# *Análise e Projeto de Sistemas II*

Aula 5: Diagrama de Classes

Prof. Fernando Xavier

fernando.xavier@udf.edu.br

# Análise e Projeto de Sistemas II

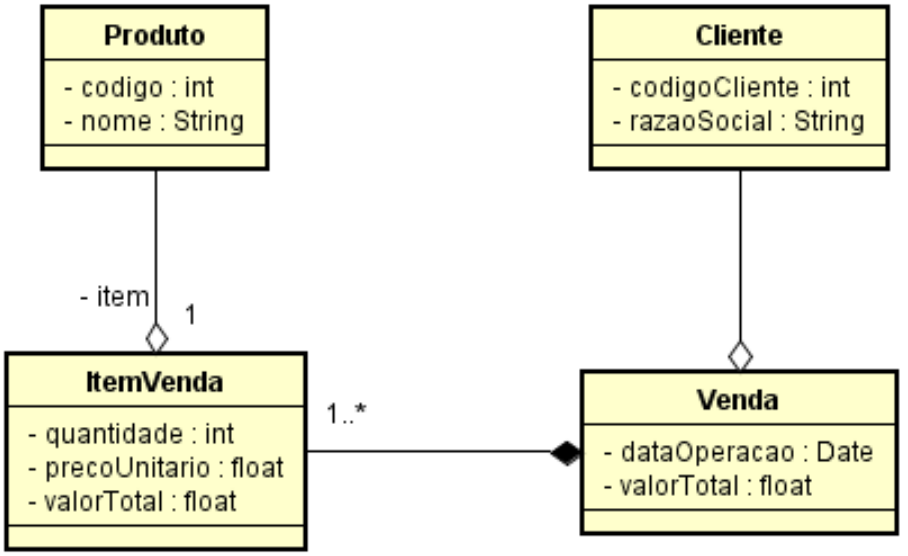
- Casos de Uso no processo de desenvolvimento
  - Perspectiva do sistema a partir de uma visão externa, ou seja, aquela mais próxima da visão dos usuários do que o sistema deve fazer
  - Para os desenvolvedores, apenas essa visão não é suficiente
  - Cada funcionalidada dessa visão é fornecida através da interação de elementos internos. Exemplos:
    - Classes/objetos
    - Telas

# Análise e Projeto de Sistemas II

- Casos de Uso no processo de desenvolvimento
  - Externamente, o ator vê um relatório de vendas
    - Interação representada pelos casos de uso
  - Internamente, elementos do sistema devem interagir para fornecer esse relatório
    - Elementos representados por outros diagramas
      - Classes
      - Pacotes
  - Lembre-se que os diagramas são complementares entre si

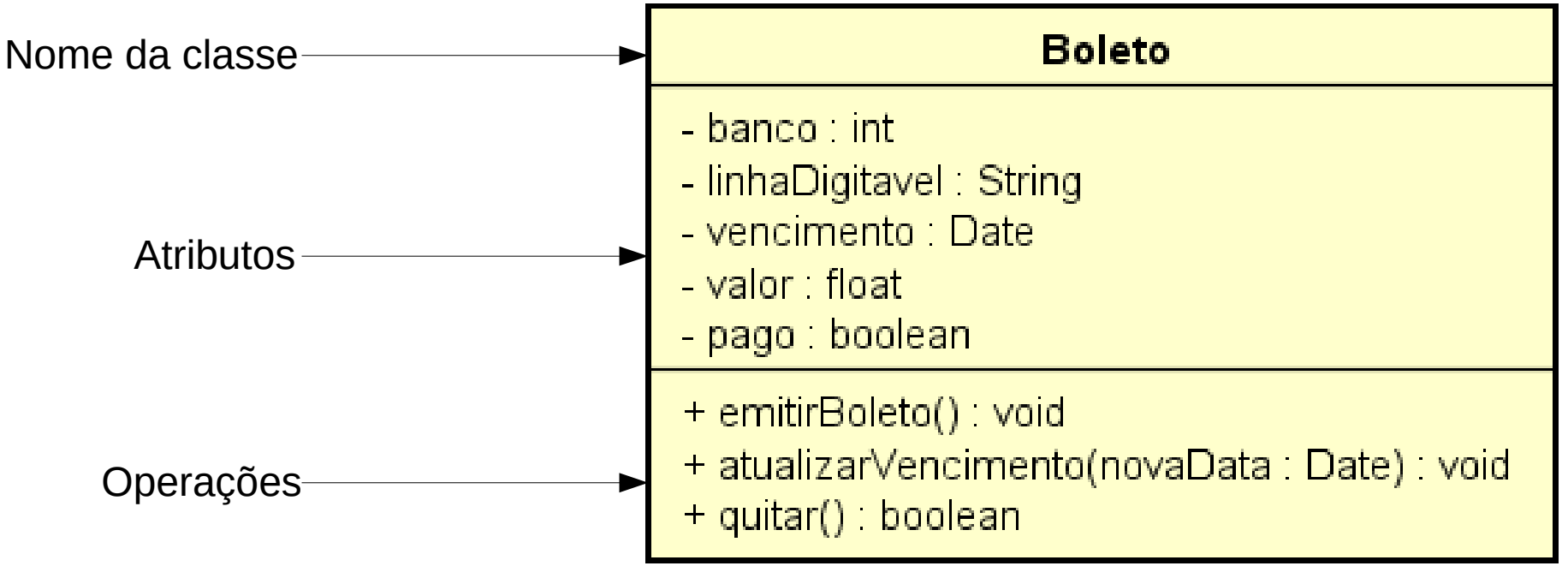
# Análise e Projeto de Sistemas II

- Diagrama de Classes
  - Diagrama utilizado para modelar sistemas orientados a objetos
  - Nesse diagrama representa-se as classes e suas interações



# Análise e Projeto de Sistemas II

- Diagrama de Classes

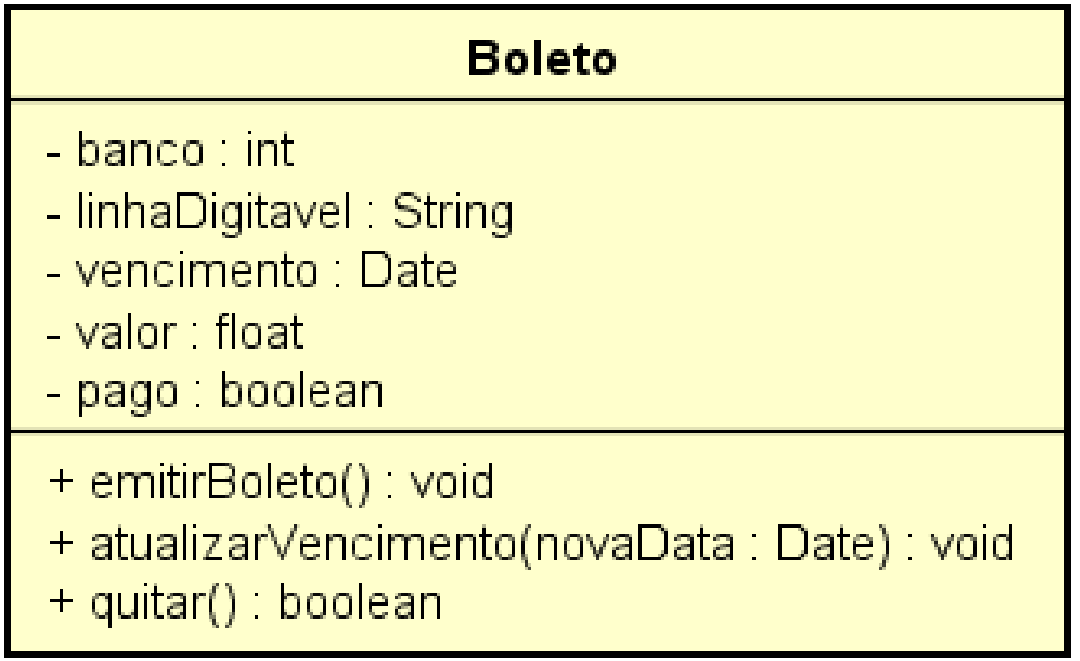


# Análise e Projeto de Sistemas II

- Diagrama de Classes: como gerar?

- Alguns elementos gerados durante a análise podem ser úteis:

- Requisitos
- Casos de uso
- Protótipos

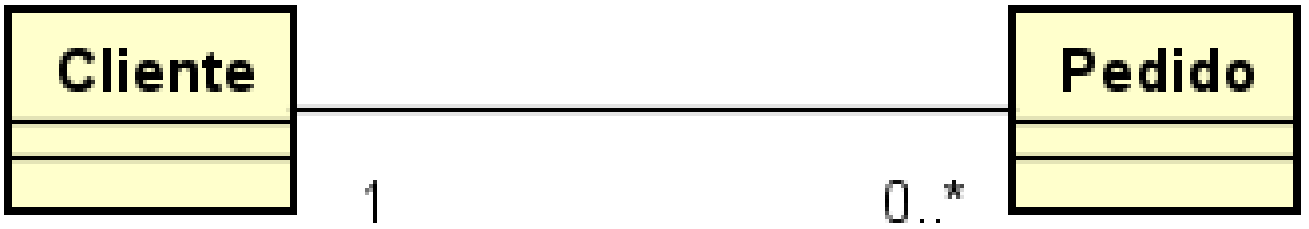


# Análise e Projeto de Sistemas II

- Diagrama de Classes: associações
  - Linhas que representam relações entre os objetos das classes

# Análise e Projeto de Sistemas II

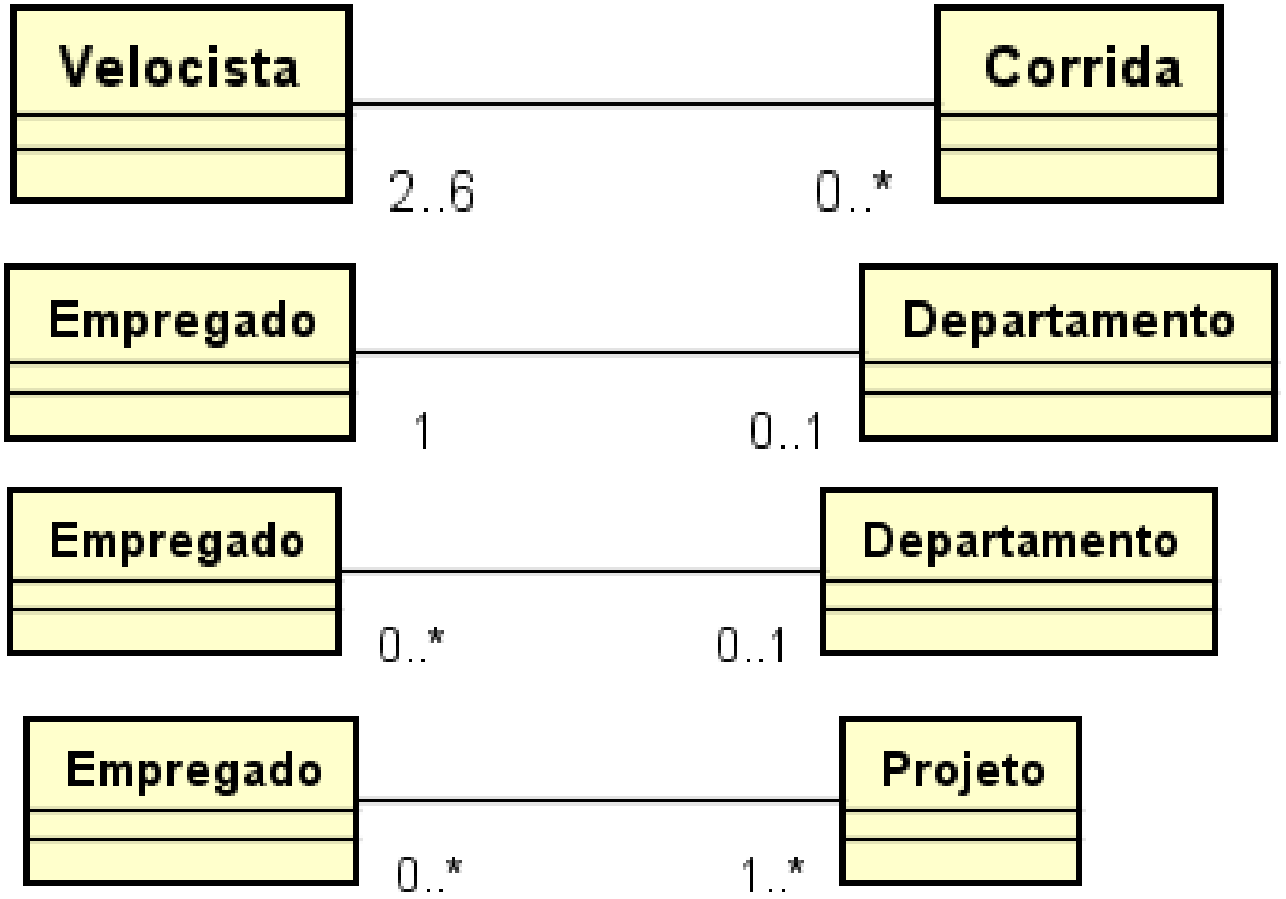
- Diagrama de Classes: multiplicidades
  - Representação dos limites superior e inferior da quantidade de objetos associados




| Nome | Símbolo |
|---|---|
| Apenas um | 1 |
| Zero ou muitos | 0..* |
| Um ou muitos | 1..* |
| Zero ou um | 0..1 |
| Intervalo | $L_{inf}..L_{sup}$ |

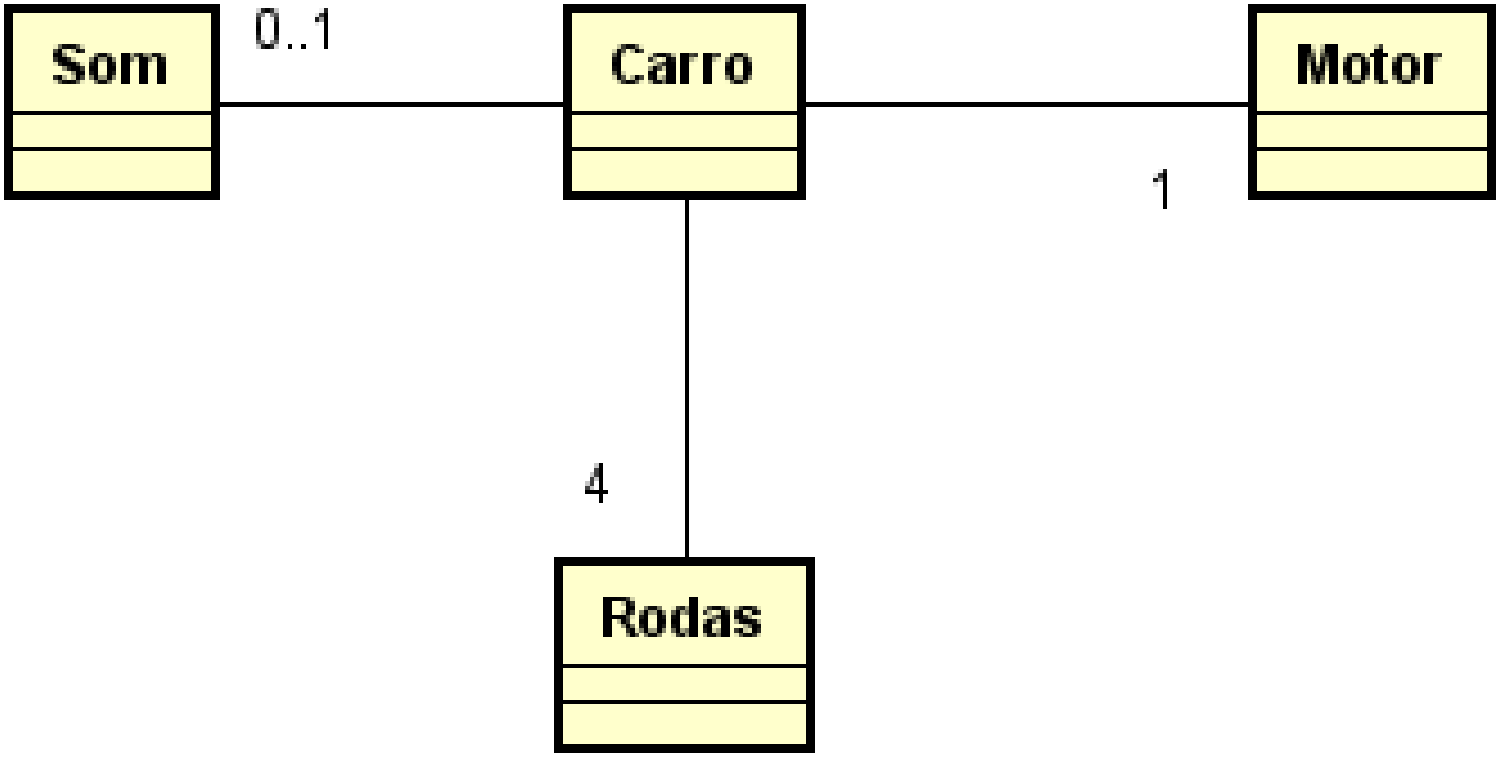
# Análise e Projeto de Sistemas II

- Explique as associações abaixo



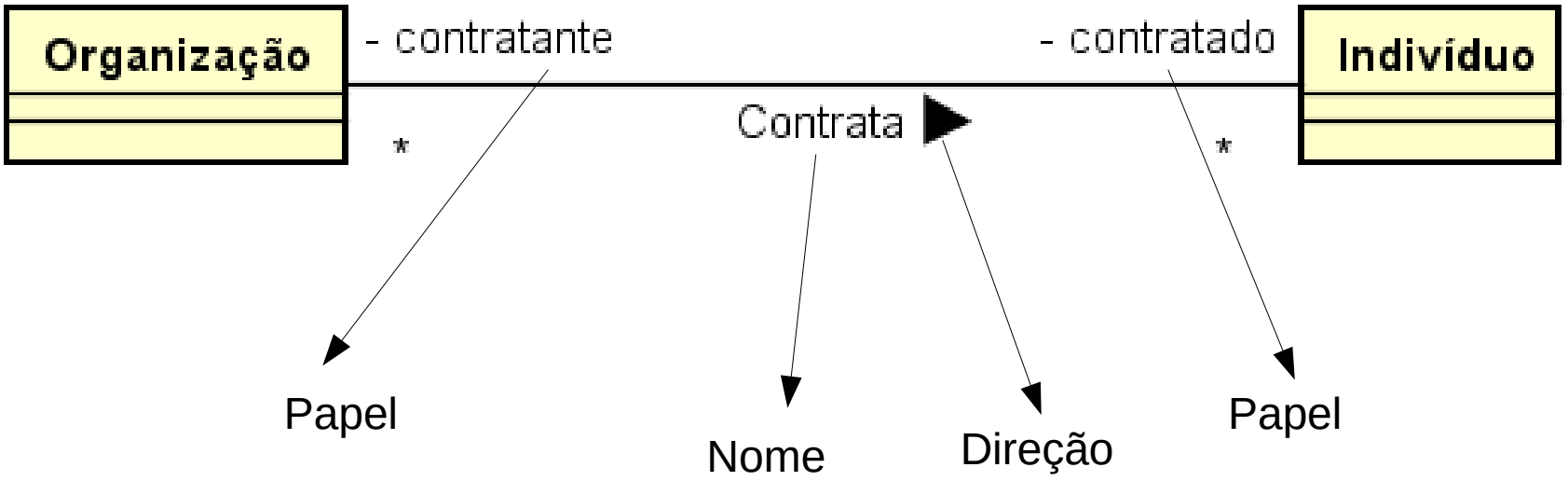
# Análise e Projeto de Sistemas II

- Explique as associações abaixo



# Análise e Projeto de Sistemas II

- Participação nas associações
  - Possibilitam melhor clareza nas associações, quando não for evidente. Cuidado com o excesso!

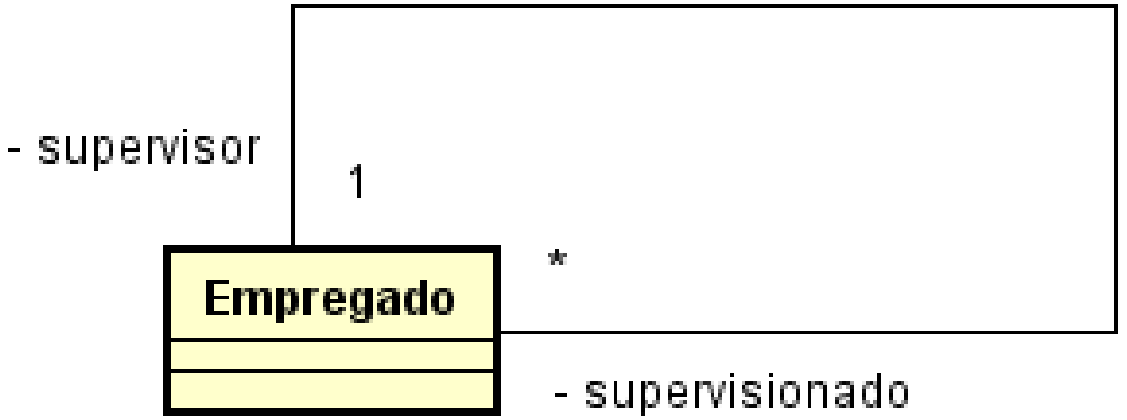


# Análise e Projeto de Sistemas II

- Duas associações com as mesmas classes

# Análise e Projeto de Sistemas II

- Associações reflexivas



# Análise e Projeto de Sistemas II

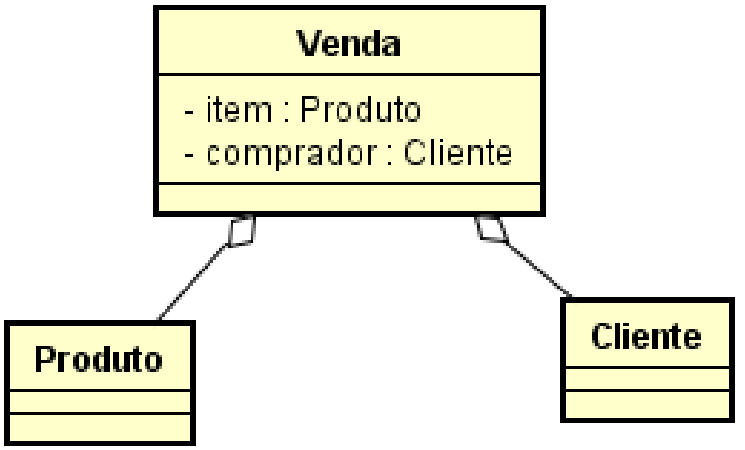
- Relações todo-parte
  - Enquanto a associação simples representa uma relação de uso (classe A usa a classe B), existem relacionamentos que envolvem o conceito de propriedade (*has-a*), em uma relação todo-parte
  - São basicamente dois tipos especiais de associação, que diferem entre si quanto ao controle de vida da parte
    - Agregação
    - Composição

# Análise e Projeto de Sistemas II

- Agregação
  - Tipo de associação Parte-Todo entre classes, onde a parte continua a existir mesmo se o todo não existir
  - O ciclo de vida da parte não é controlado pelo todo
  - Exemplos:
    - Livraria e Livro
    - Venda e Produto
    - Venda e Cliente

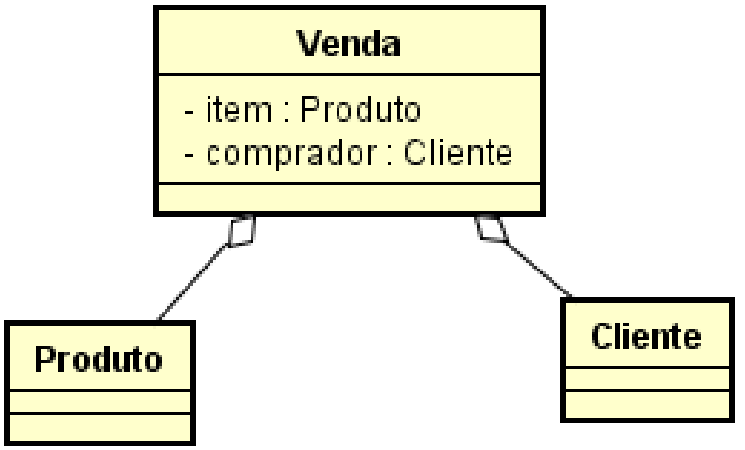
# Análise e Projeto de Sistemas II

- Agregação
  - Produto e Cliente continuam a existir mesmo se Venda não existir
  - As classes Produto e Cliente podem ser usadas em outras classes?



# Análise e Projeto de Sistemas II

- Agregação
  - Produto e Cliente continuam a existir mesmo se Venda não existir
  - As classes Produto e Cliente podem ser usadas em outras classes?
    - Sim, como Produto pode ser usada em Produção ou Cliente pode ser usada em uma classe Atendimento



# Análise e Projeto de Sistemas II

- Agregação – Exemplo em Java

```java
package agregacao;

public class Produto {

    private String nome;
    private int estoque;
    private float preco;
    public String getNome() {
        return nome;
    }
    public void setNome(String nome) {
        this.nome = nome;
    }
    public int getEstoque() {
        return estoque;
    }
    public void setEstoque(int estoque) {
        this.estoque = estoque;
    }
    public float getPreco() {
        return preco;
    }
    public void setPreco(float preco) {
        this.preco = preco;
    }

}
```

```java
package agregacao;

public class Venda {

    Produto produto;
    Cliente cliente;

}
```

Note que Venda tem atributos de Produto e Cliente. No entanto, Produto continua a existir mesmo se não houver objetos de Venda

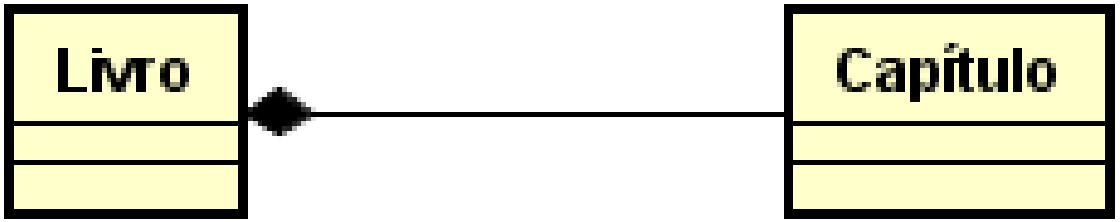
# Análise e Projeto de Sistemas II

- Agregação – Exemplo em Python

```python
class Aluno:
    def __init__(self, nomeAluno, rgm):
        self.nome = nomeAluno
        self.rgm = rgm
        self.disciplinas = []

    def inscricaoDisciplina(self, disciplina):
        self.disciplinas.append(disciplina)

class Disciplina:
    def __init__(self, nome, cargaHoraria):
        self.nome = nome
        self.cargaHoraria = cargaHoraria

disc1 = Disciplina("Programação OO", 80)
disc2 = Disciplina("Estruturas de Dados", 40)

aluno = Aluno("Fernando X", "1234567-7")
aluno.inscricaoDisciplina(disc1)
aluno.inscricaoDisciplina(disc2)

print("O aluno está cursando {} disciplinas".format(len(aluno.disciplinas)))
```

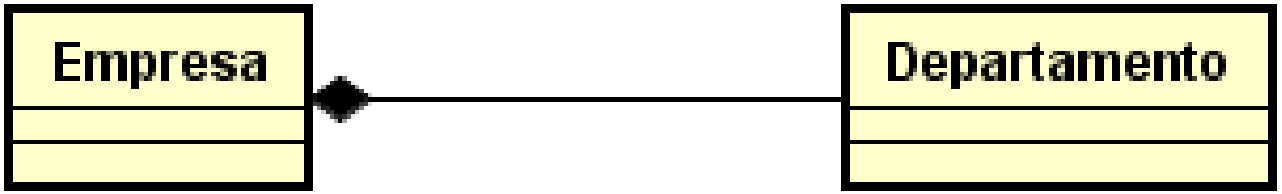
# Análise e Projeto de Sistemas II

- Composição
  - As partes não existem sem o todo e o objeto todo é responsável por criar e destruir as partes
  - O ciclo de vida da parte depende do todo
    - Livro e Capítulo
    - Pedido de Venda e Item de Venda



# Análise e Projeto de Sistemas II

- Composição
  - Uma empresa é composta de departamentos
  - A criação e destruição de departamentos depende da empresa
  - Isso significa que o departamento só existe por causa da empresa



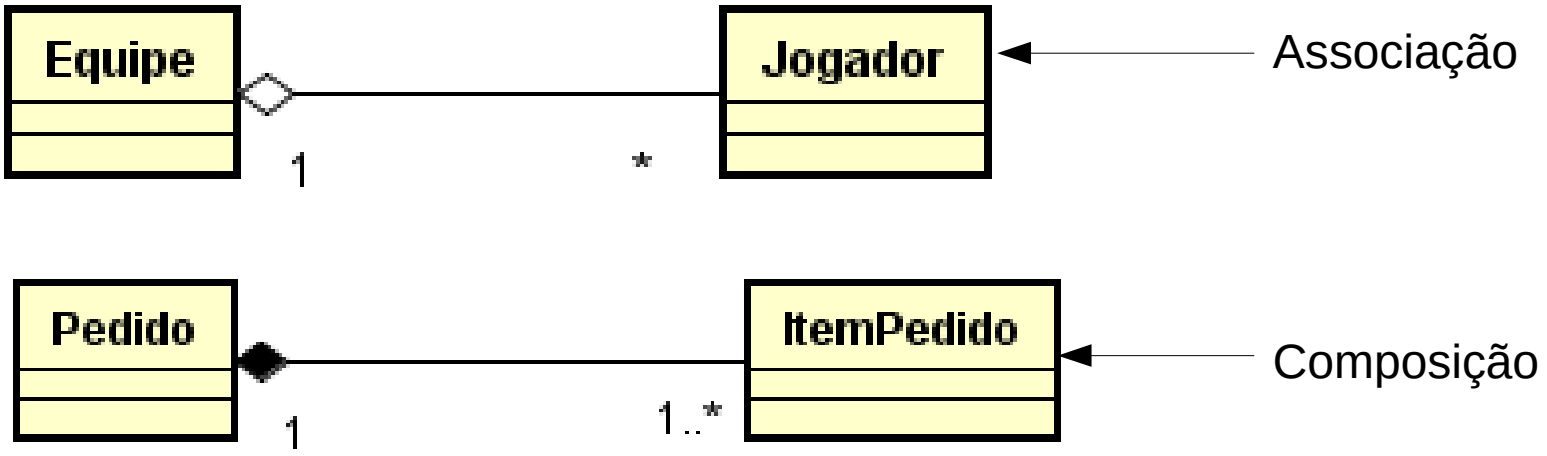
# Análise e Projeto de Sistemas II

- Composição

```
1  class livro:
2
3      def __init__(self):
4          self.titulo = None
5          self.autor = None
6          self.capitulos = []
7
8      def criarCapitulo(self, nome):
9          self.capitulos.append(capitulo(nome))
10
11 class capitulo:
12         def __init__(self, nome):
13             self.titulo = nome
14
15 livro = livro()
16 livro.criarCapitulo("1 - Introdução")
17 livro.criarCapitulo("2 - Desenvolvimento")
18 livro.criarCapitulo("3 - Conclusão")
19
20 print("O livro tem {} capítulos".format(len(livro.capitulos)))
```

# Análise e Projeto de Sistemas II

- Associações: Composição e Agregação
  - Relação do tipo todo-parte
  - Um objeto contém o outro



# *Análise e Projeto de Sistemas II*

- Sobre relações entre classes

| **Associação** | **Agregação** | **Composição** |
|---|---|---|
| Relação de Uso (*Using Relationship*) | Relação de Tem um (*Has-a relationship*) | Relação de Tem um (*Has-a relationship*) |
| Não há um objeto dono de outro | Objeto (parte) pertence a outro objeto (Todo) | Objeto (parte) pertence a outro objeto (Todo) |
| Os objetos têm seu próprio ciclo de vida | Todo-parte e as partes continuam a existir sem o Todo | Todo-parte mas as partes não existem sem oTodo |

# Análise e Projeto de Sistemas II

- Exercícios

  1) Desenhe um diagrama de classes com relacionamentos, nomes de papéis e multiplicidades para:

  - Um curso é formado por disciplinas
  - Um professor pode ser responsável por uma ou mais disciplinas
  - Uma disciplina é pré-requisito para outra disciplina

# *Análise e Projeto de Sistemas II*

- Exercícios

  2) Crie um diagrama de classes para a seguinte demanda de sistema: uma faculdade deseja implementar um sistema de gestão acadêmica, em que seja possível um departamento criar uma disciplina que, por sua vez, pode gerar oferecimentos (semestre e ano). Além disso, um aluno pode se inscrever em um oferecimento de uma disciplina.

# Análise e Projeto de Sistemas II

- Tarefa
  - Acesse o blackboard e leia o texto Conteúdo → Material Auxiliar → Identificação de Classes
  - Através da técnica de Abbott, descrita no texto acima, tente construir o diagrama de classes desse estudo de caso: Prontuário Eletrônico