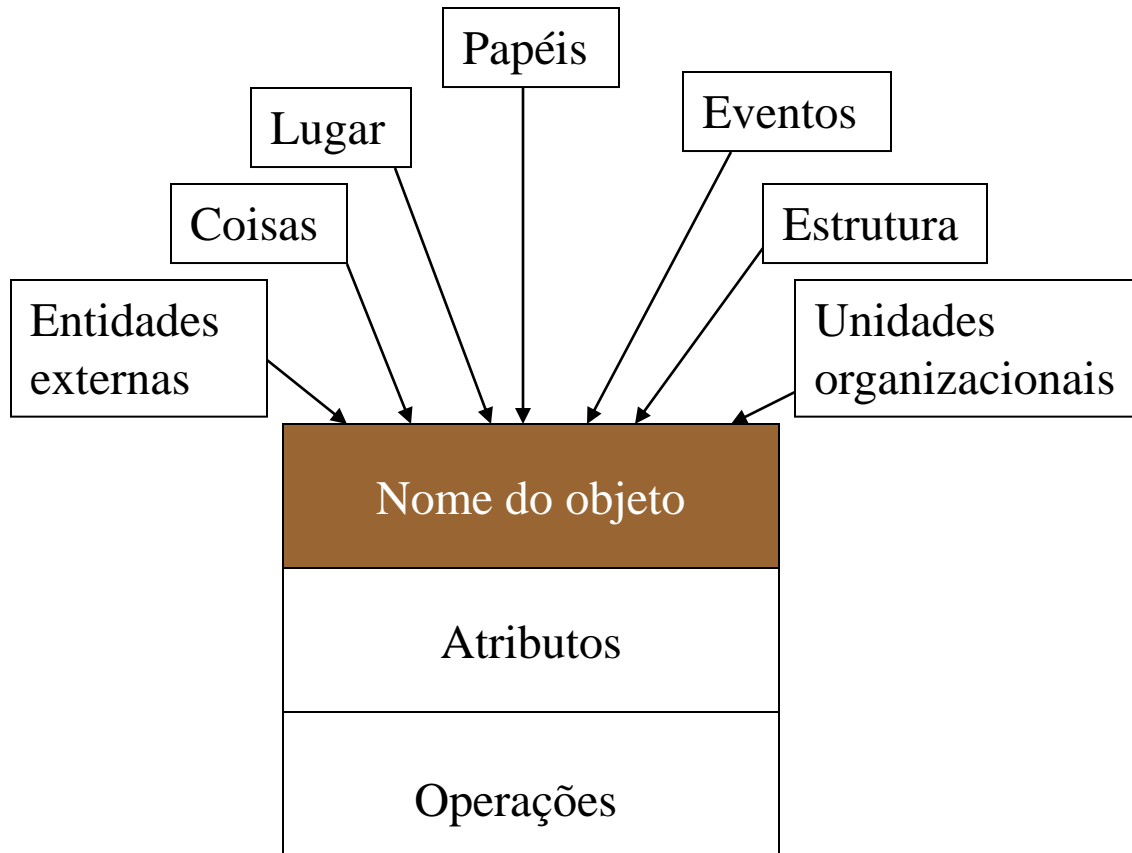


- Em uma sala existe um conjunto de objetos físicos que podem ser facilmente identificados, modelados e classificados como objetos OO.
- Mas em um problema onde o espaço é uma aplicação de software, os objetos podem não ser facilmente encontrados.
- Os objetos podem ser identificados analisando-se o problema ou fazendo um “*parser* gramatical” do texto contendo a descrição do problema
- Objetos são determinados sublinhando-se cada substantivo ou oração (parte de uma frase)

Identificando Objetos (continuação)

- Candidatos a objeto:
 - Entidade externa que produz ou consome informação para ser usada por um sistema computacional (ex.: outros sistemas, *devices*, pessoas)
 - Coisas que são parte do domínio da informação do problema (ex.: um sinal, uma carta, um display)
 - Ocorrências ou eventos que acontecem no contexto da operação do sistema (ex.: a propriedade de uma transferência, a finalização de uma série de movimentos de um robô)
 - Papeis(roles) desempenhados por pessoas que interagem com o sistema (ex.: gerente, engenheiro, vendedor)
 - Unidades de organização que são relevantes a uma aplicação (ex.: divisões, grupo, time)
 - Lugares que estabelecem o contexto do problema (ex.: galpão, estaleiro)
 - Estruturas que definem a classe de um objeto (ex.: sensor, computador, veículo 4-rodas)

Identificando Objetos (continuação)

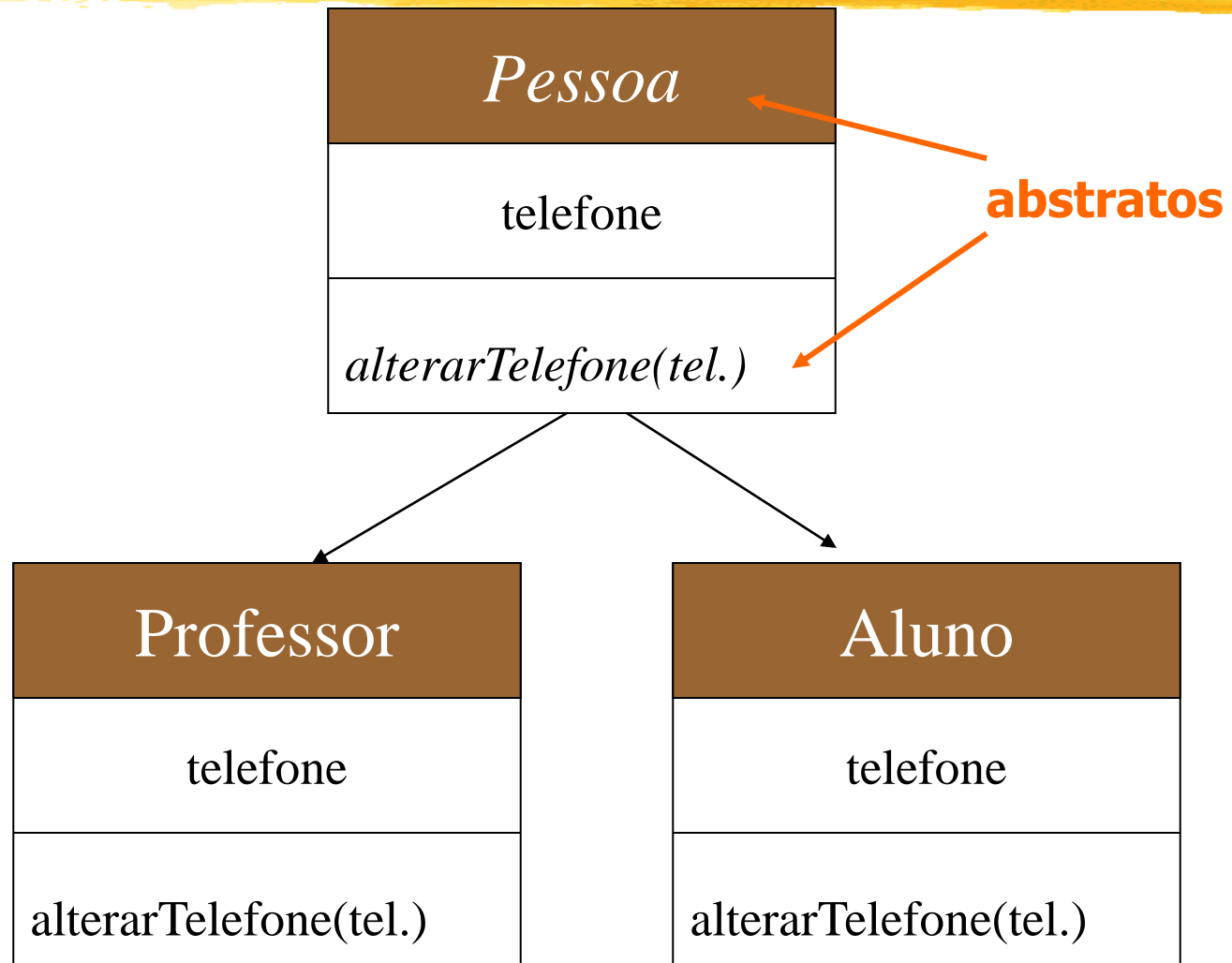


Classe abstrata



- ❑ Pelo menos um de seus métodos está *declarado* mas não têm *implementação* associada.
 - ❑ Método *abstrato*: método sem implementação
- ❑ Não gera instância
- ❑ Só pode ser usada como base para outras classes
 - ❑ herança
- ❑ Cada subclasse deverá implementar o método abstrato da superclasse (classe abstrata)

Exemplo



Polimorfismo



- ❑ *Polimorfismo* é a habilidade de diferentes instâncias, de classes diferentes, responderem a mesma mensagem de diferentes maneiras.
 - ❑ executar métodos com a mesma assinatura mas implementados de maneira diferente
- ❑ Classe polimórfica:
 - ❑ quando instâncias suas ou instâncias de classes derivadas suas possuem mensagens que *nem sempre* são respondidas da mesma maneira - a resposta irá depender do *contexto da execução*.

Exemplo

