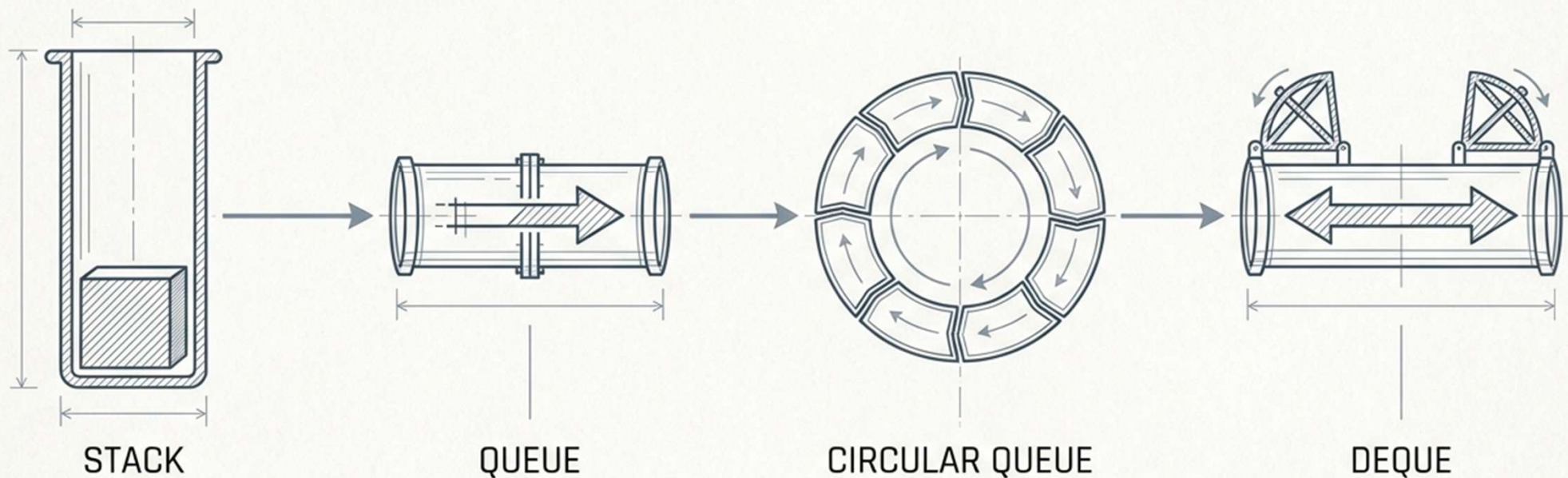


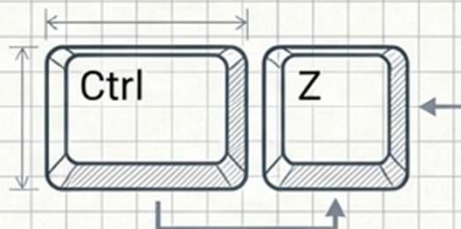
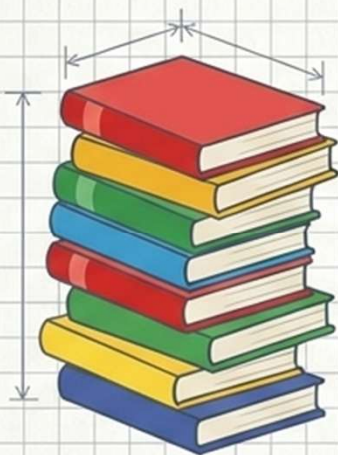
A ARQUITETURA DA INFORMAÇÃO

As estruturas de dados lineares não são tópicos isolados. Elas representam uma evolução contínua nas regras de movimentação de informações — de ambientes altamente restritivos para sistemas totalmente flexíveis.

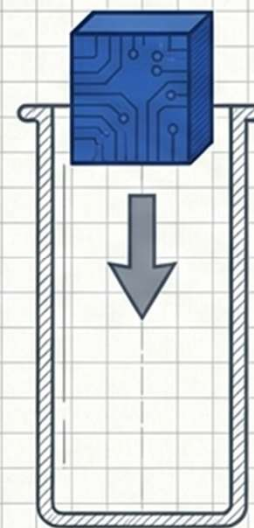


O Ponto de Partida: A Pilha (Stack)

Metáforas do Mundo Real



Regra Digital



- L.I.F.O. (Last In, First Out): O último elemento a entrar é obrigatoriamente o primeiro a sair.
- Metáforas: Pilha de livros, desfazer ações [Ctrl+Z].
- Regra de Fluxo: Inserção e remoção ocorrem exclusivamente em um único ponto (o Topo).

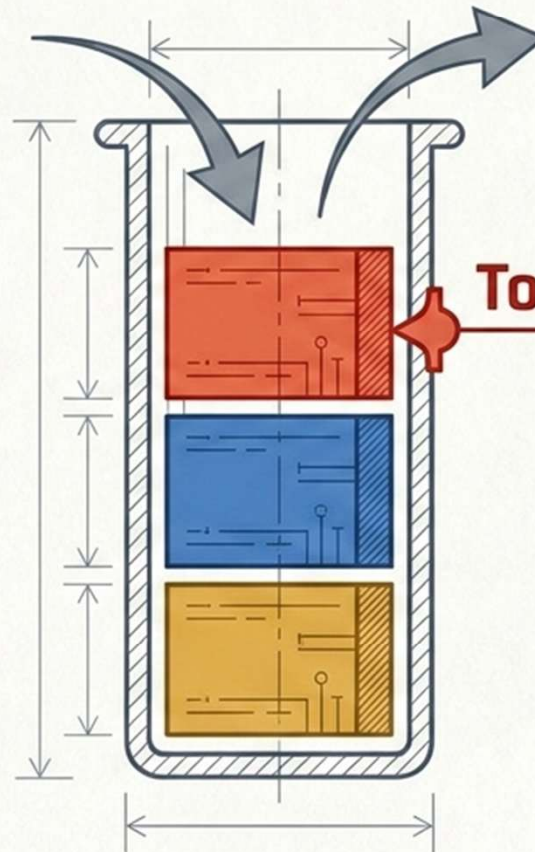
Anatomia e Controle da Pilha

empilha()

- Insere um novo bloco no topo.

desempilha()

- Remove o bloco do topo.



Topo

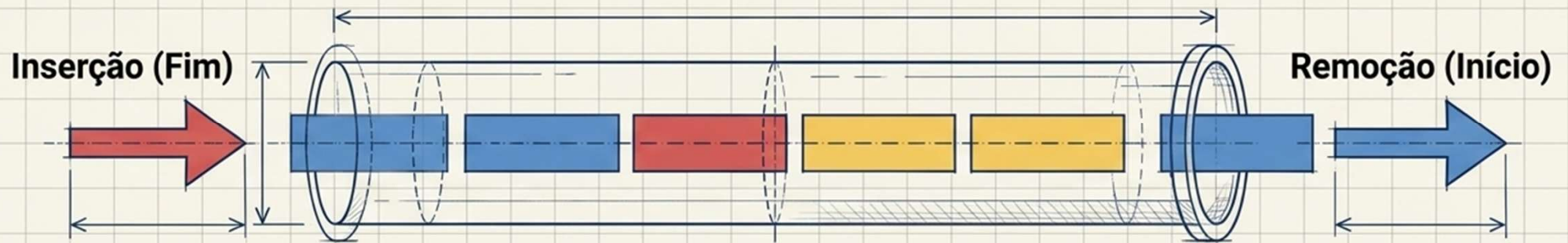
retornaElementoTopoPilha()

- Lê quem está no topo sem remover.

Controles de Estado:

- **verificarPilhavazia()**
- **verificarPilhaCheia()**
- **imprimirPilha()**

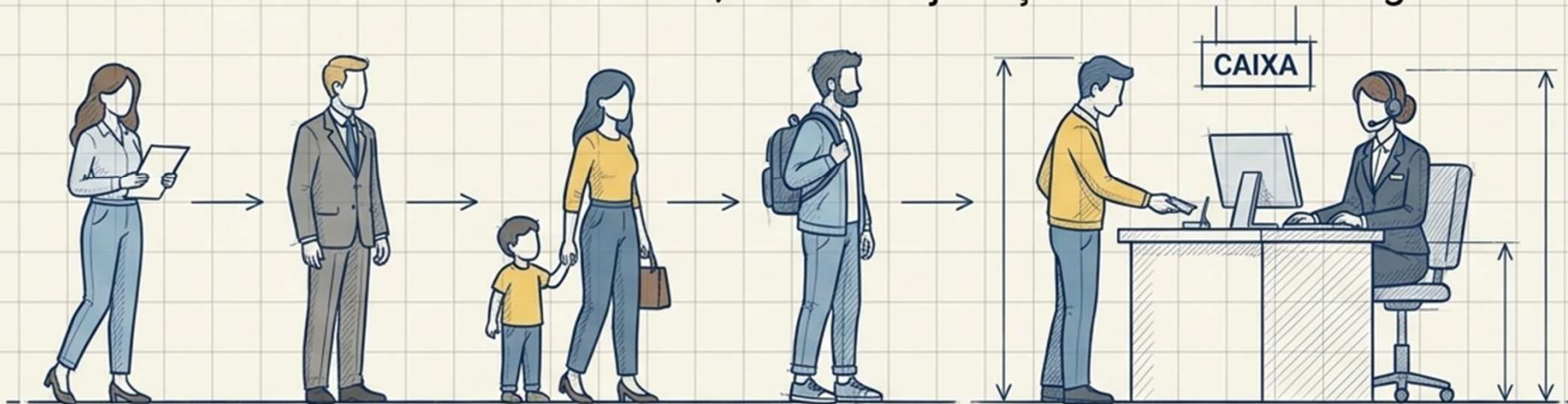
A Mudança de Paradigma: A Fila (Queue)



F.I.F.O. (First In, First Out): O primeiro a entrar é o primeiro a sair.

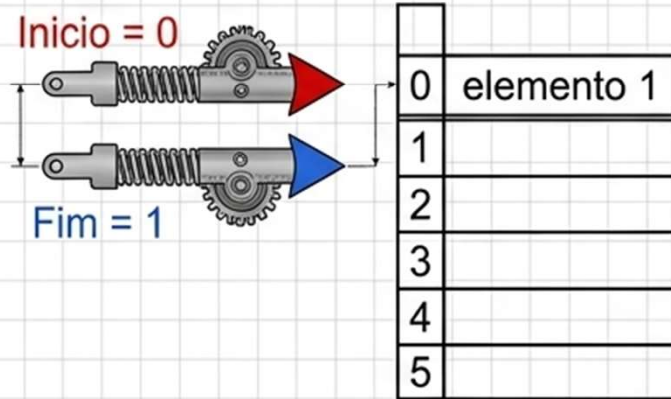
A Regra: A inserção é realizada no final da fila. A remoção é realizada no início.

Contraste: Ao contrário da Pilha, a Fila cria justiça e ordem cronológica.

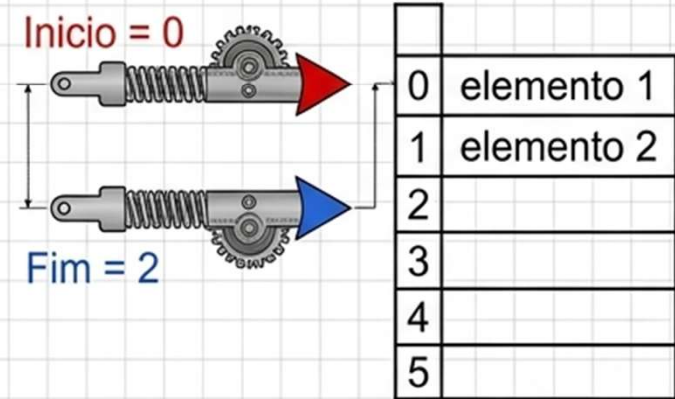


A Mecânica dos Ponteiros (Estado da Fila)

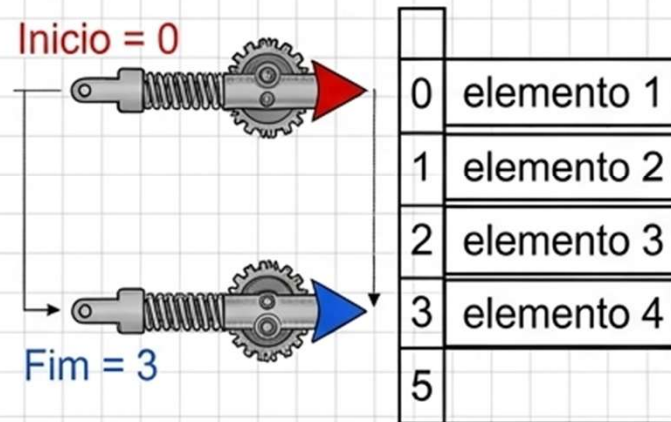
Passo 1: enfileira('elemento 1', FILA)



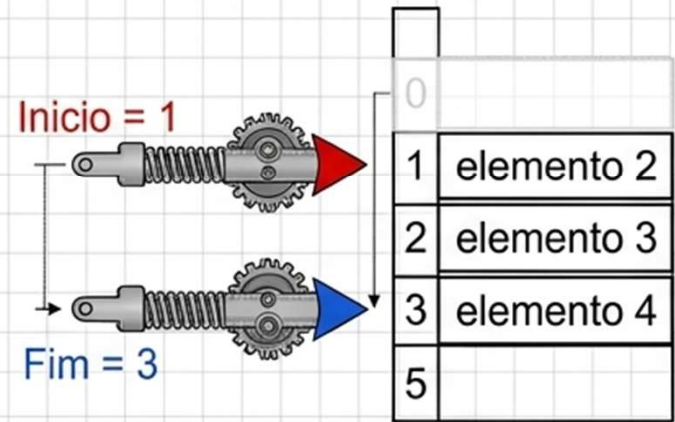
Passo 2: enfileira('elemento 2', FILA)



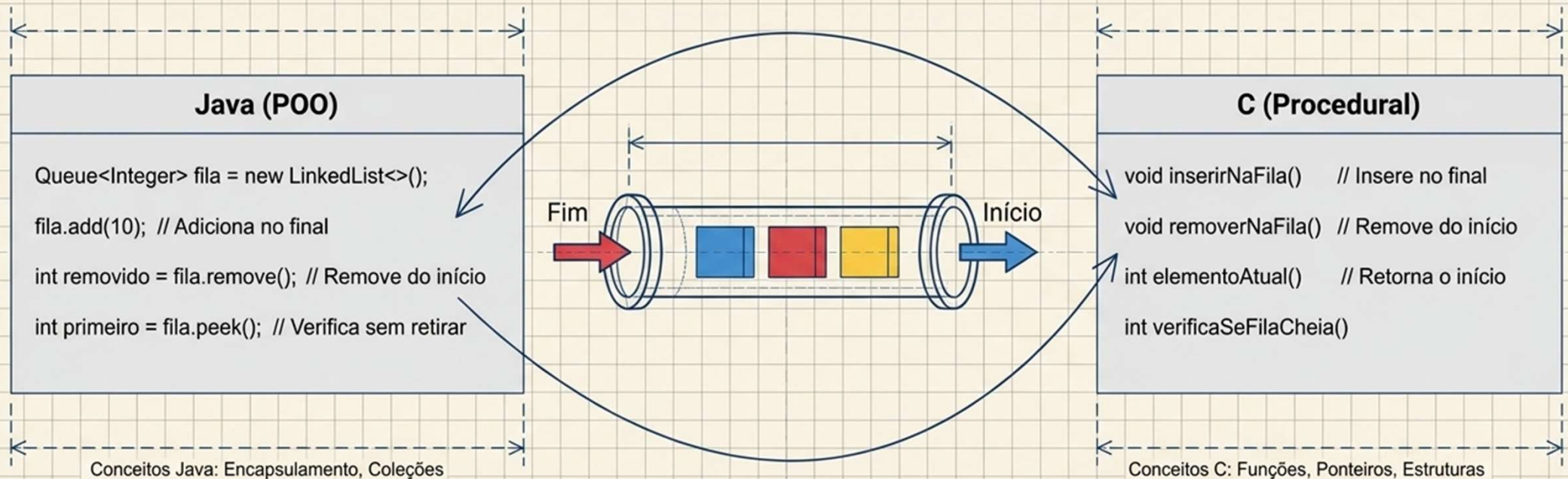
Passo 3: enfileira('elemento 3') e enfileira('elemento 4')



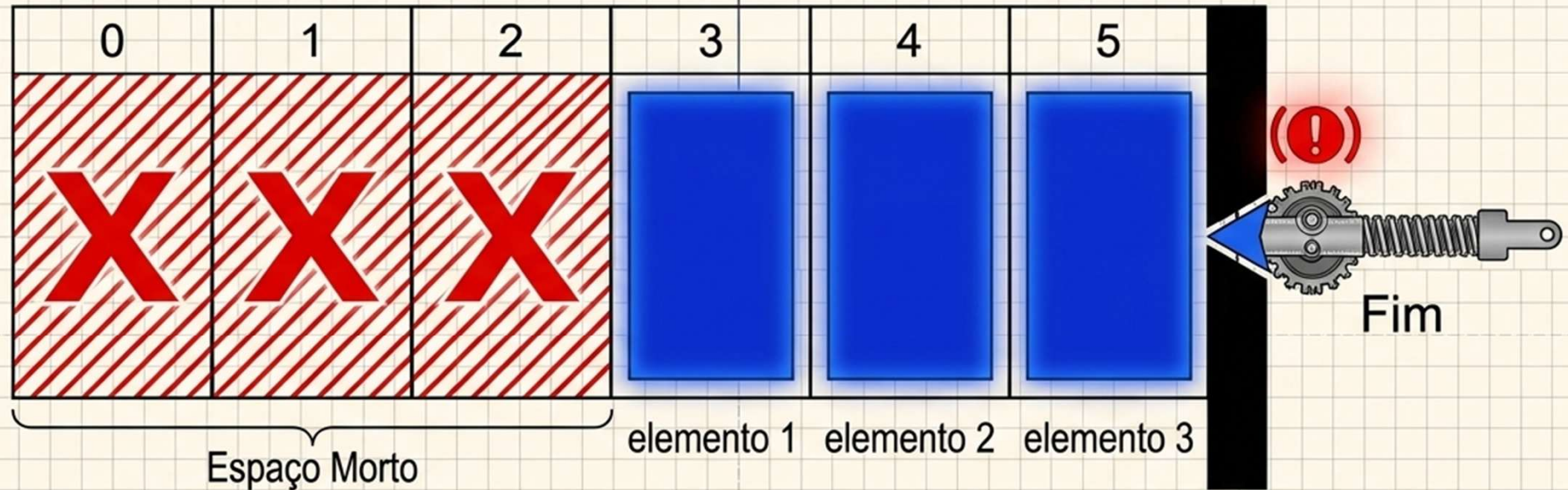
Passo 4: desenfileira(FILA)



Implementação na Prática: Java & C



O Gargalo Logístico: Vetores Estáticos

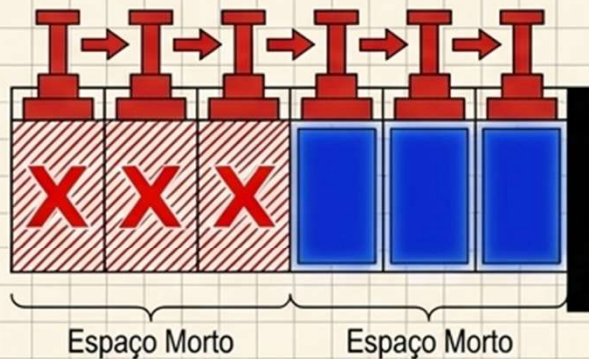


O Problema: Ao desenfilear itens, ocorrem **espaços vazios irreversíveis** o início do vetor.

O Falso 'Cheio': O ponteiro de **Fim** atinge o limite do vetor estático, impedindo novas inserções, mesmo que haja espaço livre no início.

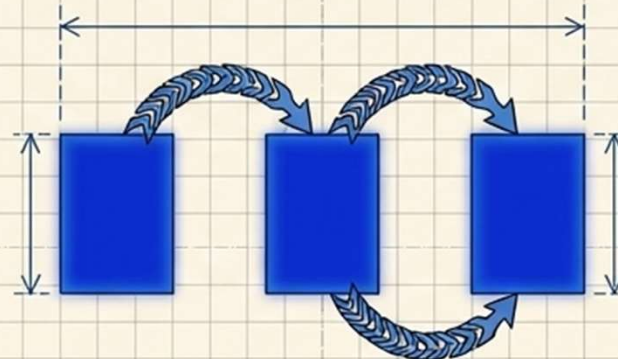
Engenharia de Soluções

Solução 1: O Shift (Ineficiente)



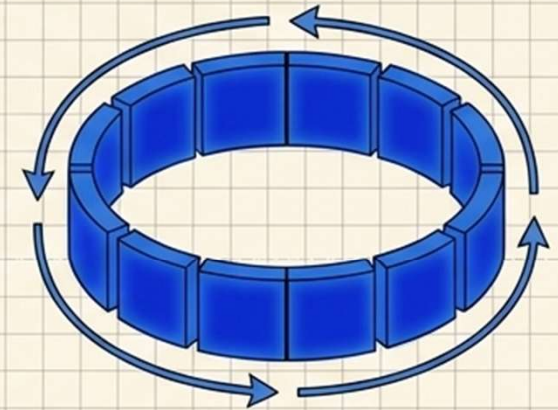
Alterar a posição de todos os itens na fila para a esquerda a cada remoção. Custo computacional extremamente alto.

Solução 2: Listas Encadeadas (A Dinâmica)



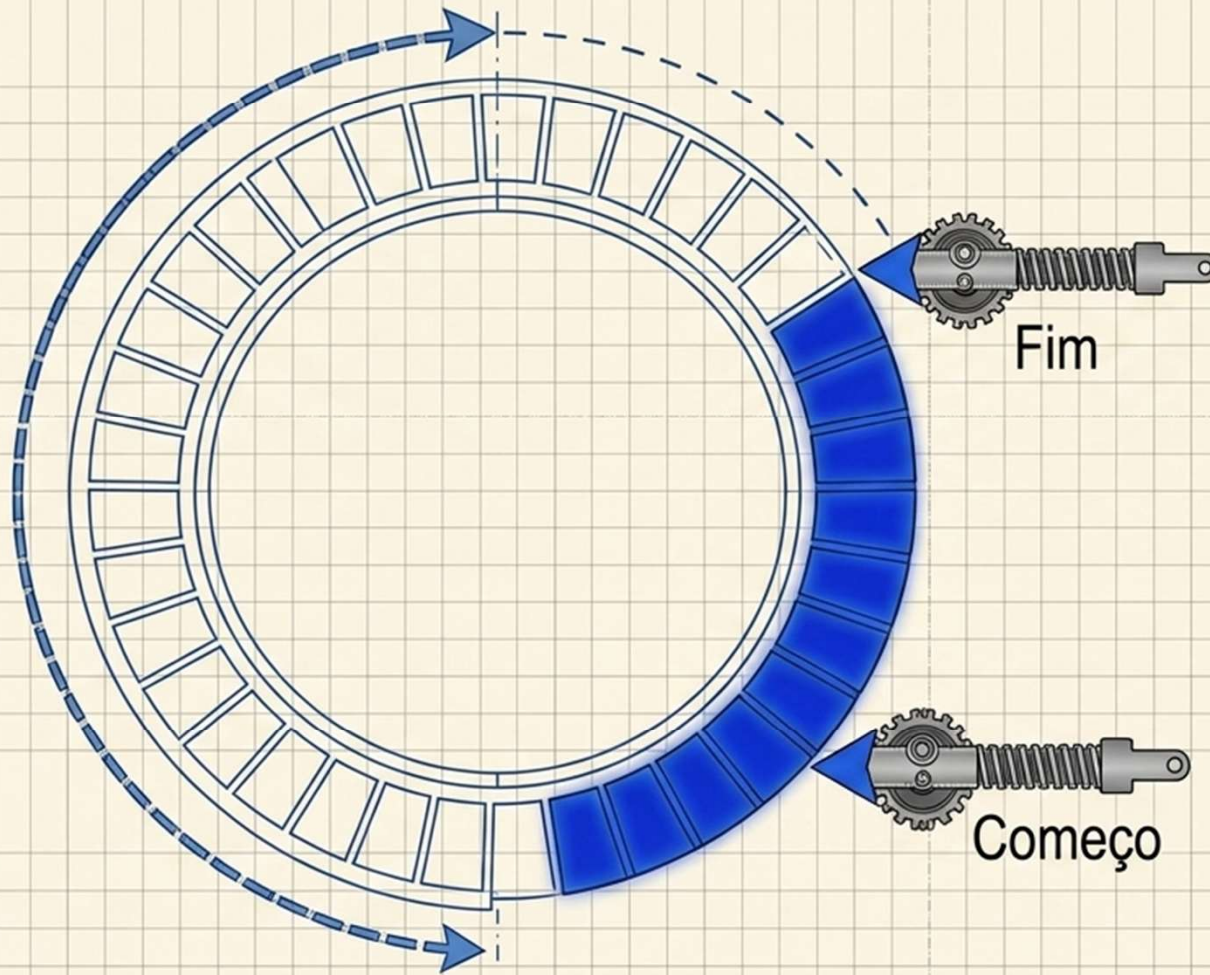
Criar a fila utilizando nós dinâmicos na memória, alocando espaço sob demanda. Melhor solução estrutural, se a linguagem permitir.

Solução 3: Fila Circular (A Abstração Matemática)



Reaproveitar os espaços iniciais transformando o vetor linear estático em um anel lógico contínuo.

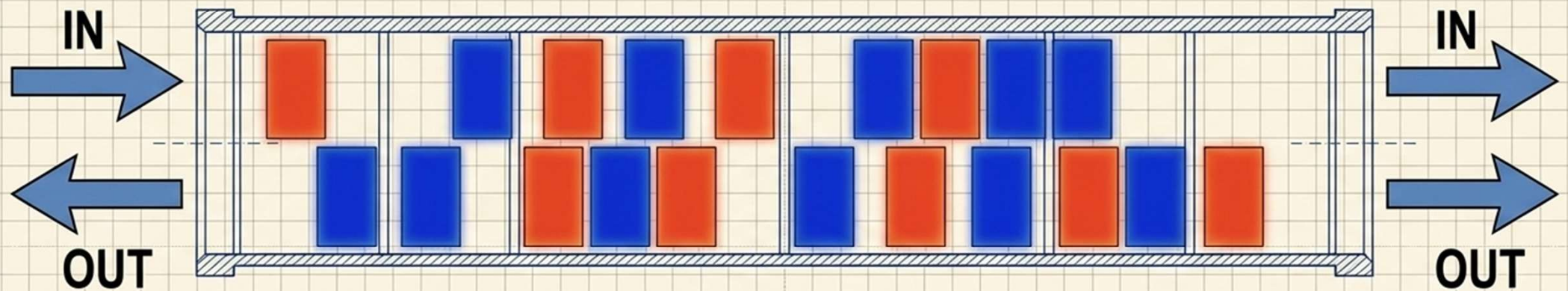
O Loop Infinito: A Fila Circular



A lógica circular **reaproveita o espaço morto**.

O ponteiro de **Fim**, ao atingir o limite físico do vetor, retorna matematicamente ao **índice 0** (se estiver **vazio**), criando um **loop perpétuo**.

A Evolução Final: O Deque



Deque (Double-Ended Queue): Diferente da fila tradicional onde a inserção e a remoção só ocorrem em lados opostos, o Deque permite incluir ou remover em ambos os lados.

A Estrutura Híbrida: É a evolução definitiva que combina as capacidades matemáticas da Pilha (LIFO) e da Fila (FIFO) em um único modelo de dados avançado.

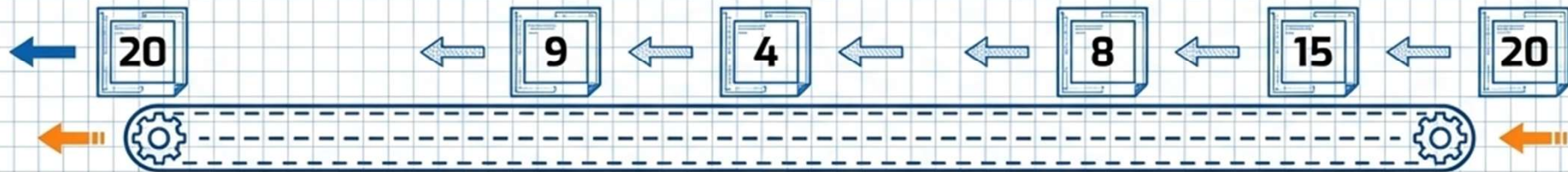
Simulação de Fluxo (Teste de Mesa)

Entrada

Sequência: **E20, E15, E9, D, E4, E8, D, D, E6, E1, D, E7**

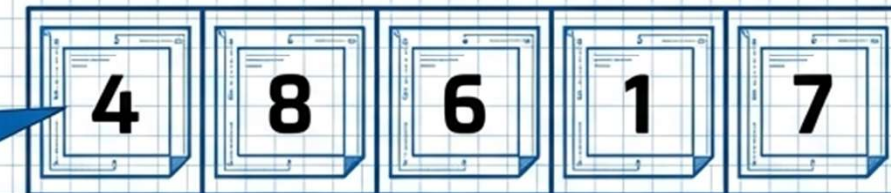
(**E** = Enfileirar, **D** = Desenfileirar)

Rastreamento Visual (Linha do Tempo)



Resultados da Saída

Próximo elemento da fila a sair: **4**



Início

Fila Final: [4, 8, 6, 1, 7]

Fim

Matriz de Decisão: Estruturas Lineares

Estrutura	Regra de Fluxo	Ponto de Inserção	Ponto de Remoção
Pilha (Stack)	L.I.F.O. (Last-In, First-Out)	Topo	Topo
Fila (Queue)	F.I.F.O. (First-In, First-Out)	Final	Início
Fila Circular	F.I.F.O. (Otimizado)	Final (Loop para Início)	Início
Deque	Híbrido (Bidirecional)	Início ou Final	Início ou Final

A **arquitetura de software** não é apenas sobre **armazenar dados** — é sobre projetar regras logísticas estritas de como a informação flui.