**Trabalho OO – Entrega 1 - Esboço Diagrama de Classes UML**

* Pacote Estacionamento
  + Classe Estacionamento
    - Atributos: nome, localização, lotação máxima, lotação atual, taxa de fração, taxa de hora, taxa de diária, taxa noturna, taxa mensal, taxa de evento
    - Métodos: adicionarVeículo(), removerVeículo(), verificarLotação(), calcularTaxa()
  + Classe Veículo
    - Atributos: placa, modelo, horaEntrada, horaSaída
    - Métodos: entrarEstacionamento(), sairEstacionamento()
* Pacote Acesso
  + Classe Acesso (classe abstrata)
    - Atributos: veículo, estacionamento
    - Métodos: calcularTarifa()
  + Classe AcessoFracao (herda de Acesso)
    - Método sobreescrito: calcularTarifa()
  + Classe AcessoHora (herda de Acesso)
    - Método sobreescrito: calcularTarifa()
  + Classe AcessoDiaria (herda de Acesso)
    - Método sobreescrito: calcularTarifa()
  + Classe AcessoNoturna (herda de Acesso)
    - Método sobreescrito: calcularTarifa()
  + Classe AcessoMensalista (herda de Acesso)
    - Método sobreescrito: calcularTarifa()
  + Classe AcessoEvento (herda de Acesso)
    - Método sobreescrito: calcularTarifa()
* Pacote Seguradora
  + Classe Seguradora
    - Atributos: nome, desconto
    - Métodos: aplicarDesconto()
  + Classe Seguro (associação entre Seguradora e Veículo)
  + Atributos: veículo, seguradora
  + Métodos: aplicarDesconto()

As relações seriam:

* Estacionamento --(possui 0..\*)-- Veículo
* Veículo --(realiza 1..\*)-- Acesso
* Seguradora --(oferece 1..\*)-- Seguro
* Seguro --(associado a 1)-- Veículo

Nesse cenário, é utilizado o encapsulamento através dos modificadores de acesso e da exposição de métodos públicos para manipular os atributos das classes. A herança é demonstrada pelas classes de Acesso, onde cada tipo específico de Acesso herda da classe Acesso. O polimorfismo é usado no método calcularTarifa(), que é implementado de maneira diferente em cada subclasse de Acesso. Os métodos set, get e construtores padrão foram omitidos para simplificação.

Detalhamento dos relacionamentos:

* Estacionamento --(possui 0..\*)—Veículo
  + Este relacionamento indica que um objeto "Estacionamento" pode possuir zero ou mais objetos "Veículo". Nesse caso, geralmente adicionamos um atributo na classe "Estacionamento" que é uma lista de "Veículo", mas como estamos criando um diagrama simplificado, apenas indicamos a relação com uma linha e anotamos a multiplicidade "0..\*".
* Veículo --(realiza 1..\*)—Acesso
  + Este relacionamento indica que um objeto "Veículo" pode realizar um ou mais "Acesso". Isso implica que um veículo pode entrar e sair várias vezes do estacionamento, cada vez contabilizando um acesso diferente.
* Seguradora --(oferece 1..\*)—Seguro
  + Este relacionamento indica que uma "Seguradora" pode oferecer um ou mais "Seguro". Em termos práticos, a seguradora pode oferecer diferentes tipos de seguros, e cada seguro é relacionado a um veículo específico.
* Seguro --(associado a 1)—Veículo
  + Este relacionamento indica que um "Seguro" está associado a um "Veículo". Isso mostra que cada seguro é específico para um veículo.