UNICAMP

UNIVERSIDADE ESTADUAL DE CAMPINAS
INSTITUTO DE COMPUTAÇÃO

INSTITUTO DE COMPUTAÇÃO

Laboratório 4

Seguradora - Menu Interativo

MC322 - Programação Orientada a Objetos

---

1 Descrição Geral

Nas atividades deste laboratório, iremos explorar novos conceitos de Orientação Objetos vistos em classe, tais como: utilização de enum, classes estáticas e polimorfismo. Esses conceitos proporcionarão maior robustez ao Sistema de Seguradora. Para ilustrar as novas implementações que estão inclusas neste laboratório, a Figura 1 apresenta o diagrama de classes[1]

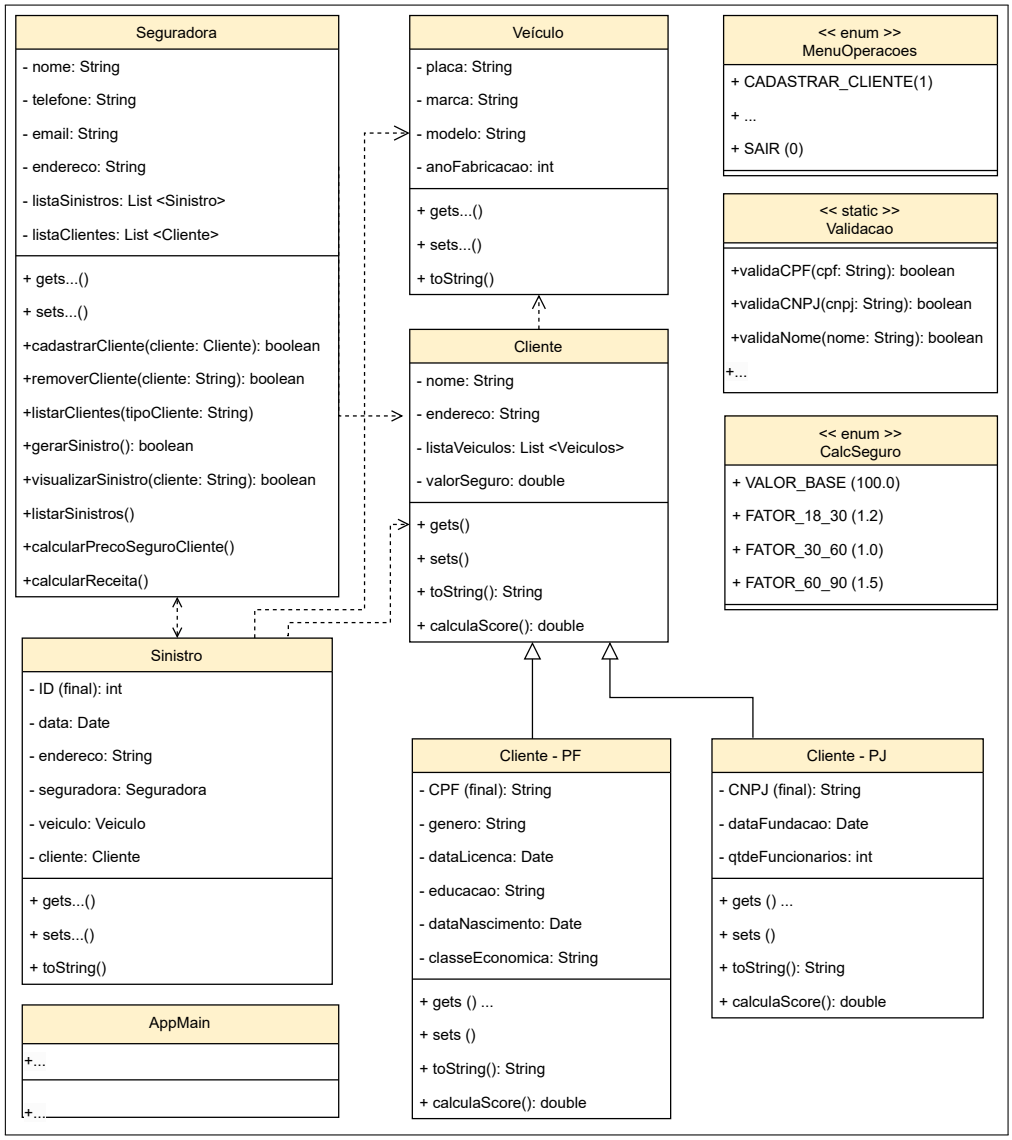


Figura 1: Diagrama de Classe - Sistema da Seguradora - Menu Interativo e Validacoes

Observando o diagrama, três novas classes foram inseridas: MenuOperacoes (Enum); CalcSeguro (Enum);

[1]Note que algumas das funcionalidades requeridas não estão explicitamente demonstradas no diagrama de classes. Tal abordagem visa um dos objetivos desse laboratório, o qual é a ampliação da capacidade de abstração por parte dos alunos em como aplicar os conceitos visto em aula para a resolução de problemas.

e Validacao (estática). MenuOperacoes é responsável por armazenar as constantes referentes ao menu interativo do seu programa; CalcSeguro armazena as constantes de valores para o cálculo do seguro. Abaixo segue um exemplo base para o Enum MenuOperacoes.

```java
public enum MenuOperacoes {
  CADASTRAR(1),
  EXCLUIR(2),
  SAIR(0);

  public final int operacao;

  MenuOperacoes(int operacao) {
    this.operacao = operacao;
  }

  public int getOperacao() {
    return this.operacao;
  }
}
```

Listing 1: MenuOperacoes.java

Com base no Laboratório anterior (03), nesse laboratório utilizaremos a mesma base de classes, contudo com novas três principais funcionalidades, sendo elas:

1. **Construção de um menu interativo** via teclado que possa fazer a realização de todas as principais operações do programa;
2. **Verificações das informações inseridas** no menu interativo. Seu programa deve assegurar a validade das informações. Além do CPF e CNPJ, seu programa, por exemplo, deve garantir que o nome do cliente contenha apenas letras;
3. **Cálculo do seguro:** após o Cliente ser cadastrado na Seguradora, ele deve receber o valor do seguro. Para o cálculo do valor, será levado em conta inicialmente os critérios de idade (ClientePF) e número de funcionários (ClientePJ). Além disso, o número de Sinistros envolvidos a esse Cliente será levado em conta.

2 Objetivos

Os objetivos principais do Laboratório 4 são os seguintes:

- Consolidação dos conteúdos vistos nos labs anteriores;
- Utilização de Enum;
- Utilização de classe estática;
- Aplicação do conceito de polimorfismo e sobrecarga de métodos;
- Capacidade de abstração de como aplicar os conceitos de orientação objetos em face das funcionalidades requeridas.

3 Atividades

As atividades a serem desenvolvidas para este Laboratório são as seguintes:

- Implementação das funcionalidades citadas anteriormente (construção de um menu iterativo; verificação das informações inseridas; cálculo do seguro);
  - Para o **Menu Interativo**, seu programa deve prover as funcionalidades demonstradas na Figura 2. Note que essa forma de organização de menu é apenas uma sugestão. Contudo, as funcionalidades do menu da Figura 2 são mandatórias.
- Atualização das classes Seguradora, Veiculo, Sinistro e Cliente com os novos métodos e atributos apresentados na Figura 1;
- Implementação do conceito de Polimorfismo através da implementação do método *calculaScore()* para as classes ClientePF e ClientePJ (veja a seção 4);

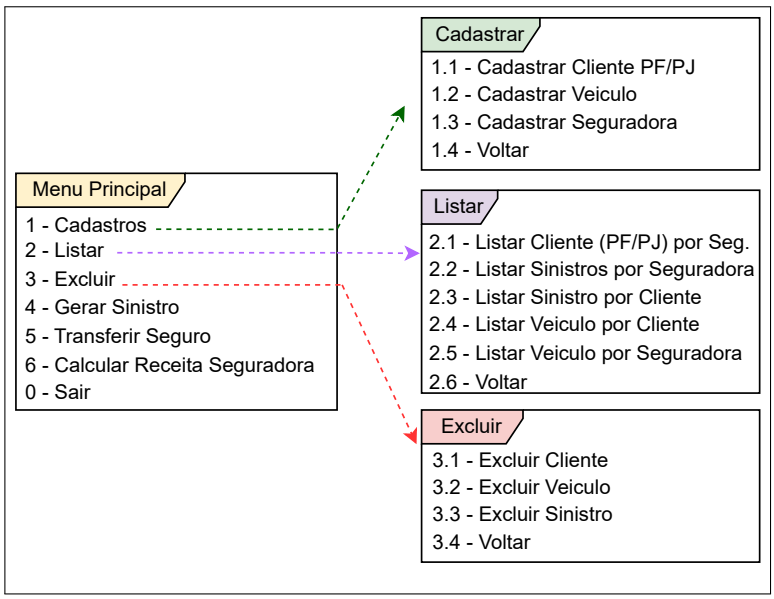


Figura 2: Menu Interativo - Sugestão

- Implementação do conceito de enumeradores através da implementação da classe MenuOperacoes e CalcSeguro;
- Implementação do conceito de classe estática através da implementação da classe Validacao.

Na classe AppMain:

- Instanciar pelo menos 2 objetos da classe Veiculo, 1 objeto da classe ClientePF, 1 objeto da classe ClientePJ, 1 objeto da classe Seguradora;
- Adicionar pelo menos 1 Veiculo em cada Cliente instanciado;
- Cadastrar pelo menos 1 ClientePF e 1 ClientePJ na Seguradora;
- Gerar pelo menos 2 objetos Sinistro;
- Chamar os métodos da classe Seguradora: *listarClientes()*; *visualizarSinistro()*; *listarSinistros()*; e *calcularReceita()* (veja a seção 4);
- Atualizar o atributo valorSeguro de cada cliente cadastrado na seguradora utilizando o método *calcularPrecoSeguroCliente()* da classe Seguradora;
- Mostrar na tela a receita total da seguradora utilizando o método *calcularReceita()*;
- Implementar uma função para criar o menu de operações usando o enum MenuOperacoes;
- Apenas ao final da execução do projeto: chamar o menu de operações.

4 Observações

Para cálculo do valor do seguro, é preciso utilizar o conceito de sobrecarga de métodos.

Para clientes do tipo PF o valor do *calculaScore()* será dado por:

```
1 VALOR_BASE * FATOR_IDADE * quantidadeCarros
```

,

Para clientes do tipo PJ o valor do *calculaScore()* será dado por:

```
2 VALOR_BASE * (1 + (quantidadeFunc)/100) * quantidadeCarros
```

,

Para o método *calcularPrecoSeguroCliente()*, da classe Seguradora o retorno deverá ser dado por:

```
calculaScore() * (1 + quantidade_de_sinistros)
```

O método *calcularReceita()* será utilizado para mostrar o balanço de seguros de todos os clientes da Seguradora. Lembre-se que o *calcularPrecoSeguraCliente()* já calcula o preço de seguros de cada cliente, então basta você iterar nesses valores e somar.

Transferência de Seguro: A operação de Transferência de Seguro da-se quando um Cliente deseja transferir seu seguro a outro Cliente. Ou seja, a lista de Veículos segurados é atribuída a outro Cliente. Quando essa transferência é realizada, o valor do seguro do Cliente deve ser atualizado.

5 Avaliação

Além da correta execução do laboratório, os seguintes critérios serão utilizados para a composição da nota do laboratório:

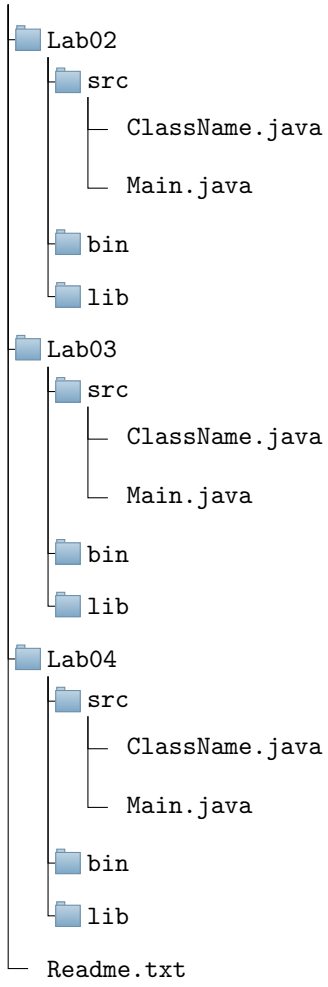
- Entrega realizada dentro do prazo estipulado;
- Execução do código;
- Qualidade do código desenvolvido (saída dos dados na tela, tabulação, comentários);
- Instanciação dos objetos, e principais métodos das classes implementadas, na classe AppMain;
- Desenvolvimento correto dos métodos e classes requisitadas;

6 Entrega

- **A entrega do Laboratório é realizada exclusivamente via Github**. Para a submissão no Github, gere um release (tag) com a identificação do laboratório no estilo <lab04-RA>. Por exemplo, para o aluno com RA 123456, a tag será: lab04-123456.
- Observação: Evite criar releases enquanto não tiver certeza que seu código está funcionando como esperado.
- Utilize os horários de laboratório e atendimentos para tirar eventuais dúvidas de submissão e também relacionadas ao desenvolvimento do laboratório.
- **Prazo de Entrega:** 02/05 - 14h

6.1 Organização das pastas do repositório

É esperado que seu repositório do Github contenha a seguinte estrutura de pastas:


Lab02
src
ClassName.java
Main.java
bin
lib
Lab03
src
ClassName.java
Main.java
bin
lib
Lab04
src
ClassName.java
Main.java
bin
lib
Readme.txt