INSTITUTO FEDERAL
Sudeste de Minas Gerais
Campus Manhuaçu

Estruturas de Dados I
Filas

Prof. Leonardo C. R. Soares - *leonardo.soares@ifsudestemg.edu.br*

Instituto Federal do Sudeste de Minas Gerais

16 de janeiro de 2025

Filas

Descrição

- Filas são um tipo abstrato de dados com a característica de que **o primeiro elemento a ser inserido é o primeiro a ser removido** (política FIFO – First in First Out).

Filas

Descrição

- Filas são um tipo abstrato de dados com a característica de que **o primeiro elemento a ser inserido é o primeiro a ser removido** (política FIFO – First in First Out).
- Considera-se uma analogia com filas de elementos, como pessoas, processos etc.

Filas

Descrição

- Filas são um tipo abstrato de dados com a característica de que **o primeiro elemento a ser inserido é o primeiro a ser removido** (política FIFO – First in First Out).
- Considera-se uma analogia com filas de elementos, como pessoas, processos etc.
- Os usos de filas incluem filas de impressão e filas de processamento em sistemas operacionais, entre outros.

Operações

O tipo abstrato de dados (TAD) fila deve, obrigatoriamente, suportar os métodos:

- enfileirar(o): Insere o objeto no fim da fila.

Operações

O tipo abstrato de dados (TAD) fila deve, obrigatoriamente, suportar os métodos:

- enfileirar(o): Insere o objeto no fim da fila.
- desenfileirar(): Retira o objeto no inicio da fila e o retorna; se a fila estiver vazia, ocorre um erro.

Operações

O tipo abstrato de dados (TAD) fila deve, obrigatoriamente, suportar os métodos:

- enfileirar(o): Insere o objeto no fim da fila.
- desenfileirar(): Retira o objeto no inicio da fila e o retorna; se a fila estiver vazia, ocorre um erro.

Adicionalmente, podemos definir os seguintes métodos auxiliares:

- tamanho(): Retorna o número de objetos na fila.

Operações

O tipo abstrato de dados (TAD) fila deve, obrigatoriamente, suportar os métodos:

- enfileirar(o): Insere o objeto no fim da fila.
- desenfileirar(): Retira o objeto no inicio da fila e o retorna; se a fila estiver vazia, ocorre um erro.

Adicionalmente, podemos definir os seguintes métodos auxiliares:

- tamanho(): Retorna o número de objetos na fila.
- vazia(): Retorna um *boolean* indicando se a fila está vazia.

Formas de implementação

Existem várias opções de estruturas de dados que podem ser usadas para representar filas. As duas representações mais utilizadas são:

- Por meio de arranjos.
- Por meio de referências.

Independente da forma de implementação, uma fila é uma lista com restrições quanto às formas de inserção e remoção (política FIFO), o que permite a reusabilidade de código.

Implementação por arranjos

- Os itens da fila são armazenados em posições contíguas de memória.

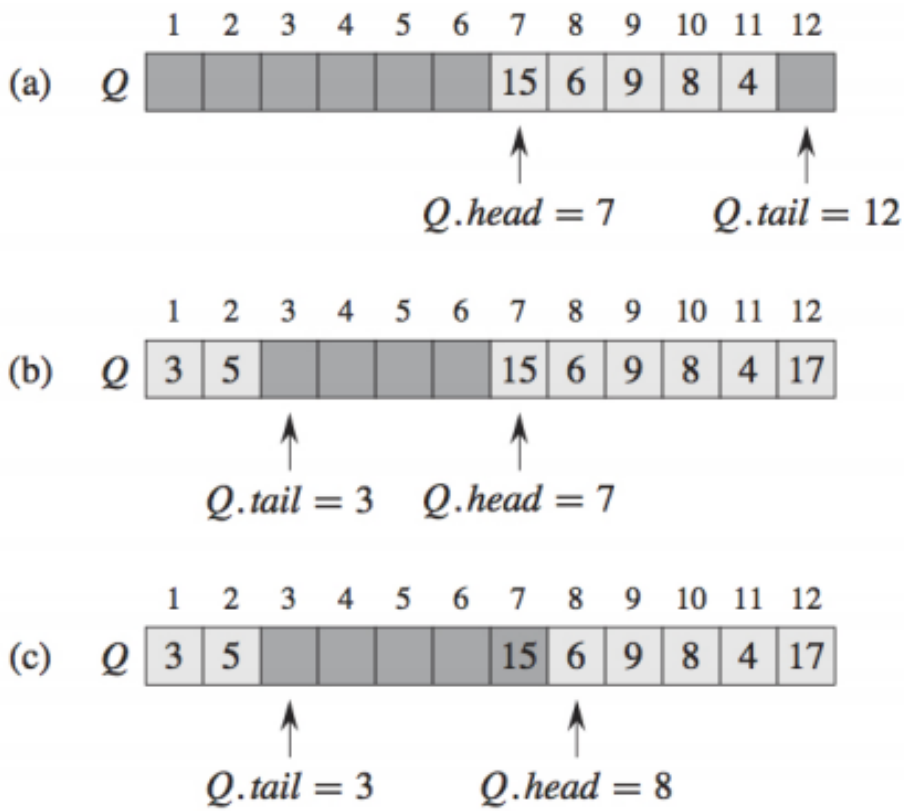
Implementação por arranjos

- Os itens da fila são armazenados em posições contíguas de memória.
- A operação Enfileira faz a parte de trás da fila expandir-se.

Implementação por arranjos

- Os itens da fila são armazenados em posições contíguas de memória.
- A operação Enfileira faz a parte de trás da fila expandir-se.
- A operação Desenfileira faz a parte da frente da fila contrair-se.

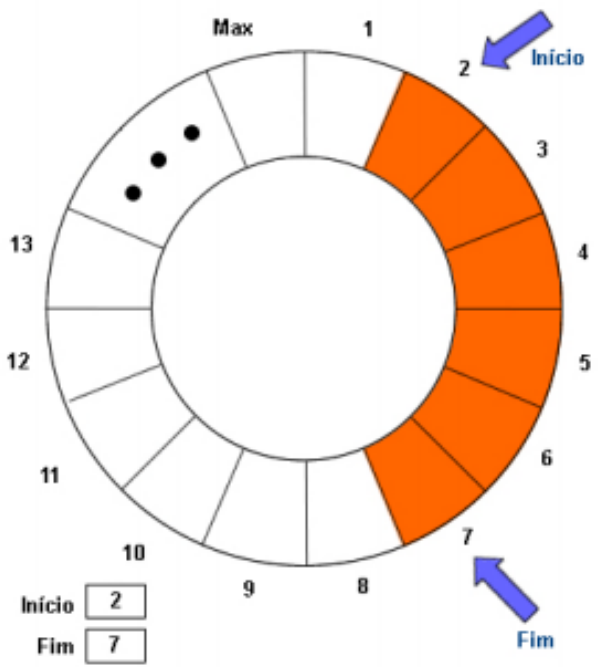
Implementação por arranjos



(a) Fila com cinco elementos. (b) Após o enfileiramento de três elementos. (c) Após desenfileirar um elemento.

Implementação por arranjos

Com poucas inserções e retiradas, a fila vai ao encontro do limite do espaço da memória alocado para ela.
Uma solução é imaginar o arranjo como um círculo, em que a primeira posição segue a última.



Implementação por arranjos

A fila se encontra em posições contíguas de memória, em alguma posição do círculo, delimitada pelos apontadores **Início** e **Fim**.

- **Início** indica a posição do primeiro elemento.
- **Fim** a primeira posição vazia (posição após o último elemento).

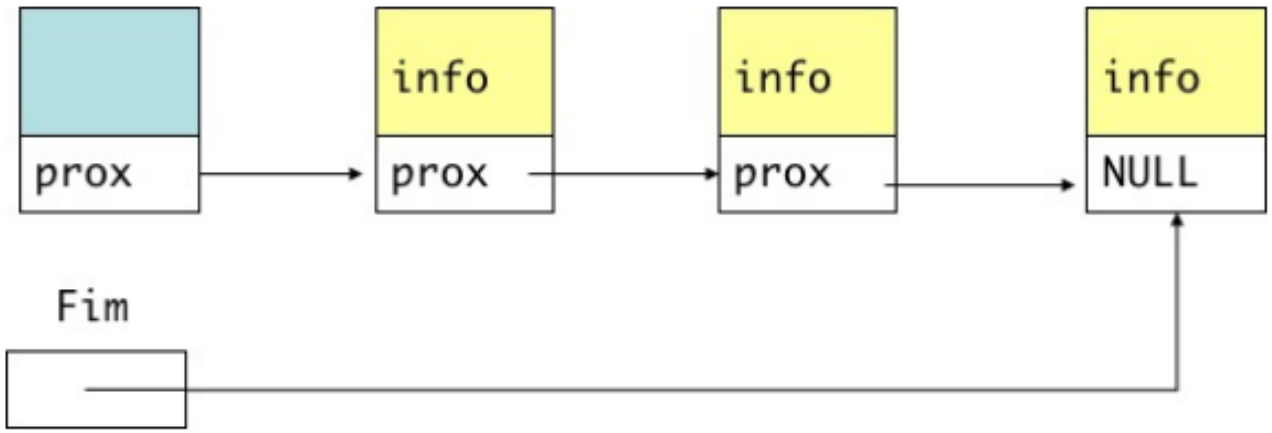
Para enfileirar, basta mover o apontador Fim uma posição no sentido horário.
Para desenfileirar, basta mover o apontador Início uma posição no sentido horário.

Implementação por referência

- A fila é implementada por meio de células, tal que cada célula contém um item da fila e um apontador para a próxima célula.

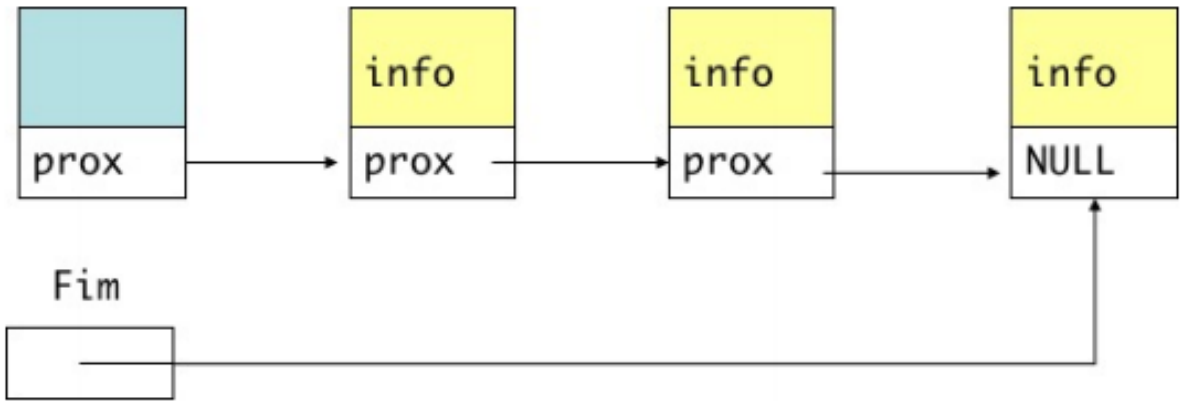
Implementação por referência

- A fila é implementada por meio de células, tal que cada célula contém um item da fila e um apontador para a próxima célula.
- A estrutura contém um apontador para a frente da fila (célula cabeça) e um apontador para a parte de trás da fila (fim).

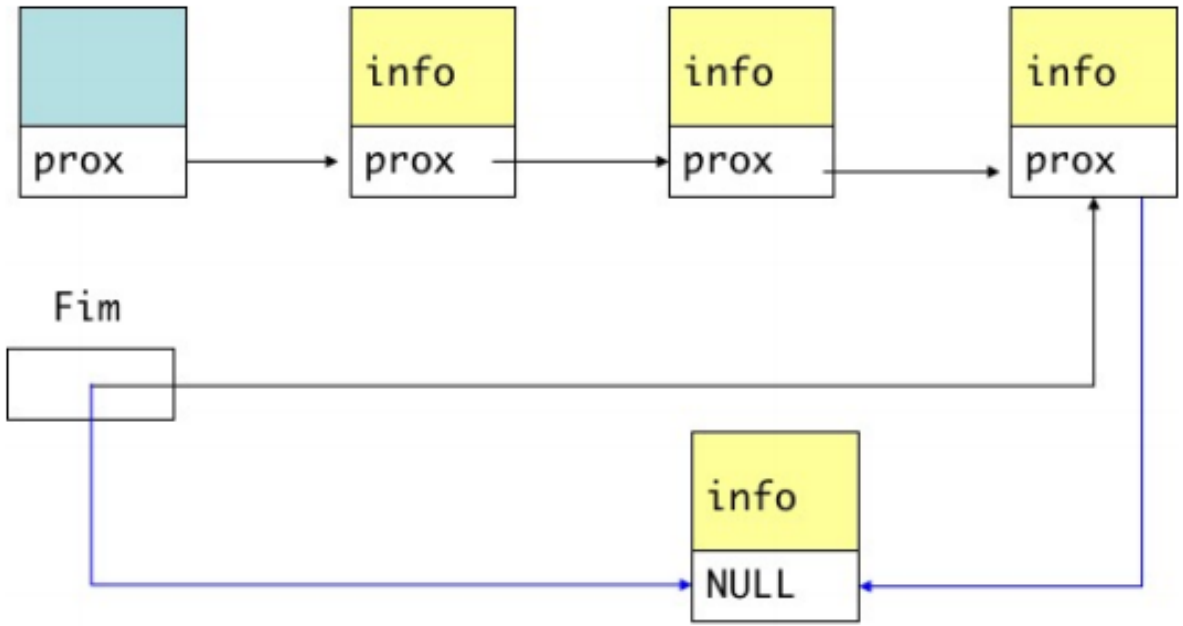


Implementação por referência - Inserção

De acordo com a política **FIFO**, há apenas uma opção de posição onde podemos inserir elementos: o fim da fila (ou seja, a última posição).

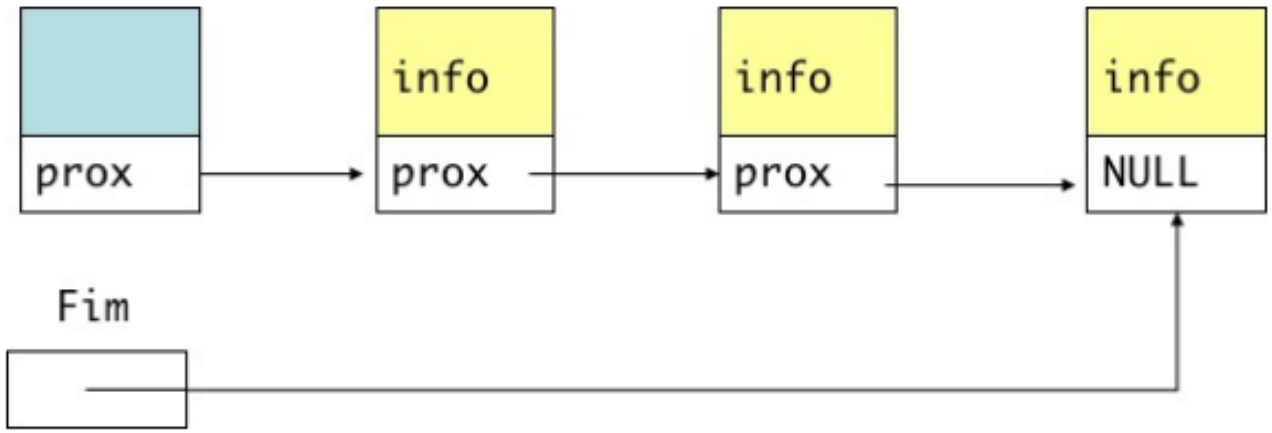


Implementação por referência - Inserção

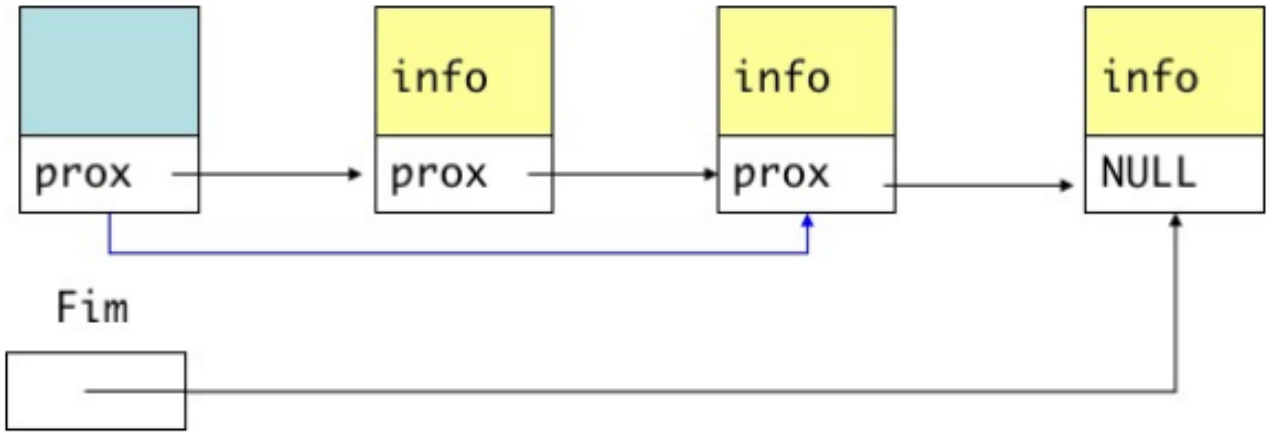


Implementação por referência - Remoção

De acordo com a política **FIFO**, há apenas uma opção de posição onde podemos remover elementos: o início da fila (ou seja, primeira posição).



Implementação por referência - Remoção



Complexidade

A complexidade de todas as operações é mantida da implementação de Lista:

- Enfileirar: $\theta(1)$
- Desenfileirar: $\theta(1)$

Perguntas?

Baixe o exemplo aqui

Exercícios

Considerando uma fila implementada por arranjos com capacidade para cinco elementos, qual o valor dos campos **inicio** e **fim** após a execução das operações: enfileirar(1), enfileirar(2), enfileirar(3), desenfileirar(), desenfileirar(), enfileirar(4), enfileirar(5), enfileirar(6), desenfileirar(), desenfileirar().

Qual o conteúdo da fila após as operações citadas no item anterior?

Exercícios

1. O exemplo apresentado implementa uma fila utilizando um arranjo "circular". Escreva um método furaFila(Pessoa p) que insere um item na primeira posição da fila (observe que neste caso estamos desrespeitando a política da FILA, FIFO). O método deve possuir complexidade $\mathcal{O}(1)$.
2. Implemente um método que **inverta** a fila.
3. **(GIT)** Desenvolva um pequeno sistema para distribuição de senhas. A cada execução do sistema serão distribuídas 50 senhas. As senhas serão nominais. O sistema deverá permitir a solicitação de senha e o chamado para atendimento.

Exercícios

4. **(GIT)** Existem partes de sistemas operacionais que cuidam da ordem em que os programas devem ser executados. Por exemplo, em um sistema de computação de tempo-compartilhado ("time-shared") existe a necessidade de manter um conjunto de processos em uma fila, esperando para serem executados. Escreva um programa que permita:
   - Incluir novos processos na fila de processos[1];
   - Retirar da fila o processo com o maior tempo de espera (o primeiro a ser incluído);
   - Imprimir o conteúdo da fila de processos;
   - Localizar um determinado processo na fila;
   - Excluir todos os processos da fila.

[1] Cada processo será representado por uma classe contendo um número identificador e um título.

GitHub (Ao vivo)

GitHub (Ao vivo)


thank you
danke
рахмат
謝謝
ngiyabonga
teşekkür ederim
gracias
merci
obrigado
спасибо
Баярлалаа
bedankt
dziękuje
sukriya
kop khun krap
go raibh maith agat
mochchakkeram
tapadh leat
arigatô
dakujem
grazie
terima kasih
감사합니다
ευχαριστώ
мерси
хвала
dank je
mersi
barka
welalin
tack
spas
faafetai lava
vinaka
blagodaram
misaotra
matondo
paldies
grazzi
mahalo
kiitos
dankie
dhanyavad
nanni
nandri
hvala
maururu
koszonom
bayarlalaa
gracie
enkosi
asante
manana
obrigada
tenki
murakoze
chokrane
djiere dieuf
tau
mamnun
дякую
chnorakaloutioun
gratias ago
gràcies
sulpay
taiku
sobodi
dėkui
sagolun
mèsi
didi madloba
najis tuke
kam sah hamnida
rahmat
ありがとう
tanemirt
rahmet
diolch
dhanyavadagalu
shukriya
merce
takk
trugarez
xièxie
akun
dankon
kia ora
спасибі

Referências

- CARVALHO, Marco Antonio Moreira de. **Projeto e análise de algoritmos**. 01 mar. 2018, 15 jun. 2018. Notas de Aula. PPGCC. UFOP
- GOODRICH, Michael T.; TAMASSIA, Roberto. **Estruturas de Dados & Algoritmos em Java.** Bookman Editora, 2013.
- ZIVIANI, Nivio. **Projeto de Algoritmos com implementações em Java e C++**, 2007.