MAC 438 - Programação Concorrente - Primeiro Semestre de 2001

Segundo Exercício-Programa: O Problema dos Barbeiros

Data de Entrega: 11 de maio de 2001

Este exercício deve ser desenvolvido em equipes de duas pessoas, a fim de suscitar discussão. Dúvidas sobre o enunciado devem ser enviadas para `reverbel-mac438@ime.usp.br`.

1 O problema

Você deve implementar uma solução para um problema clássico de comunicação entre processos: o *Problema dos Barbeiros*. Vimos em aula uma solução para uma versão desse problema na qual havia um único barbeiro e não havia limite no número de clientes esperando atendimento na barbearia.

Neste exercício você deve lidar com n barbeiros trabalhando simultaneamente (ou seja, em vez de ter um barbeiro e uma cadeira de barbeiro, a barbearia agora tem n barbeiros e n cadeiras de barbeiro). Considere também que a barbearia tem m assentos para os clientes que estão esperando que algum barbeiro os atenda. Quando um cliente chega à barbearia ele verifica se ela está lotada ou não. Se a barbearia estiver lotada (se todos os n barbeiros estiverem atendendo clientes e todos os m assentos de espera estiverem ocupados), ele sai sem ter seu cabelo cortado. Se a barbearia não estiver lotada, ele entra e espera pelo corte de seu cabelo.

2 Seu arsenal

Você deve fazer este EP em C, usando o pacote *LinuxThreads*, que implementa o padrão *pthreads* (*POSIX Threads*) para Linux 2.x. A biblioteca LinuxThreads e sua documentação estão disponíveis na nossa rede Linux. Para mais informações sobre o pacote LinuxThreads, veja

```
http://pauillac.inria.fr/~xleroy/linuxthreads/
```

Todas as funções POSIX Threads tem nome começando com "`pthread_`". Para listar essas funções, diga

```
apropos pthread_
```

Neste exercício você não precisará de todas as funções `pthread_`... Você usará funções básicas de gerenciamento de threads (`pthread_create`, por exemplo), mutexes (`pthread_mutex_`...) e variáveis de condição (`pthread_cond_`...). Não é permitido o uso de outras bibliotecas de suporte a programação multithreaded (bibliotecas de semáforos, por exemplo).

O material sobre pthreads está na pasta 48 do xerox do CAMAT. Vários tutoriais sobre pthreads estão disponíveis na Internet:

```
http://www.uq.edu.au/~cmamuys/humbug/talks/pthreads/pthreads.html
http://dis.cs.umass.edu/~wagner/threads_html/tutorial.html
http://www.llnl.gov/computing/tutorials/workshops/workshop/pthreads/MAIN.html
```

(O material na terceira URL parece ser o mais completo.)

Este é um FAQ com informações variadas sobre programação com threads (incluindo "Microsoft-style threads", caso você precise um dia...):

```
http://www.serpentine.com/~bos/threads-faq/
```

3 Organização do programa

Divida seu programa em (pelo menos) dois módulos. Um desses módulos deve implementar um monitor que encapsula todos os acessos à barbearia:

```
monitor Barbearia {

    // Campos (variáveis privadas) do monitor
    ...

    // Operação chamada pelos clientes:

    boolean cortaCabelo() { ... } // se a barbearia não estiver lotada, espera
                                  //     que o corte seja feito e retorna true
                                  // se a barbearia estiver lotada, retorna false


    // Operações chamadas pelos barbeiros:

    void proximoCliente() { ... } // pega o próximo cliente (dentro desta chamada
                                  // o barbeiro pode dormir esperando um cliente)

    void corteTerminado() { ... } // o barbeiro acorda o cliente que está na sua
                                  // cadeira e espera que ele saia da barbearia
                                  // (tome cuidado para acordar o cliente certo)
}
```

Como a linguagem C não tem monitores, você implementará o monitor `Barbearia` como um módulo C, que exporta as funções `cortaCabelo`, `proximoCliente` e `corteTerminado`. Dentro desse módulo você precisará de um mutex (para controlar os acessos ao monitor) e de variáveis de condição. Tanto o mutex como as variáveis de condição devem ser varíaveis privadas (`static`) do módulo C que implementa o monitor.

Além deste módulo passivo, seu programa deverá ter pelo menos um módulo ativo, que implementará threads "barbeiro" e threads "cliente". A saída do seu programa deve indicar que barbeiro está atendendo a que cliente (ou seja, seu programa associa uma identificação a cada barbeiro e a cada cliente).

4 Sobre a entrega

Você deverá entregar tres coisas:

- um arquivo tar.gz contendo sua solução (arquivos-fonte, makefile, README, ...);
- uma listagem impressa dos seus arquivos-fonte;
- um relatório impresso sobre sua solução, com respostas para as questões acima.

Por favor, entregue o relatório e a listagem **na secretaria do MAC**, em um saco plástico devidamente fechado, contendo também um disquete com o arquvo tar.gz.

Trabalhos atrasados não serão aceitos!

Bom trabalho!