SIMULAÇÃO DE UMA FILA DE IMPRESSÃO COM PRIORIDADES

Arthur Reina Lyra, Bruno de Menezes Sales, Bruno Sales Fiaes Carneiro, Vitor Moreira dos Santos

Resumo: Este artigo apresenta uma simulação de fila de impressão com prioridades, utilizando a estrutura de dados PriorityQueue da linguagem Java. O sistema desenvolvido permite organizar documentos de acordo com três níveis de urgência — urgente, intermediário e comum — e ainda considera a ordem de envio em casos de empate. A implementação foi realizada por meio de classes orientadas a objetos, com destaque para o uso da interface Comparable para garantir a ordenação correta dos documentos. A simulação mostrou que o sistema consegue processar os documentos de forma justa e eficiente, priorizando os mais importantes e respeitando a ordem de chegada quando necessário. O artigo também discute os benefícios e limitações da abordagem adotada, sugerindo melhorias futuras.

Palavras-chave: fila de prioridade; PriorityQueue; heap; estrutura de dados; impressão.

1 Introdução

Em muitos ambientes de trabalho, como escritórios, escolas e empresas, é comum que várias pessoas usem a mesma impressora. Nessas situações, diferentes documentos são enviados ao mesmo tempo, mas nem todos têm a mesma urgência. Um relatório importante pode acabar esperando enquanto um documento menos prioritário é impresso, simplesmente porque foi enviado antes. Esse tipo de problema pode causar atrasos e prejudicar o andamento das tarefas.

Pensando nisso, este projeto simula uma fila de impressão com prioridade, utilizando a linguagem Java. A ideia é organizar os documentos de acordo com seu nível de urgência, classificados como urgente, intermediário e comum, e considerar a ordem em que foram enviados, para que a impressão siga uma lógica justa e eficiente.

Para isso, foi usada a estrutura de dados PriorityQueue, que permite ordenar os documentos automaticamente conforme suas prioridades. O sistema é simples, interativo e funciona direto no console, permitindo ao usuário enviar documentos, visualizar a fila e entender como a prioridade afeta a ordem de impressão.

1.1 Objetivo

O projeto tem como objetivo simular o funcionamento de uma fila de impressão priorizada, utilizando os conceitos de estruturas de dados, especialmente a fila de prioridade (PriorityQueue) da linguagem Java.

Ele visa:

1. Gerenciar documentos de diferentes níveis de prioridade:
    - Urgente (1)
    - Intermediário (2)
    - Comum (3)

Esses níveis determinam a ordem de impressão, sendo que documentos mais urgentes devem ser processados antes.

2. Manter a ordem de envio entre documentos da mesma prioridade, ou seja, em caso de empate no nível de prioridade, o documento enviado mais cedo é impresso primeiro. Esse comportamento é garantido por meio do atributo tempoDeEnvio.

3. Proporcionar uma interface simples via terminal, permitindo que o usuário:
    - Envie novos documentos para impressão.
    - Visualize a ordem atual da fila de impressão.
    - Encerre o programa.

4. Aplicar conceitos de orientação a objetos:
    - Com classes como Documento, Impressora e Menu, o projeto favorece a modularidade e reutilização de código.

5. Demonstrar a aplicação prática de comparações customizadas (Comparable), para permitir que a fila de prioridade organize os documentos corretamente com base nos critérios de prioridade e tempo.

2 Fundamentação teórica

2.1 Heap

O heap é uma estrutura de dados organizada em forma de árvore binária completa, onde os elementos seguem uma ordem específica conhecida como propriedade do heap. Essa propriedade determina como os elementos se relacionam entre si, dependendo do tipo de heap utilizado.

Existem dois tipos principais:

- Min-heap: o menor valor sempre fica na raiz, e cada nó pai é menor ou igual aos seus filhos.
- Max-heap: o maior valor fica na raiz, e cada nó pai é maior ou igual aos seus filhos.

Uma das principais vantagens do heap é a possibilidade de acessar rapidamente o elemento com maior ou menor prioridade — o que o torna ideal para aplicações onde esse tipo de seleção precisa ser feita com frequência, como é o caso das filas de prioridade.

Apesar de ser uma estrutura em forma de árvore, o heap costuma ser implementado usando arrays. Nesse formato, é possível calcular facilmente a posição dos filhos e do pai de um elemento a partir de seu índice: os filhos de um elemento no índice i ficam nas posições 2i + 1 e 2i + 2, e o pai está em (i - 1) / 2.

2.2 Fila de prioridade

A fila de prioridade é uma estrutura de dados que se diferencia da fila tradicional por levar em conta a prioridade de cada elemento, e não apenas a ordem de chegada. Ou seja, em vez de seguir o modelo FIFO (primeiro a entrar, primeiro a sair), a fila de prioridade sempre remove o elemento com maior prioridade, independentemente do momento em que ele foi inserido.

Na prática, esse tipo de estrutura costuma ser implementado com base em heaps, justamente para garantir um bom desempenho nas operações principais:

- Inserção de elementos: O(log n)
- Remoção do elemento com maior prioridade: O(log n)
- Acesso ao elemento prioritário: O(1)

Essa eficiência torna as filas de prioridade bastante úteis em sistemas onde diferentes tarefas ou itens precisam ser processados de acordo com seu nível de importância, como em gerenciamento de tarefas, algoritmos de busca e sistemas de impressão.

2.3 Filas de prioridade em java

A linguagem Java oferece suporte nativo para filas de prioridade por meio da classe PriorityQueue, presente no pacote java.util. Essa classe implementa, por padrão, um min-heap, o que significa que o menor elemento (de acordo com a ordem natural ou um comparador definido pelo programador) é sempre o primeiro a ser removido.

O funcionamento da PriorityQueue depende da forma como os elementos são comparados entre si. Para isso, eles podem implementar a interface Comparable (definindo uma ordem natural) ou utilizar um Comparator externo. Essa comparação determina quem tem mais "prioridade" na fila.

Em resumo, a PriorityQueue automatiza todo o trabalho de organização e priorização, permitindo que os elementos sejam inseridos e processados conforme a lógica de prioridade definida, com alta eficiência e simplicidade de uso no código.

3 Metodologia

3.1 Classe Documento

Esta classe representa um documento que será enviado à fila de impressão.

Atributos:

nome: nome do documento

tipo: nível de prioridade (1 = urgente, 2 = intermediário, 3 = comum).

tempoDeEnvio: timestamp de quando o documento foi criado, usado como critério de desempate.

Interface:

A classe implementa Comparable<Documento>, permitindo que objetos sejam ordenados automaticamente na PriorityQueue. O método compareTo define a ordem com base em:

1. Prioridade (tipo): quanto menor o valor, maior a prioridade.
2. Tempo de envio: em caso de empate na prioridade, o documento mais antigo é impresso primeiro.

3.2 Classe Impressora

Responsável por gerenciar a fila de documentos a serem impressos.

Atributo:

filaImpressao: uma PriorityQueue<Documento> que organiza automaticamente os documentos pela prioridade definida no compareTo.

Métodos:

enfileira(String nome, int tipo): cria um Documento e o insere na fila.

verFila(): exibe os documentos na ordem em que serão impressos, sem removê-los permanentemente da fila. Para isso, é usada uma lista temporária para armazenar e re-inserir os documentos.

3.3 Classe Menu

É a interface de texto (console) que permite interação com o usuário.

Exibe um menu com três opções:

1. Enviar um novo documento.
2. Visualizar a ordem atual da fila de impressão.
3. Sair do programa.

Ao adicionar um documento, o usuário informa o nome e a prioridade, que será convertida para um número (1, 2 ou 3).

4 Resultados

Para validar o funcionamento do sistema de fila de impressão priorizada, foram realizados testes com o envio de diferentes documentos, visualização da fila e verificação da ordem correta de atendimento. A seguir, apresentamos simulações via terminal que mostram o comportamento esperado do sistema.

Envio de documentos:

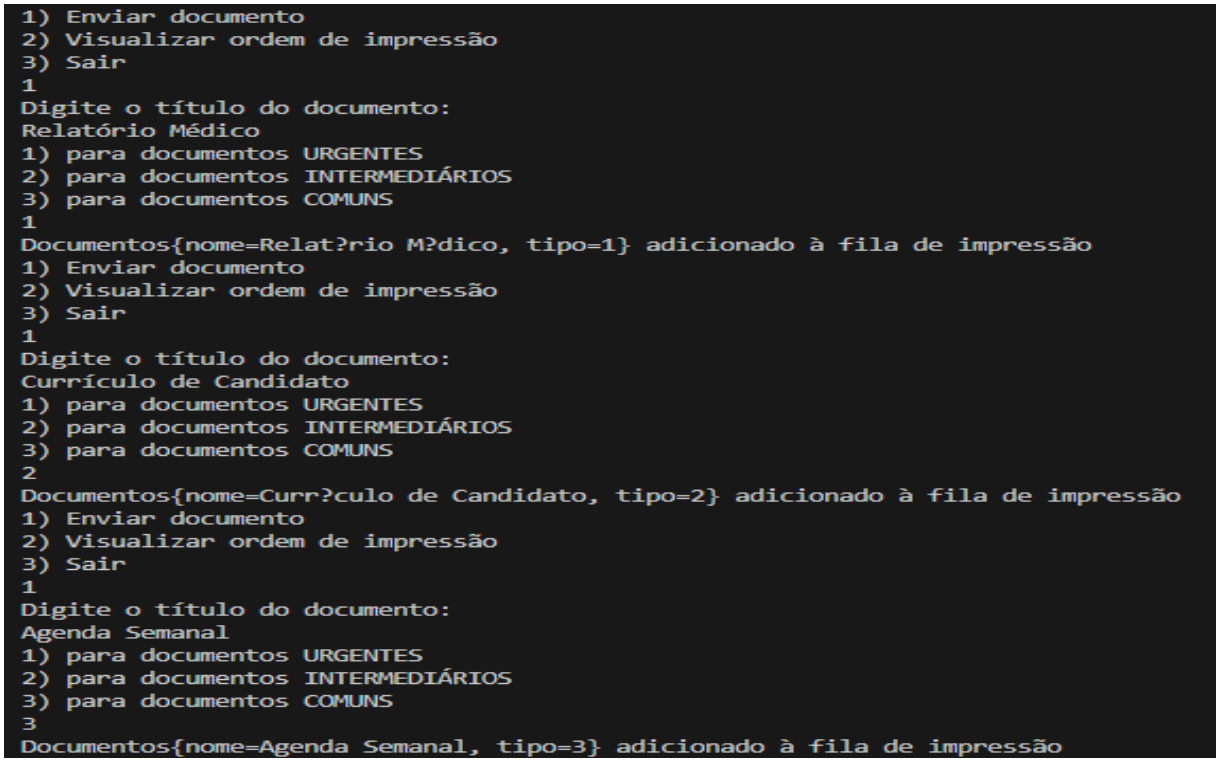

Figura 1: Envio de documentos para a fila de impressão

```
1) Enviar documento
2) Visualizar ordem de impressão
3) Sair
1
Digite o título do documento:
Relatório de Incidente de Segurança da Informação
1) para documentos URGENTES
2) para documentos INTERMEDIÁRIOS
3) para documentos COMUNS
1
Documentos{nome=Relat?rio de Incidente de Seguran?a da Informa??o, tipo=1} adicionado à fila de impressão
1) Enviar documento
2) Visualizar ordem de impressão
3) Sair
1
Digite o título do documento:
Atualização do Plano de Projeto
1) para documentos URGENTES
2) para documentos INTERMEDIÁRIOS
3) para documentos COMUNS
2
Documentos{nome=Atualiza??o do Plano de Projeto, tipo=2} adicionado à fila de impressão
1) Enviar documento
2) Visualizar ordem de impressão
3) Sair
1
Digite o título do documento:
Lista de Compras
1) para documentos URGENTES
2) para documentos INTERMEDIÁRIOS
3) para documentos COMUNS
3
Documentos{nome=Lista de Compras, tipo=3} adicionado à fila de impressão
```

Figura 2: Envio de documentos para a fila de impressão

Fila de impressão

```
1) Enviar documento
2) Visualizar ordem de impressão
3) Sair
2
Documentos na fila de impressão:
Documentos{nome=Relat?rio M?dico, tipo=1}
Documentos{nome=Relat?rio de Incidente de Seguran?a da Informa??o, tipo=1}
Documentos{nome=Curr?culo de Candidato, tipo=2}
Documentos{nome=Atualiza??o do Plano de Projeto, tipo=2}
Documentos{nome=Agenda Semanal, tipo=3}
Documentos{nome=Lista de Compras, tipo=3}
```

Figura 3: Fila de impressão

Ao visualizar a fila de impressão, os documentos são exibidos de acordo com a prioridade: os urgentes aparecem primeiro, depois os intermediários, e por último os comuns. Em caso de empate na prioridade, a ordem de envio é respeitada.

A simulação demonstra que o sistema atende corretamente aos critérios definidos de prioridade.

5 Discussão

A utilização da PriorityQueue se mostrou bastante eficaz para simular uma fila de impressão com diferentes níveis de prioridade. A lógica de ordenação, que considera o tipo do documento (urgente, intermediário ou comum) e o tempo de envio, garante que os documentos mais importantes e mais antigos sejam processados primeiro, o que está de acordo com o funcionamento de sistemas reais de impressão.

Como a PriorityQueue é baseada em uma estrutura de heap, as operações de inserção e remoção são rápidas e eficientes. No entanto, foi necessário esvaziar a fila temporariamente para exibir sua ordem, o que não é a solução mais prática ou performática, principalmente em cenários com muitos documentos.

Outro ponto a ser considerado é o uso de números inteiros para definir a prioridade. Isso funciona, mas pode causar confusão para o usuário. Utilizar um enum deixaria o código mais claro e menos propenso a erros. Além disso, o sistema ainda é simples e não oferece funcionalidades extras, como a remoção ou reordenação de documentos já enfileirados, o que poderia deixá-lo mais completo.

6 Conclusão

O projeto conseguiu cumprir seu objetivo principal: simular uma fila de impressão com prioridade usando a estrutura PriorityQueue da linguagem Java. A implementação se mostrou eficiente ao organizar os documentos de forma justa, priorizando os mais urgentes e, em casos de empate, considerando a ordem de envio.

A lógica de comparação personalizada, feita com a interface Comparable, foi essencial para garantir esse comportamento. Além disso, a divisão do sistema em classes como Documento, Impressora e Menu ajudou a tornar o código mais organizado, reutilizável e fácil de entender, reforçando o uso de conceitos de orientação a objetos.

Por fim, a interface simples via terminal torna o sistema acessível e direto, permitindo que o usuário interaja facilmente com a fila de impressão. No geral, o projeto mostra na prática como

estruturas de dados podem ser aplicadas para resolver problemas do dia a dia de forma clara e eficiente.

7 Referências

https://joaoarthurbm.github.io/eda/posts/heap/

https://www.estrategiaconcursos.com.br/blog/filas-prioridades-heap-cnu-ti/

https://www.geeksforgeeks.org/priority-queue-set-1-introduction/