Sistemas Distribuídos - Trabalho de Implementação 1
Francisco Romes da Silva Filho, 409976

Link do repositório com a implementação:
<https://github.com/romesdev/sd-2020-2>

Descrição da solução implementada
O trabalho foi implementado na linguagem Python.

Parte 1: Questão 1 e Questão 2
Os arquivos que representam a solução da parte 1 são:
client.py, server.py, Pessoa.py, PessoaOutputStream.py, PessoaInputStream.py e pessoas.txt.

Partindo do ponto que o servidor está em execução para existir comunicação.
OBS: o client está inicializado com uma mensagem padrão.

Os arquivos client e server implementam a ideia de um Cliente e Servidor, utilizando sockets TCP. A ideia é utilizá-los para uma comunicação “remota”.

PessoaOutputStream: recebe como parâmetro uma lista de pessoas (objetos do tipo Pesssoa) e o destino (por convenção passei um arquivo .txt para ser o destino padrão dos dados).
OBS: O CPF de Pessoa foi definido como String, pois o Python não trata o zero à esquerda no tipo Inteiro.

A ideia trabalhada aqui é simples. Recebe-se uma lista, serializa (com pickle) essa lista, então, encaminha para um arquivo destino, um socket e também é possível mostrar essa lista utilizando o método __str__() padrão. Durante a execução do socket, é possível ver os dados do Objeto sendo enviado e o registro do socket sendo recebido, além disso, é registrado quantos objetos estão sendo transmitidos e o número de bytes dos objetos.

PessoaInputStream: só vai utilizar do que é produzido na classe de Output. Fará a leitura do conteúdo passado.

Parte 2: Questão 3
Os arquivos que representam a solução da parte 2 são:
BancoContas.py e contas.txt.
Ideia simples. A classe Banco vai receber um conjunto de contas, gravar em um arquivo de forma serializada e da mesma forma conseguir ler o arquivo, carregar para a memória (e tornar legível) e então, poderá imprimir essas informações.

Parte 3: Questão 4 e Questao 5
Os arquivos que representam a solução da parte 3 são:
client.py, server.py, InvertCaseWriter.py, InvertCaseReader.py, texto.txt.

InvertCaseWriter.py: Com o destino da informação sendo passado por meio de parâmetro (um arquivo .txt) e realizando uma leitura via linha de comando de um texto. Então, o texto tem suas cases invertidas, é gravado em um arquivo e é realizado o envio do texto invertido para o socket.

InvertCaseReader.py: Recebendo o arquivo de origem de leitura. Esse arquivo pode realizar a leitura do texto já invertido.