Universidade Federal do Ceará (UFC) - Campus Quixadá
Disciplina: Sistemas Distribuídos - Período 2024:2
Prof. Marcos Dantas Ortiz - mdo@ufc.br
Lista prática - Servidor TCP ST vs MT - entrega – 26/11/24 - 10h

Questão 1: escreva um código de teste de carga para ambos servidores TCP:

- ***multithread*** - múltiplos clientes concorrentes
- ***singlethread*** - atende um cliente por vez . A abertura de conexão (accept()) está na mesma *thread* que trata a requisição (thread main).

O código de teste deve gerar **100 clientes (*threads*) simultâneos**. Cada cliente fará uma única requisição ao servidor e encerrará. Faça uma rodada para o servidor *multithread* e outra para o *singlethread*. **Compare os tempos de execução.**

Não faça interação com o usuário no cliente (entrada por teclado). Todas as requisições podem ser iguais e definidas no próprio código: exemplo - "ADD;10;11".

Como a calculadora não oferece carga suficiente para a *thread* perder a CPU antes de terminar sua execução, adicione uma pausa no próprio código. Em java, por exemplo: **Thread.sleep(100)** ("a ***thread*** será suspensa por 100 milissegundos"). Dessa forma, o servidor *single thread* vai levar pelo menos 100 x 100 milissegundos para tratar todas as 100 requisições.

Dica: **evite mandar mensagem de saída para o console pois ele será uma gargalo para as *threads* concorrentes** (println(), print(), System.out.println()).

Dica: O tempo de fim só deve ser contabilizado quando todas as threads estiverem encerradas. Utilize o join() para sincronizar a thread main do teste de carga com as demais threads que fazem as requisições. - **threads[i].join();**

Questão 2 – Transforme os Objetos que fornecem o serviço (calculadora, por exemplo) em ***Singletons***

- ***java:***https://www.devmedia.com.br/padrao-de-projeto-singleton-em-java/26392
- ***python:*** https://refactoring.guru/pt-br/design-patterns/singleton/python/example