Projeto: Locadora de Carros.

Equipe: Alcides Ribeiro Sampaio Neto – 402138.
Carlos Eduardo da Silva Ferreira – 397611.

Funcionalidades:

- Cadastrar carros e clientes.
- Mostrar todos os carros e clientes cadastrados.
- Mostrar os carros que estão disponíveis para serem alugados.
- Alugar carros
- Devolver carros
- Mostrar os carros que foram alugados e o CPF de quem os alugou.
- Calcular o preço que o cliente deve pagar, dada a data de devolução (os carros possuem preço por hora).

Classes:

- Lado do Servidor:
  - As classes Carro, Cliente, Alugado e Historico são referentes as tabelas do banco de dados, elas foram criadas para podermos instanciar objetos sobre as tabelas. Todas possuem apenas um construtor, getters e um método toString().
  - Já Locadora é a classe que gerencia as funções do banco. Todas as funcionalidades são implementadas aqui. O construtor inicia a conexão com o banco e os métodos fornecem os serviços, além de salvar cada alteração no banco.
  - Despachante é a classe que recebe a mensagem do cliente, a desempacota pegando o methodID e os argumentos. O methodID diz qual função da Locadora queremos e argument os argumentos necessários para o método funcionar. Depois que o serviço é processado, o despachante cria um objeto do tipo Mensagem, preenche os campos, onde o campo arguments recebe o resultado e manda para a classe Server. Em despachante também checamos se o serviço solicitado é a “Locadora”, se não for, retornamos “Fracasso”.

    - Server é a classe que cria o socket, instancia um despachante e recebe as requisições do cliente. Server recebe como json pelo método getRequest, converte para String e manda a mensagem já convertida para o despachante, então recebe do despachante a String resultado, converte para json e manda a resposta para o cliente com o método sendResponse.

- Lado do Cliente:
    - A classe Cliente inicia a conexão com o servidor, seus métodos são doOperation, sendRequest e getResponse. doOperation recebe o nome do serviço procurado, o id do método e os argumentos e cria um objeto do tipo mensagem com tais atributos, então converte mensagem para string e a string para json e por fim manda a requisição para o servidor com o método sendRequest. O método getReply recebe a resposta do servidor, converte de json e retorna a string correspondente.
    - A classe User representa a interação com o usuário, aqui temos o menu interativo com o switch case para os opções. Um objeto do tipo cliente é instanciado no construtor de user. Também existe uma variável global chamada “id” que representa o id do usuário conectado, logo, sempre que o construtor de user é chamado na main, id é incrementado garantindo um id diferente para cada pessoa conectada. Além disso, em cada case verificamos se a resposta que o servidor trouxe é realmente direcionada para o cliente certo, comparando o requestID enviado com o id recebido. Aqui também desempacotamos a mensagem acessando o campo que seria dos argumentos mas que agora possui a resposta do servido a mostramos na tela.