Relatório: Controle de Recursos com Semaphore

1. Objetivo

Este exemplo demonstra o uso do `Semaphore` em Java para controlar o acesso concorrente a uma fila de impressão. Utilizando um Semaphore, o sistema permite que até dois trabalhos de impressão ocorram simultaneamente, enquanto garante que as threads respeitem os limites de acesso ao recurso compartilhado.

2. Estrutura do Código

- **Trabalho**:
  Responsável por encapsular a tarefa de impressão. Cada instância de `Trabalho` solicita a impressão de um documento através da `FilaImpressao`.
- **FilaImpressao**:
  Gerencia a impressão de trabalhos utilizando um `Semaphore` para controlar o número de threads que podem acessar a impressora simultaneamente. O método `imprimirTrabalho` simula a impressão exibindo o horário de início e término da operação, além de simular um tempo aleatório de impressão.
- **Principal**:
  Inicializa o sistema criando uma instância de `FilaImpressao` e 10 threads, cada uma executando um `Trabalho`. As threads são iniciadas para demonstrar o acesso concorrente à impressora.

3. Fluxo de Execução

1. A classe `Principal` cria uma instância de `FilaImpressao` e um array de 10 threads, cada uma instanciando um objeto `Trabalho` que utiliza a mesma `FilaImpressao`.
2. Cada `Trabalho` chama o método `imprimirTrabalho` da `FilaImpressao`, que adquire uma permissão do `Semaphore` para iniciar a impressão.
3. A `FilaImpressao` simula a impressão exibindo o horário de início, aguardando um tempo aleatório (simulando a duração da impressão) e exibindo o horário de término.
4. Após a impressão, o `Semaphore` é liberado, permitindo que outras threads iniciem suas operações de impressão.
5. O uso do `Semaphore` com 2 permissões garante que até dois trabalhos possam ser processados simultaneamente, controlando o acesso ao recurso compartilhado.

4. Exemplo de Execução

Saída do Console:

```
Thread 0 :: em : 12:34.567
Thread 0: FilaImpressao: Imprimindo um trabalho durante 3 segundos
Thread 0 :: em : 12:34.570

Thread 1 :: em : 12:34.567
Thread 1: FilaImpressao: Imprimindo um trabalho durante 5 segundos
Thread 1 :: em : 12:34.572
...
```

5. Conclusão

Este exemplo ilustra como o uso de Semaphore possibilita o controle eficiente do acesso a recursos compartilhados em um ambiente concorrente. Ao limitar o número de threads que podem acessar a FilaImpressao simultaneamente, o sistema previne conflitos e assegura uma execução ordenada das operações de impressão. Essa abordagem é fundamental para aplicações que precisam gerenciar recursos limitados com alta concorrência.