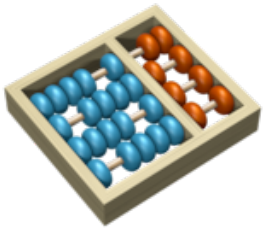

Exercício 4: multiplexação de Entrada/Saída

Aluna: Naomi Takemoto

RA: 184849

Instituto de Computação

Universidade Estadual de Campinas

Dezembro de 2020

Instruções para execução do programa

Para facilitar a correção deste exercício, foi criado um script: **run.sh**. localizado na pasta src
Este script:

- Compilará o código fonte
- Iniciara o servidor na porta 5200 (para alterar basta mudar a escolha da porta no script)
- Iniciará o cliente
- Apagará os arquivos binários gerados durante a execução.

Para o caso de o script acima apresentar problema, para iniciar manualmente cliente e servidor, basta executar na pasta src:

make

./server <#Port>

./cliente 127.0.0.1 <#Port>

OBS: O script run.sh alguns testes apresentou o problema de ficar travado, mas o output era gerado corretamente.

Arquivos de test

- Os arquivos input de teste estão na pasta input_files
- O arquivo de saída é salvo no diretório output_files

Exercício 1

1. Modifique o programa cliente da atividade 2 para que este receba como entrada e envie ao servidor linhas de um arquivo texto qualquer (O arquivo será passado utilizando o caracter de redirecionamento '<').

- ○ O cliente continuará recebendo o eco enviado pelo servidor, que deverá ser escrito em um arquivo (O arquivo será criado utilizando o caracter de redirecionamento '>'). Seu programa deverá necessariamente utilizar ou a função select ou a função poll.
- ○ Cada linha deve ser enviada separadamente para o servidor. O servidor irá enviá-las de volta para o cliente, então cuidado com os ‘\n’ e ‘\t’.
- ○ O cliente deve finalizar sua execução assim que tiver recebido **todo o arquivo** ecoado pelo servidor.

Dica: Uma forma de verificar se o seu programa está enviando e recebendo o texto da forma correta é executar o comando diff no cliente entre o arquivo de entrada (passado pelo '<') e o arquivo de saída (gerado pelo '>').

***IMPORTANTE*:** O código será testado com arquivos texto de cerca de 20000 contendo

podendo conter palavras com acento e caracteres especiais. Para certificar que seu código está funcionando corretamente, use arquivos com tamanhos próximos a esses durante os testes.

1. Dica 1: Uma boa ajuda para desenvolver esta atividade poderá ser encontrada nos exemplos do livro-texto da disciplina (**capítulo 6**) e nos programas utilizados na segunda atividade parte 1 e parte. Links úteis: select e poll

Dica 2: Como teste poderá ser utilizado os arquivos disponibilizados em

https://www.gutenberg.org/

A solução para este exercício foi baseada no código apresentado no livro texto da disciplina, na página 228.

Exercício 2

Cite um exemplo para uso de multiplexação de entrada e saída.

Existem diversos modelos para I/O são eles:

- Bloqueante
- Não bloqueante
- Multiplexação de entrada e saída
- Orientado a Sinais
- Assíncrono

No modelo de multiplexação de entrada e saída, que pode ser implementado usando as chamadas de sistema select ou poll, o que acontece é que o processo não bloqueia na chamada de I/O, mas sim em select ou poll.

Usando por exemplo a função select é possível instruir o Kernel a acordar o processo quando um I/O estiver pronto, sendo possível monitorar vários descriptors simultaneamene. Caso a multiplexação de I/O não houvesse sido utilizada no exercício acima por exemplo, o programa bloquearia esperando o input da entrada padrão stdin, assim seria difícil fazer com que o programa mandasse uma linha de cada vez para o servidor sem o uso de theads por exemplo.

Dessa forma a multiplexação de I/O pode ser uma maneira interessante para impedir que processos que lidam com diversos descriptor fiquem bloqueados esperando algum deles, como no exercício 1.