LABORATÓRIO 1: THREADS E MONITORES

Descrição

Nesta atividade de laboratório você deve implementar um sistema Escritor/Leitor usando sincronização através de um Monitor.

Implementação

Existem 240 threads, sendo 120 Escritor e 120 Leitores, e um objeto `Buffer` de tamanho 1 como dois métodos:

```
class Buffer {
   synchronized Escrever(int i) {...}
   synchronized int Ler() {...}
}
```

(a) As threads Escritor devem ser escalonadas usando **ScheduledExecutorService** (caso 2 em Formas de Escalonamento Java), ou seja, todas essas 120 threads devem ser controladas por esse escalonador. Esse escalonador deve liberar uma thread Escritor a cada 1 milissegundo (utilize `scheduleAtFixedRate`) que irá escrever um valor inteiro incremental no Buffer.

(b) As 120 threads Leitor devem ser controladas por um **ExecutorService** (caso 1, em Formas de Escalonamento Java) usando um pool de tamanho fixo que deve permitir até 4 threads Leitor em estado de pronto durante a execução do código. As threads Leitor em estado de pronto devem tentar ler o valor escrito no objeto Buffer. Verifique a quantidade de threads Leitor que leram o mesmo valor inteiro.

(c) Na terceira parte do trabalho, a primeira thread Leitor que conseguir ler o valor escrito deve configurar a variável do Buffer para zero. Para que as outras threads Leitor não leiam zero, você deve utilizar monitor juntamente com `wait/notify`. Isto é, quando uma thread Leitor verificar que o buffer está vazio, este invoca o método `wait()`. E quando uma thread Escritor tiver escrito um dado, este invoca o método `notify()`. Verifique o comportamento do programa em relação ao anterior.

(d) Considere a implementação com a situação em que 1 escritor escreve um valor no Buffer, e no máximo 4 leitores lêem.