Relatório: Uso de ReentrantLock com Fairness para Controle de Impressão

1. Objetivo

Este exemplo demonstra o uso do `ReentrantLock` em Java para controlar o acesso a uma fila de impressão. A implementação utiliza a opção de fairness (justiça) para garantir que as threads obtenham acesso à impressora de maneira ordenada. O exemplo divide a tarefa de impressão em duas seções, cada uma protegida por bloqueios, simulando um cenário real de uso concorrente.

2. Estrutura do Código

- **PrintQueue**:
  - Gerencia a impressão de documentos.
  - Utiliza um `ReentrantLock` com a opção `true` para fairness, garantindo que as threads sejam atendidas na ordem de solicitação.
  - Possui duas seções de impressão, cada uma envolvendo a aquisição e liberação do lock.
- **Trabalho**:
  - Implementa a interface `Runnable` e representa uma tarefa de impressão.
  - Cada instância de `Trabalho` usa a mesma instância de `PrintQueue` para enviar um documento para impressão.
- **Principal**:
  - Inicializa a aplicação criando uma instância de `PrintQueue` e 10 threads, cada uma executando uma tarefa de impressão.
  - Inicia as threads com um pequeno atraso para escalonar o acesso à fila.

3. Fluxo de Execução

1. **Inicialização**:
   - A classe `Principal` cria uma instância de `PrintQueue`.
   - Um array de 10 threads é criado, cada uma instanciando um objeto `Trabalho` que utiliza a mesma `PrintQueue`.
2. **Execução**:
   - Cada thread imprime uma mensagem indicando o início da escrita do trabalho.
   - Ao chamar o método `printJob` de `PrintQueue`, a thread adquire o lock e executa a primeira seção de impressão, simulada por um `Thread.sleep` com duração aleatória.

- Após a primeira seção, o lock é liberado e, em seguida, re-adquirido para a segunda seção de impressão.
- Após concluir ambas as seções, a thread imprime uma mensagem de conclusão.

3. **Sincronização**:
   - O uso do `ReentrantLock` com fairness garante que as threads sejam atendidas na ordem de chegada, evitando a possibilidade de starvation (fome de acesso).

4. Exemplo de Execução

Saída do Console:

```
Thread 0: Escrevendo tabalho.
Thread 0: PrintQueue: Printing a Job during 3 seconds in first code block
Thread 1: Escrevendo tabalho.
Thread 0: PrintQueue: Printing a Job during 2 seconds in second code block
Thread 0: O documento foi escrito.
Thread 1: PrintQueue: Printing a Job during 4 seconds in first code block
...
```

5. Conclusão

O exemplo evidencia a eficácia do ReentrantLock com fairness para gerenciar o acesso concorrente a um recurso compartilhado – neste caso, a fila de impressão. Ao dividir a impressão em duas seções, o código demonstra como múltiplas operações podem ser sincronizadas adequadamente, garantindo que as threads obtenham acesso de forma ordenada. Esta abordagem é crucial em ambientes onde a ordem de execução e a equidade entre threads são fundamentais para o desempenho e a confiabilidade do sistema.