Laboratório 13
Exercícios com monitores (em Java)

Computação Concorrente (MAB-117)
Prof. Silvana Rossetto

[1]DCC/IM/UFRJ — 21 de janeiro de 2016

Introdução

O objetivo deste Laboratório é praticar o uso dos mecanismos de sincronização em Java. Para cada atividade, siga o roteiro proposto e responda às questões colocadas. Este laboratório não valerá nota.

Atividade 1

Objetivo: Reapresenta o problema do produtor/consumidor.

Roteiro: Abra o arquivo **PC.java** e siga o roteiro abaixo.

1. Complete a implementação dos métodos `Insere` e `Remove` da classe `Buffer`.
2. **Inclua código adicional para geração de log da execução de modo que seja possível verificar a sua corretude.**
3. Execute o programa **várias vezes** e verifique se a execução está correta.
4. Varie o número de threads consumidoras e produtoras, fazendo: (a) um produtor e um consumidor; (b) um produtor e vários consumidores; (c) vários produtores e um consumidor; (d) vários produtores e vários consumidores. **Verifique se a execução do programa está sempre correta**.
5. Mostre o programa executando para a professora.

Atividade 2

Objetivo: Propõe uma variação na implementação do problema produtor/consumidor.

Roteiro: Implemente a seguinte variação do problema produtor/consumidor: a cada execução de um consumidor, ele consome o buffer inteiro, e não apenas um único item (para isso ele deve esperar o buffer ficar completamente cheio). O produtor continua com a mesma lógica, i.e., insere um item de cada vez. Varie o número de threads consumidoras e produtoras, fazendo: (a) um produtor e um consumidor; (b) um produtor e vários consumidores; (c) vários produtores e um consumidor; (d) vários produtores e vários consumidores. **Inclua código adicional para geração de log da execução de modo que seja possível verificar a sua corretude.** Mostre o programa executando para a professora.

Atividade 3

Objetivo: Implementar e testar uma solução para o problema do "banheiro unissex", visto na última aula teórica, garantindo **ausência de starvation**.

Um escritório contém um banheiro que deve ser usado por homens e mulheres, mas não por ambos ao mesmo tempo. Se um (ou mais) homem está no banheiro, outros

homens podem entrar, as mulheres devem esperar até o banheiro ficar vazio. Se uma (ou mais) mulher está no banheiro, outras mulheres podem entrar, os homens devem esperar até o banheiro ficar vazio. A solução deve permitir qualquer número de homens ou qualquer número de mulheres (mas não ambos) no banheiro ao mesmo tempo, e garantir ausência de **deadlock**.

Roteiro:

1. Implemente uma classe (Monitor) Java para gerenciar o acesso ao banheiro oferecendo 4 métodos: **EntraHomem()**, **SaiHomem()**, **EntraMulher(), SaiMulher()**.
2. Use o protótipo da solução disponível no arquivo "Banheiro.java" (complete os trechos "...").
3. Teste sua aplicação variando o número de threads "homens" e "mulheres". Certifique-se que ela funciona em todos os casos.
4. Mostre uma execução do programa para a professora onde fica claro que a situação de starvation foi evitada