

ED62A-COM2A

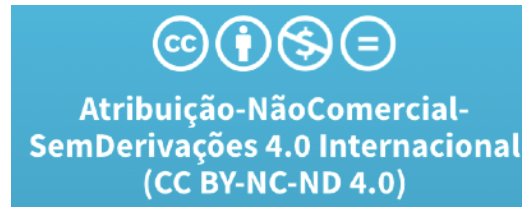
ESTRUTURAS DE DADOS

Aula 03A - Filas
(Implementação estática)

Prof. Rafael G. Mantovani

Licença

Este trabalho está licenciado com uma Licença CC BY-NC-ND 4.0:



maiores informações:

https://creativecommons.org/licenses/by-nc-nd/4.0/deed.pt_BR

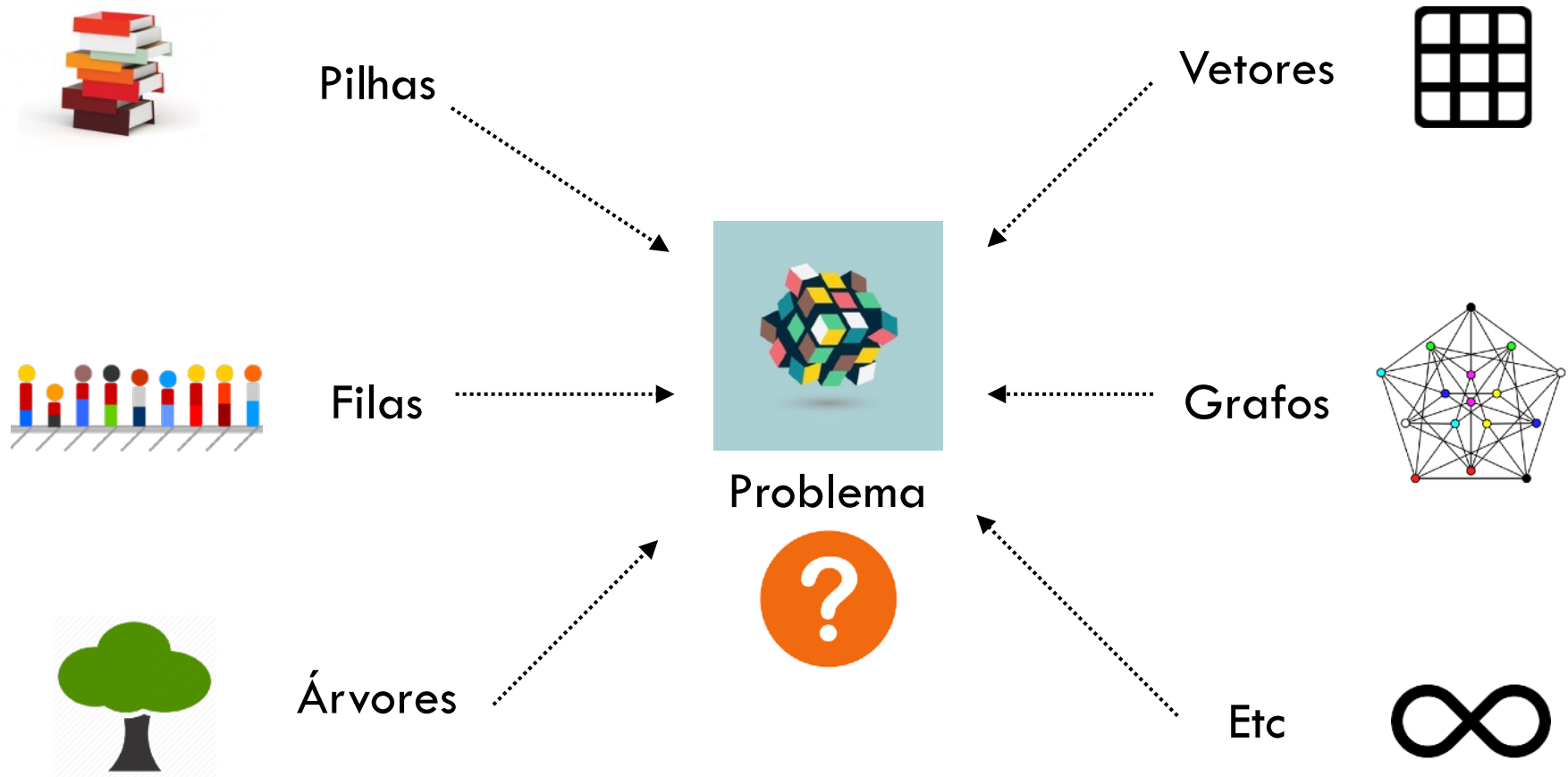
Roteiro

- 1** Introdução
- 2** Filas
- 3** Operações gerais
- 4** Inserção de elementos
- 5** Remoção de elementos
- 6** Referências

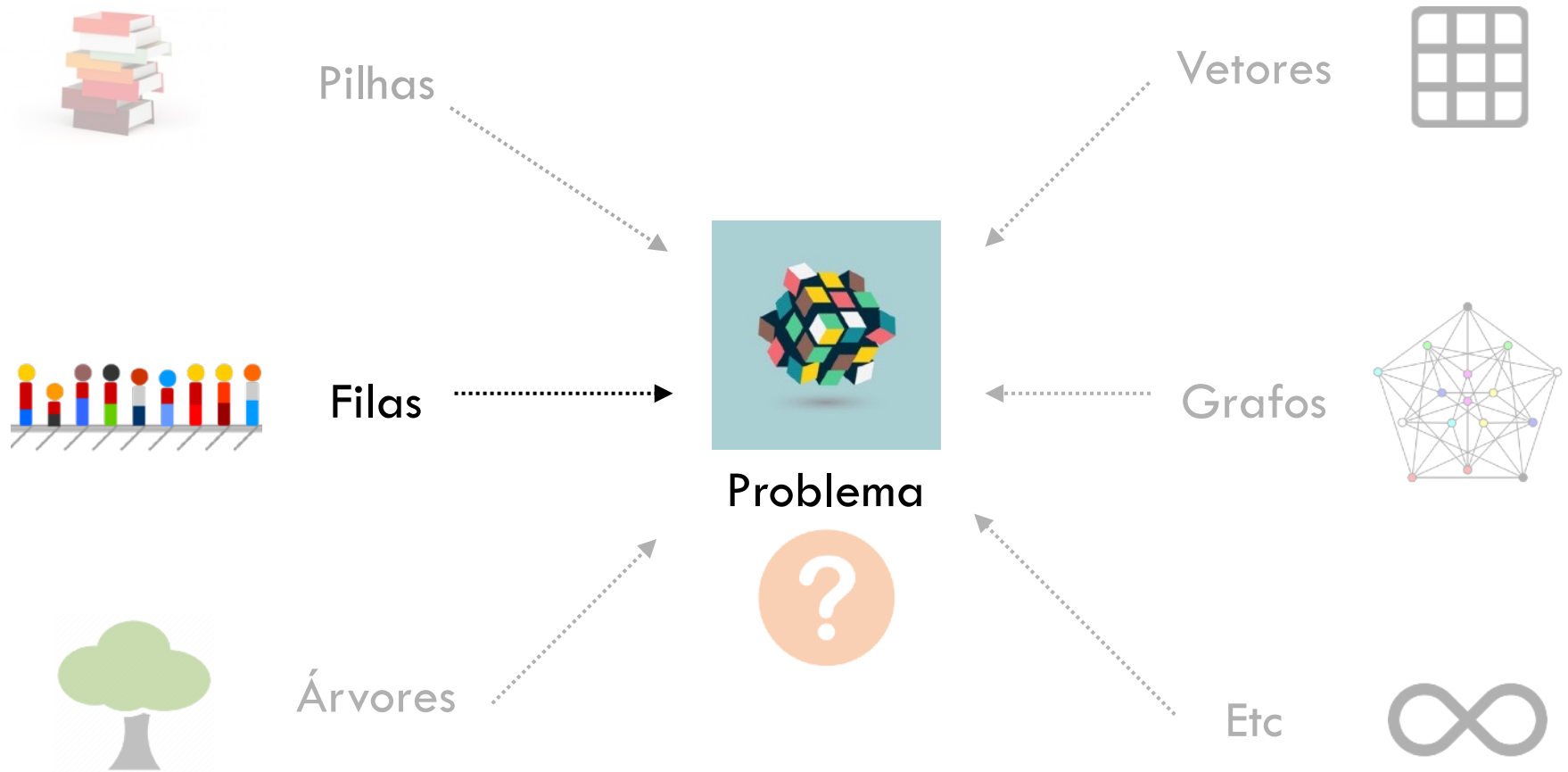
Roteiro

- 1 Introdução**
- 2 Filas**
- 3 Operações gerais**
- 4 Inserção de elementos**
- 5 Remoção de elementos**
- 6 Referências**

Introdução



Introdução



Roteiro

- 1 Introdução
- 2 Filas
- 3 Operações gerais
- 4 Inserção de elementos
- 5 Remoção de elementos
- 6 Referências

Filas



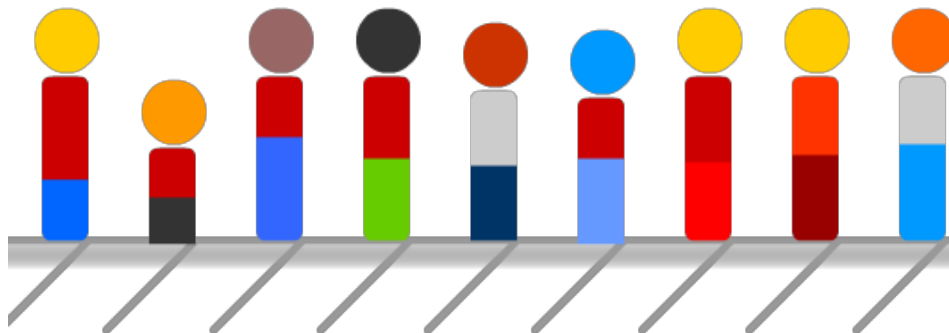
Filas



- FIFO (First In, First Out)

“Primeiro elemento a entrar é o primeiro a sair”

Filas

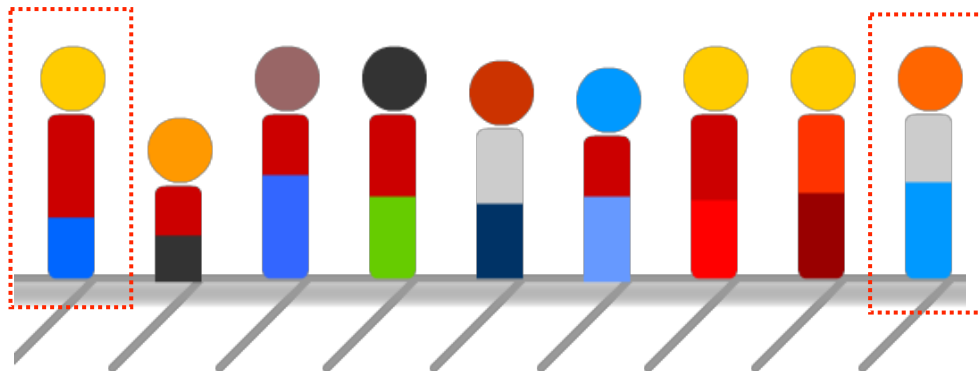


Fila de pessoas
(Queue)

Filas

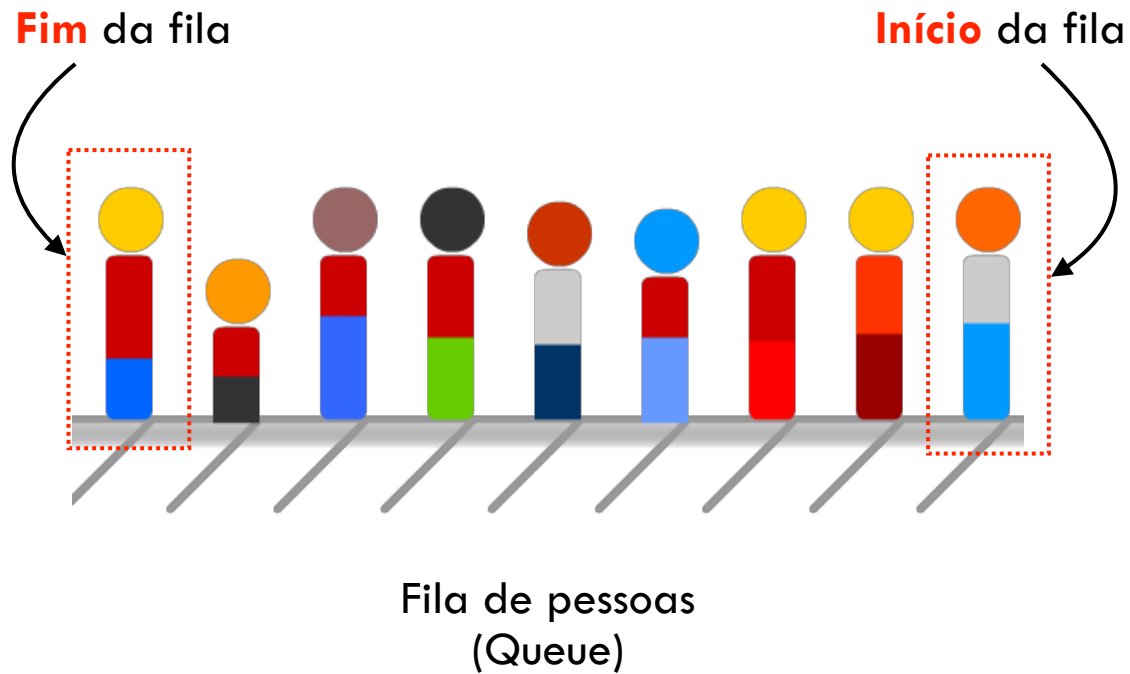
Fim da fila

Início da fila



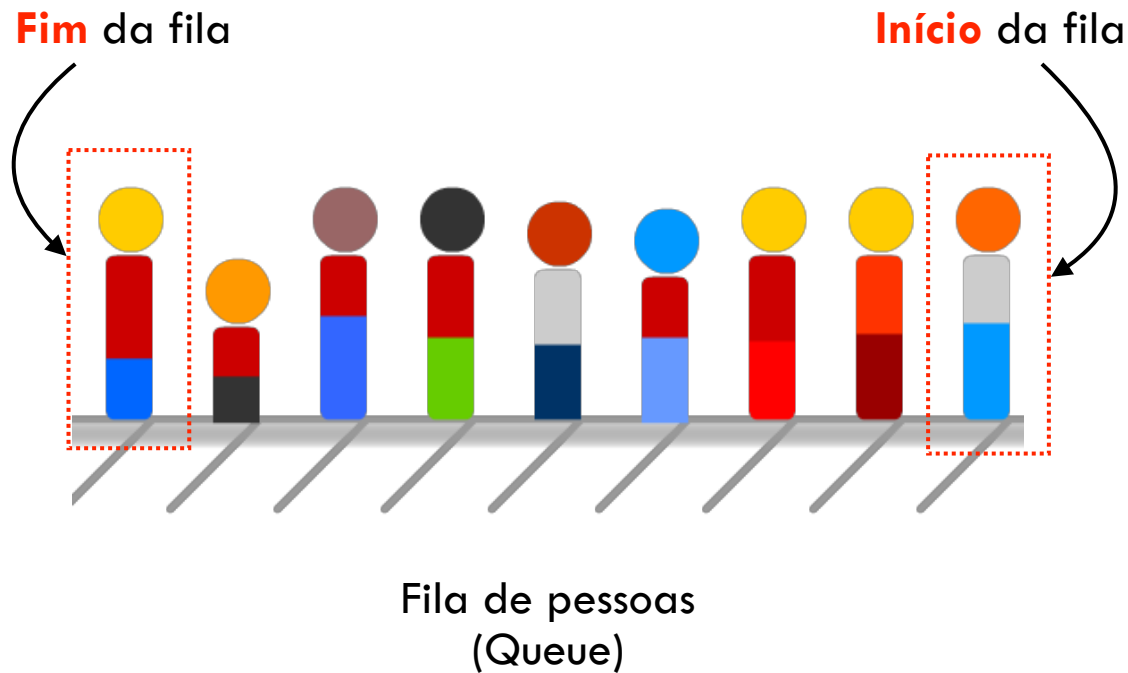
Fila de pessoas
(Queue)

Filas

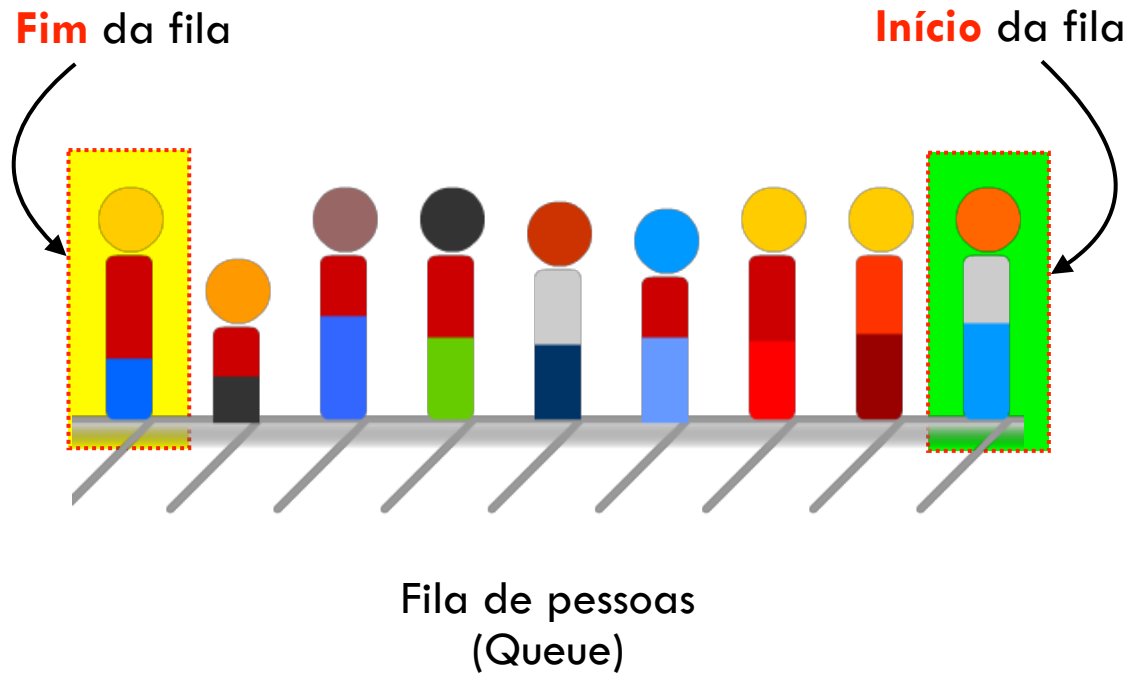


Filas

Diferente das Pilhas, nas Filas iremos manipular **as duas** extremidades do conjunto de elementos

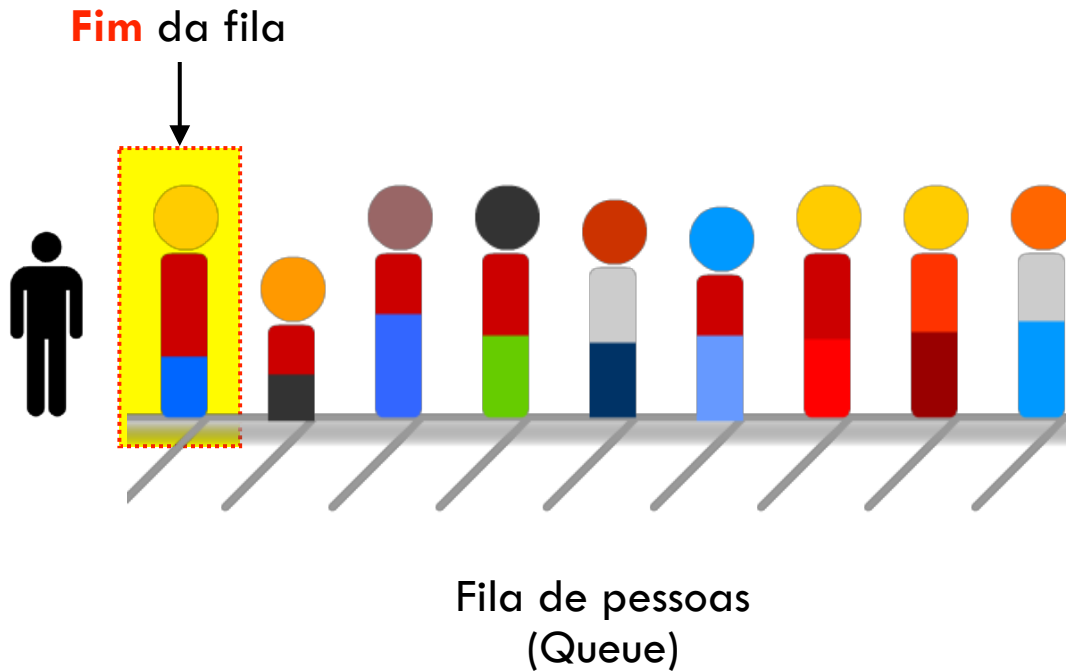


Filas



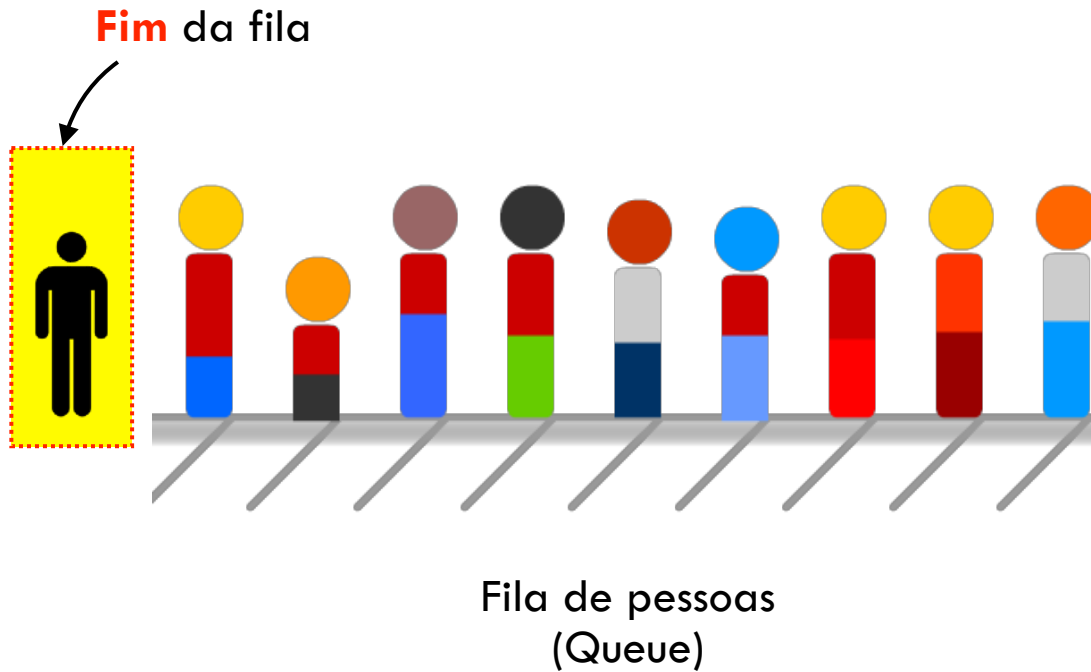
Filas

- Inserindo novo elemento: adiciona-o ao **final** da fila



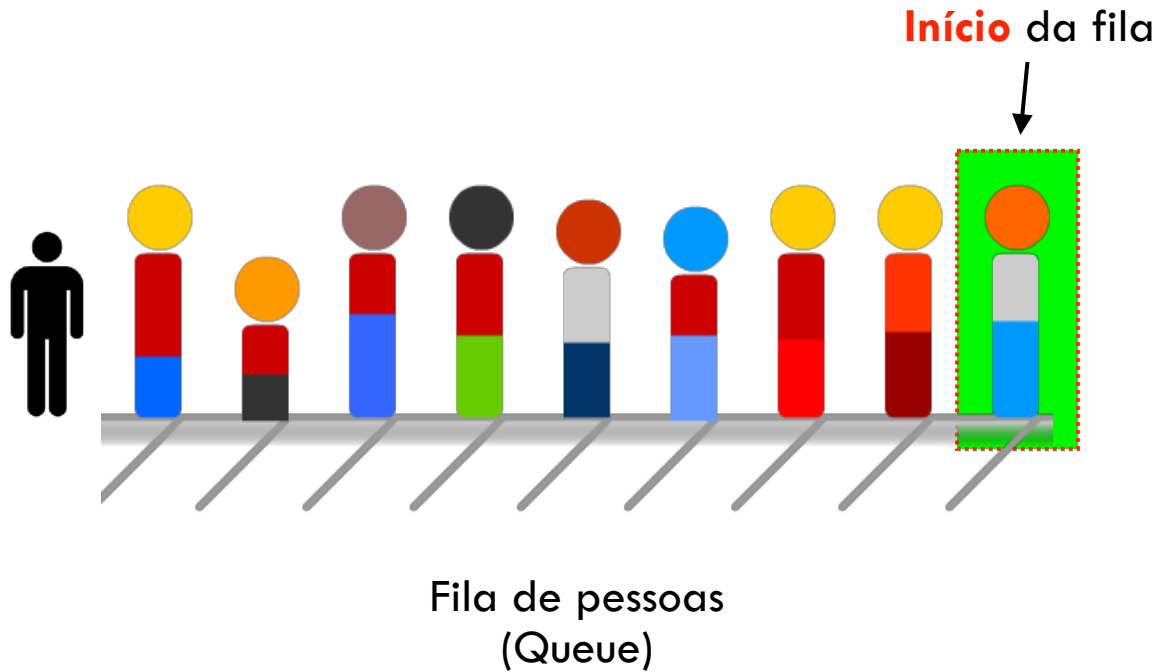
Filas

- Inserindo novo elemento: adiciona-o ao **final** da fila



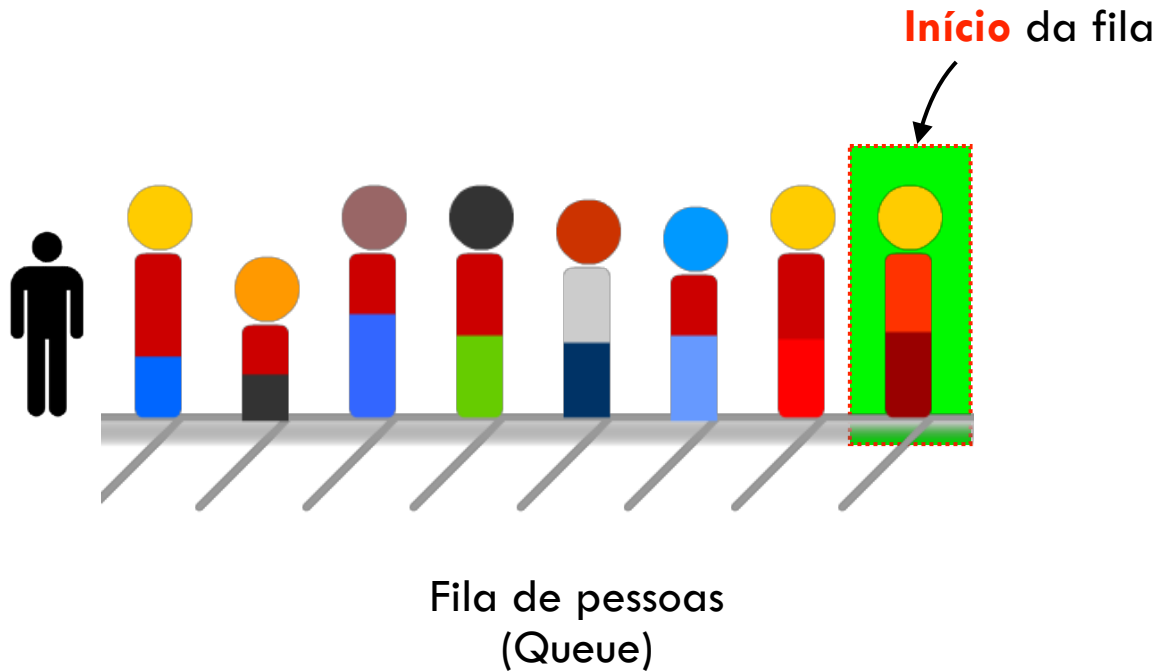
Filas

- Removendo elemento: remove-o do **início** da fila



Filas

- Removendo elemento: remove-o do **início** da fila



Filas



- Onde usamos?
 - buffer da análise léxica (Compiladores)
 - paginação de memória (Sistemas Operacionais)
 - fila de processos (Sistemas Operacionais)
 - algoritmos de árvores/grafos (Grafos, Inteligência Artificial)

Filas

- Onde usamos?
 - buffer da análise léxica (Compiladores)
 - paginação de memória (Sistemas Operacionais)
 - fila de processos (Sistemas Operacionais)
 - algoritmos de árvores/grafos (Grafos, Inteligência Artificial)

“Quando queremos estabelecer ordem”

Roteiro

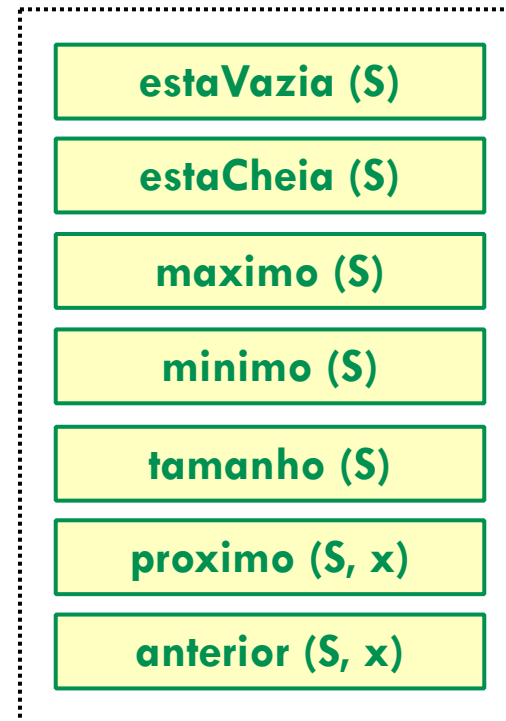
- 1 Introdução
- 2 Filas
- 3 Operações gerais
- 4 Inserção de elementos
- 5 Remoção de elementos
- 6 Referências

Operações em Filas Estáticas

Dada uma estrutura S , chave k , elemento x :



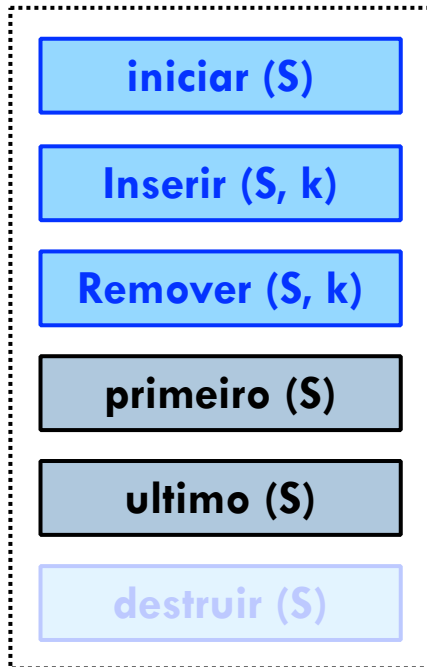
**Operações de
modificação**



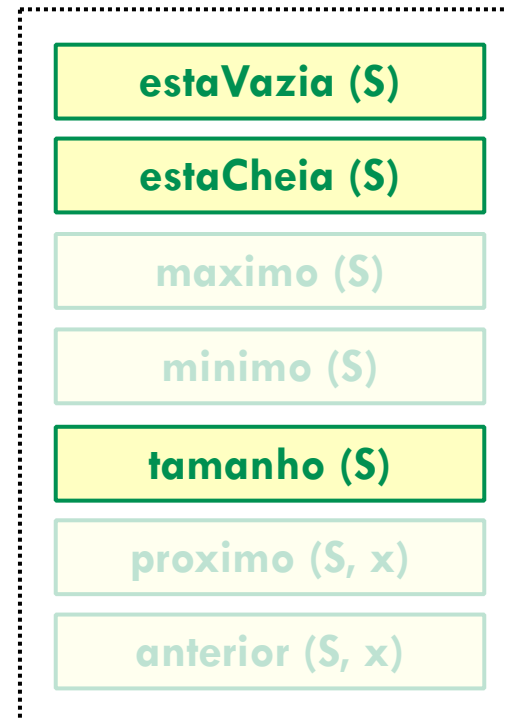
**Operações adicionais
de consulta**

Operações em Filas Estáticas

Dada uma estrutura S , chave k , elemento x :



**Operações de
modificação**



**Operações adicionais
de consulta**

Operações em Filas Estáticas

iniciar (S)

Inicializa a fila e suas variáveis

Inserir (S, k)

Inserir objeto na fila (enfileirar)

Remover (S, k)

Remover objeto da fila (desenfileirar)

primeiro (S)

Retorna o primeiro elemento da fila, sem remover

ultimo (S)

Retorna o último elemento da fila, sem remover

estaVazia (S)

Retorna booleano indicando se a fila está vazia

estaCheia (S)

Retorna booleano indicando se a fila está cheia

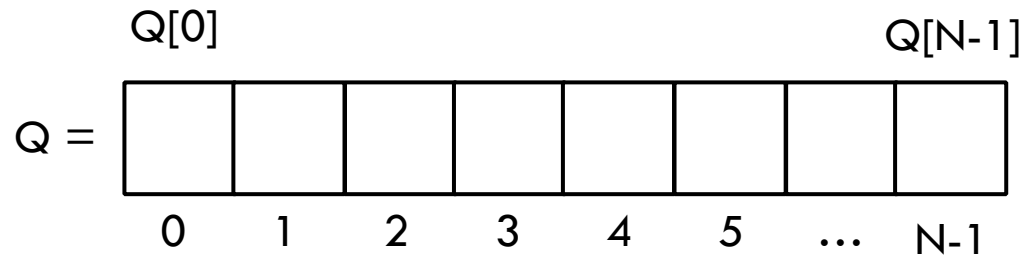
tamanho (S)

Retorna a quantidade de elementos na fila

Fila estática

Fila (queue) Q = Arranjo de N elementos

Número de elementos :



Início

Indexa a posição
inicial da fila

Fim

Indexa a posição
final da fila

Fila estática

Fila (queue) Q = Arranjo de N elementos

Número de elementos :

$Q[0]$

$Q[N-1]$

O controle da Fila é realizado por duas variáveis indexadoras (**ints**)

Início

Indexa a posição
inicial da fila

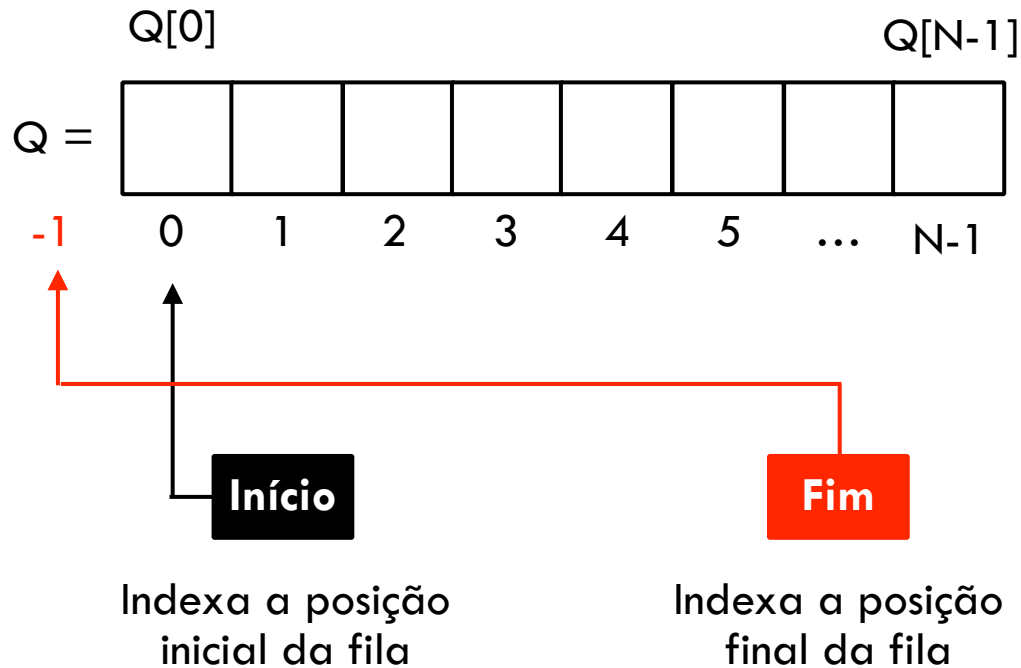
Fim

Indexa a posição
final da fila

Fila estática

- Inicialização

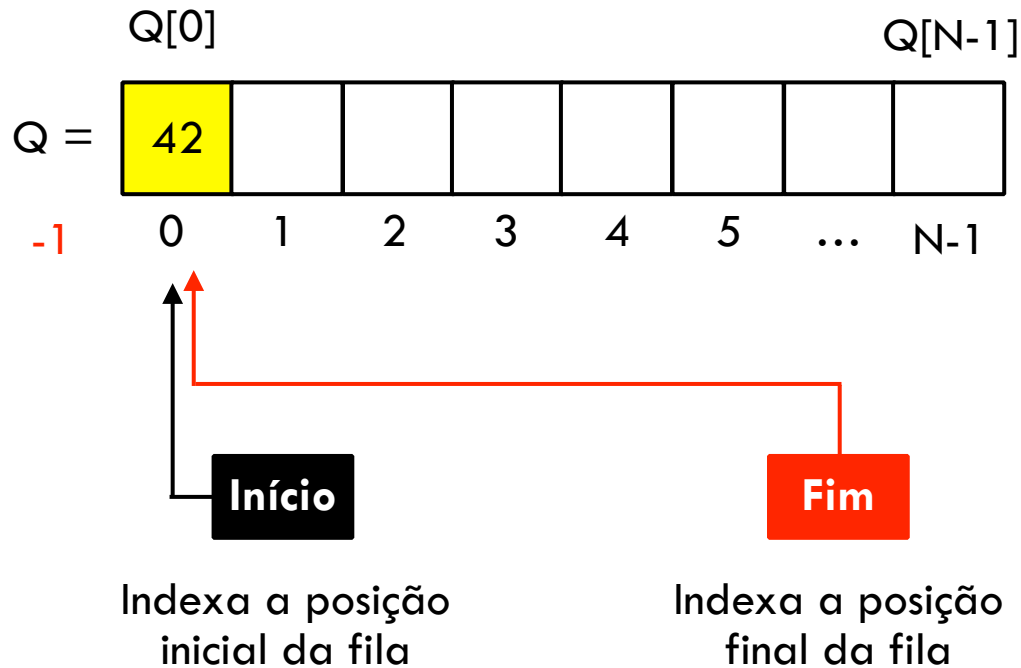
Número de elementos : 0



Fila estática

- inserindo: 42

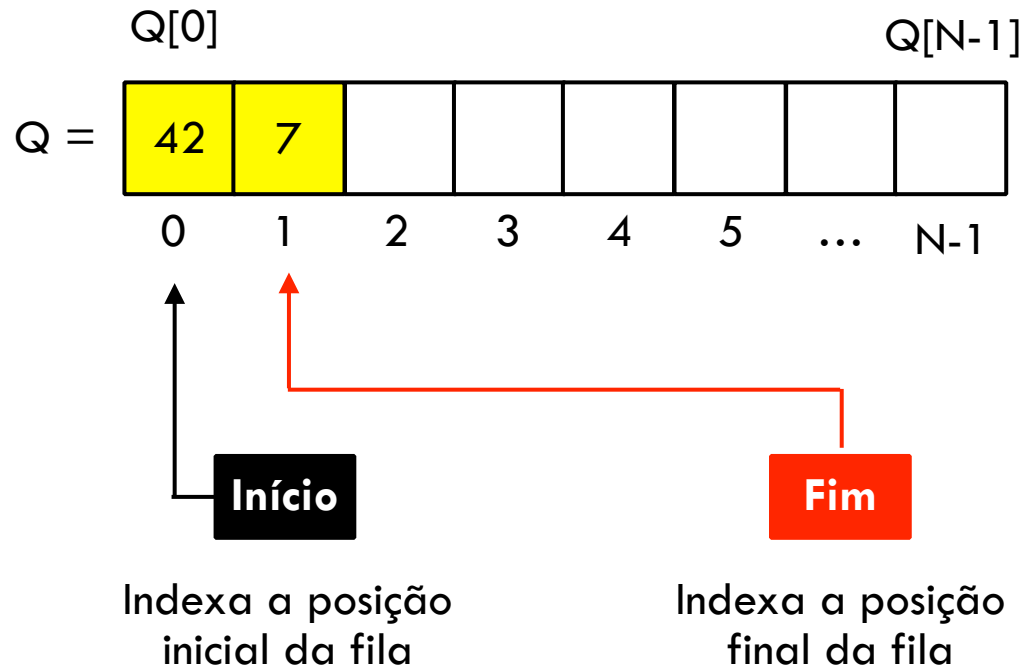
Número de elementos : 1



Fila estática

- inserindo: 7

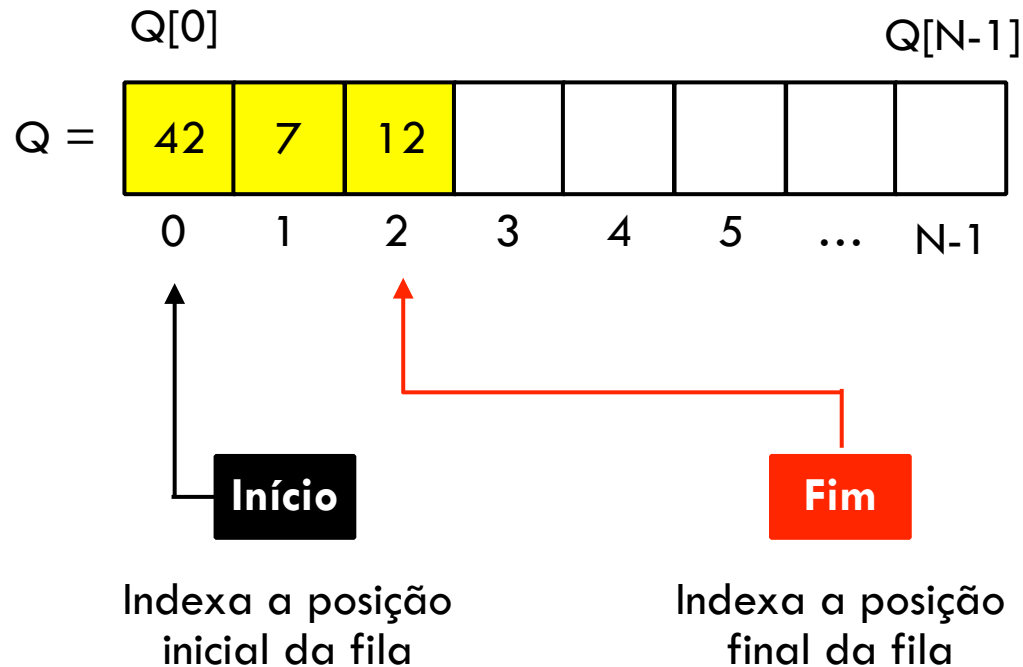
Número de elementos : 2



Fila estática

- inserindo: 12

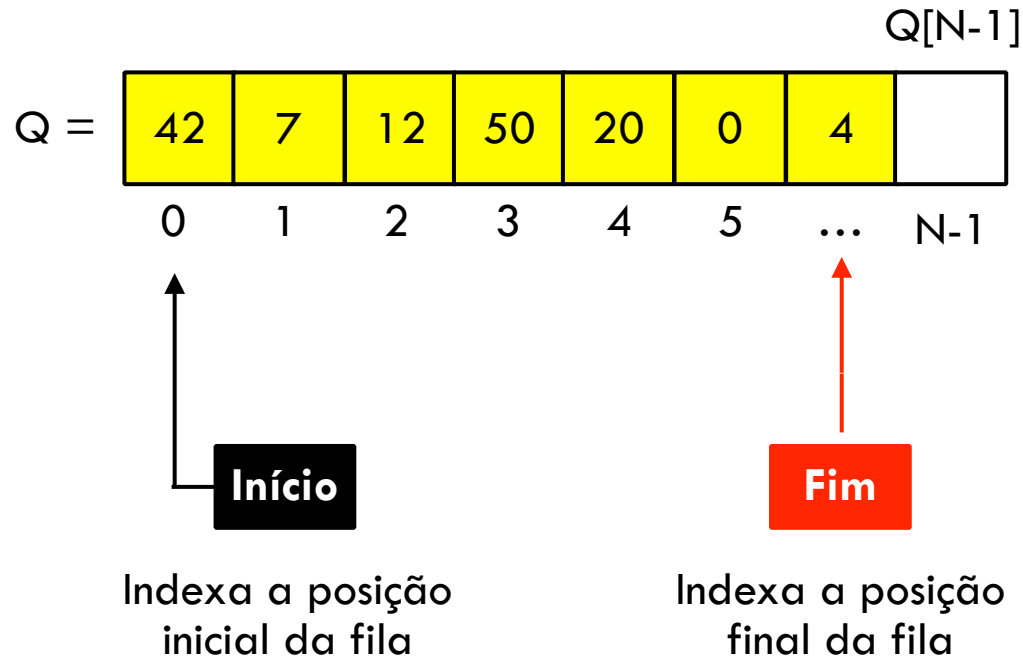
Número de elementos : 3



Fila estática

- Inserindo os elementos: 50, 20, 0, 4, 7

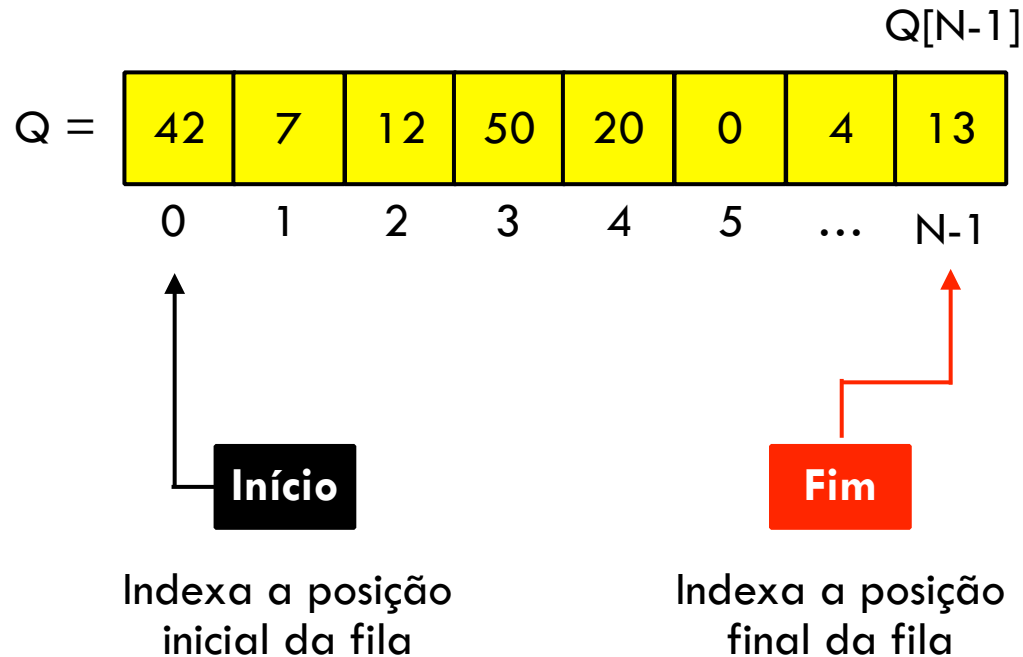
Número de elementos : 7



Fila estática

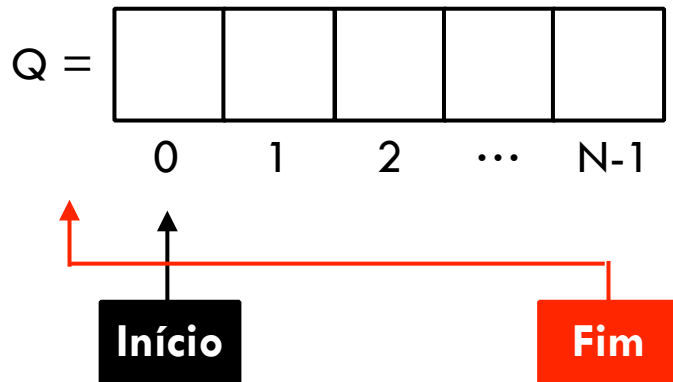
- Inserindo o elemento: 13 (fila cheia)

Número de elementos : 8



Fila estática

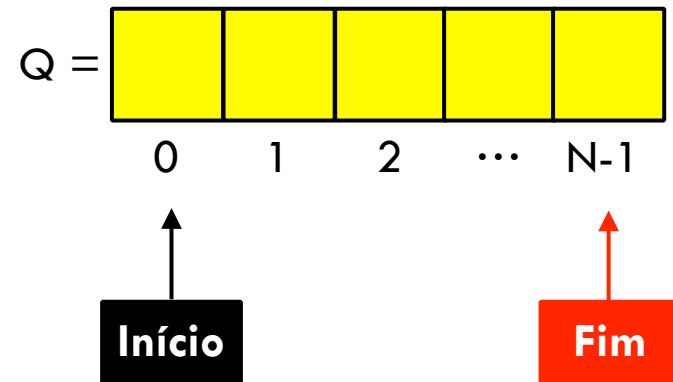
- $Q[Q.\text{contador}] == 0 \rightarrow$ fila vazia



estaVazia (Q)

1. `return(Q.contador == 0)`

- $Q[Q.\text{contador}] == N \rightarrow$ fila cheia



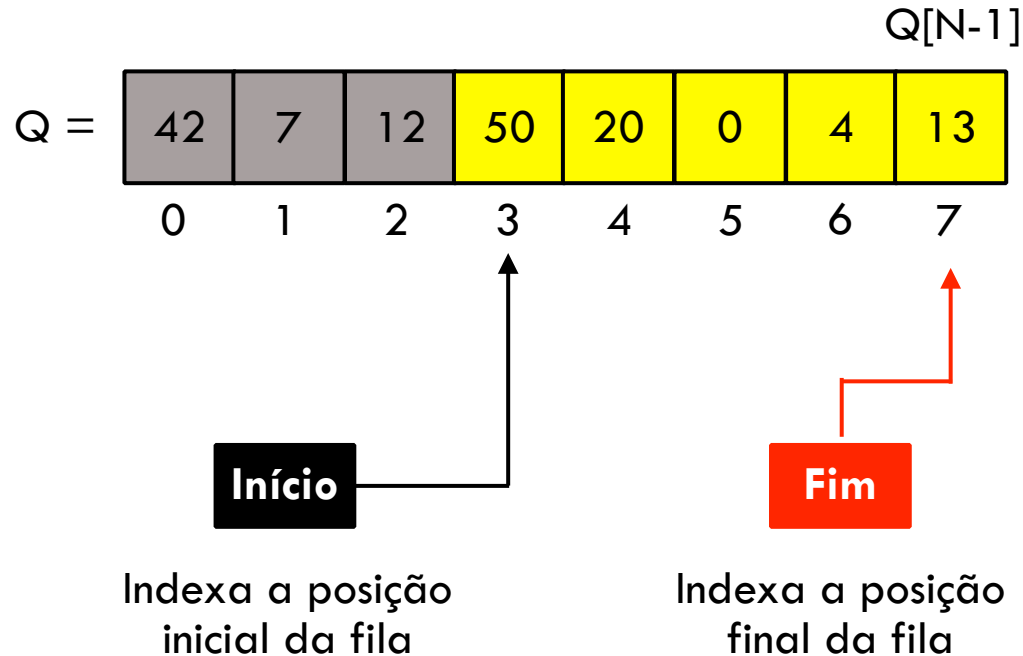
estaCheia (Q)

1. `return(Q.contador == N)`

Fila estática

- Removendo 3 elementos

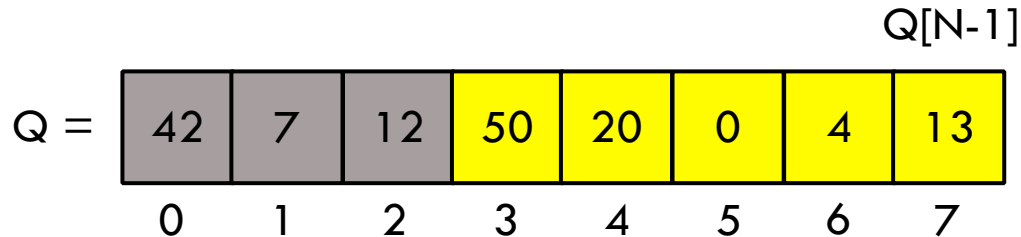
Número de elementos : 8



Fila estática

- Removendo 3 elementos

Número de elementos : 8



Remover os elementos consiste em **invalidá-los (cinza)**

Início

Fim

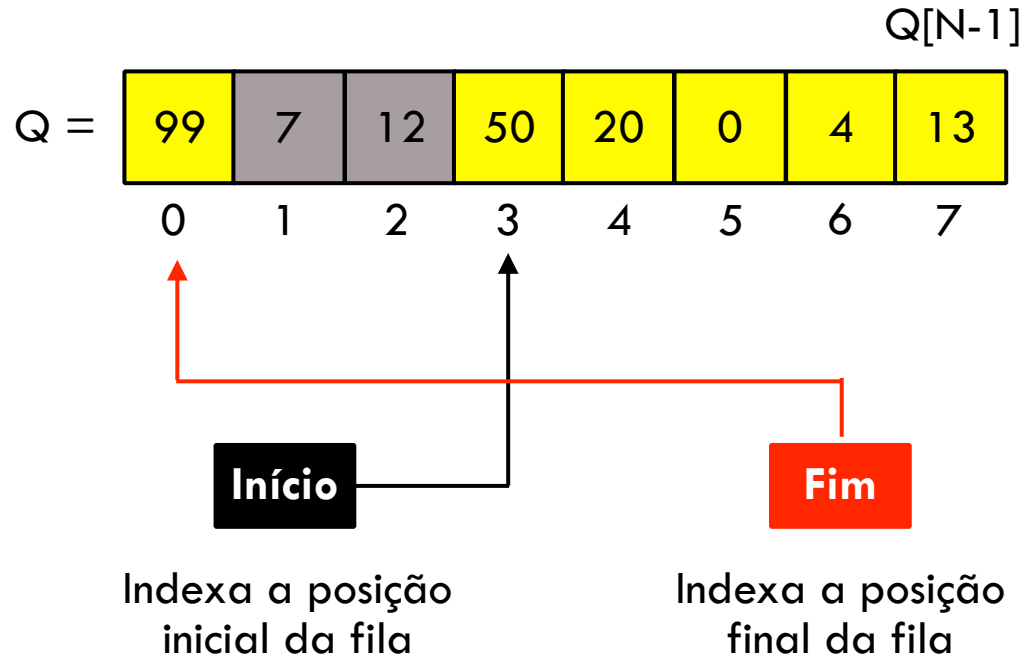
Indexa a posição
inicial da fila

Indexa a posição
final da fila

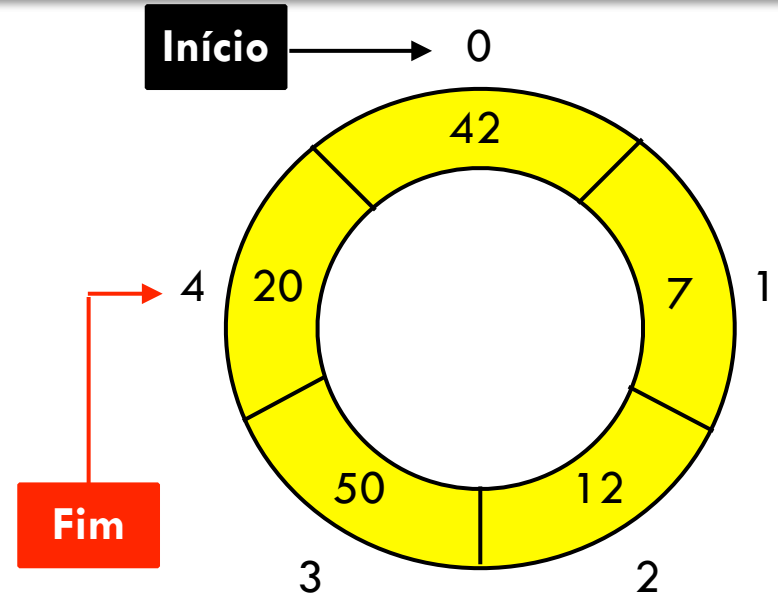
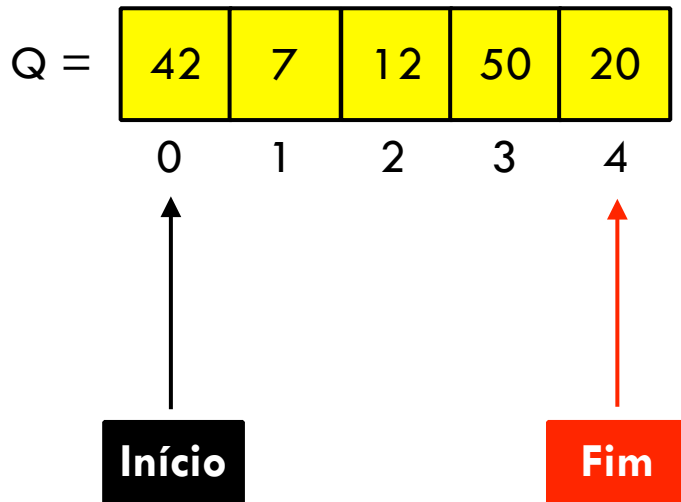
Fila estática

- Inserindo elemento 99

Número de elementos : 8



Fila estática



buffer circular

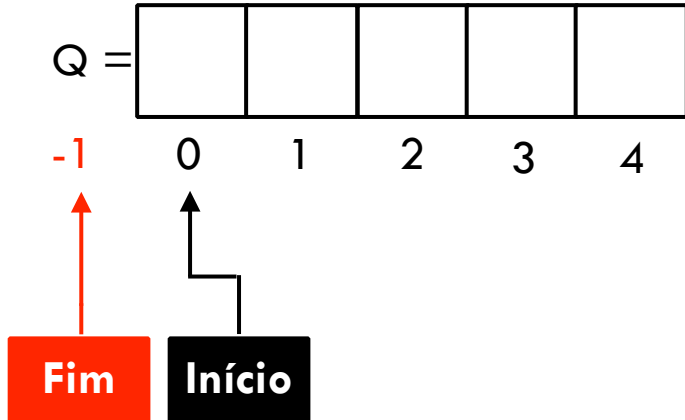
Roteiro

- 1 Introdução
- 2 Filas
- 3 Operações gerais
- 4 Inserção de elementos
- 5 Remoção de elementos
- 6 Referências

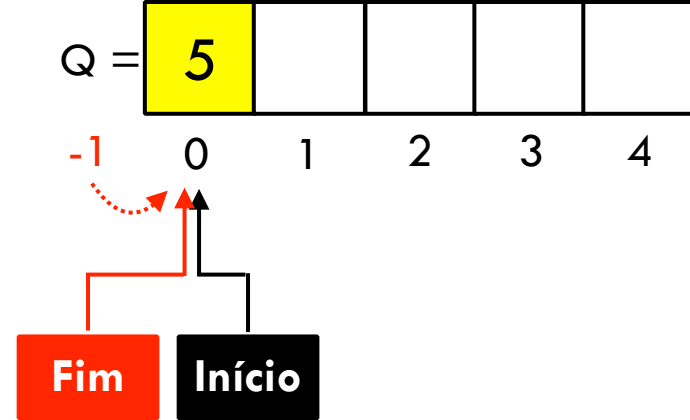
Enfileirar (enqueue)

- Inserir elemento $x = 5$

antes



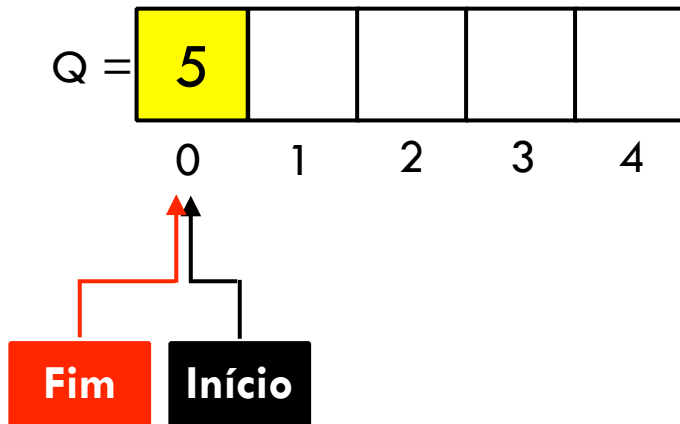
depois



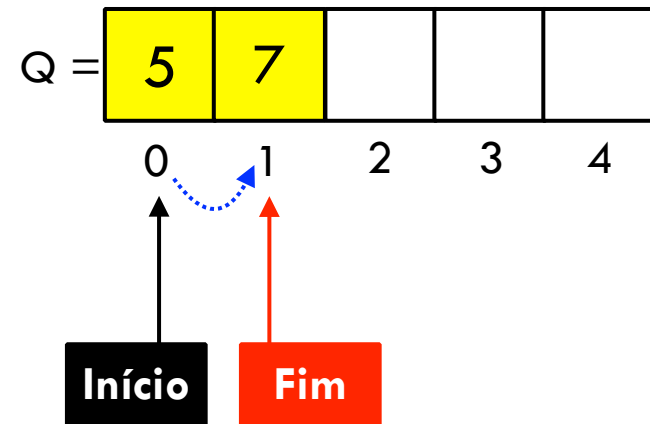
Enfileirar (enqueue)

- Inserir elemento $x = 20$

antes

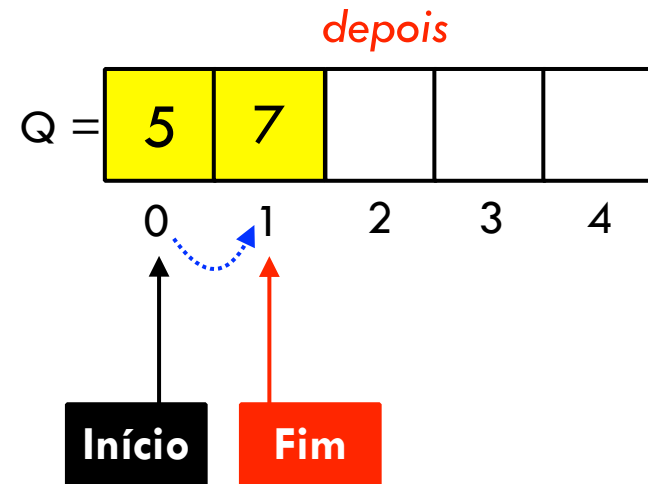
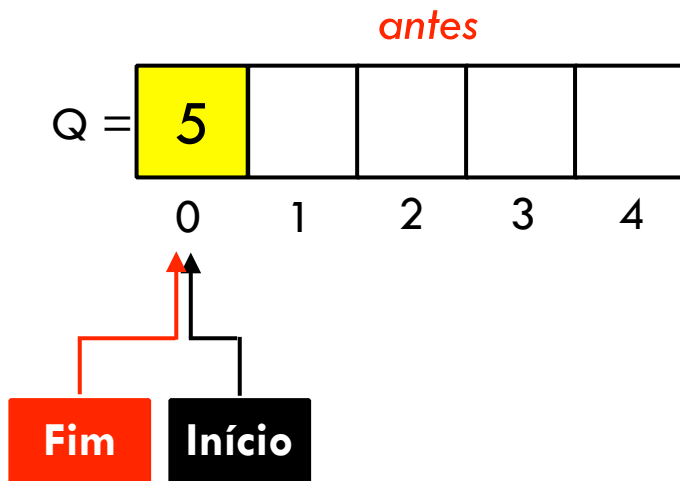


depois



Enfileirar (enqueue)

- Inserir elemento $x = 20$



O que aconteceu?

1. Incrementamos o **Fim**
2. Atribuímos o novo elemento na posição **Q[Fim]**
3. Incrementamos o contador de elementos

Enfileirar (enqueue)

Enqueue (Q, x)

1. se Q não está cheia:
2. Incrementar a variável Fim
3. Q[Fim] recebe x
4. Incrementa o contador

ou

Enqueue (Q, x)

1. if(estaCheia(Q)==0)
2. Q.fim = **incrementaIndice**(Q.fim);
3. Q.array[Q.fim] = x
4. Q.contador++;

Enfileirar (enqueue)

Enqueue (Q, x)

1. se Q não está cheia:
2. Incrementar a variável Fim
3. Q[Fim] recebe x
4. Incrementa o contador

ou

**Função auxiliar
comportamento circular**



Enqueue (Q, x)

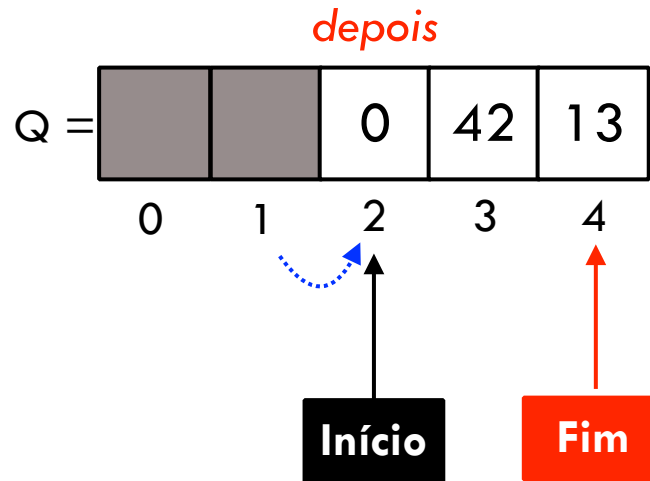
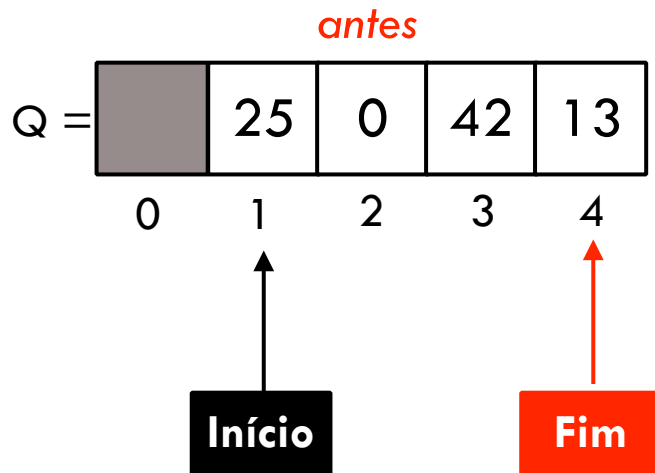
1. if(estaCheia(Q)==0)
2. Q.fim = **incrementaIndice**(Q.fim);
3. Q.array[Q.fim] = x
4. Q.contador++;

Roteiro

- 1 Introdução
- 2 Filas
- 3 Operações gerais
- 4 Inserção de elementos
- 5 Remoção de elementos
- 6 Referências

Desenfileirar (dequeue)

- Remover elemento

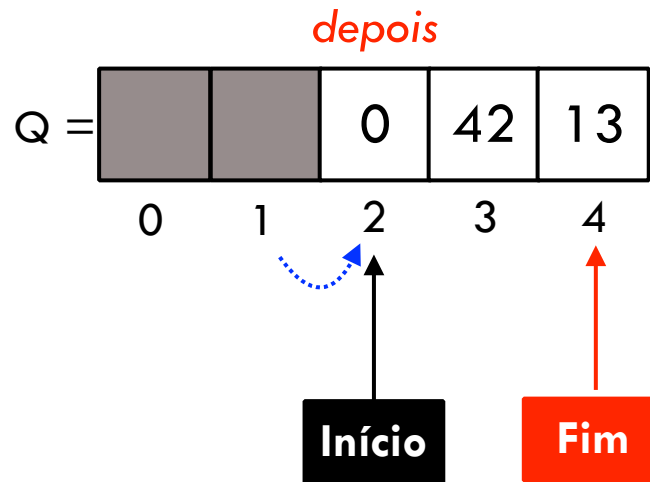
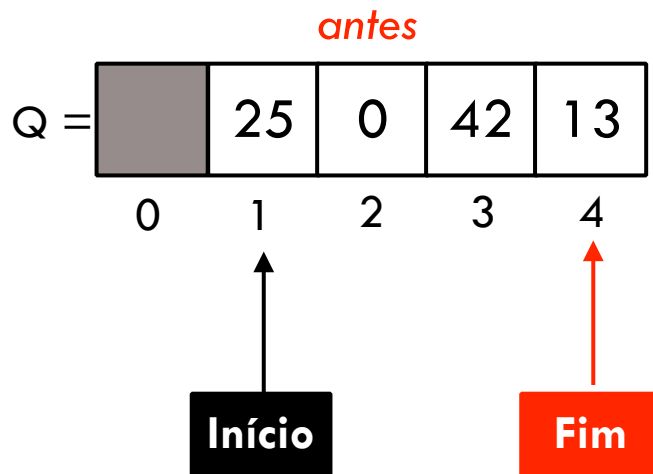


elemento
retornado

25

Desenfileirar (dequeue)

- Remover elemento



O que aconteceu?

1. Salvamos o valor da posição **Q[Início]** em uma variável auxiliar
2. Incrementamos o valor da variável **Início**
3. Decrementamos o contador de elementos
4. Retornamos o valor da variável auxiliar

25

elemento
retornado

Desenfileirar (dequeue)

Dequeue (Q)

1. se Q não está vazia:
2. variável auxiliar **aux** recebe Q[Q.Inicio]
3. Incrementa o valor de Q.Inicio
4. Decrementa o contador de elementos
5. Retorna **aux**

ou

Dequeue (Q)

1. if(estaVazia(Q) == 0):
2. **aux** = Q.array[Q.Inicio];
3. Q.Inicio = **incrementaIndice**(Q.Inicio);
4. Q.contador = Q.contador - 1;
5. return(**aux**);

Desenfileirar (dequeue)

Dequeue (Q)

1. se Q não está vazia:
2. variável auxiliar **aux** recebe Q[Q.Inicio]
3. Incrementa o valor de Q.Inicio
4. Decrementa o contador de elementos
5. Retorna **aux**

ou

**Função auxiliar
comportamento circular**



Dequeue (Q)

1. if(estaVazia(Q) == 0):
2. **aux** = Q.array[Q.Inicio];
3. Q.Inicio = **incrementaIndice**(Q.Inicio);
4. Q.contador = Q.contador - 1;
5. return(aux);

Funções adicionais?



- Quais outras funções podem ser úteis para o tipo Fila?

Exercício 01

- Ilustre cada estado de uma fila após realizar as seguintes operações (em ordem)
 - Enqueue(Q, 4)
 - Enqueue(Q, 1)
 - Enqueue(Q, 3)
 - Dequeue(Q)
 - Enqueue(Q, 8)
 - Dequeue(Q)
- Considere que a fila está inicialmente vazia e é armazenada em um arranjo $Q[1 \dots 6]$

Exercício 02

- Mãos a obra: implemente um TDA para Fila com alocação estática, e as funções de manipulação.
- Quais TDAs serão necessários?

Tipos Abstratos para Fila Estática

```
/* manipulando inteiros */
```

```
typedef struct {  
    int array[MAXTAM];  
    int inicio;  
    int fim;  
    int tamanho;  
} filaEstatica;
```

Tipos Abstratos para Fila Estática

```
void iniciaFila(filaEstatica *fila);
void enfileira(int obj, filaEstatica *fila);
int desenfileira(filaEstatica *fila);
void imprimeFila(filaEstatica *fila);
int incrementaIndice(int i);
bool estaVazia(filaEstatica *fila);
bool estaCheia(filaEstatica *fila);
int tamanhoFila(filaEstatica *fila);
int primeiro(filaEstatica *fila);
int ultimo(filaEstatica *fila);
```

Tipos Abstratos para Fila Estática

```
/* manipulando objetos */  
typedef struct {  
    int key;  
} Objeto;  
  
typedef struct {  
    Objeto array[MAXTAM];  
    int inicio;  
    int fim;  
    int tamanho;  
} filaEstatica;
```

Tipos Abstratos para Fila Estática

```
void iniciaFila(filaEstatica *fila);
void enfileira(Objeto obj, filaEstatica *fila);
Objeto desenfileira(filaEstatica *fila);
void imprimeFila(filaEstatica *fila);
int incrementaIndice(int i);
bool estaVazia(filaEstatica *fila);
bool estaCheia(filaEstatica *fila);
int tamanhoFila(filaEstatica *fila);
Objeto primeiro(filaEstatica *fila);
Objeto ultimo(filaEstatica *fila);
```

Próximas Aulas

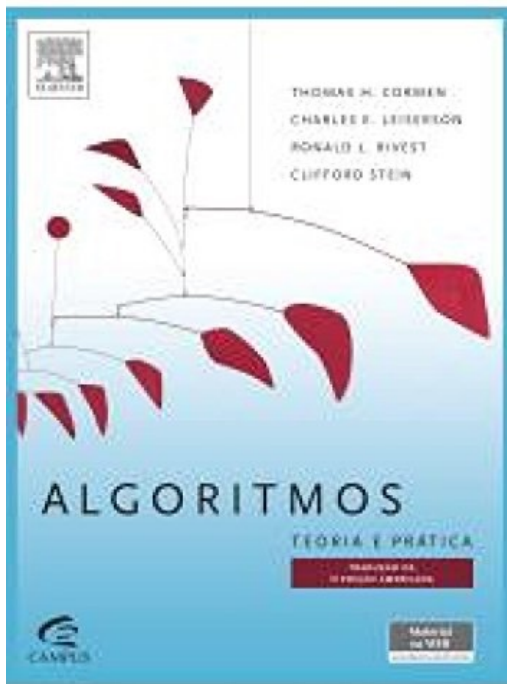


- Filas → implementação dinâmica
- Listas Lineares
 - single-linked
 - double-linked

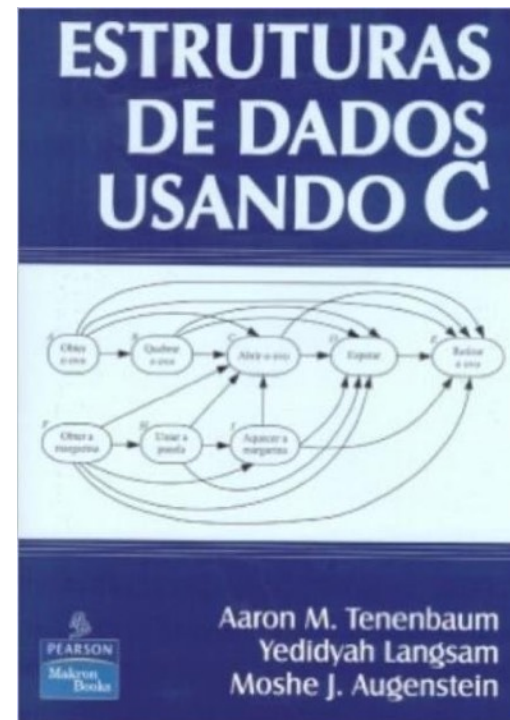
Roteiro

- 1 Introdução
- 2 Filas
- 3 Operações gerais
- 4 Inserção de elementos
- 5 Remoção de elementos
- 6 Referências

Referências sugeridas

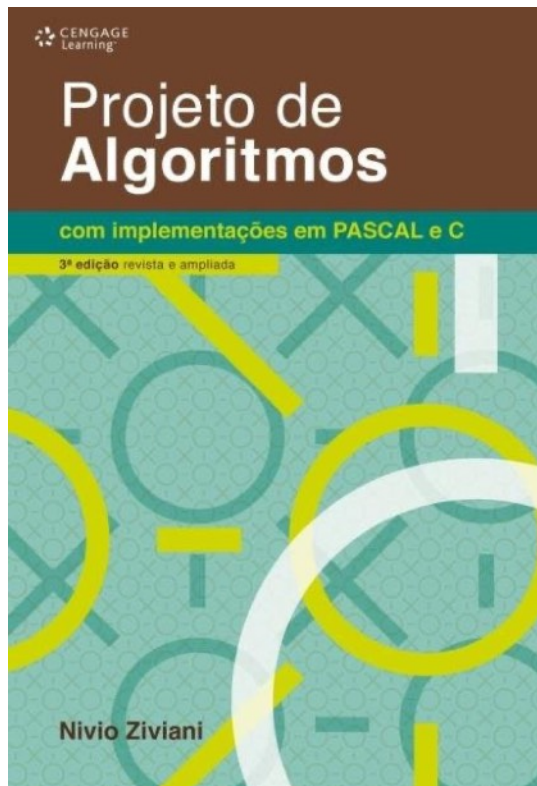


[Cormen et al, 2018]



[Tenenbaum et al, 1995]

Referências sugeridas



[Ziviani, 2010]



[Drozdek, 2017]

Perguntas?

Prof. Rafael G. **Mantovani**

rafaelmantovani@utfpr.edu.br