



**Professor:** Edwar Saliba Júnior

### Exercícios:

1. Desenvolva um *software* para controle de livros. Este *software* deverá ser capaz de armazenar, em memória principal, 300 livros. Para isto, faça uso de vetor. O seu *software* deverá possuir a classe Livro, com os seguintes atributos: título (*String*), autor (*String*), editora (*String*), ano da publicação (*int*), edição (*int*) e isbn<sup>1</sup> (*String*). Para facilitar a utilização do *software*, por parte do usuário, construa um *menu* que permita ao mesmo a: inclusão, exclusão, alteração, consulta ou impressão de relatório dos livros.
2. Necessita-se de *software* para controle de funcionários. Este *software* deverá possuir as seguintes classes: Funcionário, com seguintes atributos: cpf (*String*), nome (*String*), salário (*float*), identidade (*String*) e filhos[ ] (Filho); e também a classe Filho, com os seguintes atributos: cpf<sup>2</sup> (*String*), nome (*String*), número da certidão de nascimento (*String*) e data de nascimento (*Date*), conforme Figura 1. Lembre-se, um funcionário poderá ter diversos filhos, assim sendo o atributo “filhos” deverá ser um vetor de, no mínimo, 10 posições. Construa um *software* com uma estrutura que permita o cadastro de 50 funcionários e com um *menu* que possibilite ao usuário: incluir, excluir, alterar, consultar e imprimir relatório de: Funcionários e de Funcionários com seus respectivos Filhos.

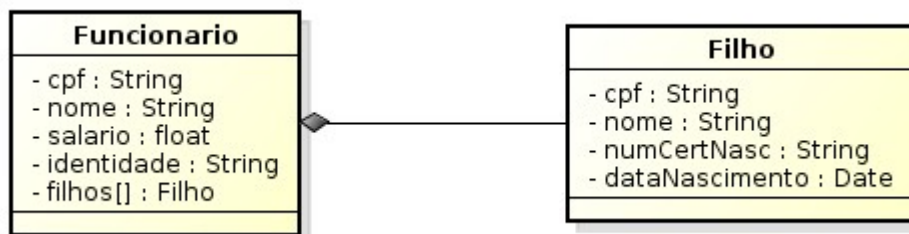


Figura 1: Diagrama de Classe - Funcionário - Filho.

3. Construa um *software* para controle de cobrança e entrega de produtos. Este *software* deverá possuir duas classes: Cliente, com os seguintes atributos: cpf (*String*), nome (*String*), valor (*float*), data (*Calendar*), endereço de cobrança (Endereço) e endereço de entrega (Endereço) e também a classe Endereço, com os seguintes atributos: logradouro (*String*), número (*int*), complemento (*String*), bairro (*String*), município (*String*), estado (*String*) e cep<sup>3</sup> (*String*), conforme mostrado na Figura 2. Seu *software* deverá ser capaz de cadastrar, em memória principal, no mínimo 200 Clientes. Para facilitar a sua utilização, pelo usuário, construa um *menu* que possibilite ao usuário: incluir, excluir, consultar, alterar ou imprimir relatório.

1 *International Standard Book Number*, do inglês, Padrão Internacional de Numeração de Livro.

2 Cadastro de Pessoa Física - número atribuído ao cidadão pela Receita Federal Brasileira.

3 Código de Endereçamento Postal - número atribuído a locais pelos Correios.

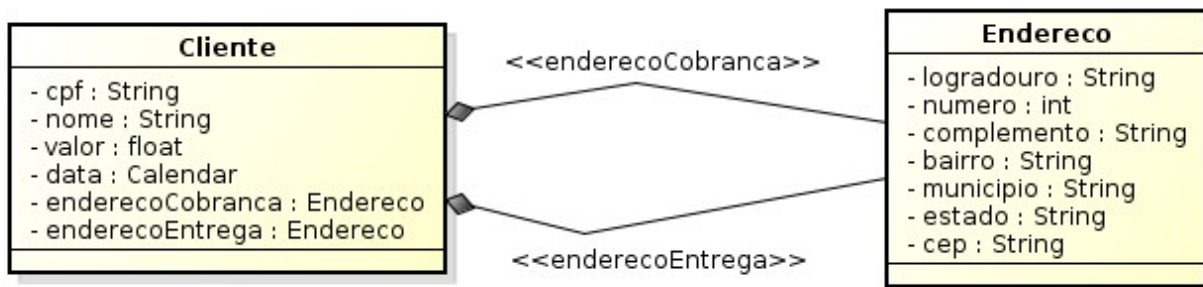


Figura 2: Diagrama de Classe - Cliente - Endereço.

4. Em um Sistema de Agendamento de Consultas de uma Clínica Médica, considere as classes Médico, Consulta e Paciente. A classe Médico possui os seguintes atributos: *crm* do tipo (*int*) e *nome* (*String*), a classe Paciente possui: *cpf* (*String*), *nome* (*String*), *endereço* (*String*) e *telefone* (*String*). Já a classe Consulta possui: *médico* (*Médico*), *paciente* (*Paciente*), *data* (*LocalDate*) e *hora* (*LocalTime*), conforme mostrado na Figura 3. Construa um *software* que seja capaz de registrar, em memória principal, até 1000 consultas, 30 médicos e 100 pacientes. Para que seu software fique organizado, então, utilize as classes de gerenciamento (vulgarmente conhecida como “fichários”). Construa um *menu* com as opções de: inclusão, exclusão, alteração, consulta e relatório; para facilitar a utilização do *software* por parte do usuário.

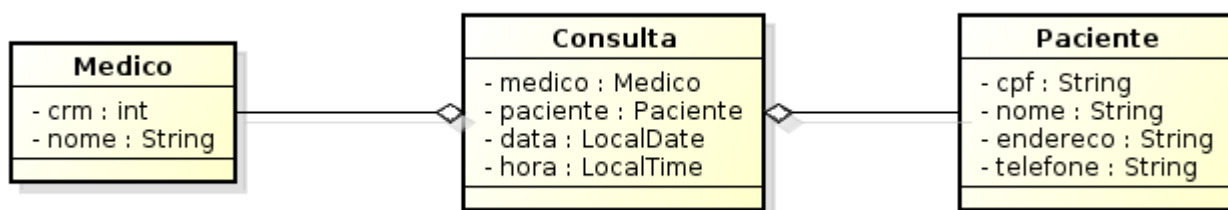


Figura 3: Diagrama de Classe - Médico - Paciente - Consulta.

### Observação:

Em Java, antes da versão 8, haviam duas maneiras de se trabalhar com datas, ou seja, podíamos usar o tipo *Date* já ultrapassado ou o tipo *Calendar*, este, um pouco mais recente, mas ainda assim meio chato de se trabalhar. Já na versão 8 do Java, foi incorporado o tipo *LocalDate*, ou seja, uma nova maneira de se trabalhar com datas em Java.

Para que você esteja ciente e saiba trabalhar com estas três maneiras existentes, os exercícios 2, 3 e 4 contemplam o uso destes três tipos de “datas” que a linguagem Java possui.

Para auxiliá-los neste primeiro momento, deixo-lhes quatro referências para serem estudadas:

FERREIRA, R.; AQUILES, A. **Conheça a Nova API de Datas do Java 8**. Disponível em: <<http://blog.caelum.com.br/conheca-a-nova-api-de-datas-do-java-8/>>. Acesso em: 05 mar. 2015.

ORACLE and/or its affiliates. **Package java.time - The main API for dates, times, instants, and durations**. Disponível em: <<https://docs.oracle.com/javase/8/docs/api/java/time/package-summary.html>>. Acesso em: 11 maio 2021.

SALIBA JÚNIOR, E. **Exemplo de Utilização das Classes Calendar e Date**. Disponível em: <<http://www.esj.eti.br/Apostilas/Programacao/Java/DataEmJava.pdf>>. Acesso em: 05 mar. 2015.

VINICIUS, Caio. **Utilizando LocalDate, LocalDateTime e LocalTime na Prática**. Disponível em: <<https://blog.cvinicius.com.br/2018/08/utilizando-localdate-localdatetime-e.html>>. Acesso em: 11 maio 2021.