Relatório: Sincronização com Lock em Fila de Impressão

1. Objetivo

Demonstrar o uso do mecanismo de Lock (ReentrantLock) para controlar o acesso a um recurso compartilhado, simulando uma fila de impressão. O exemplo coordena múltiplas threads que tentam imprimir documentos, garantindo que apenas uma thread acesse a fila de impressão por vez.

2. Estrutura do Código

2.1. Classe FilaDeImpressao

- **Descrição:**
  Gerencia o acesso à fila de impressão utilizando um Lock.
- **Funcionamento:**
  - Adquire o lock antes de iniciar a impressão.
  - Simula a impressão com uma pausa (Thread.sleep) por um tempo aleatório.
  - Libera o lock após a impressão.

2.2. Classe Tarefa

- **Descrição:**
  Implementa a interface Runnable para simular a impressão de um documento através da FilaDeImpressao.

2.3. Classe Principal

- **Descrição:**
  Cria uma instância de FilaDeImpressao e inicia 10 threads que executam a tarefa de impressão simultaneamente.

3. Fluxo de Execução

1. A classe Principal cria a FilaDeImpressao.
2. São criadas 10 threads, cada uma executando a tarefa de impressão.
3. Cada thread adquire o lock, imprime o documento simulando um tempo aleatório e, em seguida, libera o lock.
4. O lock garante que as operações de impressão não se sobreponham, mantendo a integridade do acesso ao recurso.

4. Exemplo de Execução

Saída do Console:

:: em : 12:34.567.890 Thread 0: FilaDeImpressao: Imprimindo um documento durante 3 segundos, dormindo em 12:34.567.890 :: em : 12:34.570.123 :: em : 12:34.571.456 Thread 1: FilaDeImpressao: Imprimindo um documento durante 5 segundos, dormindo em 12:34.571.456 :: em : 12:34.576.789 ...

5. Conclusão

A implementação utilizando ReentrantLock demonstra como controlar o acesso a um recurso compartilhado (a fila de impressão), permitindo que múltiplas threads operem de forma segura. O lock garante que apenas uma thread execute a impressão de cada vez, evitando conflitos e garantindo a integridade dos dados. Este mecanismo é essencial em ambientes concorrentes para evitar condições de corrida e manter a ordem de execução.