Estrutura de Dados

Filas Dinâmicas

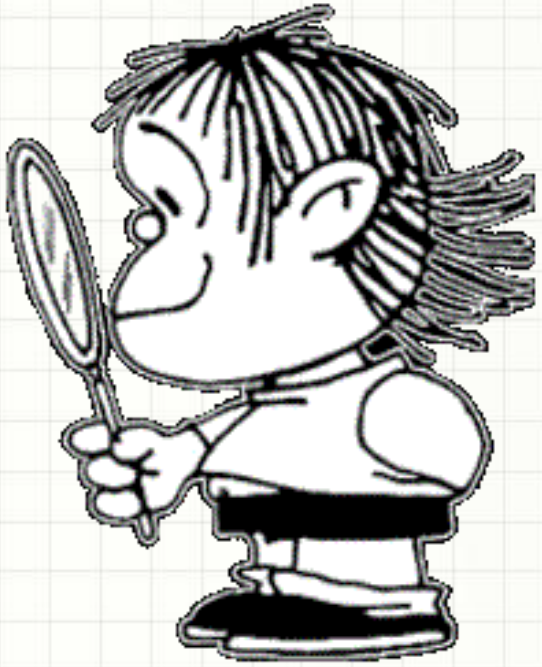
Prof. Dr. Daniel Caetano

2014 - 2

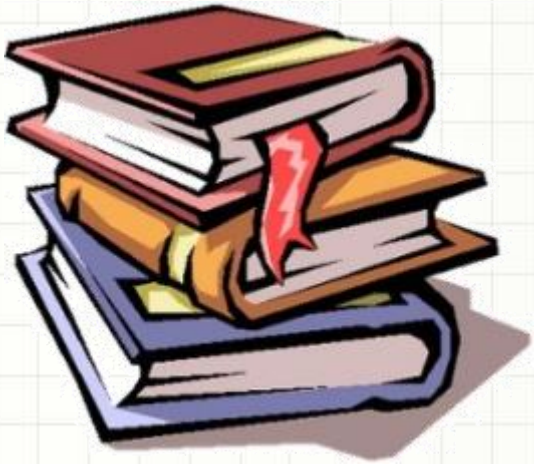
Objetivos

- Compreender o conceito de fila encadeada
- Usar filas dinâmicas para aplicações
- Conhecer as vantagens e desvantagens das filas encadeadas

- **Atividade Estruturada!**



Material de Estudo



Material	Acesso ao Material
Apresentação	http://www.caetano.eng.br/ (Aula 14)
Material Didático	Estruturas de Dados (Parte 2) – Páginas 143 a 146
Online	C Completo e Total – Páginas 540 a 550

RECORDANDO...

Recordando...

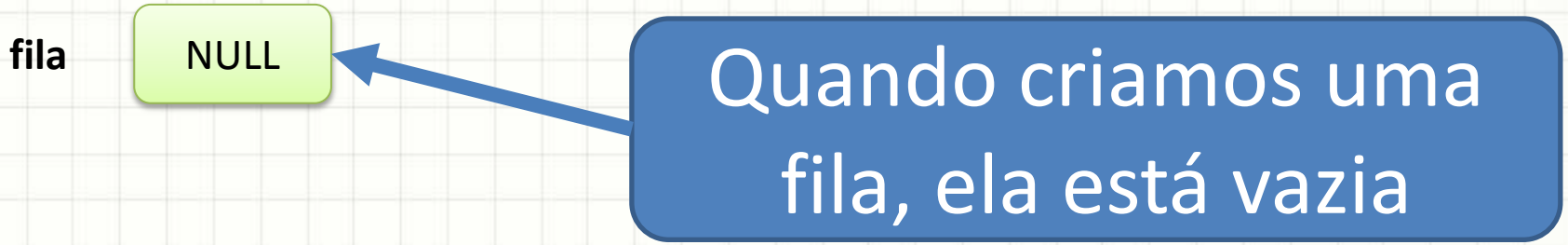
- Vimos Pilhas Encadeadas
- Uso:
  - Inserção no Topo (Início)
  - Remoção do Topo (Início)
- E se continuarmos removendo do início, mas inserirmos no fim?
- Virou uma fila?

Fila Encadeada ou Fila Dinâmica

Filas Encadeadas

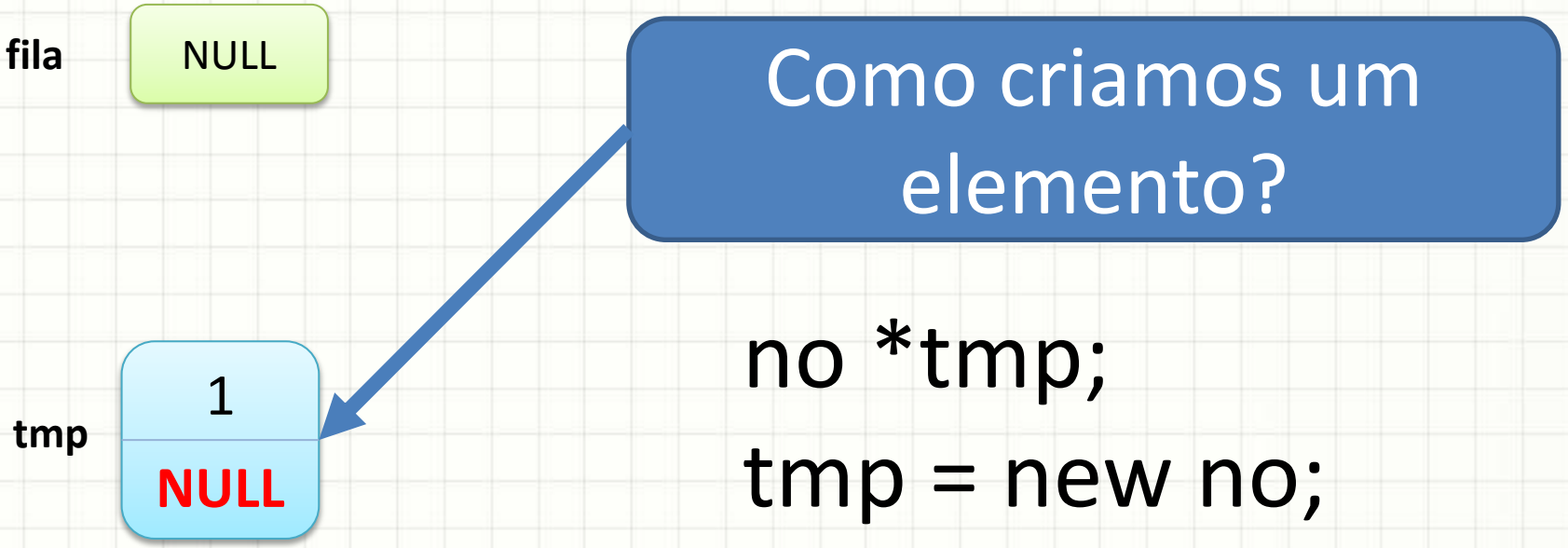
- Sempre que usarmos uma lista encadeada
  - Inserirmos no fim... e
  - Removermos do início...
- Fila Dinâmica ou Encadeada
- Exemplo

Fila: Inicializando



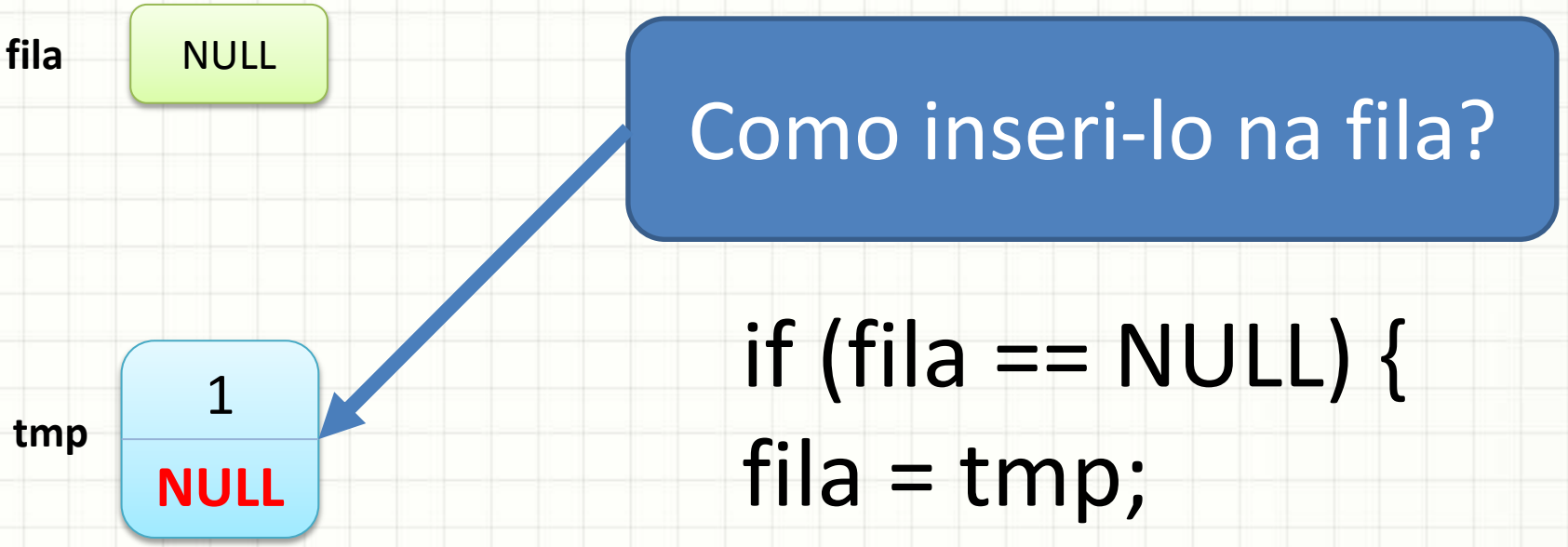
no *fila = NULL;

Fila: Inserção

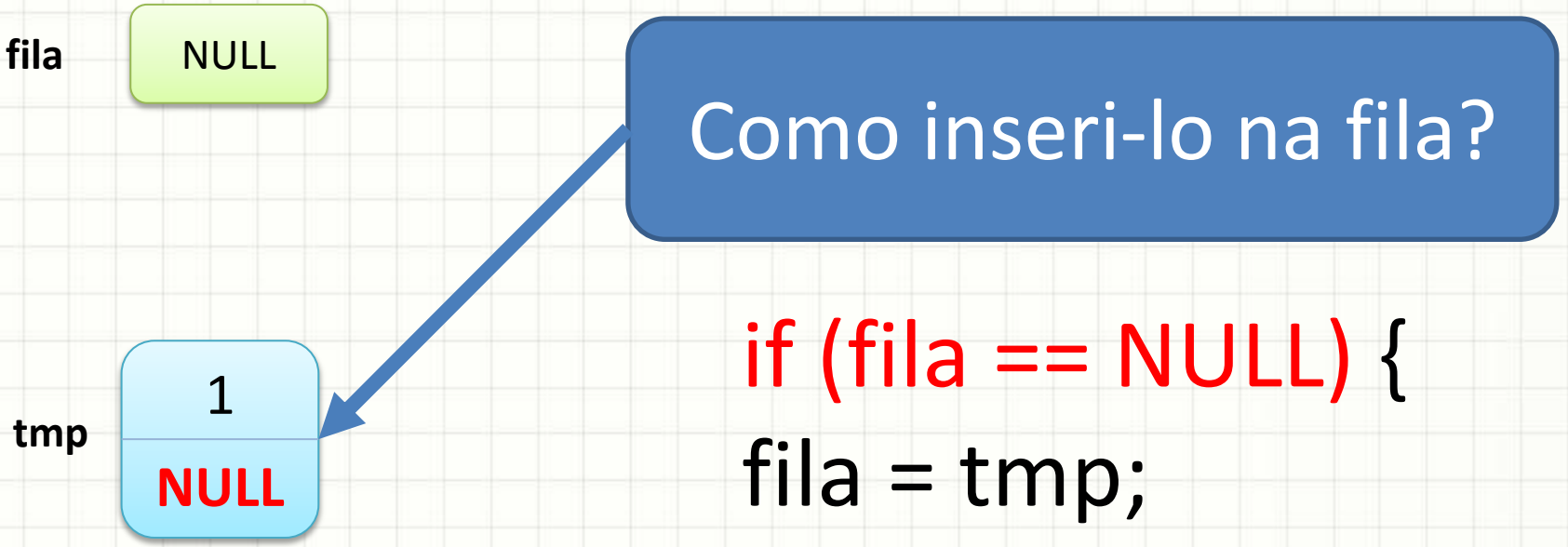



```
no *tmp;
tmp = new no;
tmp->valor = 1;
tmp->prox = NULL
```

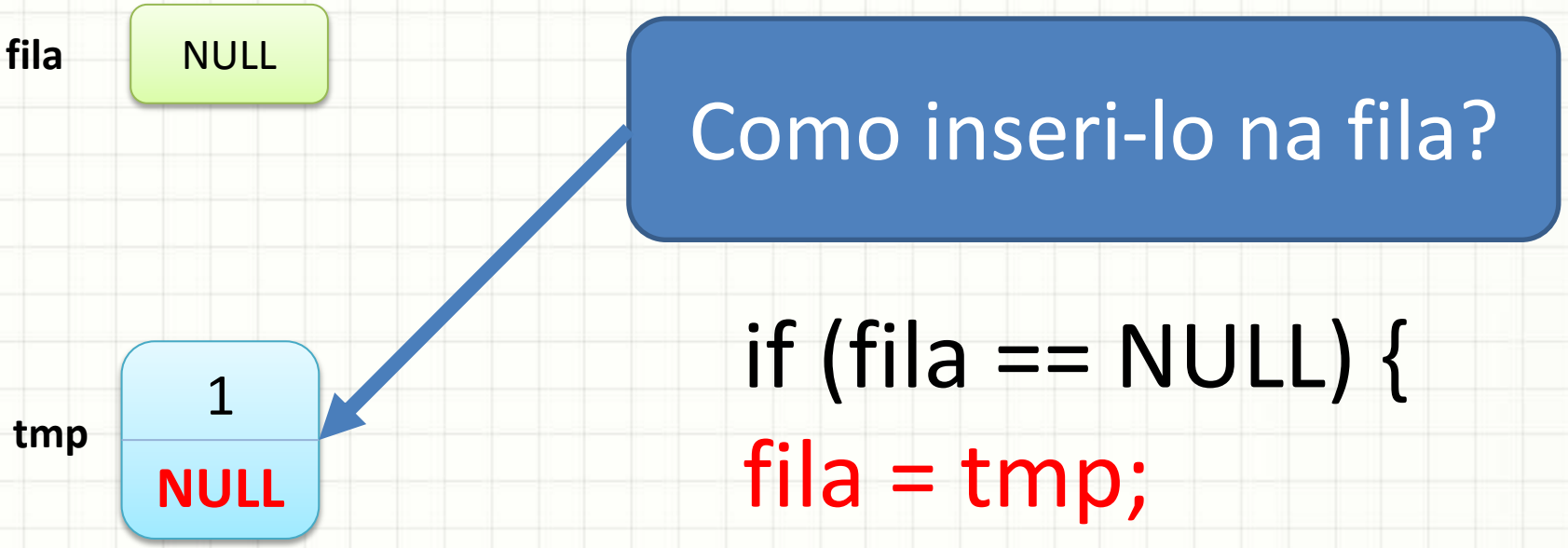
Fila: Inserção



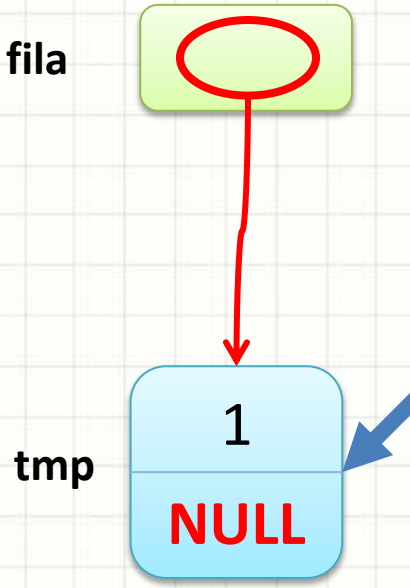
Fila: Inserção



Fila: Inserção



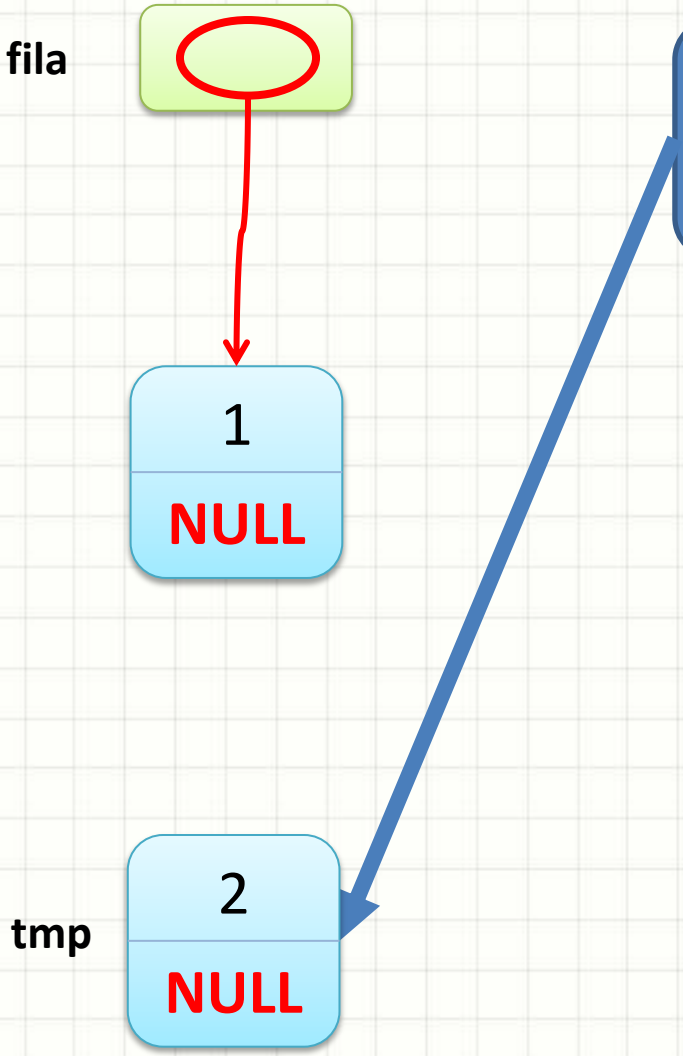
Fila: Inserção



Como inseri-lo na fila?

```
if (fila == NULL) {
fila = tmp;
}
```

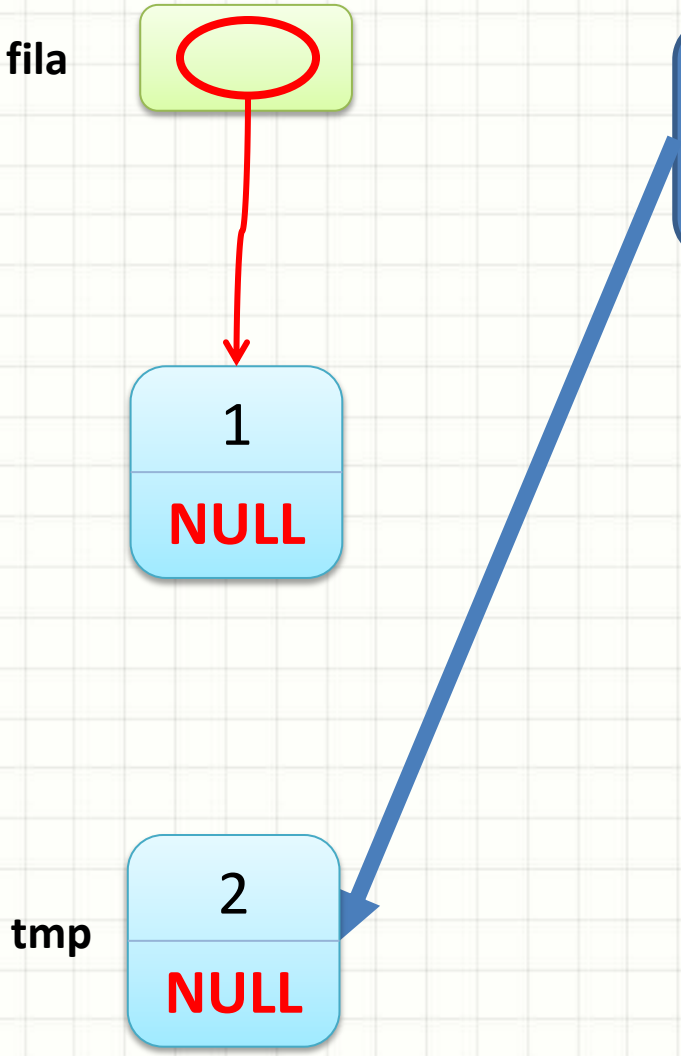
Fila: Inserção



Inserindo outro nó

```
if (fila != NULL) {
fim = fila;
while (fim->prox != NULL) {
        fim = fim->prox;
        }
fim->prox = tmp;
}
```

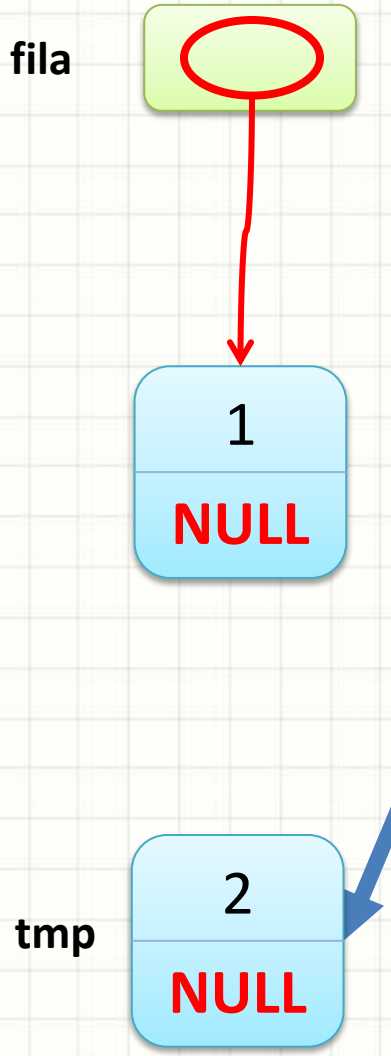
Fila: Inserção



Inserindo outro nó

```
if (fila != NULL) {
fim = fila;
while (fim->prox != NULL) {
        fim = fim->prox;
        }
fim->prox = tmp;
}
```

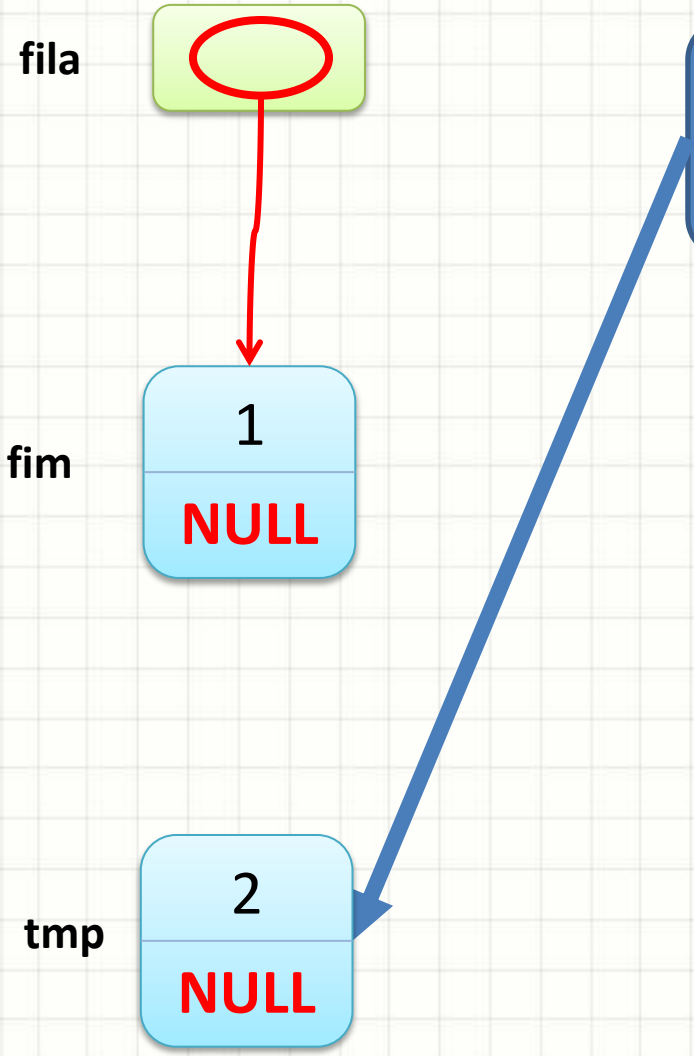
Fila: Inserção



Inserindo outro nó

```
if (fila != NULL) {
fim = fila;
while (fim->prox != NULL) {
        fim = fim->prox;
        }
fim->prox = tmp;
}
```

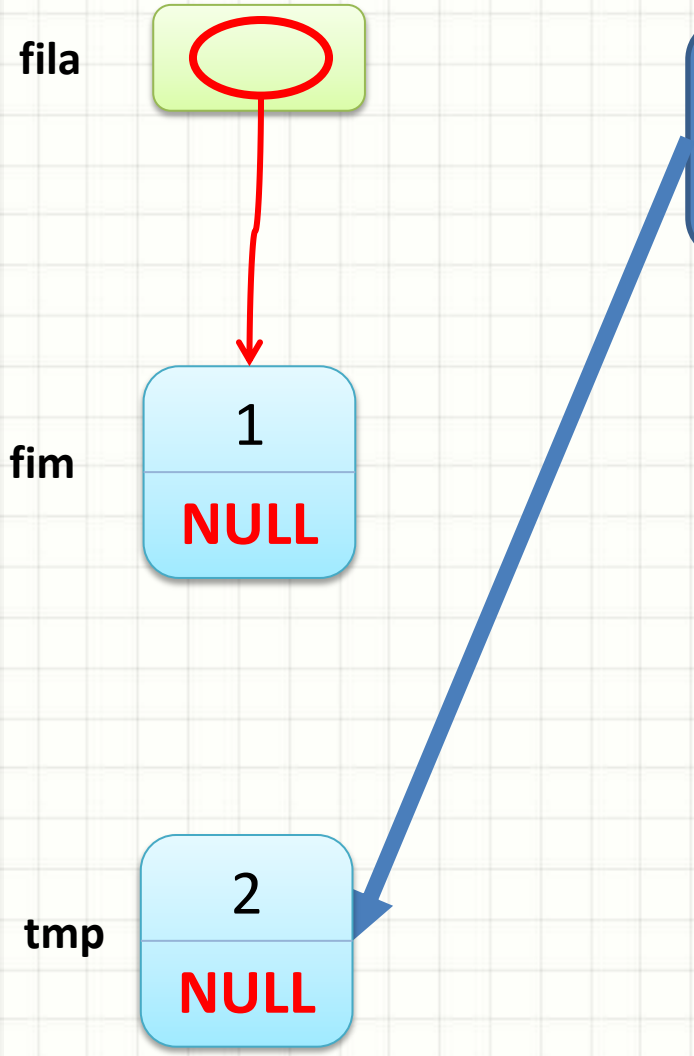
Fila: Inserção



Inserindo outro nó

```
if (fila != NULL) {
fim = fila;
while (fim->prox != NULL) {
        fim = fim->prox;
        }
fim->prox = tmp;
}
```

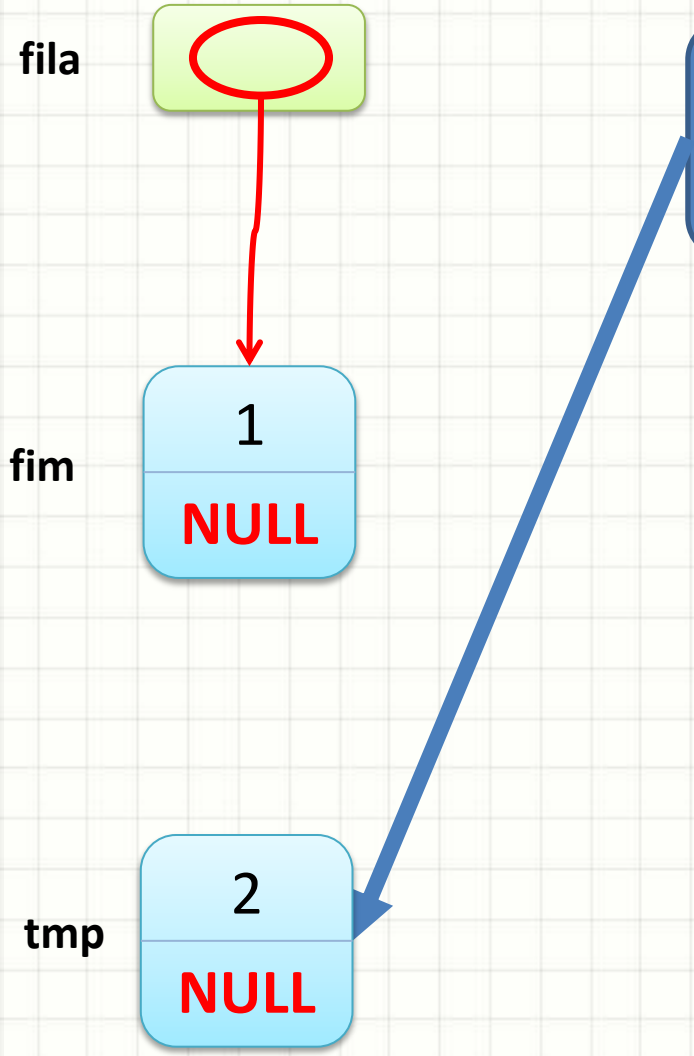
Fila: Inserção



Inserindo outro nó

```
if (fila != NULL) {
fim = fila;
while (fim->prox != NULL) {
        fim = fim->prox;
        }
fim->prox = tmp;
}
```

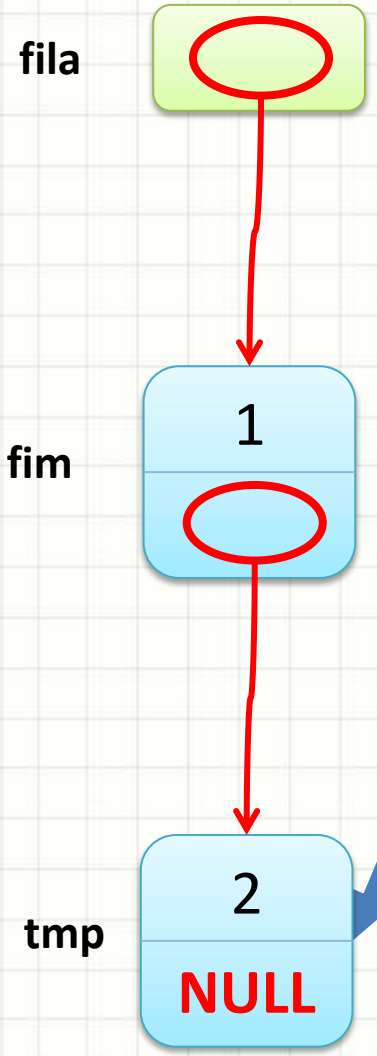
Fila: Inserção



Inserindo outro nó

```
if (fila != NULL) {
fim = fila;
while (fim->prox != NULL) {
        fim = fim->prox;
        }
fim->prox = tmp;
}
```

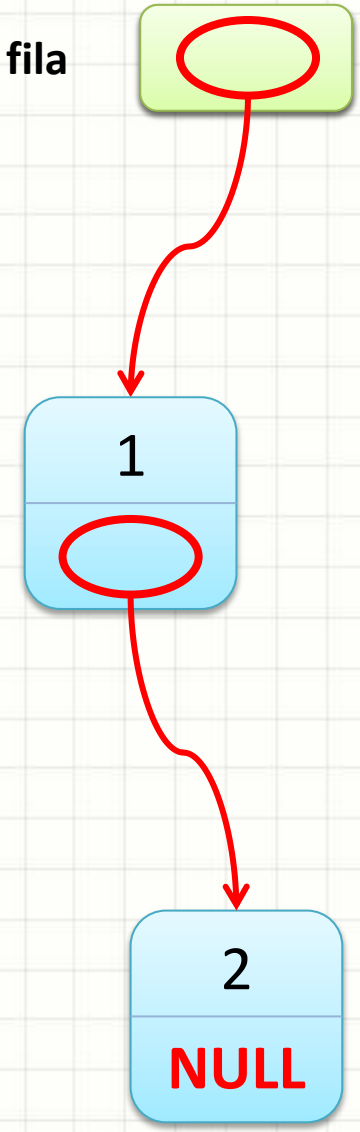
Fila: Inserção



Inserindo outro nó

```
if (fila != NULL) {
fim = fila;
while (fim->prox != NULL) {
        fim = fim->prox;
        }
fim->prox = tmp;
}
```

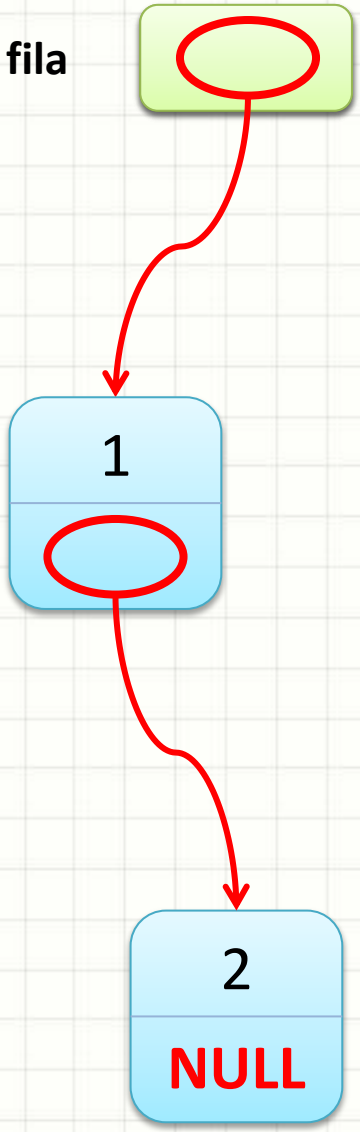
Fila: Remoção



Removendo um nó

```
no *tmp;
tmp = fila;
fila = tmp->prox;
valor = tmp->valor;
delete tmp;
```

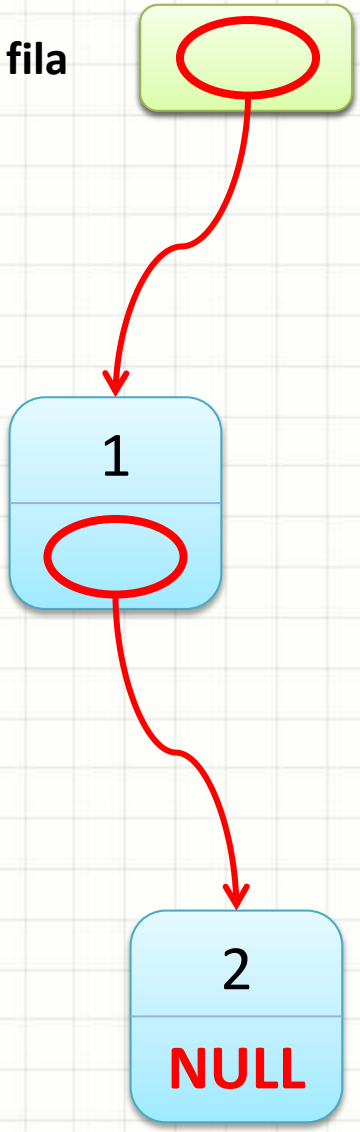
Fila: Remoção



Removendo um nó

```
no *tmp;
tmp = fila;
fila = tmp->prox;
valor = tmp->valor;
delete tmp;
```

Fila: Remoção

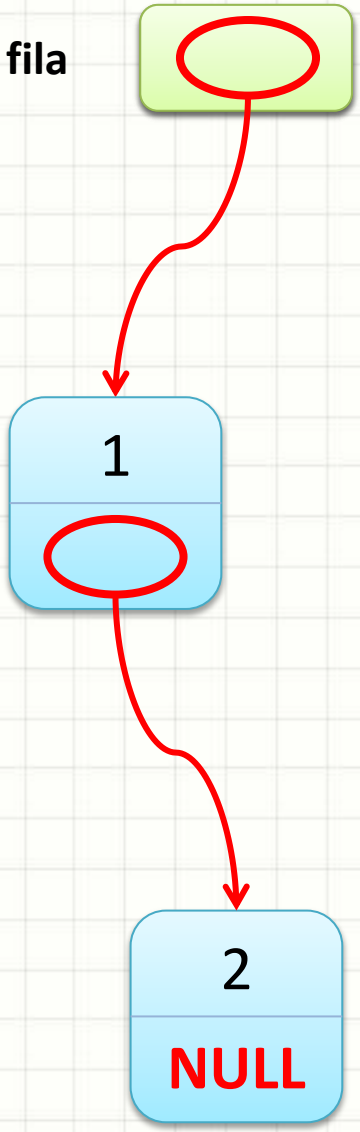


tmp

Removendo um nó

```
no *tmp;
tmp = fila;
fila = tmp->prox;
valor = tmp->valor;
delete tmp;
```

Fila: Remoção

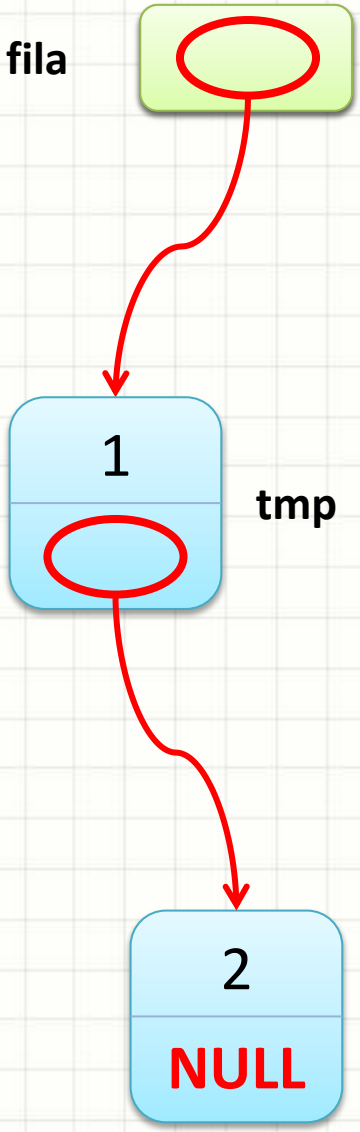


tmp

Removendo um nó

```
no *tmp;
tmp = fila;
fila = tmp->prox;
valor = tmp->valor;
delete tmp;
```

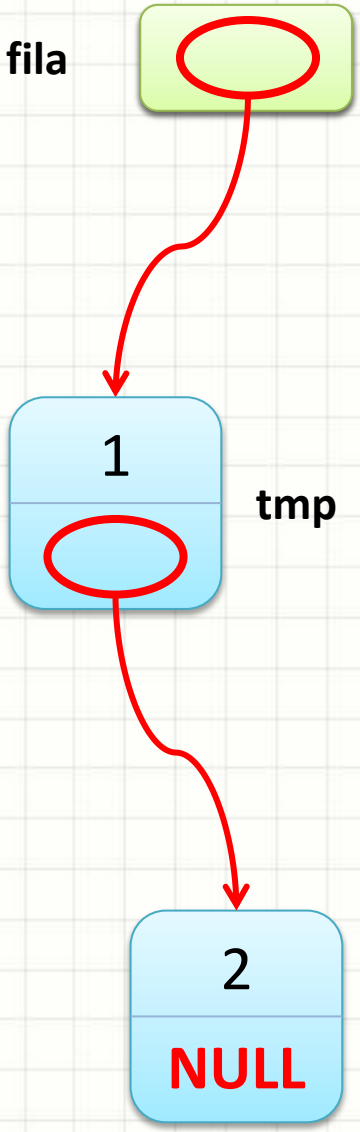
Fila: Remoção



Removendo um nó

```
no *tmp;
tmp = fila;
fila = tmp->prox;
valor = tmp->valor;
delete tmp;
```

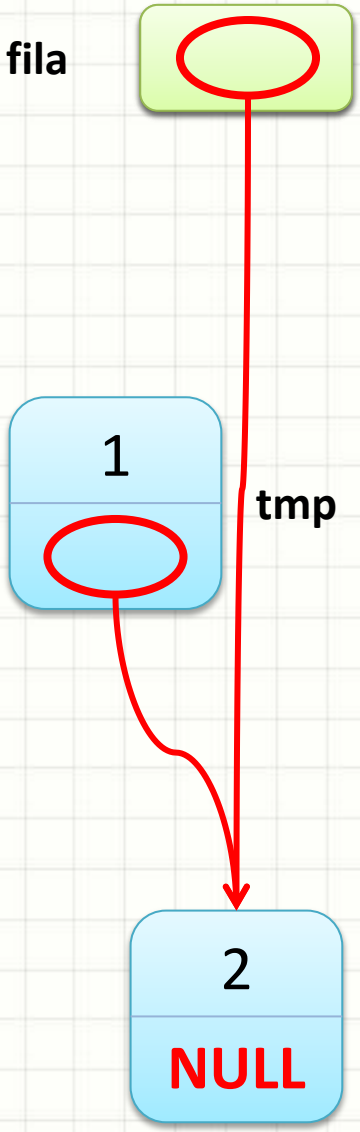
Fila: Remoção



Removendo um nó

```
no *tmp;
tmp = fila;
fila = tmp->prox;
valor = tmp->valor;
delete tmp;
```

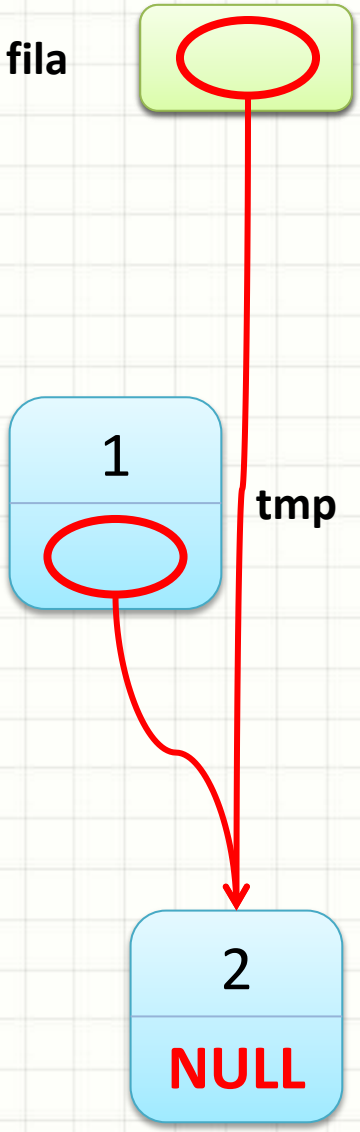
Fila: Remoção



Removendo um nó

```
no *tmp;
tmp = fila;
fila = tmp->prox;
valor = tmp->valor;
delete tmp;
```

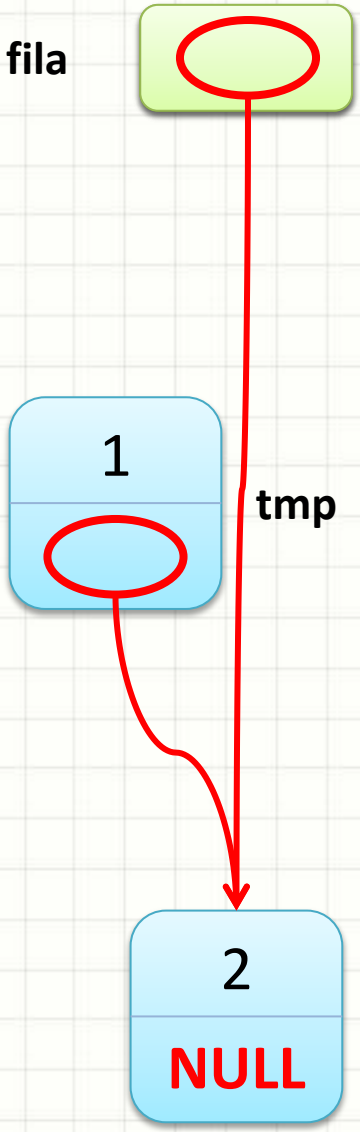
Fila: Remoção



Removendo um nó

```
no *tmp;
tmp = fila;
fila = tmp->prox;
valor = tmp->valor;
delete tmp;
```

Fila: Remoção

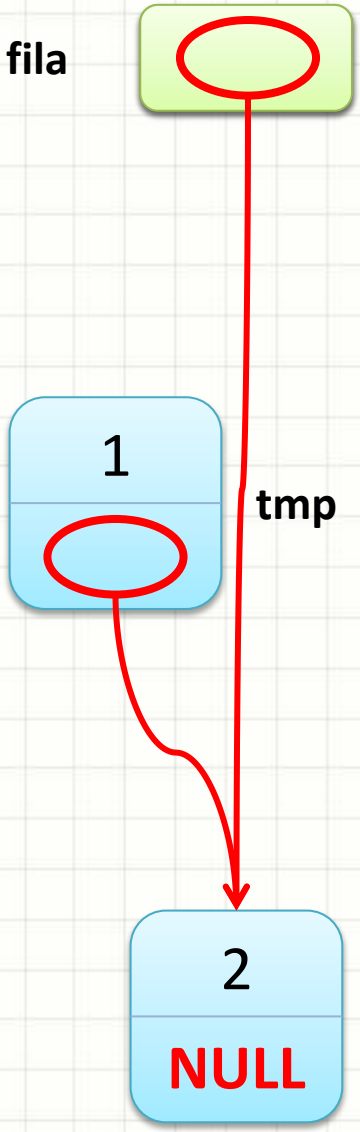


Removendo um nó

1

```
no *tmp;
tmp = fila;
fila = tmp->prox;
valor = tmp->valor;
delete tmp;
```

Fila: Remoção



Removendo um nó

```
no *tmp;
tmp = fila;
fila = tmp->prox;
valor = tmp->valor;
delete tmp;
```

Fila: Remoção

Removendo um nó

```
no *tmp;
tmp = fila;
fila = tmp->prox;
valor = tmp->valor;
delete tmp;
```

Discussão: Qualidade da Implementação

Qualidade da Implementação

- Qual o problema dessa implementação?
  - Pense na inserção e remoção...
  - Há como melhorar?

Prática:
Construindo uma Fila Encadeada

Construindo uma Pilha Encadeada

- Acompanhe o professor na construção...

EXERCÍCIOS DE FIXAÇÃO

Exercícios

- Como base em “ex1.cpp” e “filaaluno.cpp”:
  - Modifique ex1.cpp para que implemente uma fila de alunos
- Em grupos de 4 alunos, debata as vantagens e desvantagens do uso de encadeamento;
  - Elabore uma lista de vantagens e desvantagens comparando as estruturas de dados contíguas com suas versões encadeadas: pilhas, filas e listas

Vantagens e Desvantagens

	Vantagens	Desvantagens
Encadeados Em geral		
Pilhas Encadeadas		
Filas Encadeadas		
Listas Encadeadas		

Vantagens e Desvantagens

	Vantagens	Desvantagens
Encadeados Em geral	- Menor consumo de memória - Menor processamento em tarefas complexas - Tamanho máximo limitado pela memória do equipamento - Protótipos mais simples	- Maior complexidade de programação - Uso de ponteiros - Maior processamento em tarefas simples - Consumo de processamento para alocação de memória
Pilhas Encadeadas	- Apenas um apontador	
Filas Encadeadas	- Não exige implementação circular - Apenas dois apontadores	- Se não for usado apontador de fim, custo para inserção cresce com o tamanho da fila
Listas Encadeadas	- Inserção ordenada e ordenação mais eficientes - Apenas um apontador	- Acesso aleatório prejudicado (busca)

Conclusões

Resumo

- As listas encadeadas podem ser usadas como filas
- É mais interessante criar uma estrutura específica, com referências para início e fim
- Adaptar programas que usam filas contíguas para usar filas encadeadas é simples

PERGUNTAS?