ENGENHARIA DE SOFTWARE 2 – AULA 8

PROFª Mª DENILCE VELOSO

DENILCE.VELOSO@FATEC.SP.GOV.BR

DENILCE@GMAIL.COM

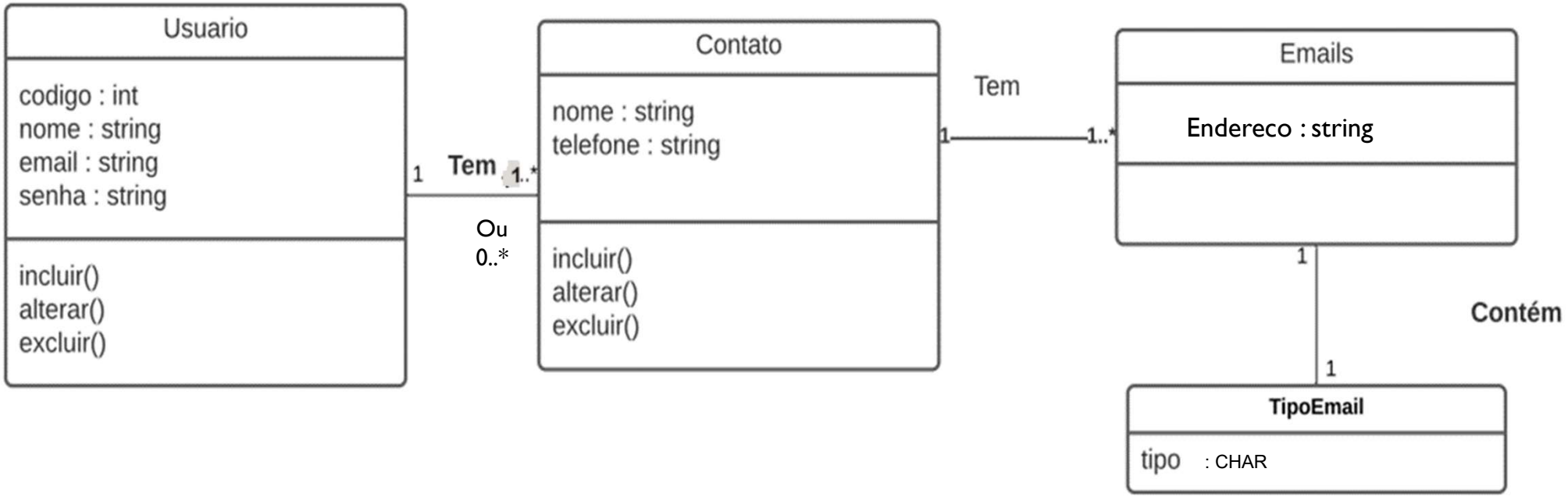
PROJETO ORIENTADO A OBJETOS COM UML CORREÇÃO ATIVIDADE DA AULA PASSADA

PROBLEMA – Cadastro de Contatos

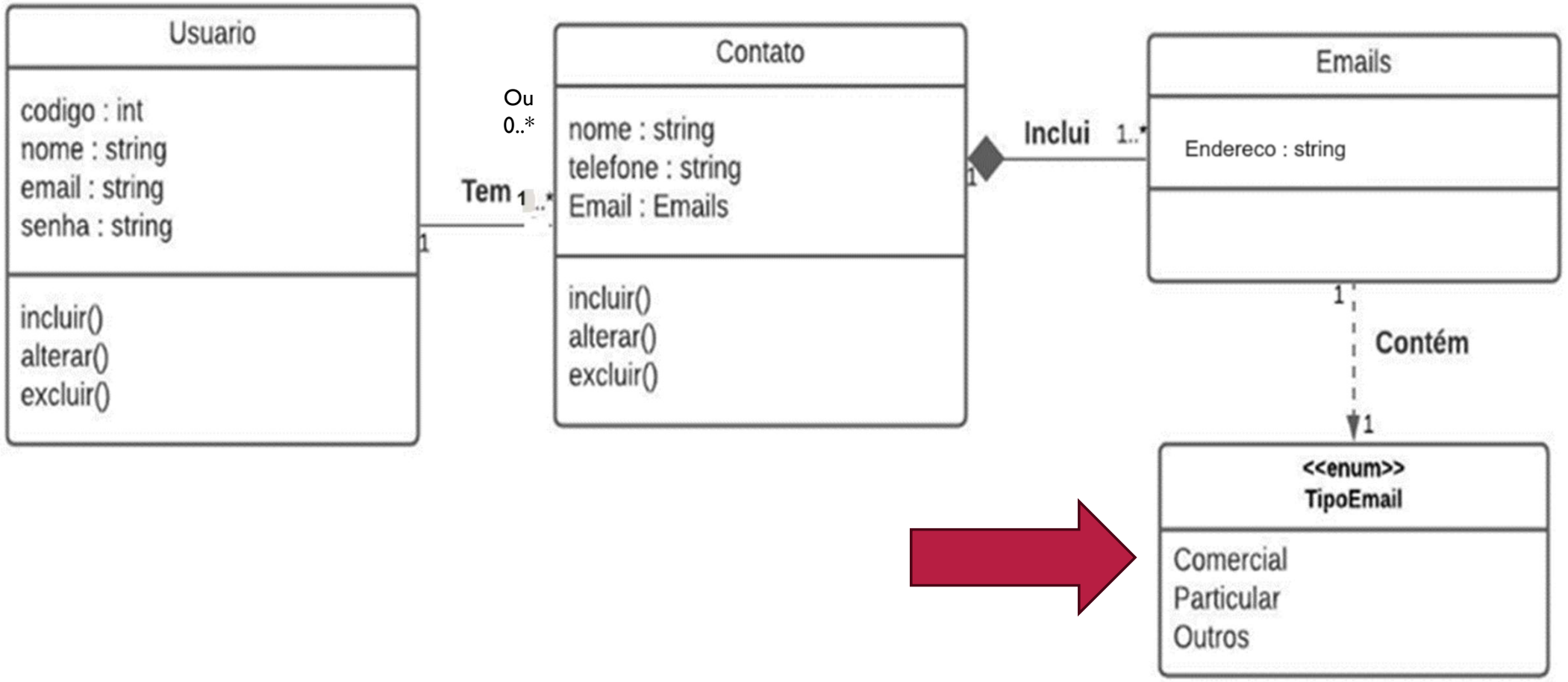
- Um usuário que faz o login (tem código, nome, e-mail e senha) pode possuir vários contatos e o sistema deverá manter os dados de cada usuário individualmente.
- Um contato pode possui vários endereços de e-mail e para cada e-mail está associado um tipo (comercial, particular ou outros). O contato deve possuir no mínimo um e-mail. Somente pode existir um e-mail de cada tipo.
- As informações associadas ao contato são: Nome, Telefone e E-mail.
- Um contato não pode estar ligado a mais de um usuário.

Adaptado de Melo (2006).

PROJETO ORIENTADO A OBJETOS COM UML – MODELOS DE PROJETO – SUGESTÕES ATIVIDADE 6 (AULA PASSADA)

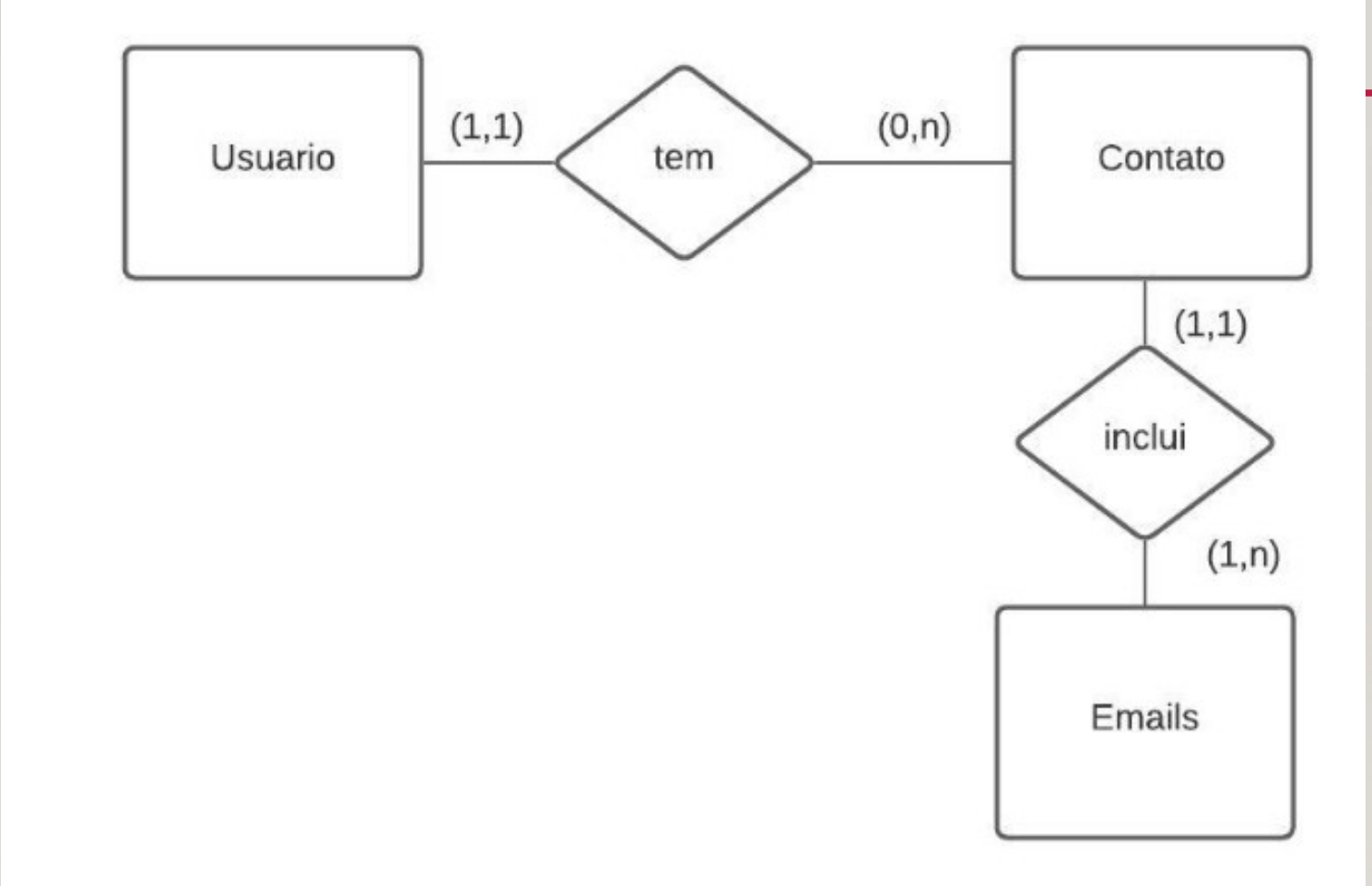


PROJETO ORIENTADO A OBJETOS COM UML – MODELOS DE PROJETO – SUGESTÕES ATIVIDADE 6 (AULA PASSADA)



** Um enum (enumeração) representa um tipo de dado personalizado que define um conjunto fixo de valores possíveis. É como um menu predefinido que limita a escolha a opções específicas. Exemplos: Pago, Pendente, Enviado, Cancelado

ATIVIDADE 6 – MODELO DE DADOS CONCEITUAL



PROJETO ORIENTADO A OBJETOS COM UML – DIAGRAMA DE CLASSES (REVISÃO – FORMAS DE REPRESENTAÇÃO)

Um **diagrama de classes** pode oferecer três perspectivas, cada uma para um tipo de observador diferente. São elas:

- **Conceitual** :
  - Representa os conceitos do **domínio em estudo**.
  - Perspectiva **destinada ao cliente**.
- **Especificação**
  - Tem foco nas principais interfaces da arquitetura, nos **principais métodos, e não como eles irão ser implementados.**
  - Perspectiva **destinada as pessoas que não precisam saber detalhes de desenvolvimento**, tais como gerentes de projeto.
- **Implementação** - *a mais utilizada* **
  - Aborda vários detalhes de implementação, tais como navegabilidade, *tipo* dos atributos, etc.
  - Perspectiva destinada ao time de desenvolvimento.

PROJETO ORIENTADO A OBJETOS COM UML – PROJETO ARQUITETURAL – IDENTIFICAÇÃO DE CLASSES DE OBJETOS – DIAGRAMA DE CLASSES

Relacionamentos – ligam as classes/objetos entre si.

Podem ser:

- Associação – normal, recursiva, qualificada, exclusiva, ordenada, classe, ternária e agregação.
- Generalização.
- Dependências e Refinamentos.

PROJETO ORIENTADO A OBJETOS COM UML – PROJETO ARQUITETURAL – IDENTIFICAÇÃO DE CLASSES DE OBJETOS – DIAGRAMA DE CLASSES

Associação de Agregação

- Indica que uma das classes do relacionamento é "uma parte", ou está contida em outra classe, o "todo".
- As palavras chaves para identificar uma agregação são: "contém", "consiste em","é parte de".
- Existem tipos especiais de agregação : compartilhadas e compostas (de composição).
- É representada por um losango em branco (compartilhada) ou preenchido (de composição).
- Atenção: a utilização de associação normal, agregação e etc DEPENDE EXCLUSIVAMENTE DAS REGRAS DE NEGÓCIO.

Slide 8

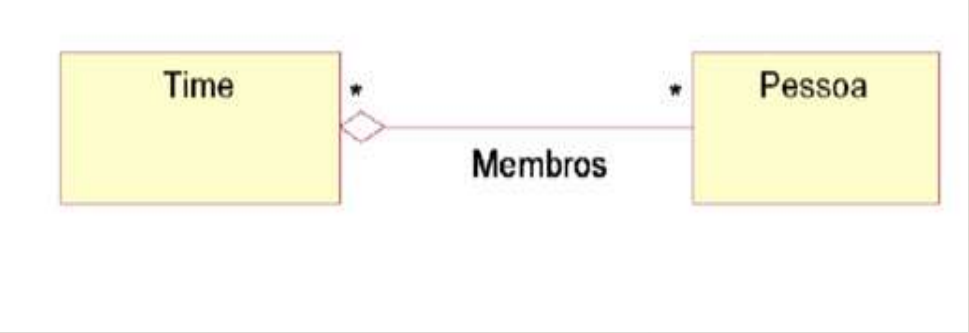
DV [2]1 A agregação é usada para modelar um relacionamento de composição entre elementos do modelo. Há muitos exemplos de relacionamentos de composição: uma Biblioteca contém Manuais, dentro de uma empresa os Departamentos são compostos de Funcionários, um Computador é composto de vários Dispositivos. Para modelar isso, o agregado (Departamento) tem uma associação de agregação para as partes constituintes (Funcionário).
DENILCE VELOSO; 17/04/2022

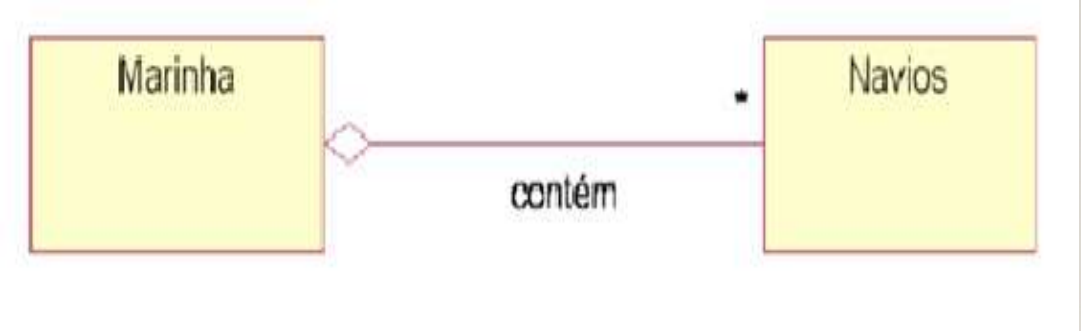
PROJETO ORIENTADO A OBJETOS COM UML – PROJETO ARQUITETURAL – IDENTIFICAÇÃO DE CLASSES DE OBJETOS – DIAGRAMA DE CLASSES

Associação de Agregação Compartilhada

- Quando uma das classes é uma parte, ou está contida na outra, mas esta parte pode estar contida na outra várias vezes em um mesmo momento.

Exemplos: Uma pessoa pode ser membro de um time ou vários times e em determinado momento. A marinha pode ter um ou vários navios.

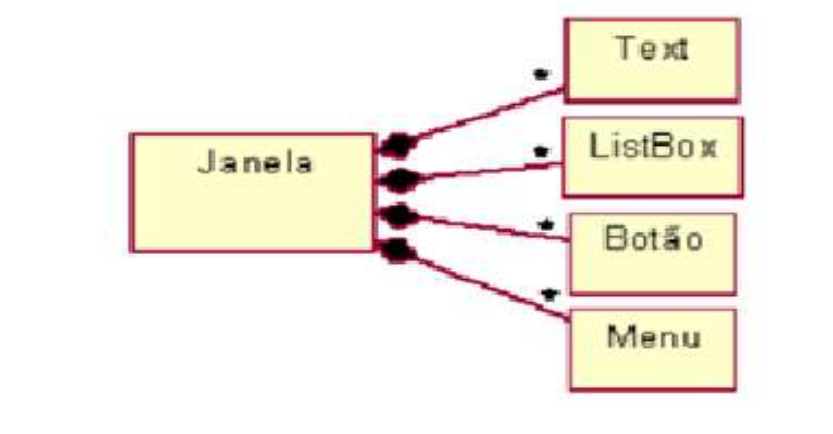




PROJETO ORIENTADO A OBJETOS COM UML – PROJETO ARQUITETURAL – IDENTIFICAÇÃO DE CLASSES DE OBJETOS – DIAGRAMA DE CLASSES

Associação de Agregação de Composição

- Agregação onde uma classe que está contida na outra "vive" e constitui a outra. Se o objeto da classe que contém for destruída, as classes da agregação de composição serão destruídas também. Exemplos: Os componentes só existem se existir a janela, os itens do pedido só existem se existir o pedido.



Fonte: http://www.facom.ufu.br/~bacala/DAW/Aula05-1%20-diagrama-de-classes.pdf

PROJETO ORIENTADO A OBJETOS COM UML – PROJETO ARQUITETURAL – IDENTIFICAÇÃO DE CLASSES DE OBJETOS – DIAGRAMA DE CLASSES

Associação de Agregação Compartilhada ou de Composição

Exemplo de implementação em C#:

Composição

```
class NotaFiscal: IDisposable {
    IList<ItemNotaFiscal> Itens {get;set;}
}

class ItemNotaFiscal: IDisposable { ... }
```

Agregação

```
class Time {
    IList<Pessoa> Integrantes {get;set;}
}

class Pessoa {}
```

Slide 11

DDAOV1 IDisposable é uma interface em C# que sinaliza que um objeto possui recursos não gerenciados que devem ser liberados explicitamente quando o objeto não for mais necessário

DENILCE DE ALMEIDA OLIVEIRA VELOSO; 20/04/2021

PROJETO ORIENTADO A OBJETOS COM UML – PROJETO ARQUITETURAL – IDENTIFICAÇÃO DE CLASSES DE OBJETOS – DIAGRAMA DE CLASSES

Generalização

- É um relacionamento entre um elemento geral e outro mais específico.
- O elemento mais específico possui todas as características do elemento geral e contém ainda mais particularidades.
- A generalização, também chamada de **herança**, permite a criação de elementos especializados em outros.
- Os tipos de generalização são: **normal, restrita**.

PROJETO ORIENTADO A OBJETOS COM UML – PROJETO ARQUITETURAL – IDENTIFICAÇÃO DE CLASSES DE OBJETOS – DIAGRAMA DE CLASSES

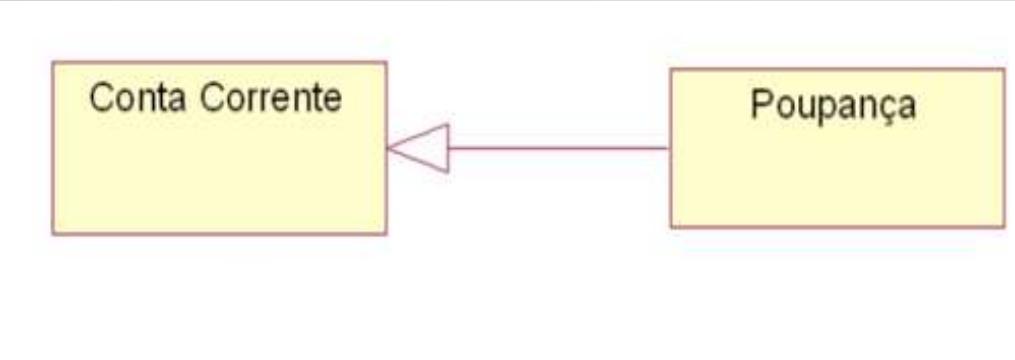
Generalização Normal

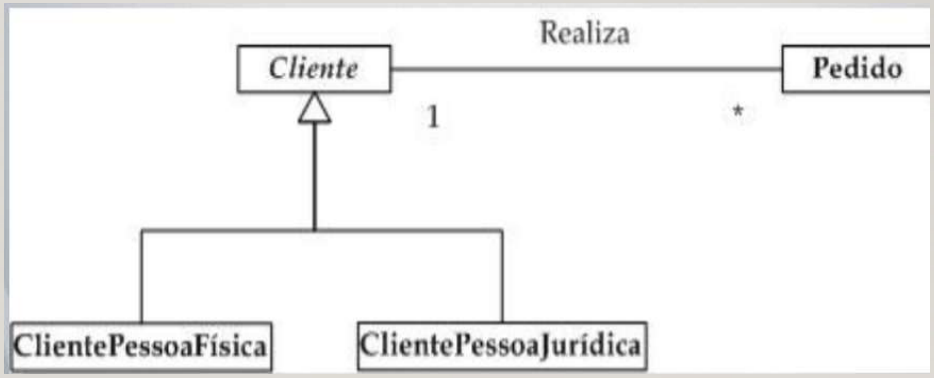
- Quando a classe mais específica (subclasse), herda tudo (atributos, operações, associações) da classe mais geral (superclasse).
- É representada por uma linha entre as duas classes e uma seta no lado onde se encontra a superclasse.

PROJETO ORIENTADO A OBJETOS COM UML – PROJETO ARQUITETURAL – IDENTIFICAÇÃO DE CLASSES DE OBJETOS – DIAGRAMA DE CLASSES

Generalização Normal

Exemplo: A Conta Corrente é uma generalização da Poupança e Cliente é generalização de ClientePessoaFisica e ClientePessoaJuridica.





Fonte: http://www.facom.ufu.br/~bacala/DAW/Aula05-1%20-diagrama-de-classes.pdf

PROJETO ORIENTADO A OBJETOS COM UML – PROJETO ARQUITETURAL – IDENTIFICAÇÃO DE CLASSES DE OBJETOS – DIAGRAMA DE CLASSES

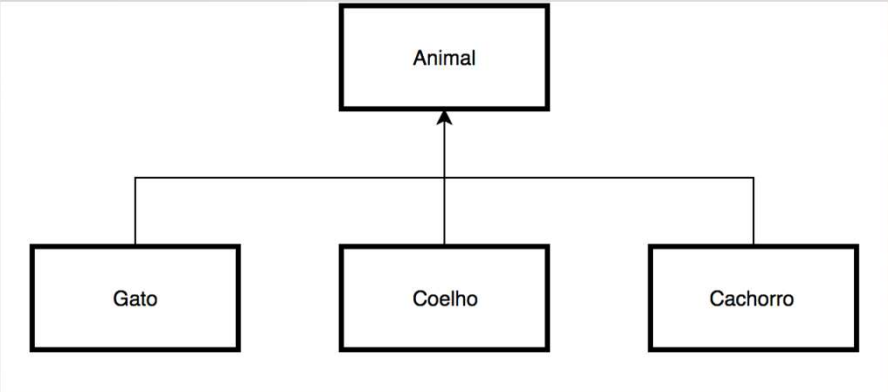
Generalização Restrita

- Uma restrição aplicada a uma generalização especifica informações mais precisas (restrições) sobre como a generalização deve ser usada e estendida no futuro.
- Generalizações restritas com mais de uma subclasse:
  - Sobreposição e Disjuntiva
  - Completa e Incompleta

PROJETO ORIENTADO A OBJETOS COM UML – PROJETO ARQUITETURAL – IDENTIFICAÇÃO DE CLASSES DE OBJETOS – DIAGRAMA DE CLASSES

Generalização Restrita - Disjuntiva

- **Herda atributos e comportamentos de uma classe pai em um momento, não posso reescrever os métodos. Exemplo:** classe Animal é a superclasse. Gato, Coelho e Cachorro compartilham todos os atributos e métodos de Animal, ou seja, um objeto só pode ser uma instância de Gato, Coelho ou Cachorro. (padrão)

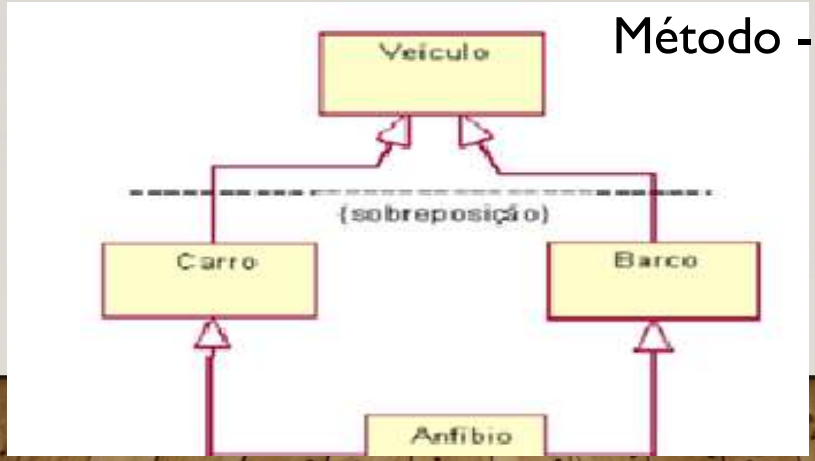


Método -> locomover()

PROJETO ORIENTADO A OBJETOS COM UML – PROJETO ARQUITETURAL – IDENTIFICAÇÃO DE CLASSES DE OBJETOS – DIAGRAMA DE CLASSES

Generalização Restrita – Sobreposição

- Sub-Classes herdam por sobreposição, e novas subclasses dessas podem herdar de mais de uma classe (herança múltipla).
- **Herda atributos e comportamentos da classe pai, pode "reescrever" métodos (mesma assinatura) em cada classe filha.** Exemplo: classe Veículo é a superclasse de Carro e Barco. Carro e Barco compartilham apenas alguns atributos e métodos de Veículo. Poderia permitir um objeto ser uma instância de Carro e Barco (anfíbio).



Método -> mover()

Slide 17

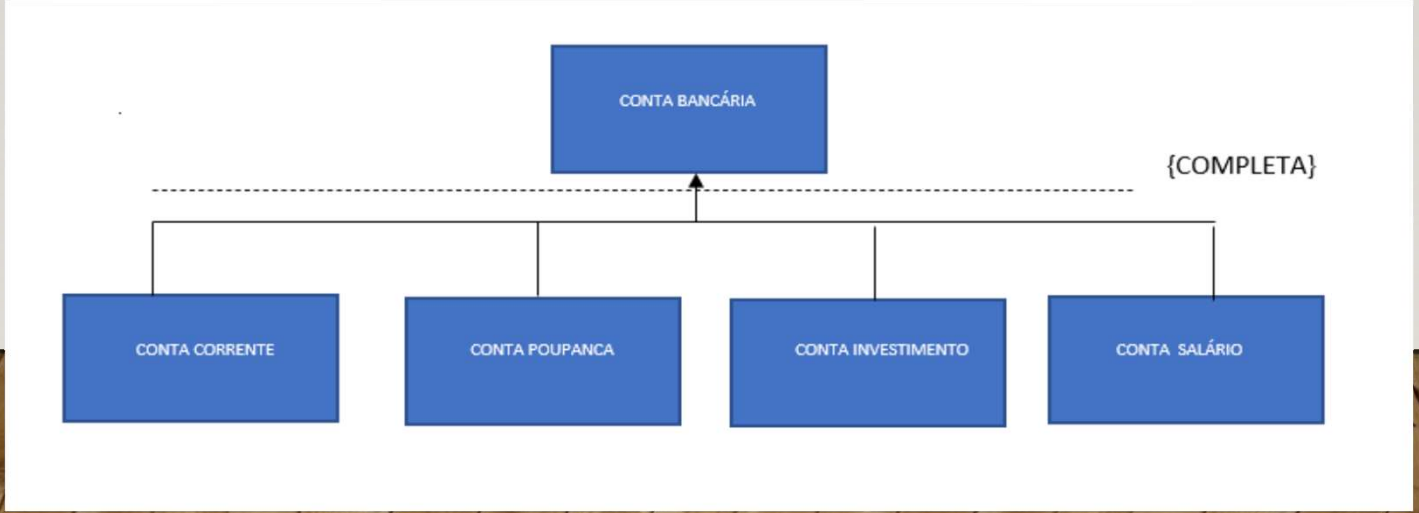
DDAOV2 C# e Java nao tem heranca multipla - para esse casos poderia utilizar Interface
DENILCE DE ALMEIDA OLIVEIRA VELOSO; 27/10/2022

DV [2]8 Linguagens que suportam herança múltipla:
C++: É uma das linguagens mais conhecidas que suporta herança múltipla. No entanto, ela também traz complexidades, como o problema do "diamante", que
exige cuidado na implementação.
Python: Também permite herança múltipla, oferecendo flexibilidade, mas exigindo atenção para evitar conflitos de nomes e métodos.
Lisp (CLOS - Common Lisp Object System): O CLOS é um sistema de objetos poderoso que suporta herança múltipla de forma robusta.
DENILCE VELOSO; 03/04/2025

PROJETO ORIENTADO A OBJETOS COM UML – PROJETO ARQUITETURAL – IDENTIFICAÇÃO DE CLASSES DE OBJETOS – DIAGRAMA DE CLASSES

Generalização Restrita – Completa e Incompleta

- Uma restrição simbolizando que uma generalização é completa, significa que **todas as subclasses já foram especificadas**, e não existe mais possibilidade de outra generalização a partir daquele ponto.
- A generalização incompleta é exatamente ao contrário da completa (é o padrão).



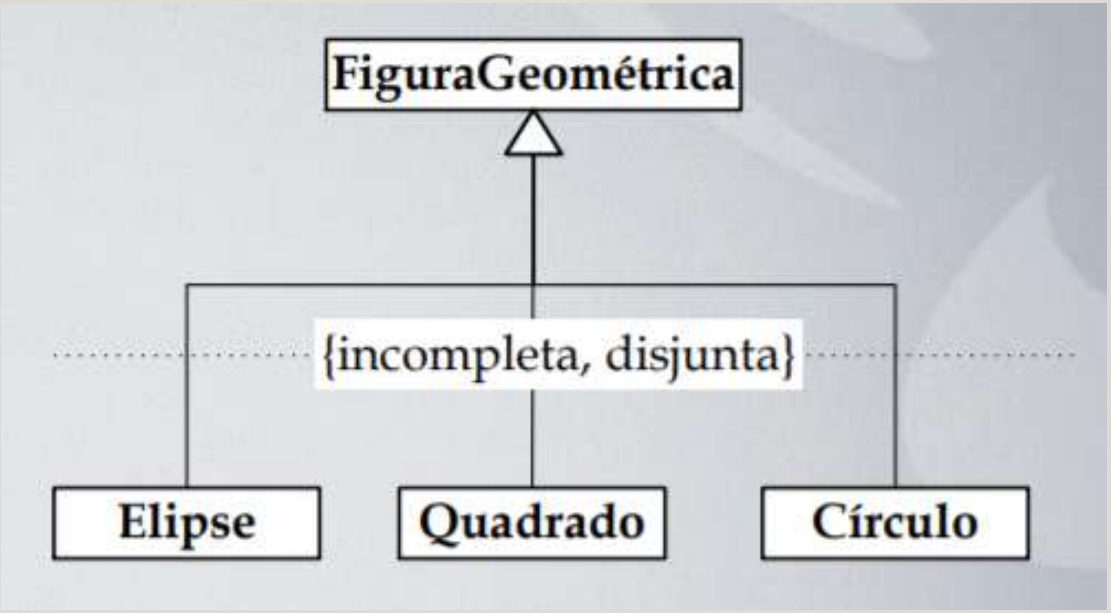
DV9

DV10

PROJETO ORIENTADO A OBJETOS COM UML – PROJETO ARQUITETURAL – IDENTIFICAÇÃO DE CLASSES DE OBJETOS – DIAGRAMA DE CLASSES

Generalização Restrita – Questão:

Por quê no exemplo é disjuntiva e incompleta?



Fonte: http://www.facom.ufu.br/~bacala/DAW/Aula05-1%20-diagrama-de-classes.pdf

Slide 19

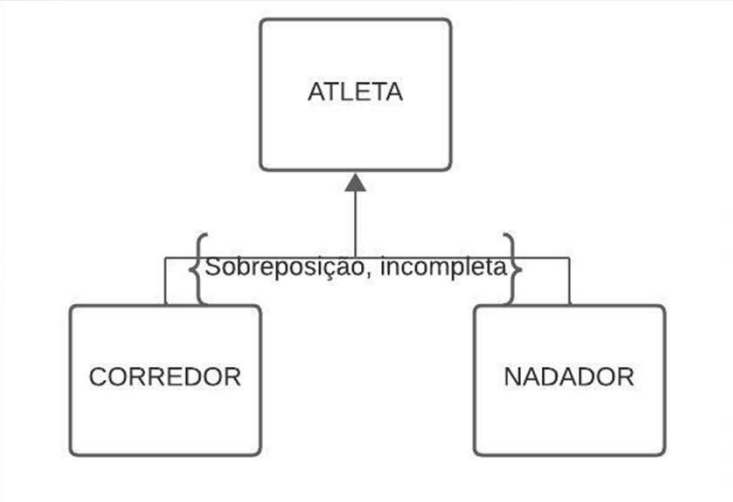
DV9 Disjunta porque a figura só pode ser uma delas e incompleta porque estáo faltando mais figuras por exemplo retângulo, losango, etc.
DENILCE VELOSO; 08/04/2021

DV10 DENILCE VELOSO; 08/04/2021

PROJETO ORIENTADO A OBJETOS COM UML – PROJETO ARQUITETURAL – IDENTIFICAÇÃO DE CLASSES DE OBJETOS – DIAGRAMA DE CLASSES

Generalização Restrita – Questão:

Por quê no exemplo é sobreposta e incompleta?



Método →mover

Poderia ter uma atleta que corre e nada

Fonte: http://www.facom.ufu.br/~bacala/DAW/Aula05-1%20-diagrama-de-classes.pdf

DV9 sobreposta porque pode ter atleta corredor e nadador
DENILCE VELOSO; 08/04/2021

PROJETO ORIENTADO A OBJETOS COM UML – PROJETO ARQUITETURAL – IDENTIFICAÇÃO DE CLASSES DE OBJETOS – DIAGRAMA DE CLASSES

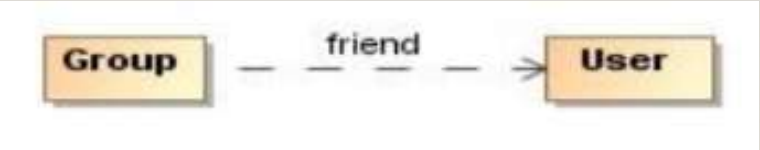
Dependência

- Dependência é uma conexão semântica entre dois modelos de elementos, um independente e outro dependente.
- Uma mudança no elemento independente irá afetar o modelo dependente. Assim como no caso anterior com generalizações, os modelos de elementos podem ser uma classe, um pacote, um caso de uso e assim por diante.
- Quando uma classe recebe um objeto de outro classe como parâmetro, uma classe acessa o objeto global da outra. Nesse caso existe uma dependência entre estas duas classes, apesar de não ser explícita.

PROJETO ORIENTADO A OBJETOS COM UML – PROJETO ARQUITETURAL – IDENTIFICAÇÃO DE CLASSES DE OBJETOS – DIAGRAMA DE CLASSES

Dependência (quanto ao uso)

- Uma relação de dependência é simbolizada por uma linha tracejada com uma seta no final de um dos lados do relacionamento. E sobre essa linha o tipo de dependência que existe entre as duas classes. Uma classe **usa** a outra classe (A recebe um objeto de B como parâmetro de um método, A cria um objeto da classe B dentro de um de seus métodos etc)
- A **modificação na especificação de uma classe afetará as classes que interagem diretamente com ela.** Exemplo:



DV [2]5 protected internal – disponíveis não apenas em todo o projeto que contém a classe que os declara, mas em todas as classes derivadas dela (mesmo que seja em outros projetos/assemblies).

DENILCE VELOSO; 18/04/2024

PROJETO ORIENTADO A OBJETOS COM UML – PROJETO ARQUITETURAL – IDENTIFICAÇÃO DE CLASSES DE OBJETOS – DIAGRAMA DE CLASSES

Em termos práticos, a existência de dependência entre classes significa que para uma classe ser **compilada** e/ou **executada** a outra classe precisa estar "linkada" a ela (às vezes a dependência ocorre apenas em tempo de execução. Por ex. quando há o uso de injeção de dependência usado em Java.

PROJETO ORIENTADO A OBJETOS COM UML – PROJETO ARQUITETURAL – IDENTIFICAÇÃO DE CLASSES DE OBJETOS – DIAGRAMA DE CLASSES

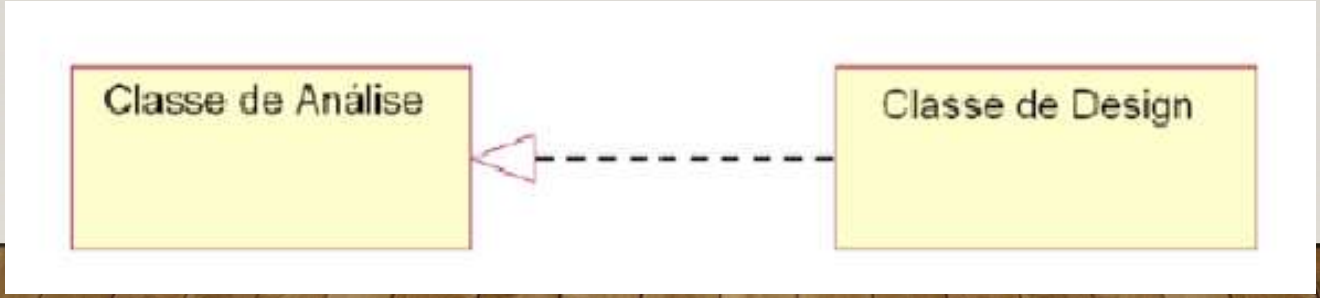
Refinamento

- Os Refinamentos são um tipo de relacionamento entre duas descrições de uma mesma coisa, mas em níveis de abstração diferentes, e podem ser usados para modelar diferentes implementações de uma mesma coisa (uma implementação simples e outra mais complexa, porém mais eficiente).
- Os refinamentos são simbolizados por uma linha tracejada com um triângulo no final de um dos lados do relacionamento e são usados em modelos de coordenação.

PROJETO ORIENTADO A OBJETOS COM UML – PROJETO ARQUITETURAL – IDENTIFICAÇÃO DE CLASSES DE OBJETOS – DIAGRAMA DE CLASSES

Refinamento

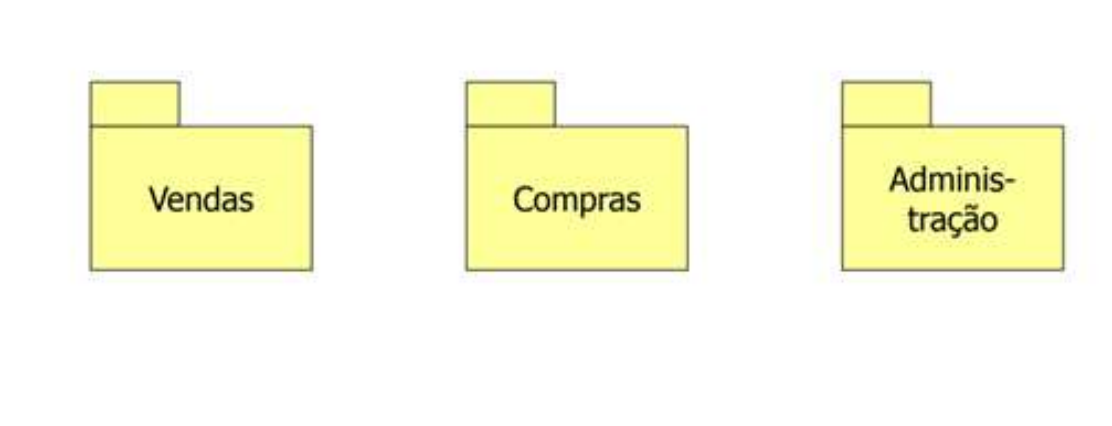
- Em grandes projetos, todos os modelos que são feitos devem ser coordenados.
- Coordenação de modelos pode ser usada para mostrar modelos em diferentes níveis de abstração que se relacionam e mostram também como modelos em diferentes fases de desenvolvimento se relacionam.



PROJETO ORIENTADO A OBJETOS COM UML – PROJETO ARQUITETURAL – IDENTIFICAÇÃO DE CLASSES DE OBJETOS – DIAGRAMA DE CLASSES

Organização de Classes em Pacotes Lógicos:

Exemplo:



** Pensando em termos práticos é possível organizar as classes em pacotes, por exemplo no Java, no C# (nameespaces) etc

DV [2]6
DV [2]7

OBJETO ORIENTADO A OBJETOS COM UML – PROJETO ARQUITETURAL – IDENTIFICAÇÃO DE CLASSES DE OBJETOS – DIAGRAMA DE CLASSES - INTERFACES

- Uma interface define um **conjunto de métodos que uma classe ou outro objeto deve implementar.** É como se fosse um "contrato" entre o implementador e o usuário da interface: o implementador garante que fornecerá os métodos definidos na interface, e o usuário pode chamar esses métodos sem precisar saber como eles são implementados. Vantagens: reuso, encapsulamento, polimorfismo.
- Ex. Sistema Bancário – métodos depositar(), sacar() podem ser implementados para diferentes tipos de contas Corrente e Poupança

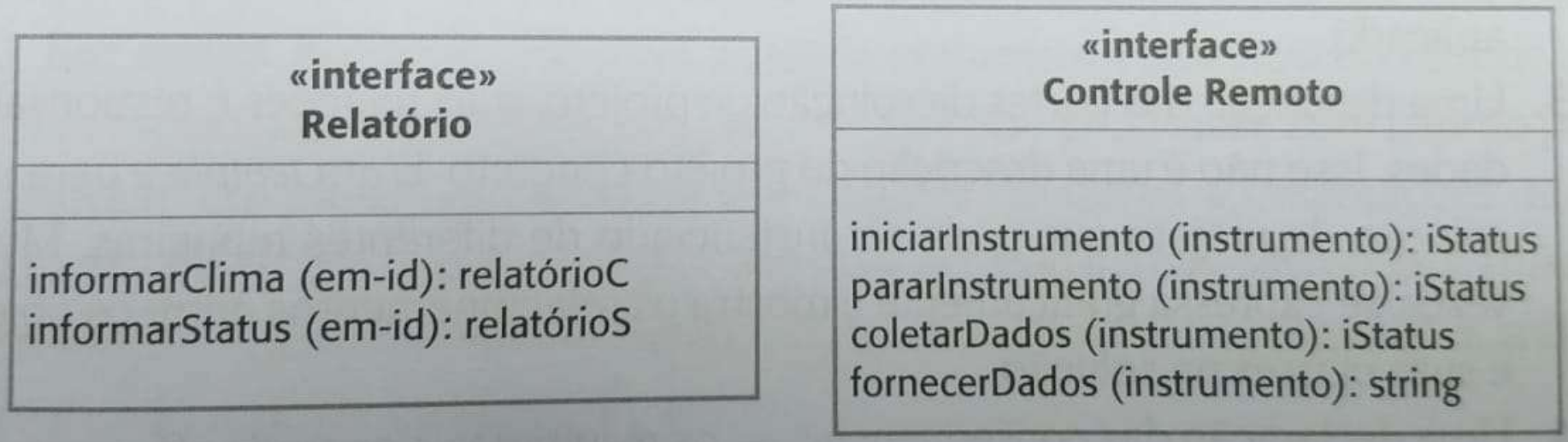
DV [2]6 DENILCE VELOSO; 18/04/2024

DV [2]7 Em um sistema bancário com classes para diferentes tipos de contas, como ContaCorrente e ContaPoupança. Ambas as classes podem implementar a interface Conta, que define métodos como depositar(), sacar() e obterSaldo(). Isso significa que qualquer código que interaja com contas bancárias, independentemente do tipo de conta, pode usar esses métodos sem precisar saber qual classe específica está sendo usada.
DENILCE VELOSO; 18/04/2024

PROJETO ORIENTADO A OBJETOS COM UML – PROJETO ARQUITETURAL – IDENTIFICAÇÃO DE CLASSES DE OBJETOS – DIAGRAMA DE CLASSES - INTERFACES

Exemplo 1:

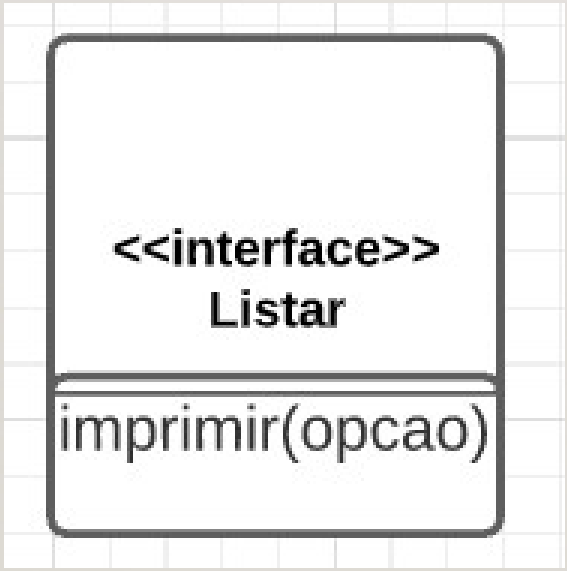
FIGURA 7.9 Interfaces da estação meteorológica.



Fonte: (Sommerville,2019. pag.187)

PROJETO ORIENTADO A OBJETOS COM UML – PROJETO ARQUITETURAL – IDENTIFICAÇÃO DE CLASSES DE OBJETOS – DIAGRAMA DE CLASSES - INTERFACES

Exemplo 2:



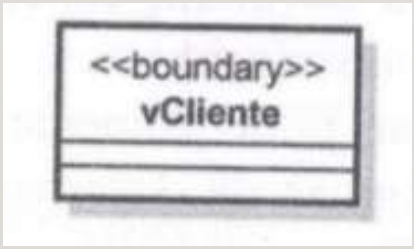
PROJETO ORIENTADO A OBJETOS COM UML – PROJETO ARQUITETURAL – IDENTIFICAÇÃO DE CLASSES DE OBJETOS – DIAGRAMA DE CLASSES

Mecanismos Gerais: Ornamentos

- Um ornamento é algo como **uma nota que adiciona texto ou algum elemento gráfico ao modelo.**
  - estereótipos
  - valores rotulados (*tagged values*)
  - restrições

PROJETO ORIENTADO A OBJETOS COM UML – PROJETO ARQUITETURAL – IDENTIFICAÇÃO DE CLASSES DE OBJETOS – DIAGRAMA DE CLASSES

Mecanismos Gerais: Estereótipos



** boundary → classe de fronteira - externo

PROJETO ORIENTADO A OBJETOS COM UML – PROJETO ARQUITETURAL – IDENTIFICAÇÃO DE CLASSES DE OBJETOS – DIAGRAMA DE CLASSES

Mecanismos Gerais: Valores Rotulados

```
Classe: Cliente
- nome (String): João Silva
- idade (int): 32
- endereço (String): Rua das Flores, 123
```

```
Classe: Produto
- id (int): persistent(primary key)
- nome (String): persistent
- descrição (String): persistent
- preço (double): persistent
- quantidadeEstoque (int): persistent
```

Por exemplo para uma melhor compreensão.
Persistent – deve ser armazenado de forma permanente

PROJETO ORIENTADO A OBJETOS COM UML – PROJETO ARQUITETURAL – IDENTIFICAÇÃO DE CLASSES DE OBJETOS – DIAGRAMA DE CLASSES

Dicas: Como identificar uma classe?

- Fazer uma lista das entidades candidatas.
- Verificar substantivos e frases nominais nas descrições textuais do domínio do problema como possíveis candidatos a entidades ou atributos.

Exemplos:
O usuário deve fazer login.
Cada estudante deve ter seu ra.
O cliente deverá fazer a reserva pelo menos 30 dias antes do embarque.

PROJETO ORIENTADO A OBJETOS COM UML – PROJETO ARQUITETURAL – IDENTIFICAÇÃO DE CLASSES DE OBJETOS – DIAGRAMA DE CLASSES

Dicas: Como identificar uma classe?

- Verificar se existem informações que devem ser armazenadas ou analisadas.
- Verificar se existem sistemas externos ao modelados (devem ser vistos como classes para interação).
- Verificar papel dos atores dentro do sistema.

Exemplos:
As leituras realizadas pelo medidor de temperatura devem ficar em um banco de dados.
As movimentações diárias de pagamento devem ser enviadas ao banco via cnab.

PROJETO ORIENTADO A OBJETOS COM UML – PROJETO ARQUITETURAL – IDENTIFICAÇÃO DE CLASSES DE OBJETOS – DIAGRAMA DE CLASSES

Dicas: Como identificar uma associação?

- Focar nas associações cujo conhecimento deve ser preservado.
- Usar nomes baseados em expressões que façam sentido quando lidas no contexto do modelo.
- Evitar mostrar associações redundantes.

Exemplos:

O estudante deve fazer a reserva do livro.

O fornecedor deve enviar o preço do produto.

O colaborador pode solicitar um treinamento quando estiver disponível.

PROJETO ORIENTADO A OBJETOS COM UML – PROJETO ARQUITETURAL – IDENTIFICAÇÃO DE CLASSES DE OBJETOS – DIAGRAMA DE CLASSES

Dicas: Como identificar uma generalização?

- Verificar se a subclasse tem atributos adicionais de interesse.
- Verificar se a subclasse tem associações adicionais de interesse.
- Verificar se a subclasse será manipulada/usada de maneira diferente da superclasse.
- Verificar se a subclasse se comporta de maneira diferente da superclasse.
- As subclasse satisfazem 100% a regra do "is-a".

Exemplos: Prova, Prova Escrita, Prova Oral, Prova Online.

PROJETO ORIENTADO A OBJETOS COM UML – PROJETO ARQUITETURAL – IDENTIFICAÇÃO DE CLASSES DE OBJETOS – DIAGRAMA DE CLASSES

Dicas: Como identificar uma agregação?

- Verificar se elas são geralmente criadas/destruídas no mesmo instante.
- Verificar se as partes possuem tempo de vida limitado ao tempo de vida do composto.
- Verificar se elas possuem relacionamentos comuns.

Exemplos:

- Um carrinho de compra pode conter vários produtos.
- Uma nota fiscal pode conter vários itens.
- O corpo deve conter coração e pulmão.
- A matrícula deve conter o aluno e a turma.

PROJETO ORIENTADO A OBJETOS COM UML – PROJETO ARQUITETURAL – IDENTIFICAÇÃO DE CLASSES DE OBJETOS – DIAGRAMA DE CLASSES

Dicas: Como identificar um atributo?

- Atributos devem preferencialmente representar tipos primitivos de dados ou de valores simples. Exemplo: Texto, Data, Número, Endereço, etc.
- Atributos não devem ser usados para representar um conceito complexo ou relacionar conceitos (ex.: atributo chave-estrangeira)

PROJETO ORIENTADO A OBJETOS COM UML – PROJETO ARQUITETURAL – IDENTIFICAÇÃO DE CLASSES DE OBJETOS – DIAGRAMA DE CLASSES

Como identificar um atributo?

- Um atributo deve ser de tipo não-primitivo quando:
  - É composto de seções separadas: Ex.: endereço, data.
  - Precisa ser analisado ou validado: Ex,: CPF, número de matrícula
  - Possui outros atributos: Um preço promocional com prazo de validade.
  - É uma quantidade com uma unidade. Ex.: valores monetários, medidas.

Exemplos: Classe Carro – colocar potência do motor como atributo, na verdade potência é um atributo da classe motor.

PROJETO ORIENTADO A OBJETOS COM UML – PROJETO ARQUITETURAL – IDENTIFICAÇÃO DE CLASSES DE OBJETOS – DIAGRAMA DE CLASSES

Como identificar um método?

- Os métodos são acrescentados na perspectiva de implementação e são derivados a partir dos diagramas de interação: colaboração e sequências.
- É útil distinguir operações de métodos. Operação é algo que se evoca sobre um objeto (a chamada do procedimento). Para realizar uma operação a classe implementa um método (o corpo do procedimento).

PROJETO ORIENTADO A OBJETOS COM UML – PROJETO ARQUITETURAL – IDENTIFICAÇÃO DE CLASSES DE OBJETOS – DIAGRAMA DE CLASSES

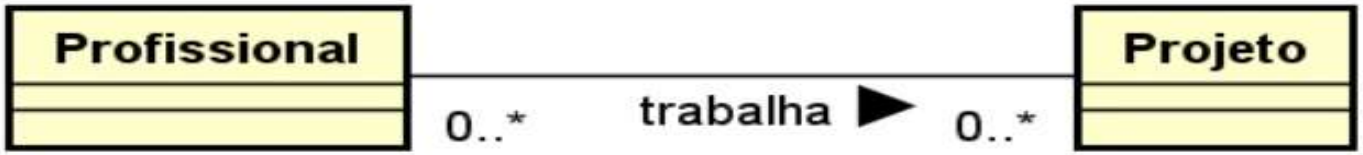
Como identificar um método?

- É útil distinguir operações que alteram ou não o estado (atributos) de uma classe.
- Não alteram: query, métodos de obtenção, *getting methods*.
- Alteram: modificadores, métodos de atribuição ou fixação, *setting methods*.

PROJETO ORIENTADO A OBJETOS COM UML – DIAGRAMA DE CLASSES (EXEMPLOS-CONCEITUAL)

► Um para Um

► Um para Muitos

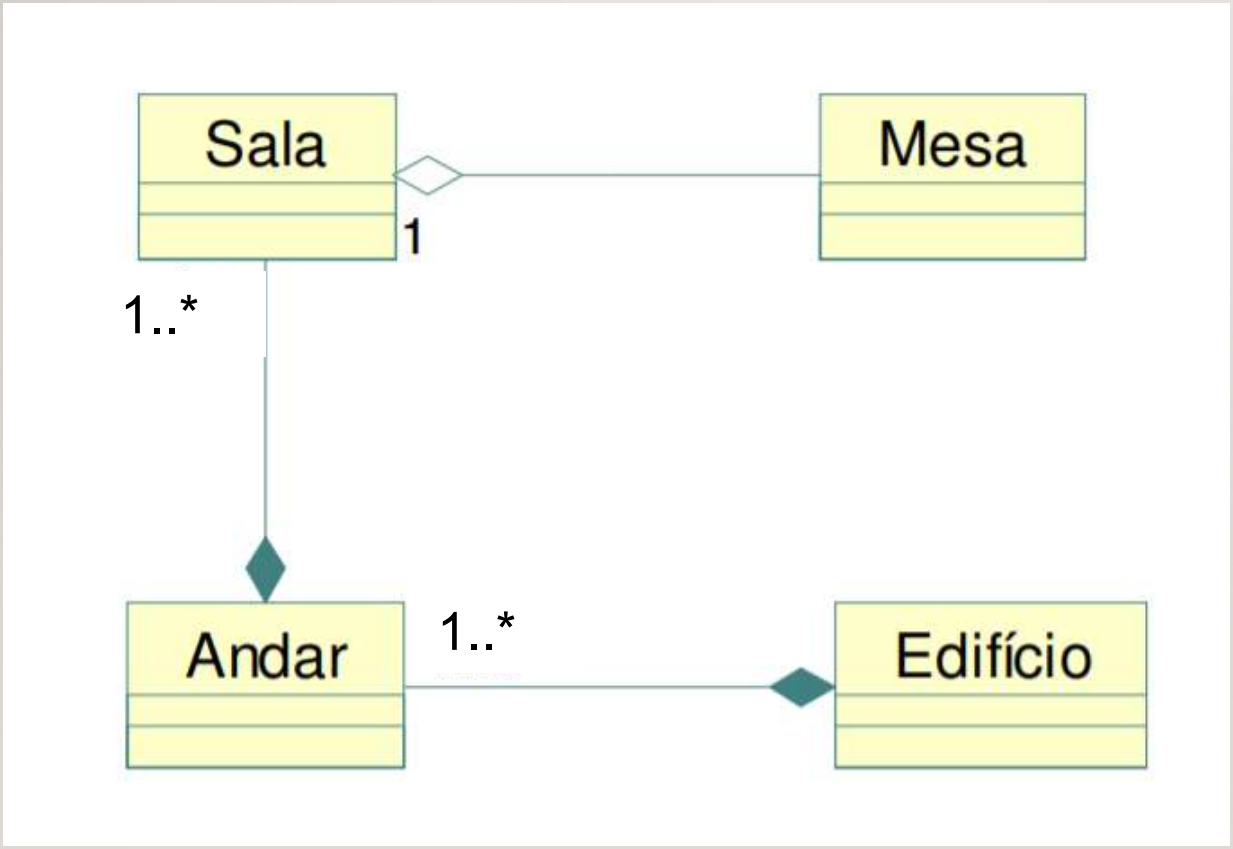


► Muitos para Muitos

http://spaceprogrammer.com/uml/compreendendo-multiplicidade-e-os-tipos-de-associacao/

DV14

PROJETO ORIENTADO A OBJETOS COM UML – DIAGRAMA DE CLASSES (EXEMPLOS - CONCEITUAL)



Losango preenchido – Associação tipo composição

Losango aberto – Associação tipo agregação

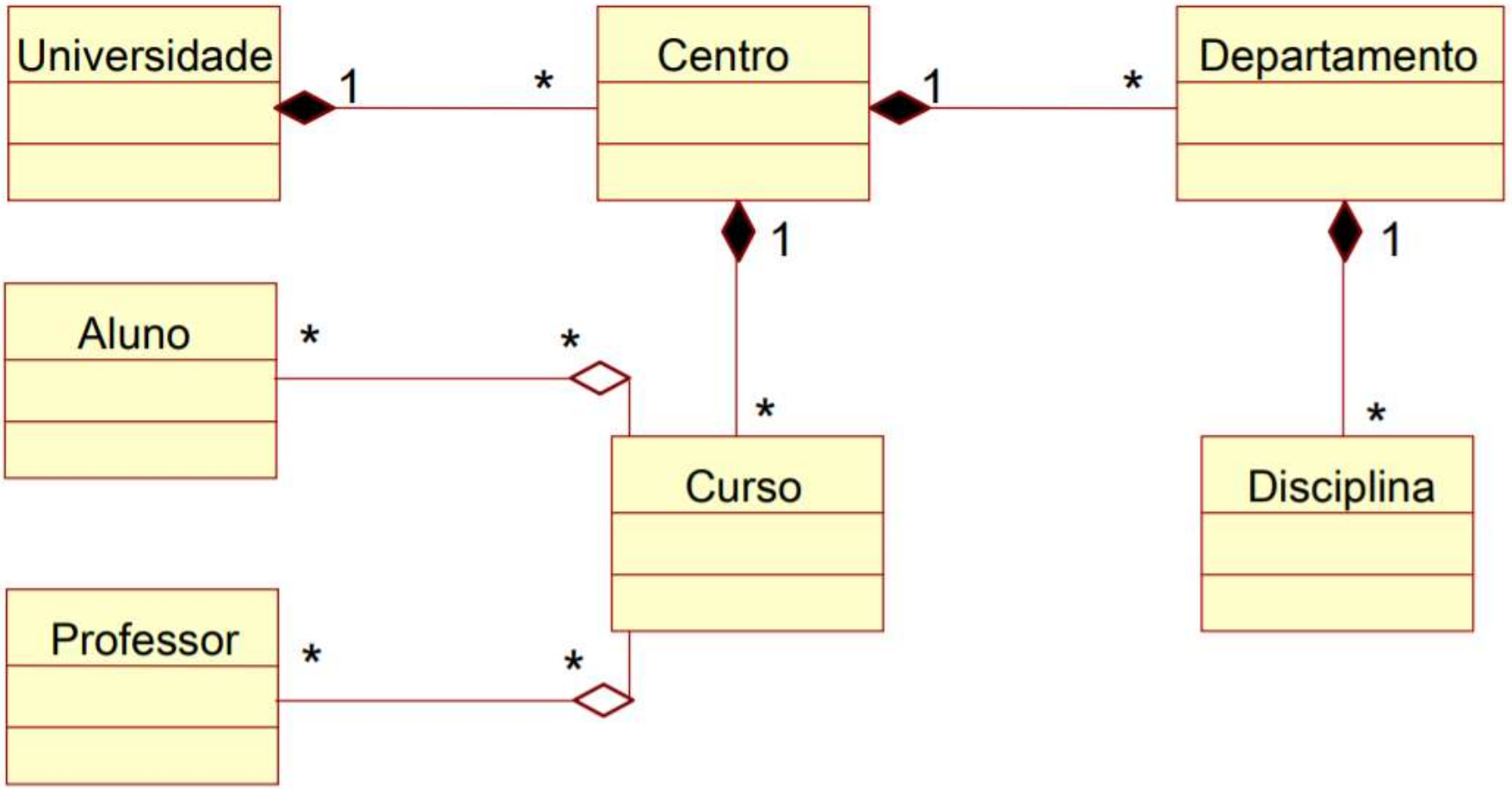
** conceitual

https://moodle.unesp.br/ava/pluginfile.php/25933/mod_resource/content/1/diagrama_classes.pdf

DV14 lembre-se na composição, quando o todo "é excluido" a outra classe agregada também deve ser excluida
DENILCE VELOSO; 15/04/2021

PROJETO ORIENTADO A OBJETOS COM UML – DIAGRAMA DE CLASSES – EXEMPLO CONCEITUAL

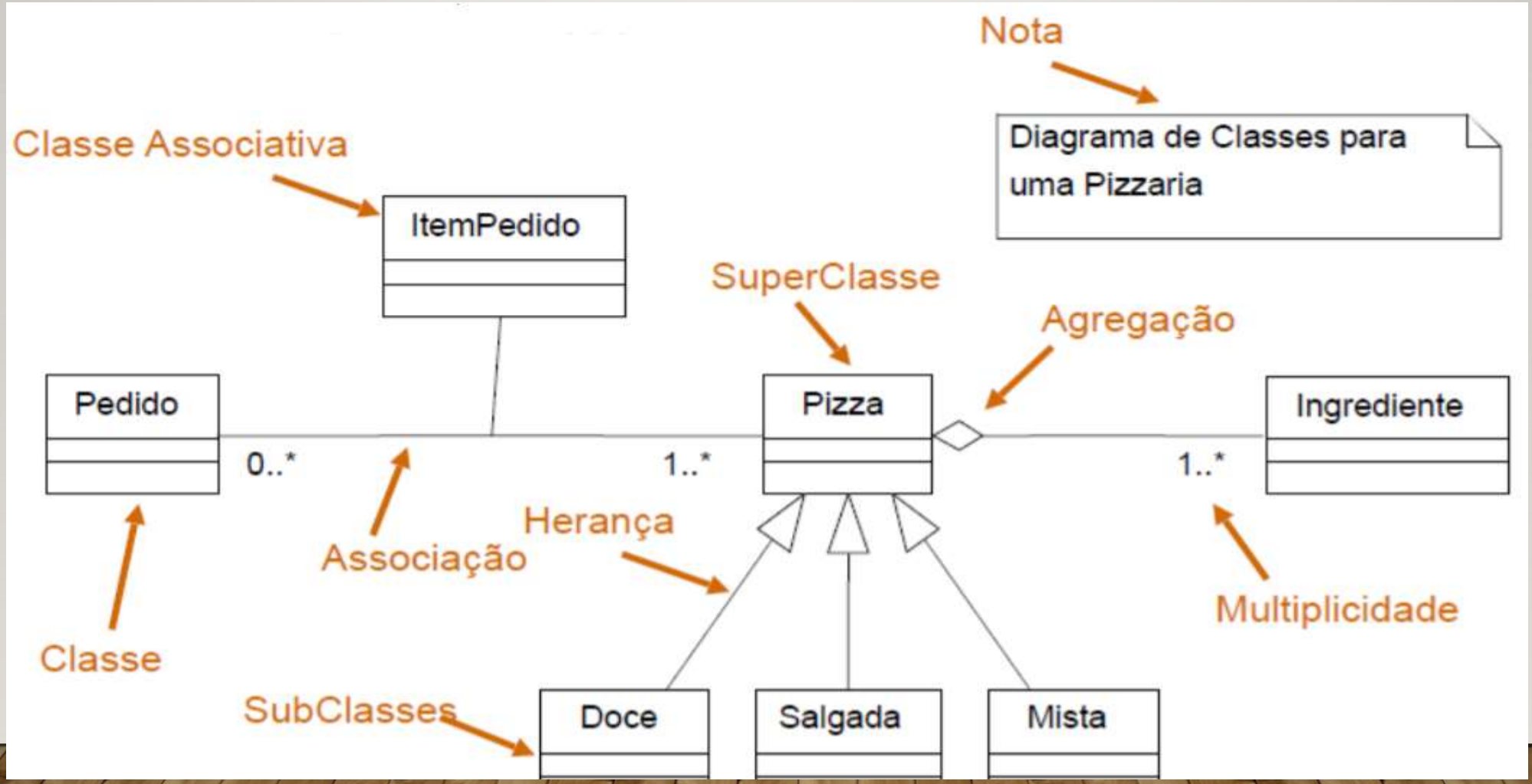
- Agregação Simples X Agregação de Composição



https://www.cin.ufpe.br/~in1008/aulas/ProjetoBD_ParteVI_Diagrama_de_Classe.pdf

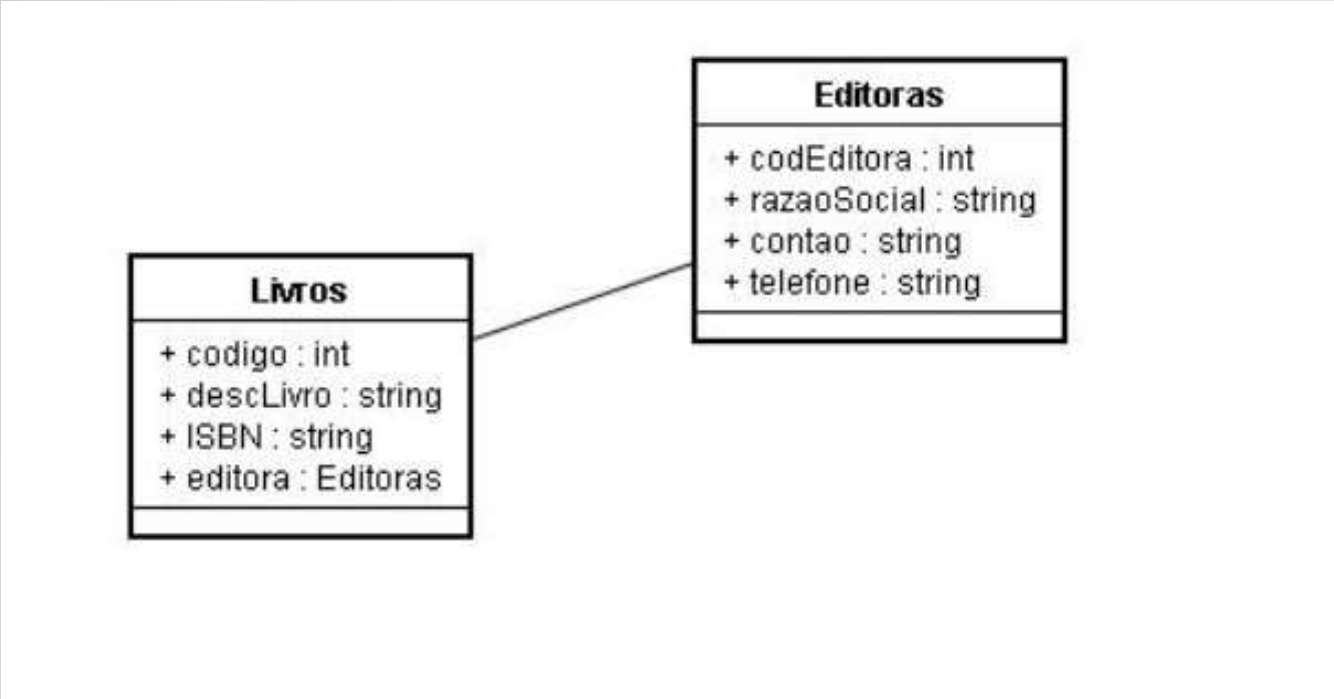
PROJETO ORIENTADO A OBJETOS COM UML – DIAGRAMA DE CLASSES – EXEMPLO CONCEITUAL

Como você lê o diagrama???



http://www.urisantiago.br/computacao/arquivos/enade/Aula%203%20-%20Diagramas%20de%20Classes.pdf

PROJETO ORIENTADO A OBJETOS COM UML – DIAGRAMA DE CLASSES (EXEMPLOS – IMPLEMENTAÇÃO)



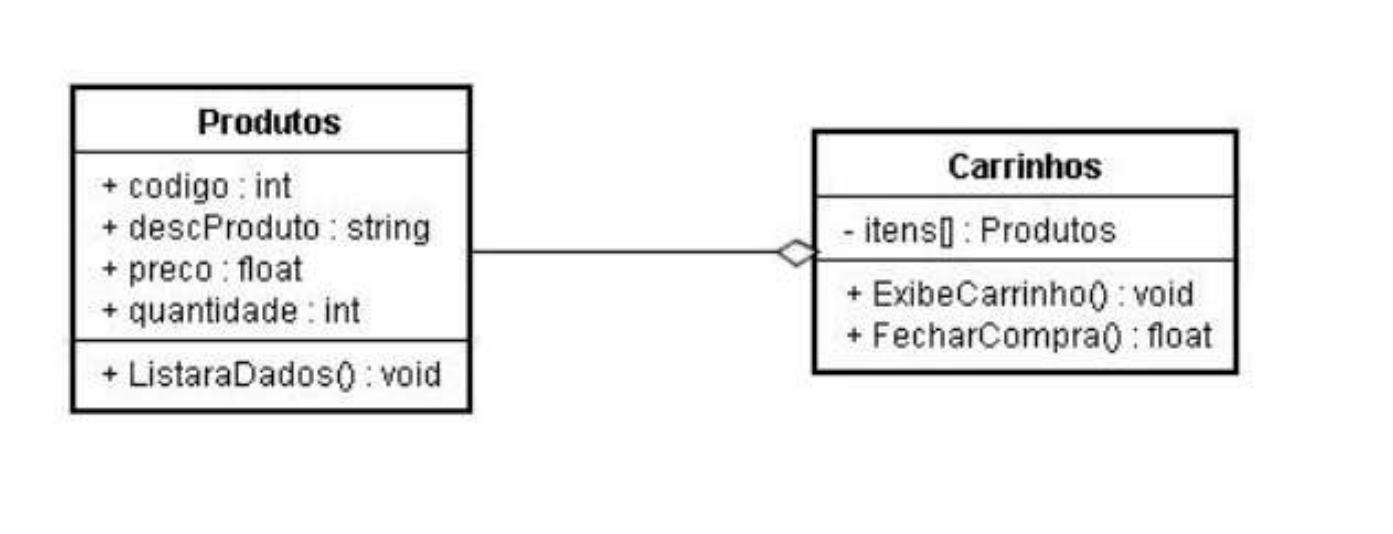
Associação Normal – linha contínua

** um livro sempre pertence somente a uma editora, se fosse livraria seria diferente

https://www.ateomomento.com.br/uml-diagrama-de-classes/

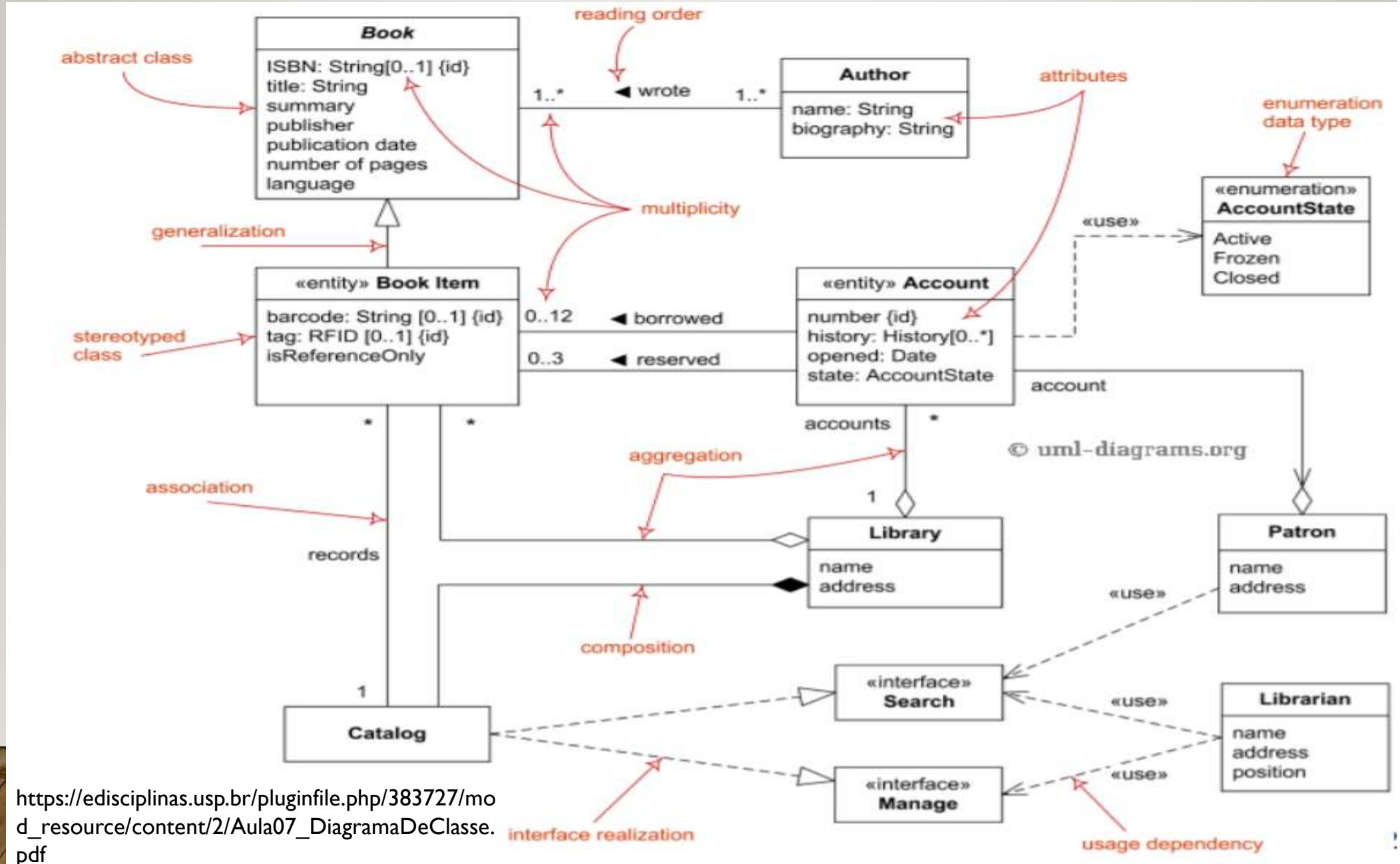
PROJETO ORIENTADO A OBJETOS COM UML – DIAGRAMA DE CLASSES – EXEMPLOS IMPLEMENTAÇÃO

Associação tipo Agregação
** implementação



https://www.ateomomento.com.br/uml-diagrama-de-classes/

PROJETO ORIENTADO A OBJETOS COM UML – DIAGRAMA DE CLASSES - EXEMPLO



https://edisciplinas.usp.br/pluginfile.php/383727/mod_resource/content/2/Aula07_DiagramaDeClasse.pdf

PROJETO ORIENTADO A OBJETOS COM UML – PROJETO ARQUITETURAL – IDENTIFICAÇÃO DE CLASSES DE OBJETOS – DIAGRAMA DE CLASSES

Vantagens e desvantagens do Diagrama de Classes:

- Importante utilizar os diagramas de classes para a construção de softwares que necessitam orientação a objetos.
- São muitos ricos na descrição do sistema.
- Podem ser tornar complexos na hora da utilização.
- Risco de ficar preso em detalhes de implementação.

ATIVIDADE 7 – **CRIAR DIAGRAMA DE CLASSES**

PROBLEMA : Uma pessoa controla em Excel uma planilha com a sua lista de compras mensal. Ela cadastra o nome do produto, a unidade de compra, a quantidade prevista para um mês, a quantidade que efetivamente será comprada e o preço estimado (atualizado todo mês)

Produto	Unidade de compra	Qtd Mês	Qtd Compra	Preço Estimado
Arroz	Kg	8	7	1,80
Feijão	Kg	6	6	2,10
Açúcar	Kg	3	2	1,05
Carne	Kg	6	7,5	8,00
...				
Total Estimado				150,00

- A quantidade de compra pode variar em virtude de sobra de um mês para o outro, ou da necessidade de um gasto maior no mês.
- As compras são feitas pela própria pessoa. Por esse motivo, ela não vê necessidade de relacionar as marcas dos produtos. Mensalmente, a pessoa analisa o quanto comprou e pagou por cada produto, e se achar necessário, atualiza a quantidade prevista e preço estimado (previsto) de cada produto.

Adaptado de Melo (2006).

PRÓXIMO PASSO PARA O PROJETO

HOJE

Discussão sobre DIAGRAMA DE CASO DE USO v.2.0, RESULTADOS DAS PESQUISA e REQUISITOS FUNCIONAIS V.3.0

PRÓXIMA AULA

- Versão 5.0 Requisitos Funcionais
- Versão 3.0 Diagrama de Caso de Uso (alto nível)
- Versão 1.0 Descrição de Caso de Uso (baixo nível)
- AINDA NÃO FIZEMOS AO NÃO FUNCIONAIS

*** usar SEMPRE os mesmos nomes da lista de requisitos funcionais no diagrama de caso de uso e descrição de caso de uso*

REFERÊNCIAS

- BOOCH, Grady et al. ***The Unifed Modeling Language User Guide***. Addison Wesley, 2005.
- GUEDES, Gilleanes T. A. **UML: Uma abordagem prática**. São Paulo: Novatec, 2006.
- MEDEIROS, Ernani. Desenvolvendo Software com UML 2.0: Definitivo. Makron Books, 2006.
- PRESSMAN, Roger S**. Engenharia de software : uma abordagem profissional**. 7. ed. Porto Alegre: McGraw-Hill, 2011.
- SOMMERVILLE, Ian**. Engenharia de software**. 10. ed. São Paulo: Pearson, 2019.
- _____. Materiais Fornecidos pelo autor. Disponível em: <https://www.slideshare.net/software-engineering-book/ch4-req-eng> Acesso em: 01 de mar. de 2021.