POO – Relacionamento entre Classes – Associação

Disciplina: Programação I

Tópicos da Aula

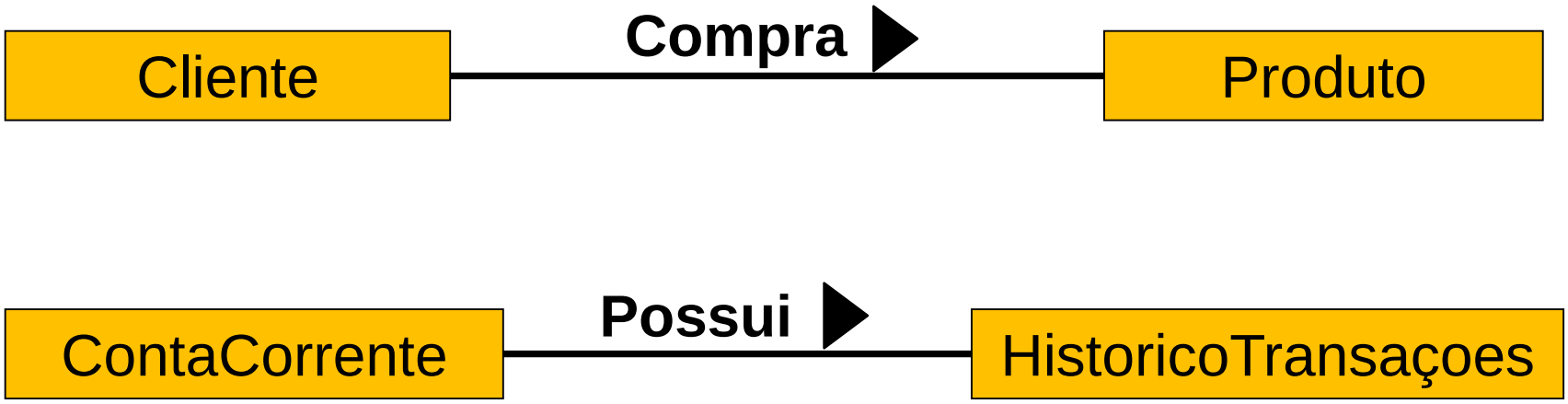
- Relacionamento entre Classes – Associação
  - Definição
  - Cardinalidade / Multiplicidade
  - Conectividade
  - Tipos de Associação
  - Exemplos de Associação Exercícios

Relacionamento entre Classes – Associação

- É uma conexão entre classes.
- Os objetos de uma classe estão ligados a objetos de outras classes, podendo haver troca de informações (mensagens) e compartilhamento de métodos.
- Ocorre normalmente entre duas classes (binária),entre uma classe com ela mesma (unária) e entre várias classes (ternária/N-ária).
- “Equivale” aos relacionamentos E-R.

Relacionamento entre Classes – Associação

- Exemplos



Cardinalidade / Multiplicidade entre Classes

- Consiste na quantidade mínima e máxima de objetos que podem ser conectados pela instancia de uma associação.

- Exemplo:

Cardinalidade / Multiplicidade entre Classes

- Exemplos:

Cardinalidade / Multiplicidade entre Classes

- Exemplos:

  - Pode haver **um cliente** que esteja associado a **vários pedidos**.
  - Pode haver **um cliente** que **não** esteja associado a **pedido** algum.
  - **Um pedido** está associado a **um**, e **somente um**, **cliente**.

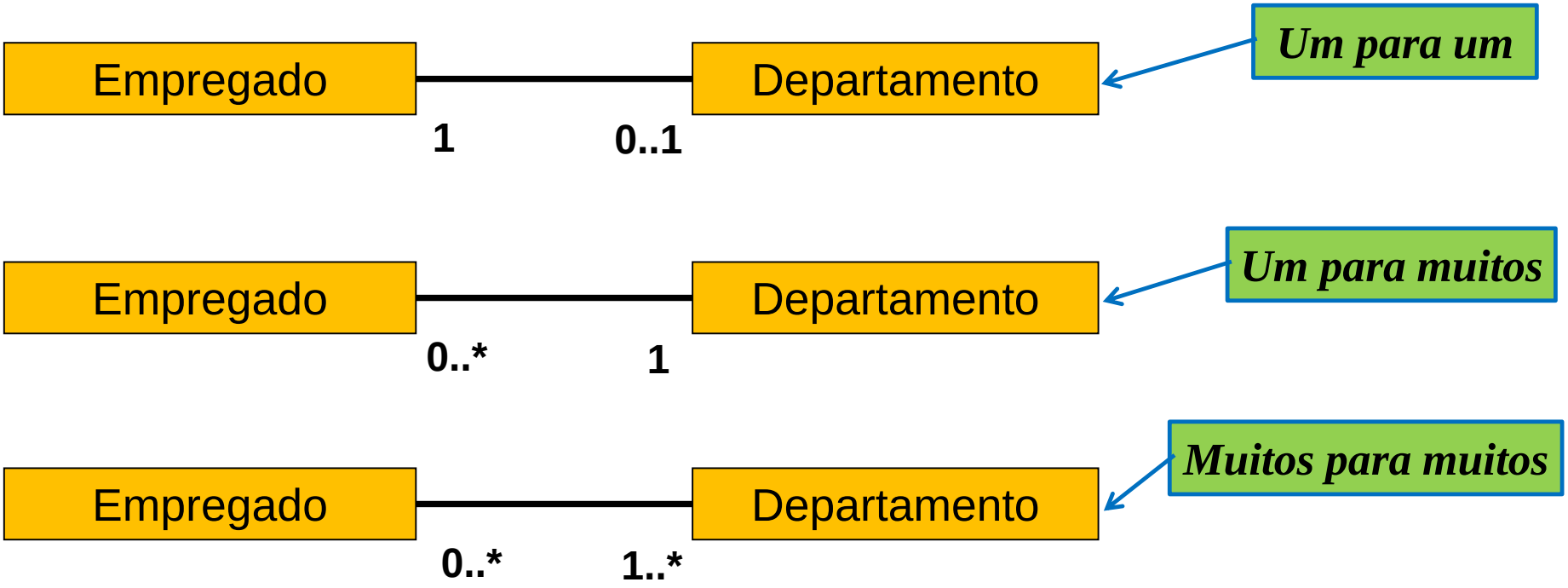
Conectividades entre Classes

- Corresponde ao tipo de associação existentes entre classes:
  - “muitos para muitos”, “um para muitos” e “um para um”.

Conectividade	Em um Extremo	No outro Extremo
Um para um	**0..1** **1**	**0..1** **1**
Um para muitos	**0..1** **1**	* **1..*** **0..***
Muitos para muitos	* **1..*** **0..***	* **1..*** **0..***

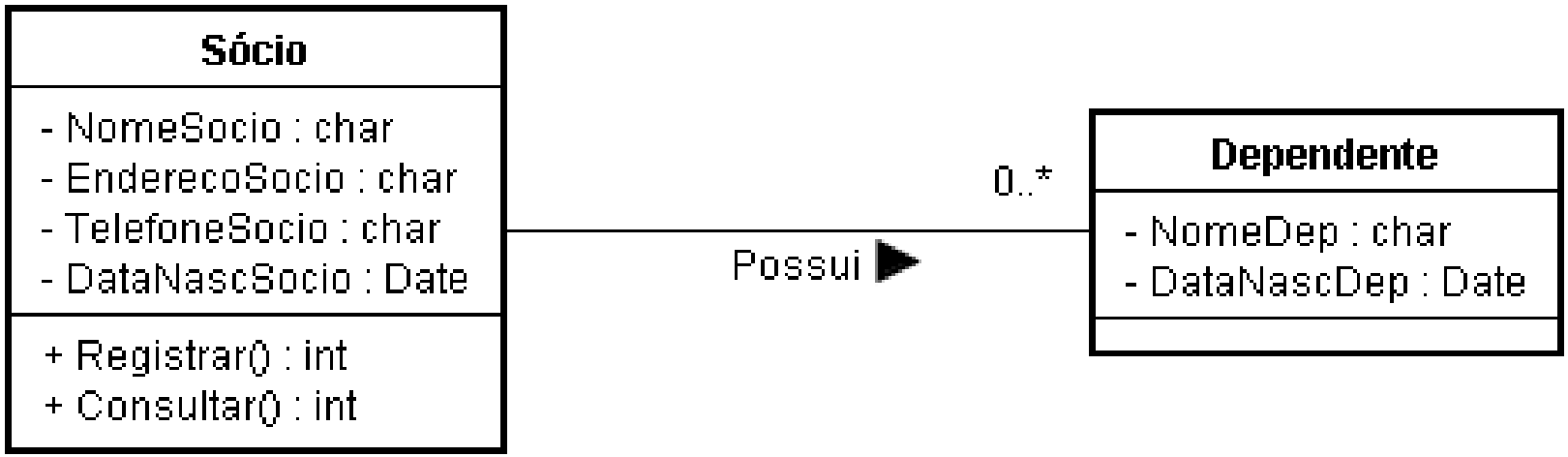
Conectividades entre Classes

- Exemplos:



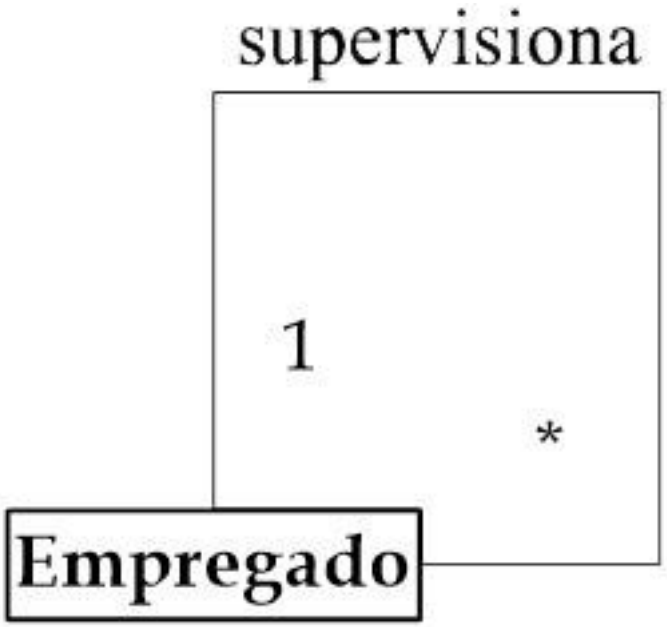
Tipos de Associação

- Associação Binária
  - Associações entre duas classes
  - Mais comum
  - Exemplo:

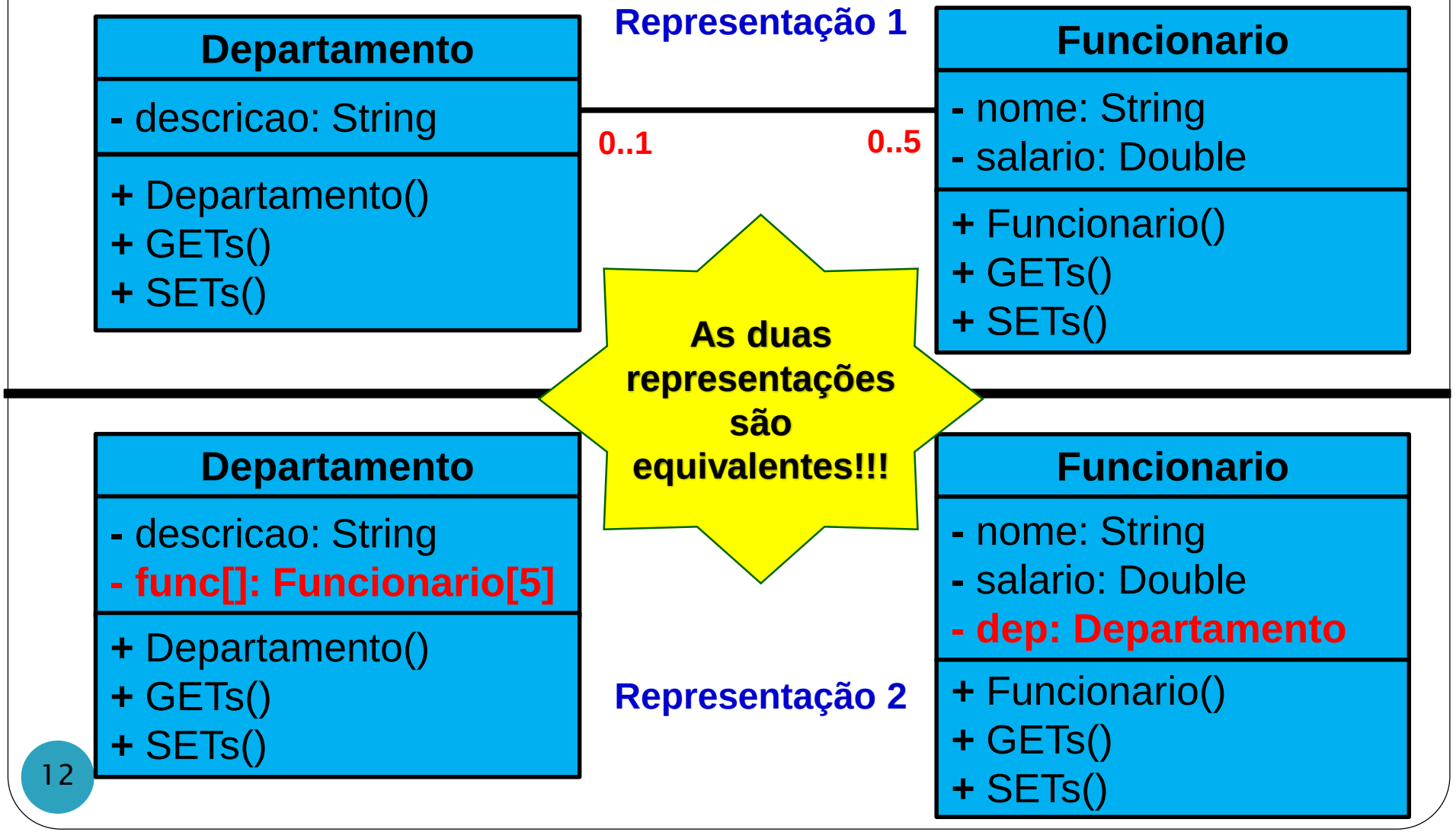


Tipos de Associação

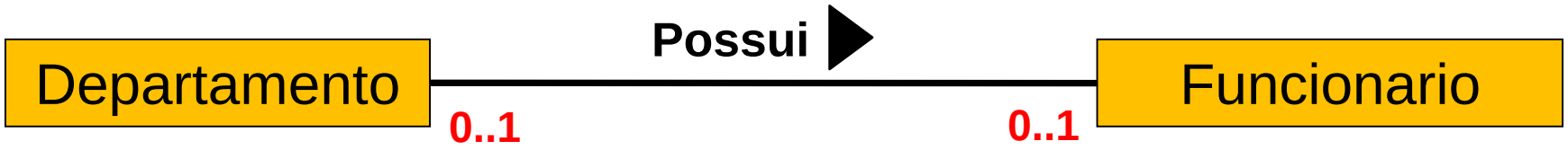
- Associação Unária (ou Reflexiva)
  - Ocorre quando uma classe relaciona com si mesmo.
  - Exemplo:



Representação Gráfica de associação entre Classes



Associação *"um para um"*



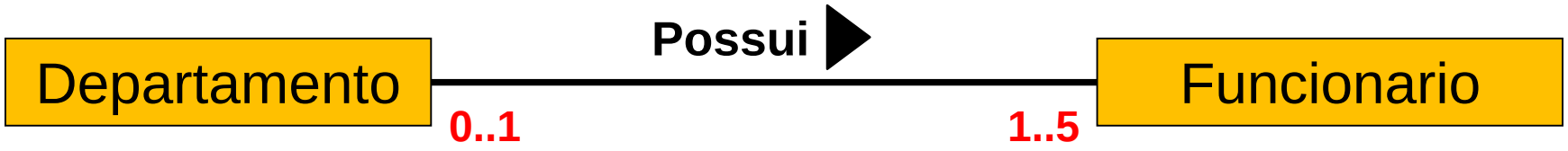
- Departamento

```
public class Departamento{
    private Funcionario func = new Funcionario();
}
```

- Funcionario

```
public class Funcionario{
    private Departamento dep = new Departamento();
}
```

Associação *"um para muitos"*



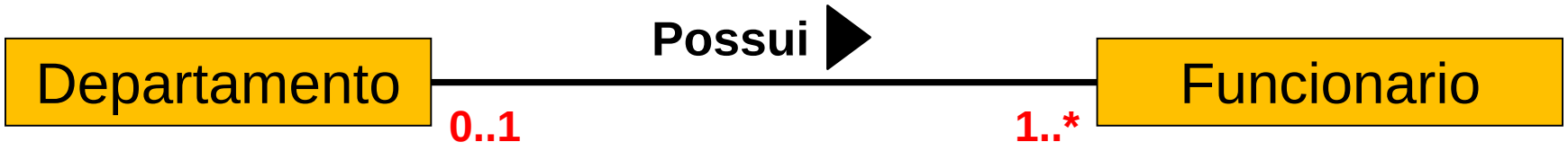
- Departamento

```
public class Departamento{
   private Funcionario[] func = new Funcionario[5];
}
```

- Funcionario

```
public class Funcionario{
   private Departamento dep = new Departamento();
}
```

Associação *"um para muitos"*



- Departamento

```
public class Departamento{
 private ArrayList<Funcionario> funcionarios =
                    new ArrayList<Funcionario>();
}
```

- Funcionario

```
public class Funcionario{
   private Departamento dep = new Departamento();
}
```

Associação *"muitos para muitos"*

- Departamento

```
public class Departamento{
   private Funcionario[] funcs = new Funcionario[5];
}
```

- Funcionario

```
public class Funcionario{
   private Departamento[] deps = new Departamento[2];
}
```

Associação *"muitos para muitos"*

- Departamento

```
public class Departamento{
    private ArrayList<Funcionario> funcs =
            new ArrayList<Funcionario>();
}
```

- Funcionario

```
public class Funcionario{
   private ArrayList<Departamento> deps =
            new ArrayList<Departamento>();
}
```

Dúvidas?

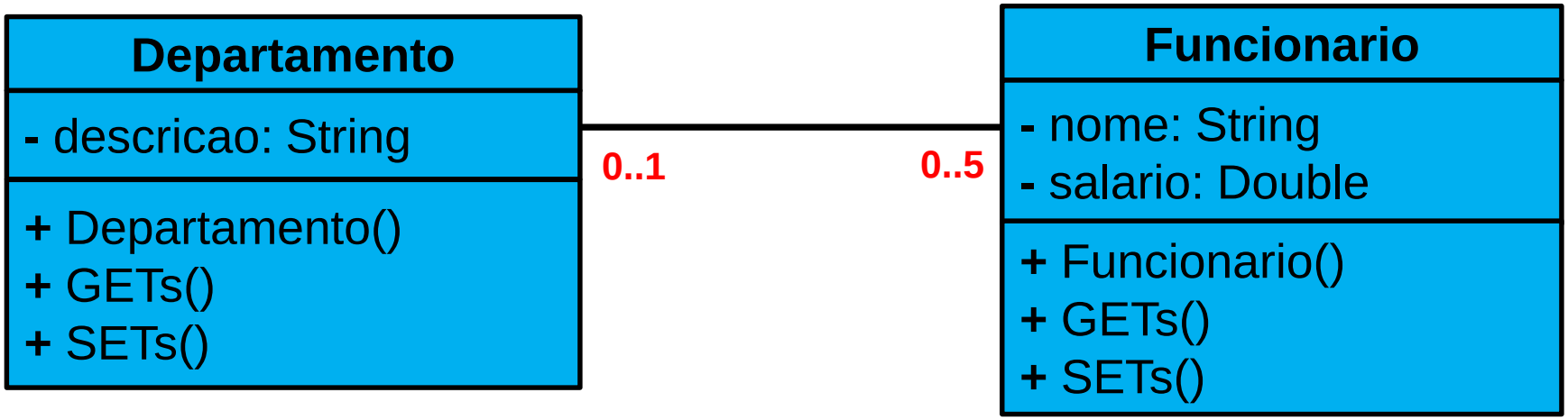
Links Interessantes

- https://plleon.wordpress.com/2009/02/22/associacoes-entre-classes-de-objetos-uml/
- https://www.youtube.com/watch?v=vJvRhQ6Ggt0

Vamos para a Prática!!!

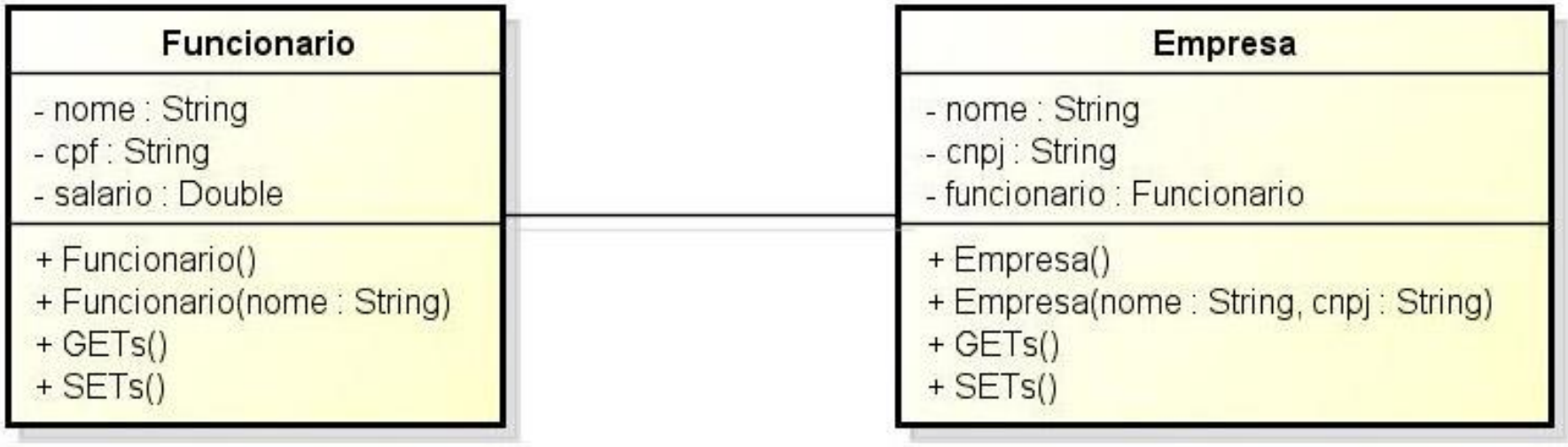
Exemplo Prático

- Escreva um programa em que implemente o Diagrama de Classes apresentado abaixo.



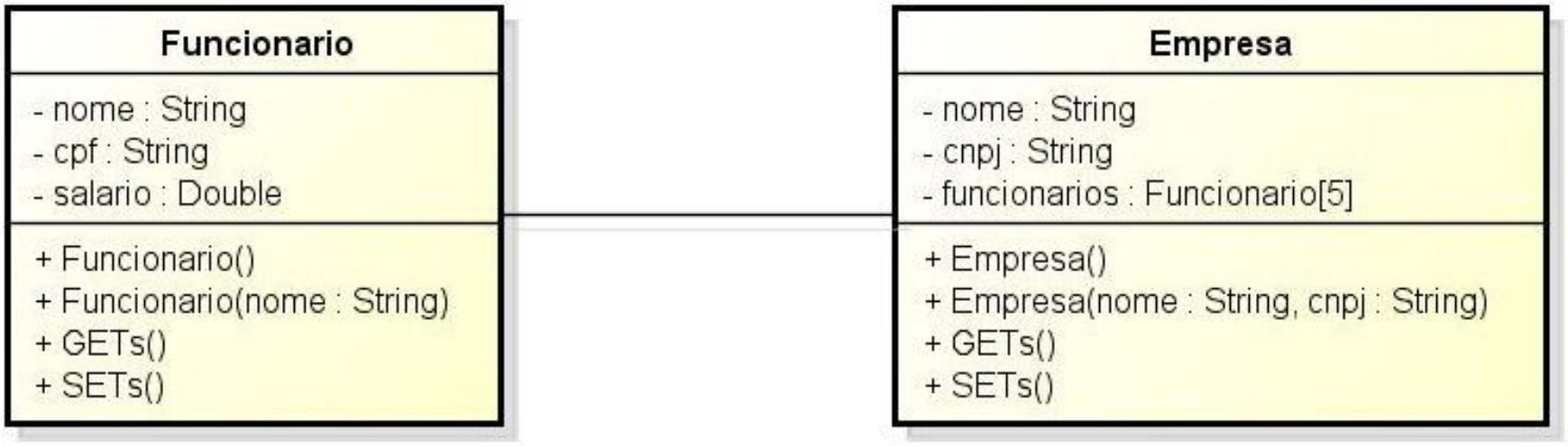
Exercício 01

- Escreva um programa em que implemente o Diagrama de Classes apresentado abaixo.



Exercício 02

- Altere as classes do exercício considerando o Diagrama de Classes abaixo.



Exercício 03

- Desenvolva uma aplicação em utilizando os conceitos da POO, para cadastro e controle de Clientes e seus Pedidos, onde:
  - Um Cliente pode ter, zero, um ou muitos Pedidos;
  - Um Pedido é de um único Cliente.

- Desenvolva uma classe principal contendo o método “*main*” para testar o programa.

Exercício 04

- Desenvolva uma aplicação utilizando os conceitos da POO, para cadastro e controle de Clientes e seus Endereços, onde:
  - Um Cliente possui um ou vários Endereços;
  - Um Endereço pode ter, zero, um ou muitos Clientes.

- Desenvolva uma classe principal contendo o método “*main*” para testar o programa.