Associações entre classes

Paulo Ricardo Lisboa de Almeida

UDESC
UNIVERSIDADE
DO ESTADO DE
SANTA CATARINA

Associações

- Lembrem-se que começamos a construir relações entre nossas classes
  - A classe *Disciplina* possui
    - Um objeto pessoa para representar o professor
    - Vários objetos pessoa para representar os alunos
- O que fizemos com as classes foi uma **associação**

Relações entre classes

- As classes podem se relacionar de diversas formas
  - **Associações** ← Tema da aula de hoje
  - Agregações fracas
  - Agregações fortes (composições)
  - Herança
  - Dependência

Antes de avançar

- Antes de avançar nos slides, leia o seguinte
  - Seção 10.5 do livro “Pressman, R.;Maxim, B. **Engenharia de Software: uma abordagem Profissional.** McGraw Hill Brasil, 2016. 8 ed. ISBN 9788580555349.”
    - A seção está **disponível gratuitamente** em books.google.com

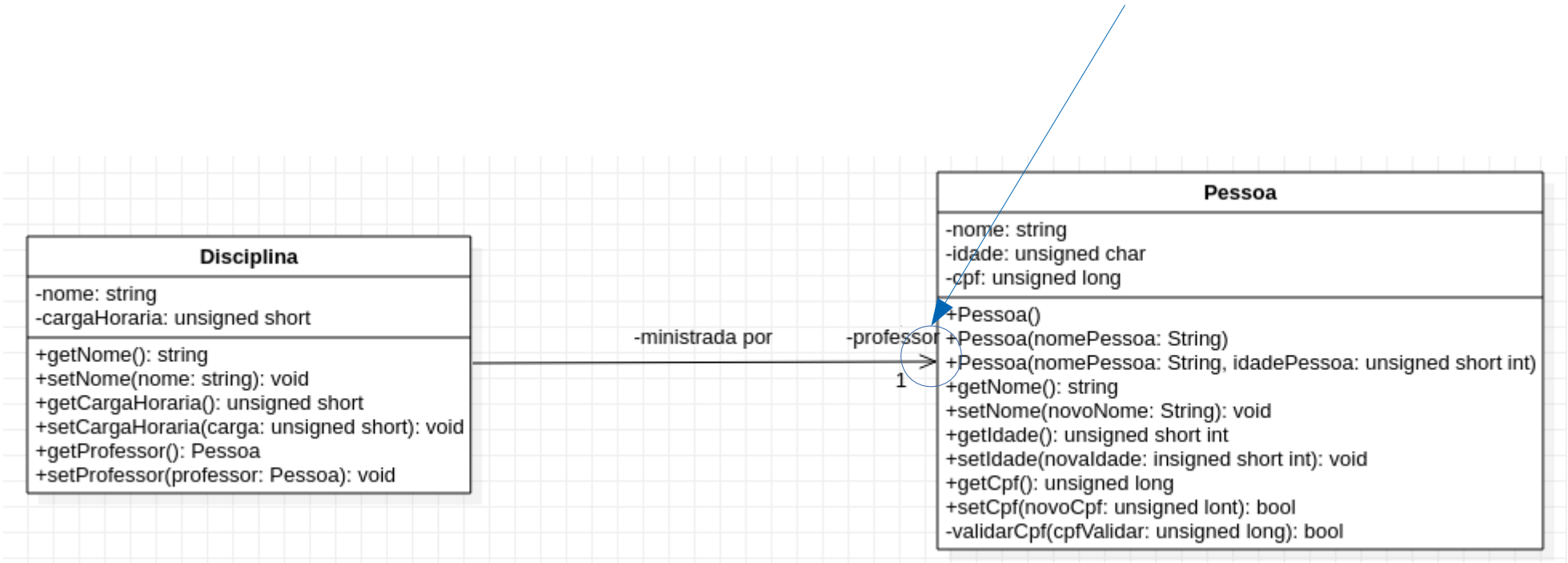
Propriedades

- Algumas propriedades de uma **associação**
  1. O dado membro associado (Pessoa) não tem relação direta com a classe (Disciplina)
  2. O dado membro associado (Pessoa) pode pertencer a mais de uma classe ao mesmo tempo
     - Por exemplo, a mesma pessoa é professor da *Disciplina x*, e também é coordenador do *Laboratório y*
  3. A classe (Disciplina) não gerencia a existência do dado membro
     - A pessoa que representa o professor da disciplina existe independentemente da classe e objetos *Disciplina*
  4. O dado membro associado (Pessoa) **pode ou não** saber da existência da classe (Disciplina)
     - Exemplo: a classe Disciplina tem um ponteiro para o professor, mas a classe Pessoa **pode ou não** ter ponteiros para as disciplinas do professor

No Diagrama de Classe UML

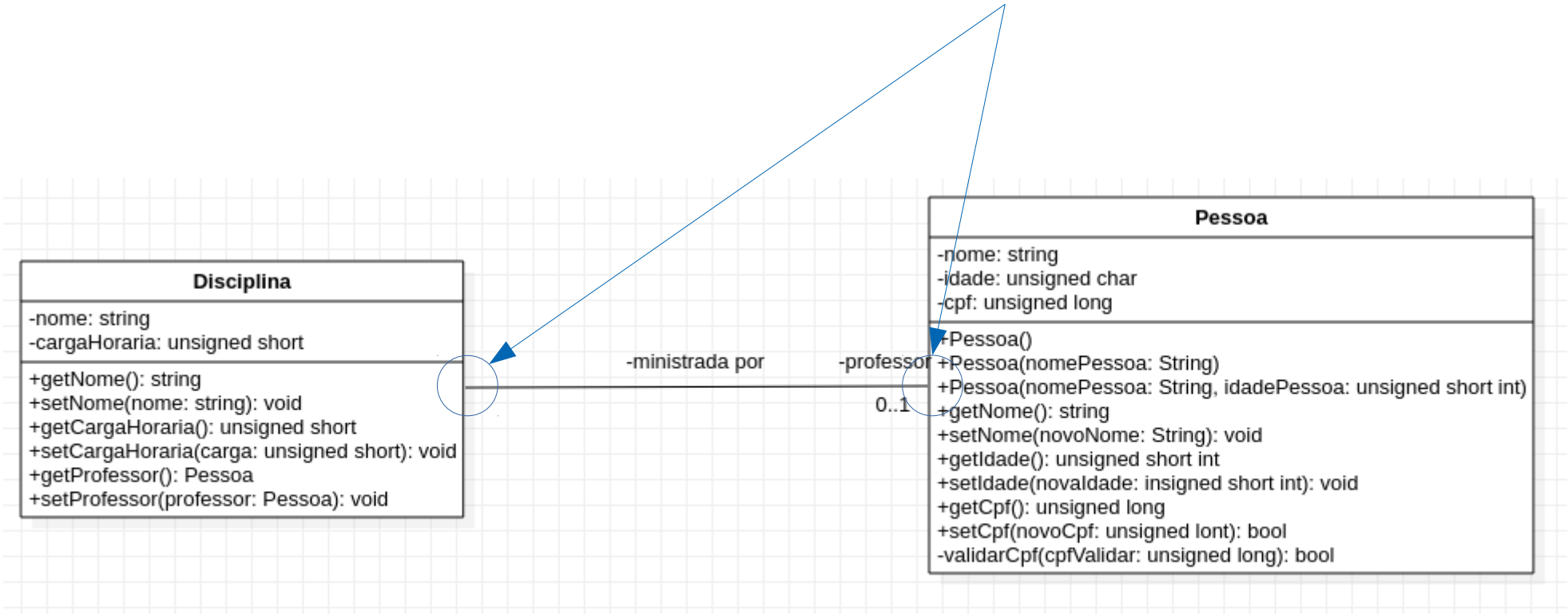
Associação Direcionada

A seta significa uma **associação direcionada.** Nesse caso, a classe Disciplina “Conhece” a classe pessoa, mas a classe Pessoa não Conhece a Disciplina. Note que isso está de acordo com o item 4 das “Propriedades”.



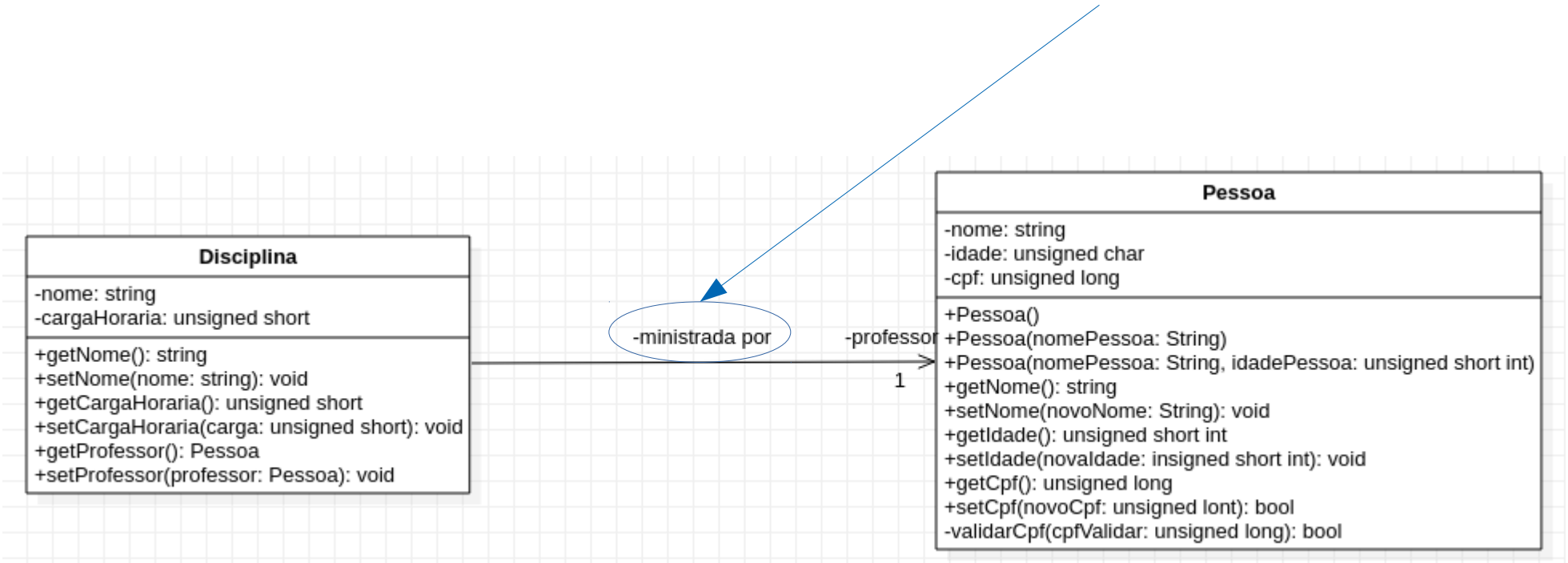
Associação

Esse é um exemplo de associação não direcionada, onde os dois lados “se conhecem”. Note a falta de setas. No momento vamos criar apenas **associações direcionadas**.



Papel

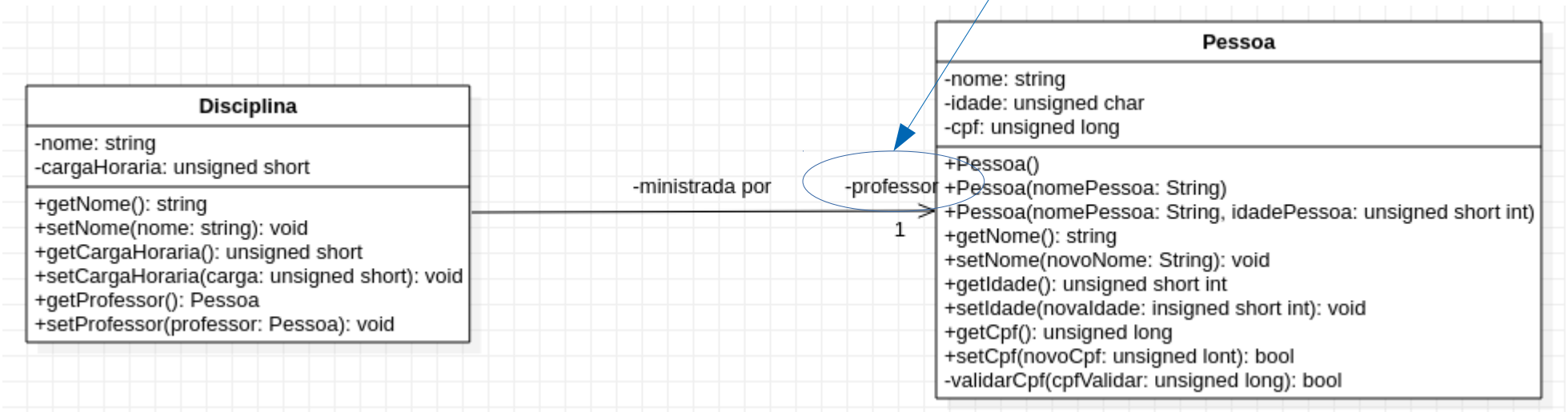
Aqui especificamos o **Papel**. Indica qual a função do objeto associado na classe.
O **papel** é um item **opcional**, mas que pode nos ajudar a entender melhor as relações



UDESC

Vamos por partes

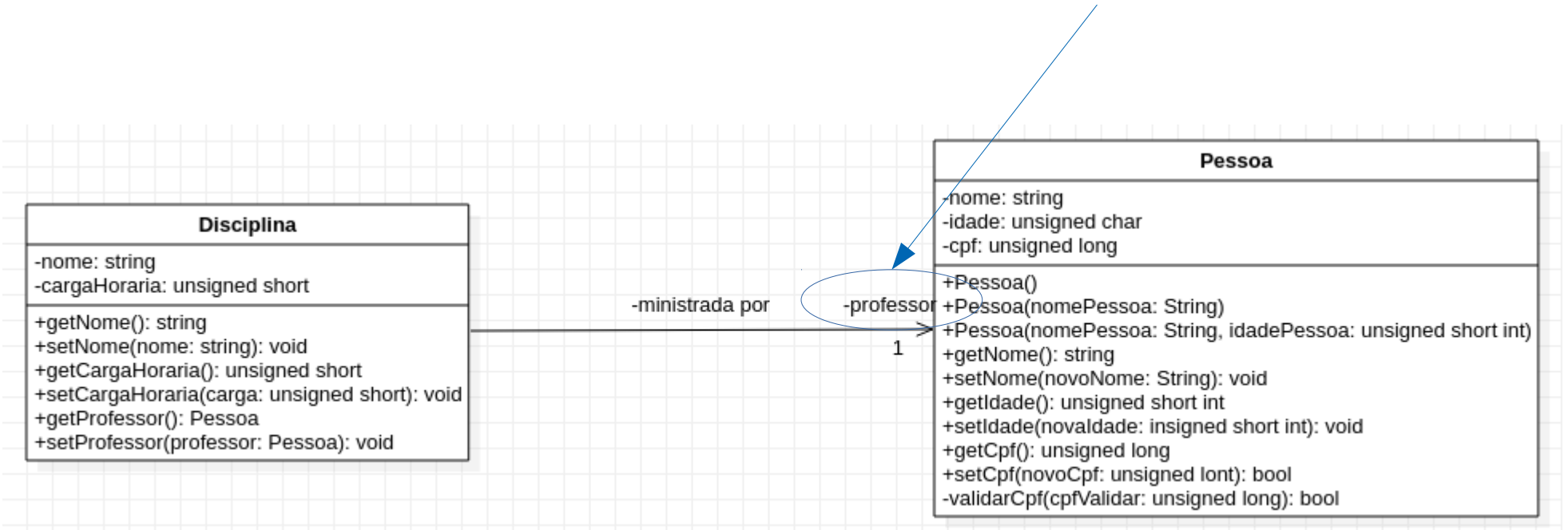
Aqui podemos especificar o nome do dado membro que vai realizar a associação. Note que **não precisamos** especificar o seu tipo, já que o tipo é implicitamente definido pela associação.



UDESC

Vamos por partes

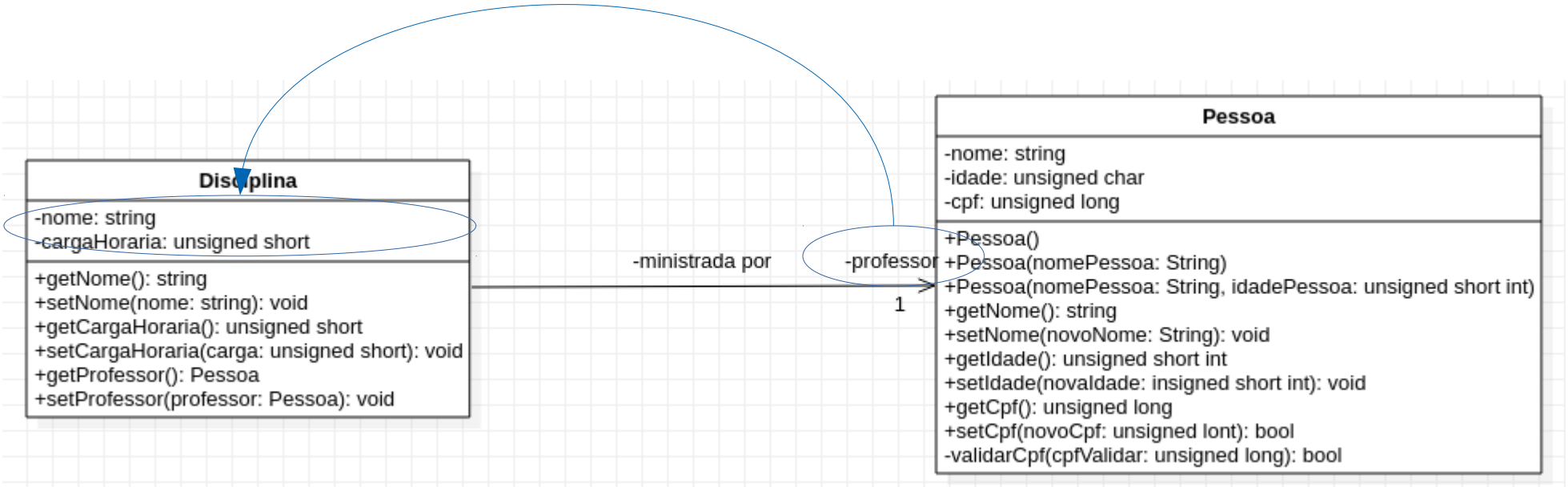
Geralmente não é possível definir que, por exemplo, temos um ponteiro para pessoa, e não uma pessoa. Se é um ponteiro ou não o programador vai precisar definir de acordo com a associação. Isso se dá pelo fato do Modelo UML ser agnóstico de linguagem (por exemplo, Java não tem ponteiros).



UDESC

Vamos por partes

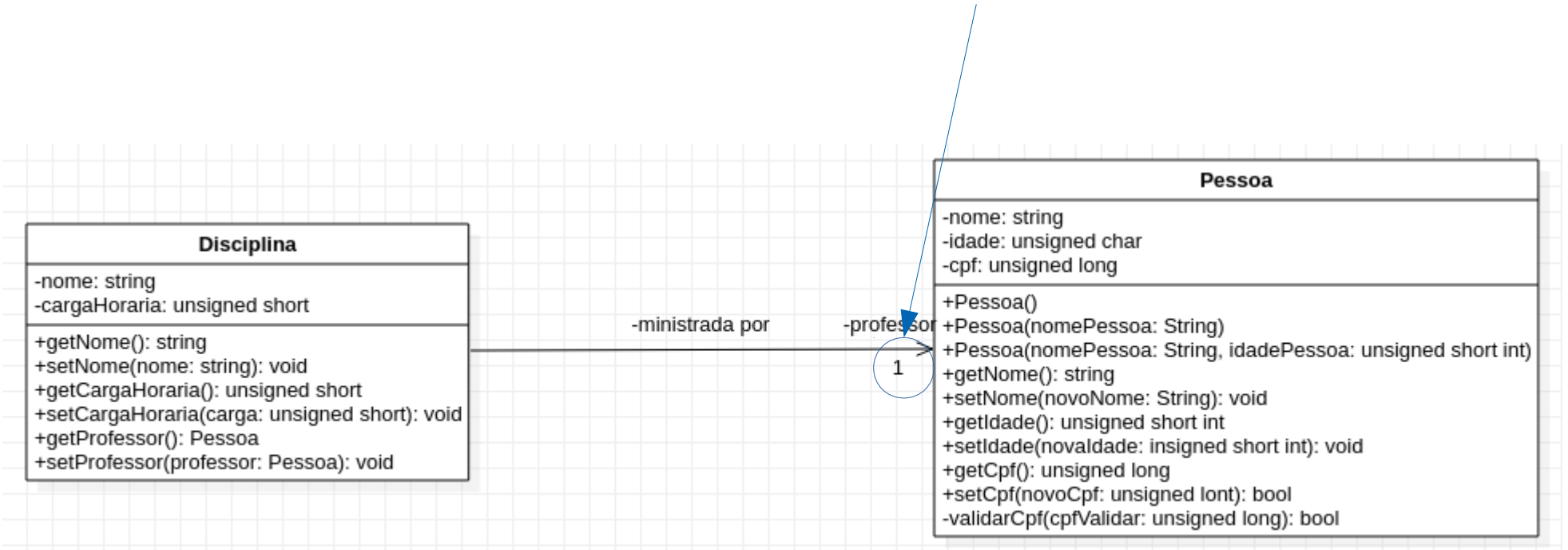
Não temos explicitamente dado membro professor em Disciplina. Esse dado membro **existe,** mas é definido de acordo com a associação (o programador deve olhar para a associação e criar esse dado membro de acordo com ela).



UDESC

Vamos por partes

Essa é a **multiplicidade** da relação. Está indicando que toda disciplina possui **exatamente 1** professor.

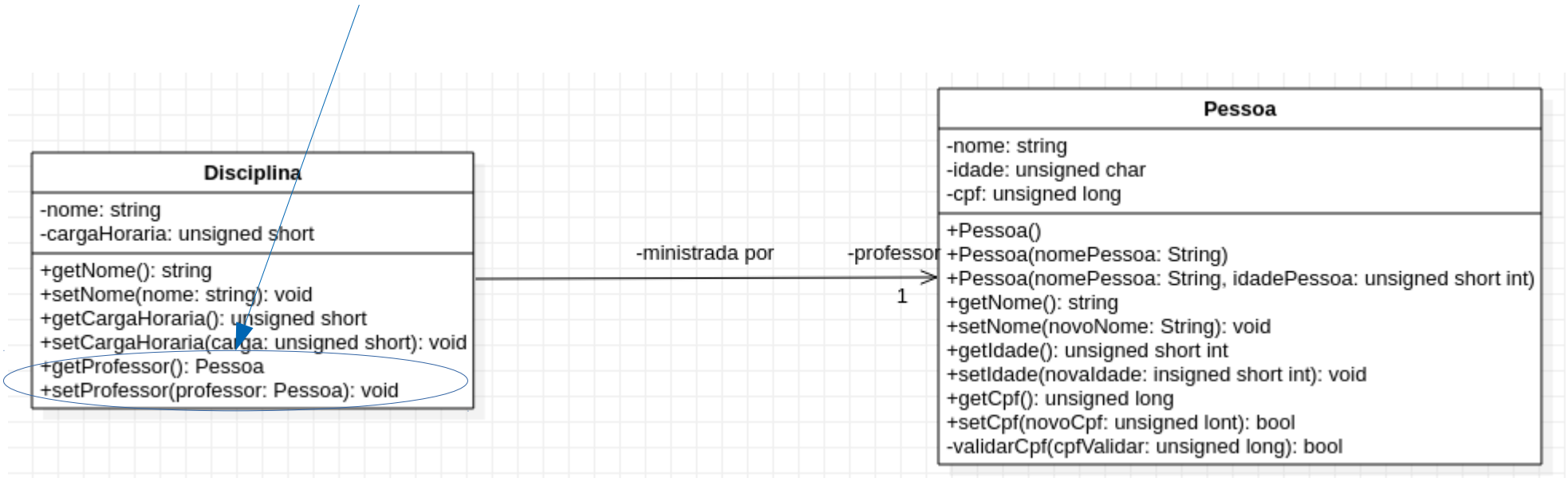


Multiplicidades válidas

- Considere *m* e *n* como números naturais
- A multiplicidade pode ser
  - n → Relacionamento com **exatamente *n*** objetos
    - Exemplo 5 → Relacionamento com exatamente 5 objetos
  - 0 .. 1 → Relacionamento com **0 ou 1** objetos
  - n .. m → Relacionamento com um valor **entre *n* e *m*** objetos
    - Exemplo 2 ..4 → Relacionamento com 2, 3 ou 4 objetos
  - 0 .. * → Relacionamento com um valor **entre 0 e infinitos** objetos
  - 1 .. * → Relacionamento com um valor **entre 1 e infinitos** objetos

Vamos por partes

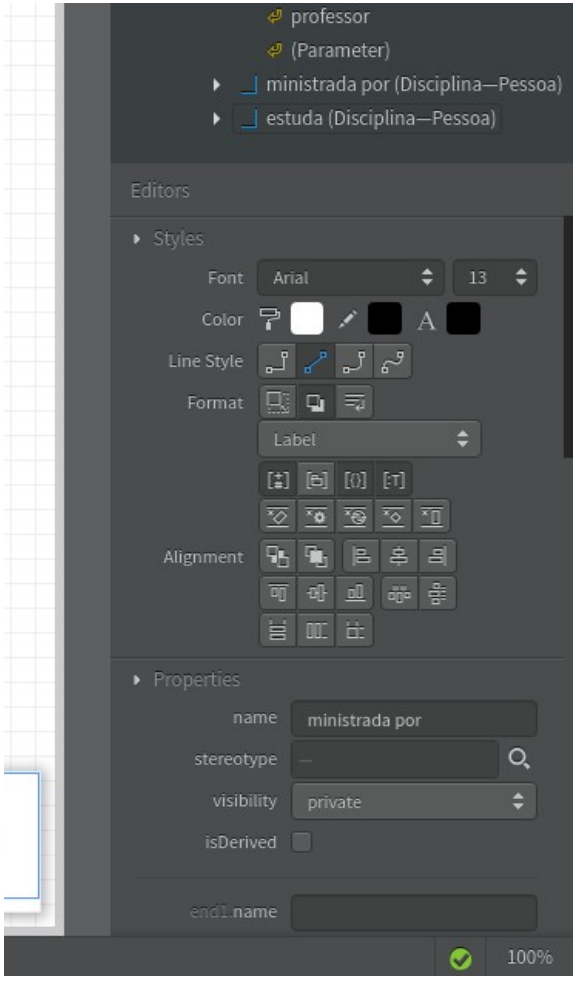
Note que ainda podemos definir os gets e sets



UDESC

StarUML

- Para editar as propriedades da associação no StarUML, clique na associação criada, e edite no menu que aparece no canto inferior direito

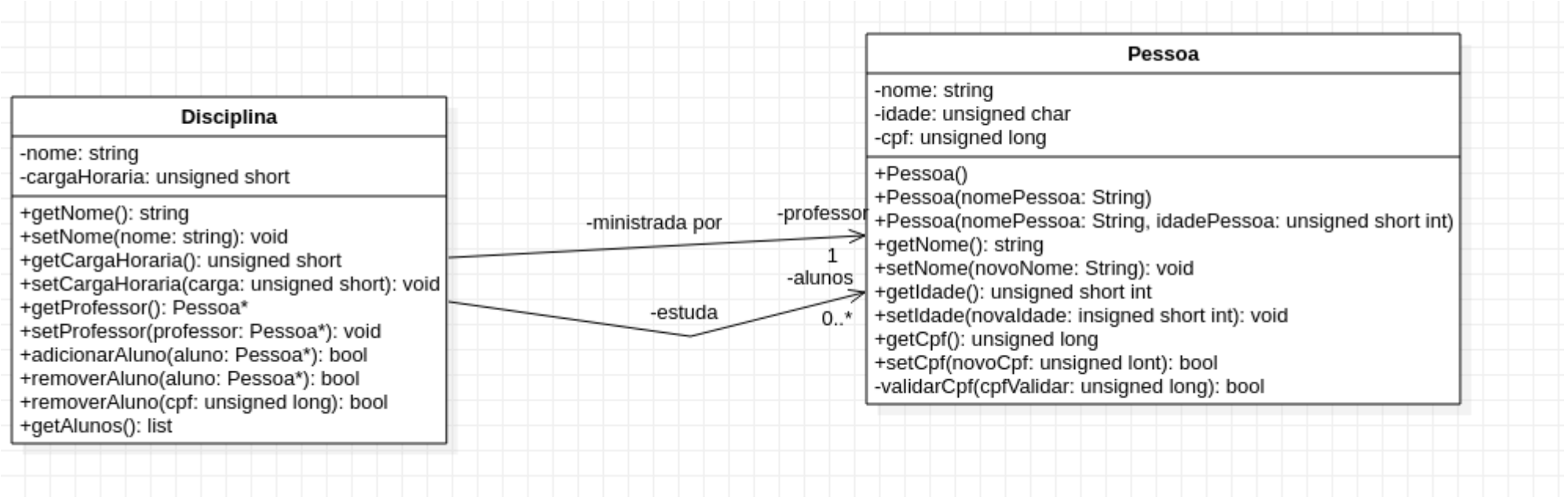


UDESC

Alunos

- Note que temos duas associações com Pessoa
  - Uma para professor, e uma para os alunos
  - O processo para criar a associação “alunos” é o mesmo

Alunos

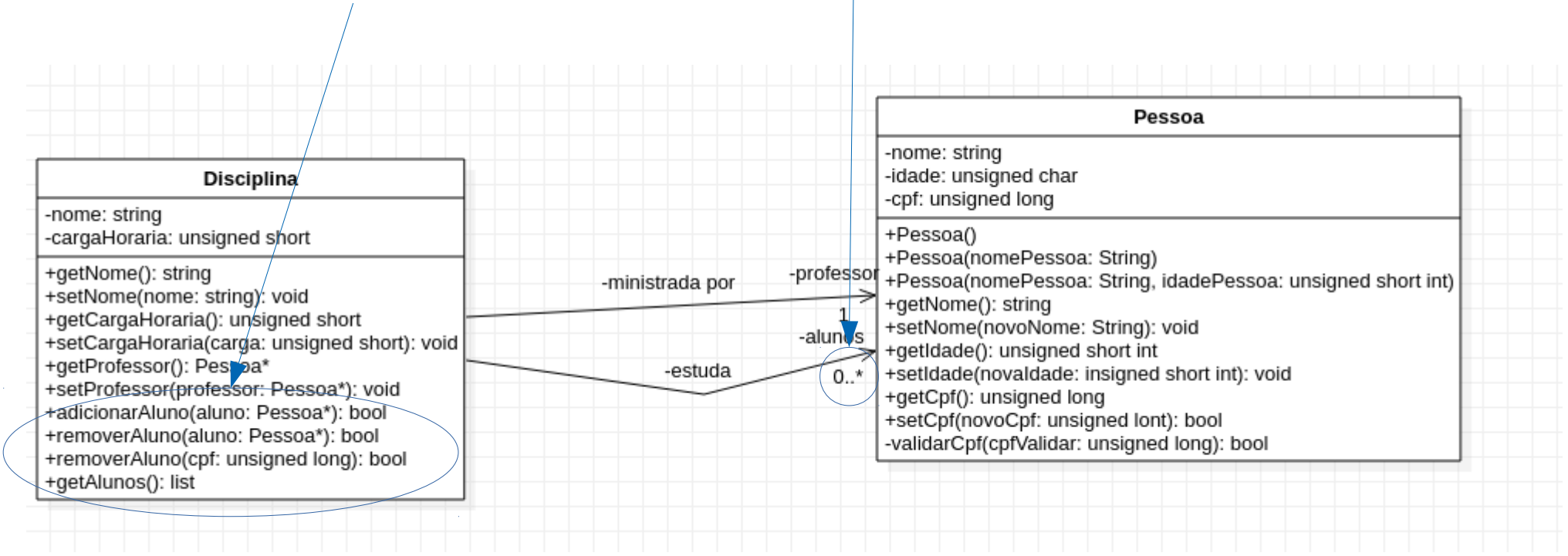


UDESC

Alunos

Uma disciplina tem entre 0 e infinitos alunos

Funções membro que lidam com os alunos



UDESC

Exercícios

1. Adicione no **diagrama de classe** uma associação associação entre Curso e Disciplina (se você não criou uma classe Curso ainda, crie agora)
   - Faça uma associação direcionada
   - O curso conhece suas disciplinas, mas a disciplina não conhece o curso a qual pertence
   - Adicione as seguintes funções membro em Curso
     - adicionarDisciplina(Disciplina* disciplina);
     - removerDisciplina(Disciplina* disciplina);
2. Implemente essa associação no Programa

Referências

- DEITEL, P.; DEITEL, H. **C++ how to Program**. [S.l.]: Pearson, 2017. ISBN 9780134448237
- STROUSTRUP, B. **The C++ Programming Language**. Pearson Education, 2013. ISBN 9780133522853.
- https://www.learncpp.com/cpp-tutorial/10-4-association/
- Pressman, R.;Maxim, B. **Engenharia de Software: uma abordagem Profissional.** McGraw Hill Brasil, 2016. 8 ed. ISBN 9788580555349.