

Algoritmos e Estrutura de Dados



Aula 09 – Estrutura de Dados: Filas
e Pilhas

Prof. Tiago A. E. Ferreira

Fila

□ Definição

- Estrutura de dados comumente chamada de FIFO = *First In-First Out*.
- O primeiro elemento a entrar na fila será o primeiro elemento a sair da fila



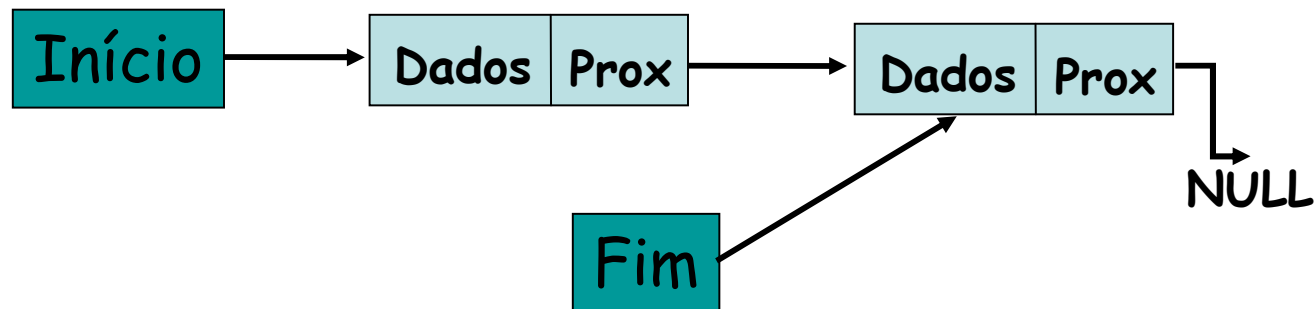
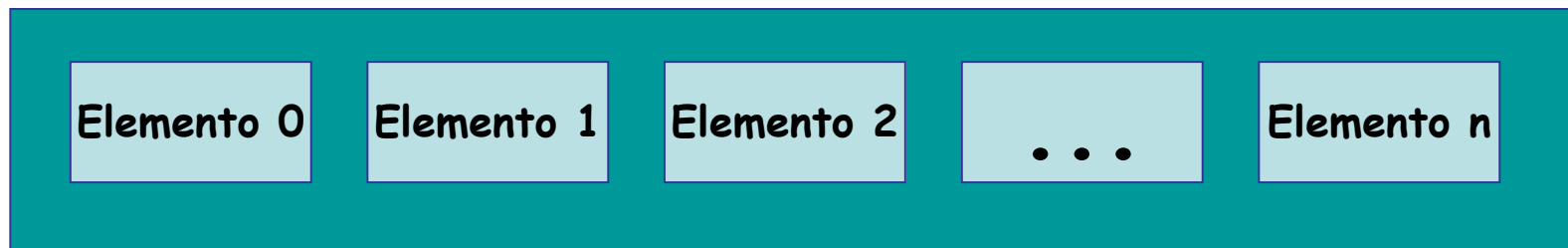
Filas - Aplicações

- ❑ Fila de processos do sistema operacional
- ❑ Fila de um banco
- ❑ Fila para o *check-in* de um vôo
- ❑ Tratamento de teclas acionadas no teclado do computador

Filas - Implementação

- Uma fila pode ser implementada usando uma lista dinâmica com as seguintes características:
 - O novo elemento sempre é inserido no final (ordem de chegada)
 - O elemento removido é sempre o que chegou há mais tempo na fila
 - A consulta retorna o primeiro elemento da fila

Filas – Representação Gráfica



Filas – Operações Básicas

- ❑ Criar a Fila
- ❑ Inserir Elemento
- ❑ Remover Elemento
- ❑ Consultar Primeiro Elemento
- ❑ Listar Todos Elementos

Criando uma Fila

- Inicialmente, declara-se dois ponteiros: um para o início e outro para o fim da fila.
 - O ponteiro para o fim da fila permite realizar inserções sem que seja necessário percorrer toda a fila.
 - No início, como a fila está vazia, ambos apontam para **NULL** ou **None**.

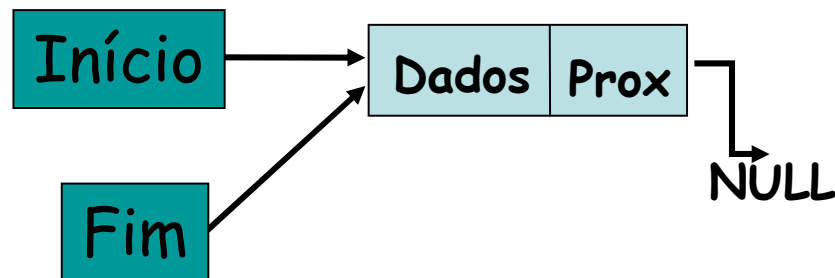


Inserindo Elemento no Lista

□ Inserindo Elementos

■ Caso 1: Fila vazia

- Cria-se o novo elemento que aponta para **NULL** ou **None**;
- Ponteiros de início e fim apontam para o novo elemento

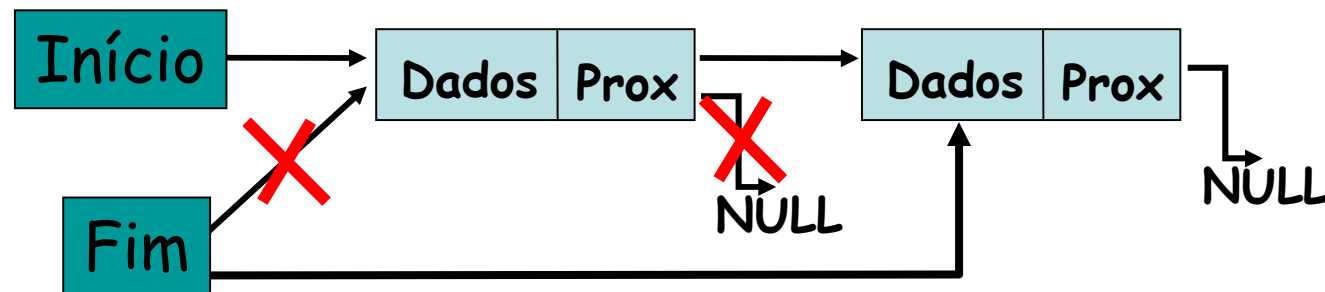


Inserindo Elemento no Lista

□ Inserindo Elementos

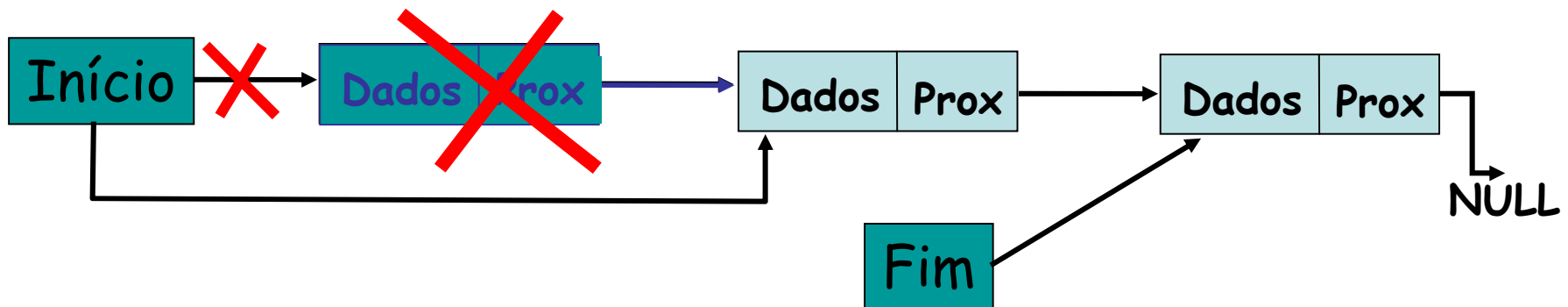
■ Caso 2: Fila com pelo menos 1 elemento

- Cria-se o novo elemento que aponta para **NULL** ou **None**;
- O último elemento da fila aponta para o novo nó;
- O ponteiro de fim aponta para o novo elemento.



Removendo Primeiro Elemento da Fila

- Removendo o Primeiro Elemento
 - O elemento a ser removido é marcado.
 - O ponteiro do início aponta para o próximo elemento.
 - A memória é liberada.



Lista Dinâmica - Inserção

□ Inserindo Elementos

- **Caso 1:** Lista Vazia

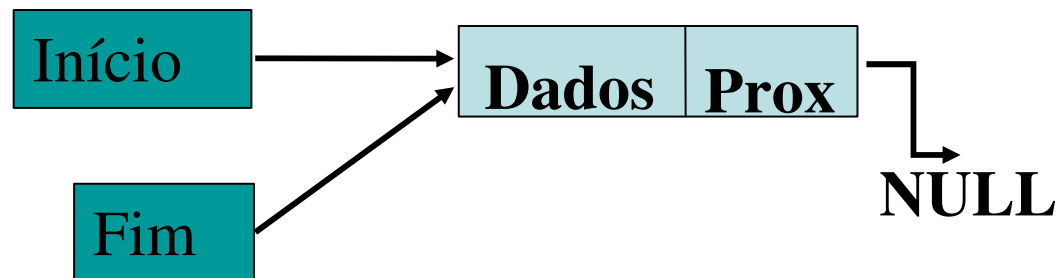
- Cria-se o novo nó e os dois ponteiros apontam para o novo nó inserido na lista, que por sua vez aponta para NULL.

- **Caso 2:** Lista Não Vazia – Inserção no final da lista

- Cria-se o novo nó que aponta para NULL; o último nó da lista aponta para o novo nó; e, ponteiro de fim aponta para o novo nó

Lista Dinâmica - Inserção

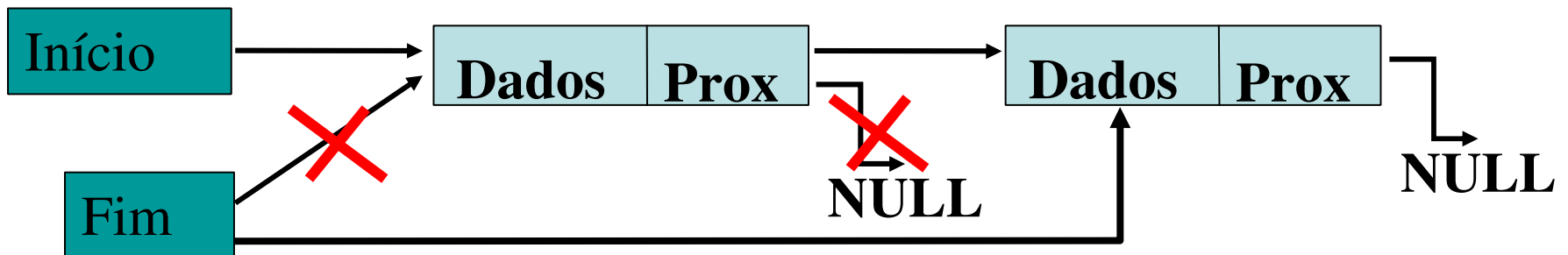
- **Inserindo Elementos**
 - **Caso 1: Lista Vazia:**



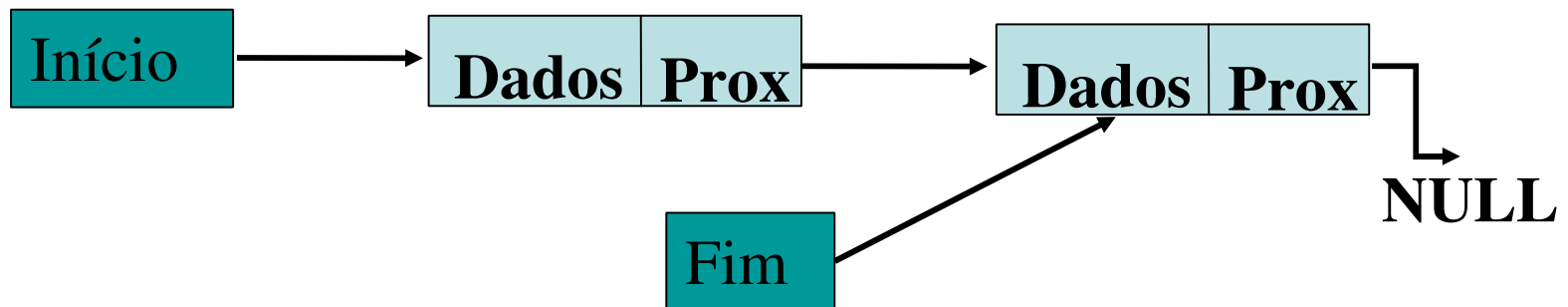
Lista Dinâmica - Inserção

□ Inserindo Elementos

- Caso 2: Lista Não Vazia – Inserção no final da Lista



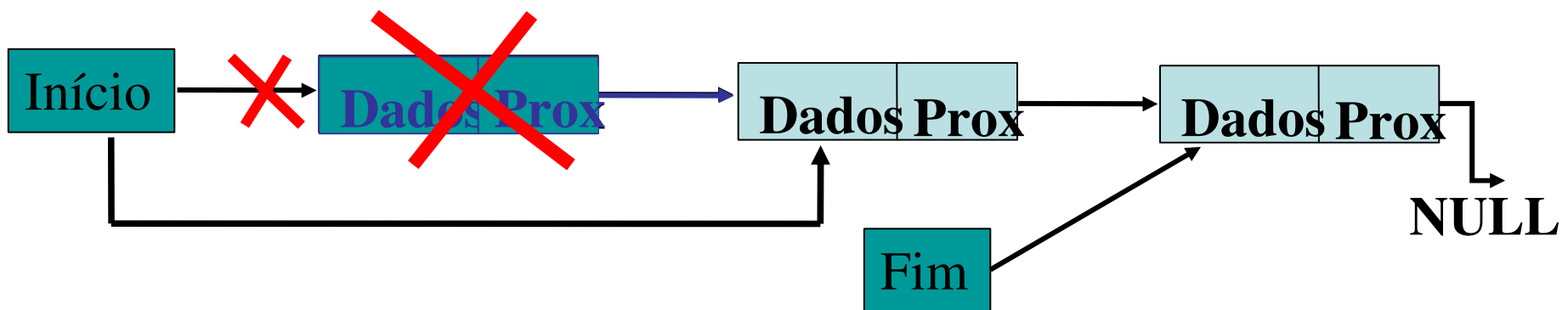
- Resultado Final:



Lista Encadeada - Remoção

Removendo Elementos

- **Caso 1:** Remover primeiro elemento da lista
 - O elemento a ser removido é marcado.
 - O ponteiro do início aponta para o próximo elemento.
 - A memória é liberada

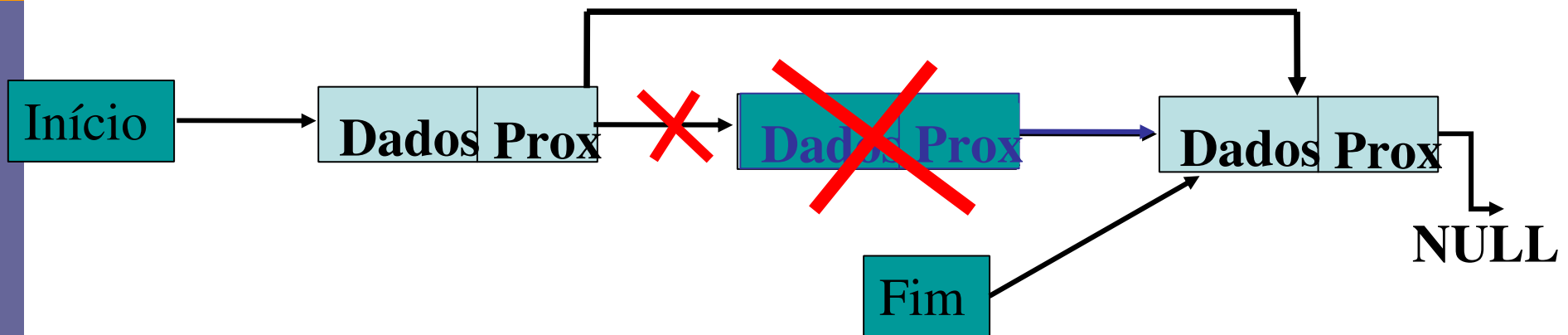


Lista Encadeada - Remoção

- Removendo Elementos

- Caso 2: Remover elemento do meio da lista

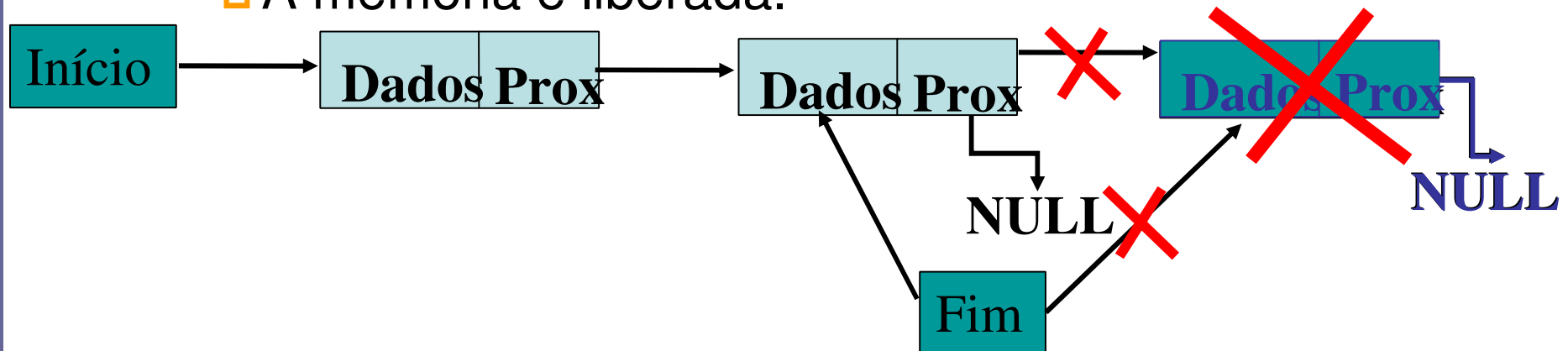
- ▣ O elemento a ser removido é marcado.
- ▣ O elemento anterior ao removido aponta para onde o removido apontava.
- ▣ A memória é liberada.



Lista Encadeada - Remoção

Removendo Elementos

- Caso 3: Remover elemento do final da lista
 - O elemento a ser removido é marcado.
 - O elemento anterior ao removido aponta para NULL.
 - O ponteiro para fim aponta para o anterior.
 - A memória é liberada.

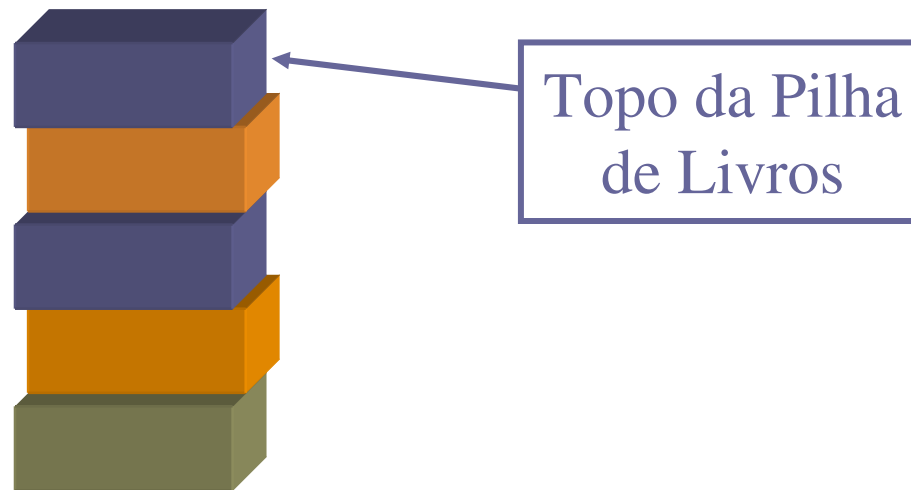


Pilha – Definição

- Uma pilha pode ser implementada usando uma lista ligada com as seguintes características:
 - O novo elemento sempre é inserido no topo da pilha (ordem de chegada)
 - O elemento removido é sempre o que chegou a menos tempo na pilha (o último inserido)
 - A consulta retorna o elemento no topo da pilha

Pilha - Definição

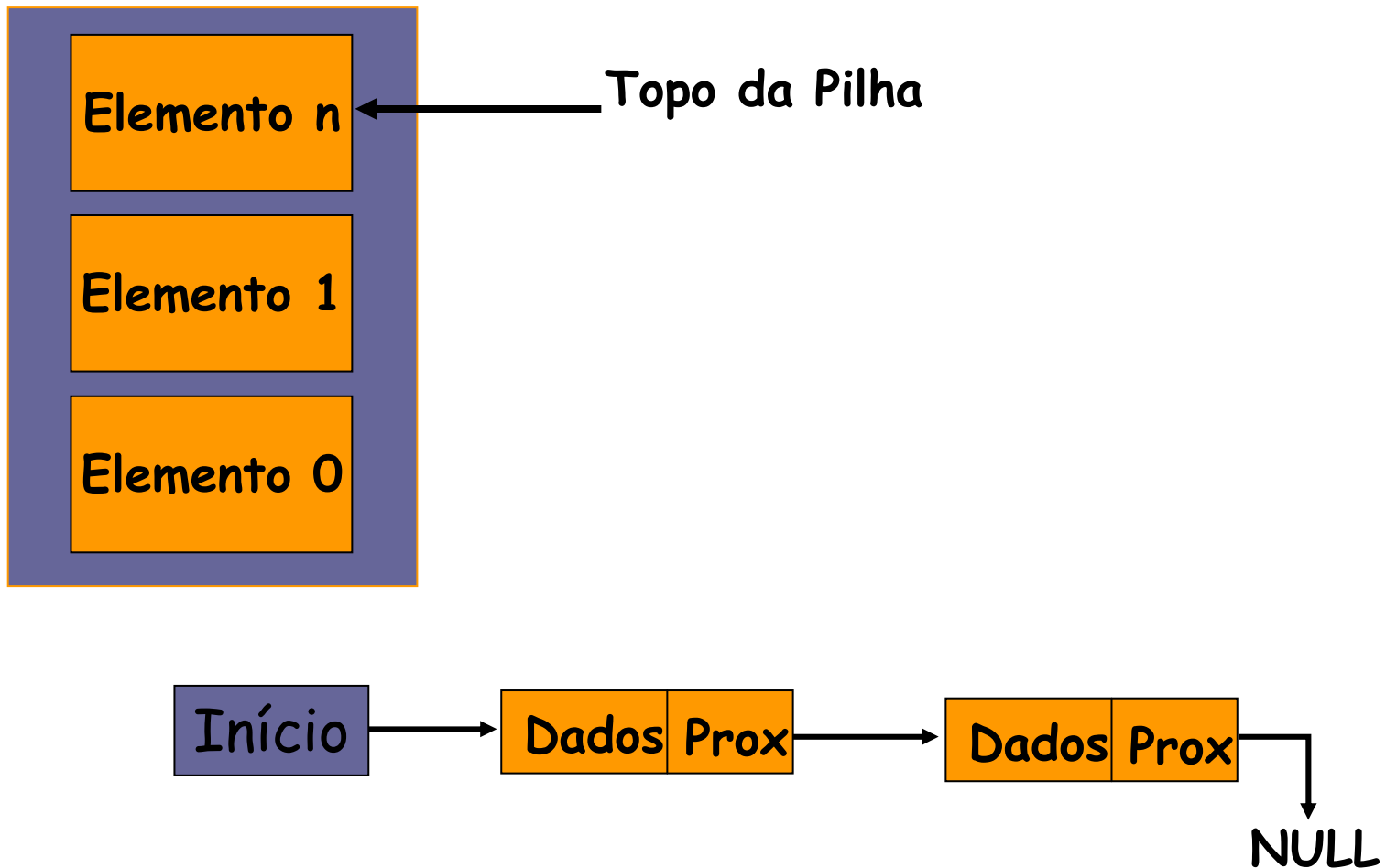
- ❑ Estrutura de dados comumente chamada de LIFO = *Last In. First Out*.
- ❑ O último elemento a entrar na pilha será o primeiro elemento a sair da pilha



Pilhas - Aplicações

- ❑ Verificar se um código fonte está bem estruturado
- ❑ *Parser* de expressões aritméticas
- ❑ O controle da seqüência de chamadas de funções
- ❑ Recursividade
- ❑ Processamento de quaisquer estruturas aninhadas de profundidade imprevisível

Pilhas – Representação Gráfica

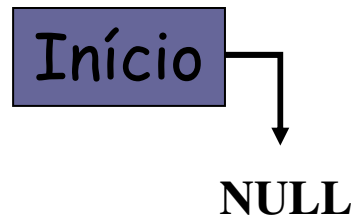


Pilhas – Operações Básicas

- ❑ Criar a Pilha
- ❑ Empilhar elemento (Push)
- ❑ Desempilhar elemento (Pop)
- ❑ Consultar elemento no topo da pilha (Peek)
- ❑ Listar os elementos

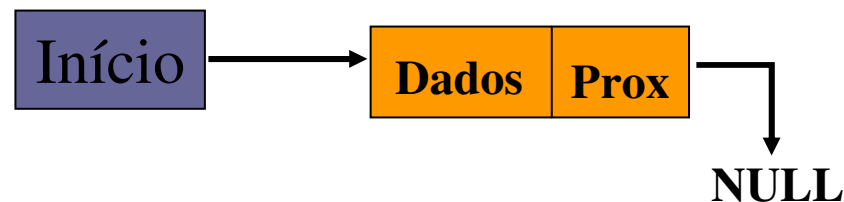
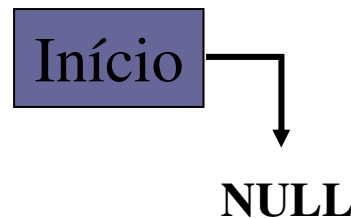
Criando uma Pilha

- ❑ Inicialmente, declara-se um ponteiro para o início da pilha.
- ❑ Só é preciso de um ponteiro, pois tem-se um ponto único de inserção e remoção de elementos da pilha
- ❑ Como a pilha está vazia inicialmente, o ponteiro inicio aponta para NULL ou None.



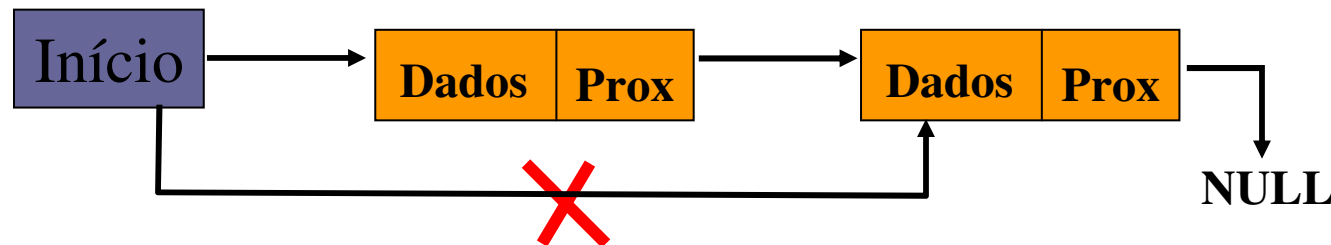
Empilhando um Elemento - PUSH

- Com a pilha vazia: Cria-se o novo elemento e o ponteiro de início aponta para o novo elemento inserido na pilha, que por sua vez aponta para NULL ou None.



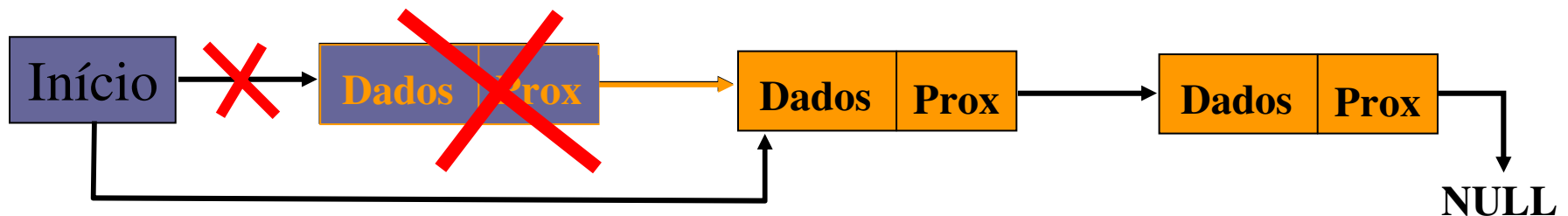
Empilhando um Elemento - PUSH

- Com elementos na pilha: Cria-se o novo elemento que aponta para o elemento do topo da pilha; o ponteiro de início da pilha aponta para o novo elemento.



Desempilhando um Elemento - POP

- ❑ O elemento a ser removido é marcado.
- ❑ O ponteiro do início aponta para o próximo elemento.
- ❑ A memória é liberada.



Exercícios Práticos

- **Exercício 1:** Implementar o exemplo da pilha de números.
- **Exercício 2:** Implemente um programa que utiliza a estrutura de dados pilha para ler uma string do teclado e imprimir a string reversa. OBS: Utilize as funções *push* e *pop*