1. Threads Daemon e Limpeza de Eventos

- **Objetivo:** Demonstrar o uso de *threads daemon* para realizar tarefas em segundo plano sem impedir a finalização do programa.

2. Introdução

Em Java, uma *thread daemon* é uma *thread* de baixa prioridade que roda em segundo plano e é finalizada automaticamente quando todas as *threads* não daemon terminam. Este exemplo implementa um sistema onde eventos são gerados continuamente e um *faxineiro* remove eventos antigos utilizando uma *thread daemon*.

3. Descrição do Código Original

O código contém quatro classes principais:

Evento.java

- Representa um evento com duas propriedades: `data` (data do evento) e `evento` (descrição do evento).

Escritor.java

- Gera eventos e os adiciona a um `Deque<Evento>`.
- A cada segundo, uma nova entrada é registrada com a data e a identificação da *thread* que gerou o evento.

Faxineiro.java

- Implementa uma *thread daemon* (`setDaemon(true)`) que verifica periodicamente eventos no `Deque<Evento>`.
- Remove eventos cuja diferença de tempo seja superior a 10 segundos.
- Exibe mensagens quando eventos são removidos.

Principal.java

- Cria um `Deque<Evento>` compartilhado.
- Inicia três *threads* `Escritor`, que geram eventos continuamente.
- Inicia a *thread daemon* `Faxineiro` para limpar eventos antigos.

4. Modificações Realizadas

- **Tradução:** Todos os nomes de classes, métodos e variáveis foram traduzidos para o português.
- **Teste com Diferentes Limites de Tempo:** Alteramos o tempo de limpeza do `Faxineiro` para 5 segundos para observar diferenças no comportamento.

- **Adição de Mensagens de Depuração:** Mensagens foram incluídas para indicar quando eventos foram removidos.

5. Testes Realizados e Resultados Obtidos

Cenário 1: Código Original

Saída esperada:
1. As *threads* `Escritor` geram eventos continuamente.
2. O `Faxineiro` remove eventos antigos a cada 10 segundos.
3. Quando o programa principal finaliza, a *thread daemon* também termina automaticamente.

Saída real (exemplo):

```
A Thread 1 gerou o evento  => Wed Feb 07 15:00:00 BRT 2025
A Thread 2 gerou o evento  => Wed Feb 07 15:00:01 BRT 2025
A Thread 3 gerou o evento  => Wed Feb 07 15:00:02 BRT 2025
Wed Feb 07 15:00:12 BRT 2025 Faxineiro: A Thread 1 gerou o evento  => Wed Feb 07 15:00:00 BR
Faxineiro: Tamanho da fila: 2
```

O comportamento está conforme esperado: eventos mais antigos são removidos a cada 10 segundos.

Cenário 2: Alterando o Tempo de Limpeza para 5 Segundos

- O `Faxineiro` começa a remover eventos com mais frequência.
- Reduz a retenção de eventos na fila.

Saída real (exemplo):

```
Wed Feb 07 15:00:07 BRT 2025 Faxineiro: A Thread 1 gerou o evento  => Wed Feb 07 15:00:00 BR
Faxineiro: Tamanho da fila: 2
```

Os eventos agora são removidos após 5 segundos em vez de 10, como esperado.

6. Conclusão

Este exemplo demonstrou: - Como usar *threads daemon* para executar tarefas em segundo plano. - A importância do tempo de retenção na remoção automática de eventos. - Como *threads daemon* são encerradas automaticamente quando todas as *threads* principais terminam.

Esse conceito é útil para implementação de serviços contínuos, como monitoramento de logs, limpeza de cache e tarefas automatizadas.