ALGORITMOS E ESTRUTURA DE DADOS

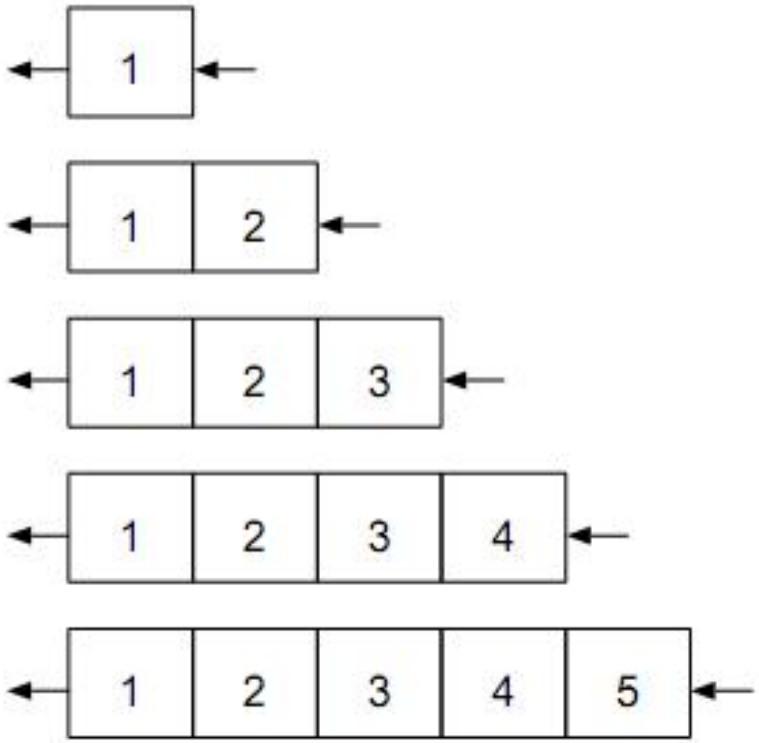
Curso de Engenharia de Software
Lucas Sampaio Leite

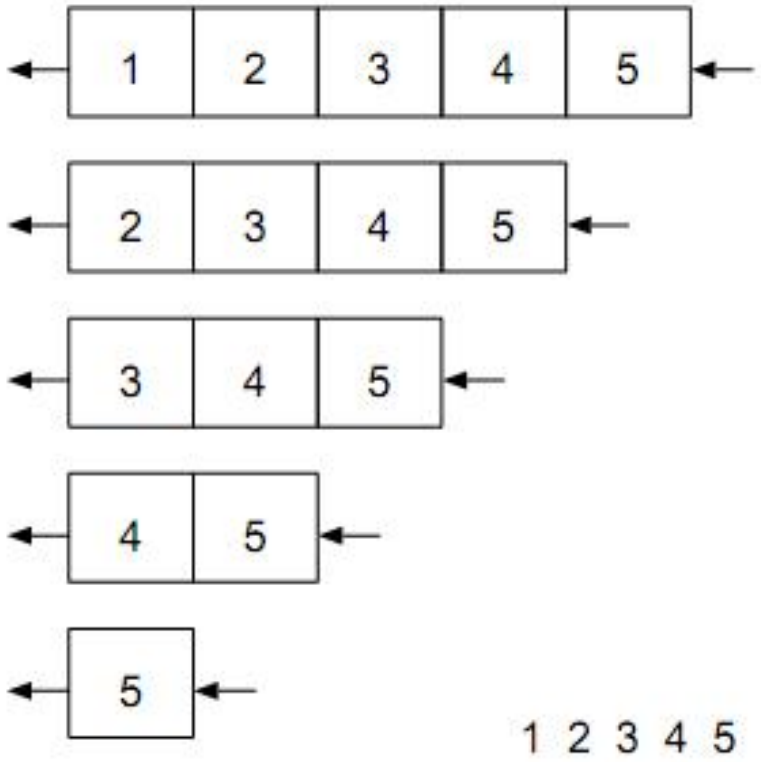
Filas

- A estrutura de dados fila é uma coleção linear de elementos que segue a política FIFO (First In, First Out), isto é, o primeiro elemento inserido na fila será também o primeiro a ser removido.
    - Esse comportamento é semelhante ao de uma fila de pessoas em um atendimento, onde quem chega primeiro é atendido primeiro.

Filas

INSTITUTO FEDERAL
Pernambuco





Filas

INSTITUTO FEDERAL
Pernambuco

- Operações comuns em uma fila:
  - void enqueue(Fila *f, T elemento); → insere um elemento no final da fila;
  - T dequeue(Fila *f); → remove e retorna o elemento do início da fila;
  - T front(Fila *f); → retorna o elemento do início sem removê-lo;
  - int isEmpty(Fila *f); → verifica se a fila está vazia;
  - int size(Fila *f); → retorna a quantidade de elementos armazenados;
  - void clear(Fila *f); → remove todos os elementos da fila.
- Implementações:
  - Baseada em vetores;
  - Baseada em listas encadeadas.

Filas

INSTITUTO FEDERAL
Pernambuco

- Definindo uma fila:

```c
typedef struct No{
    T elemento;
    struct No *prox;
} No;

typedef struct {
    No *inicio;
    No *fim;
    int tamanho;
} Fila;

void inicializar(Fila *f) {
    f->inicio = NULL;
    f->fim = NULL;
    f->tamanho = 0;
}
```

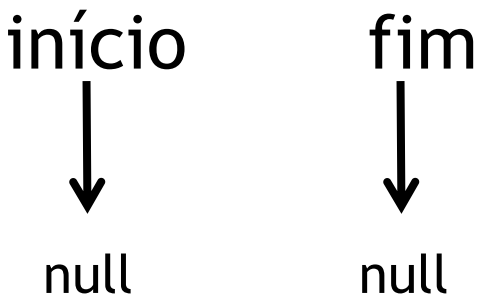
```c
typedef struct {
    T dados[MAX];
    int tamanho;
} Fila;

void inicializar(Fila *f) {
    f->tamanho = 0;
}
```

Qual a diferença entre as definições?

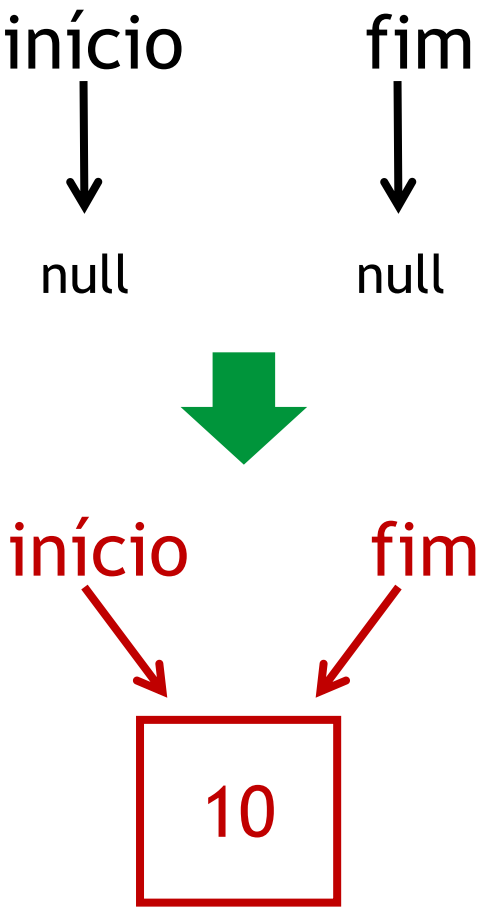
Filas

- Operação enfileirar (enqueue):



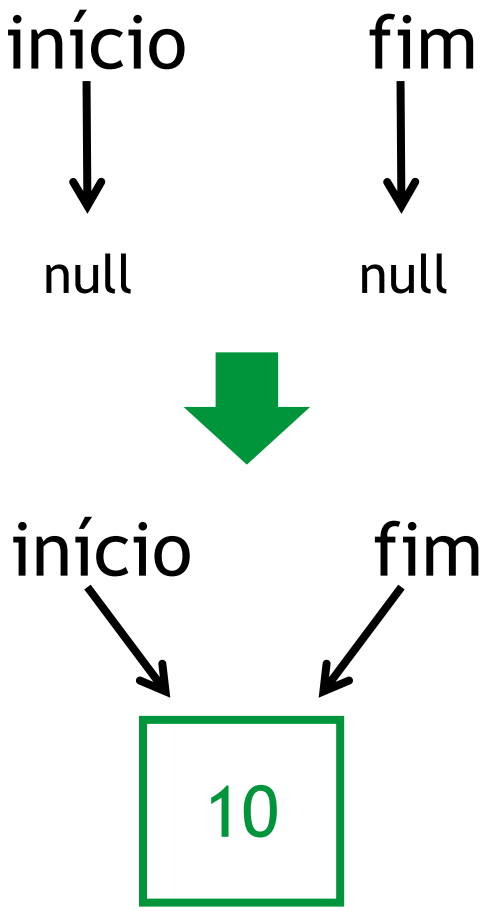
Filas

- Operação enfileirar (enqueue):



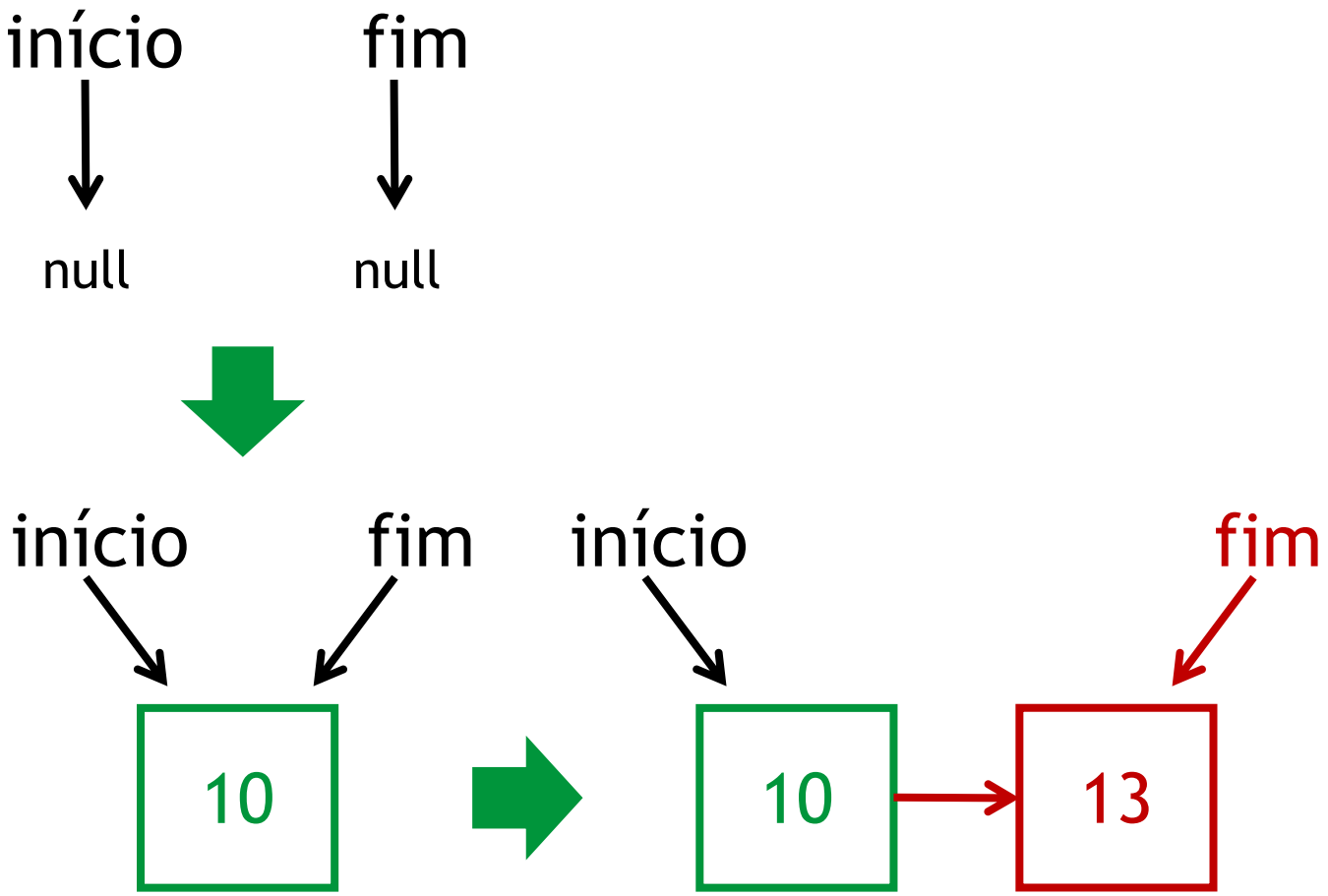
Filas

- Operação enfileirar (enqueue):



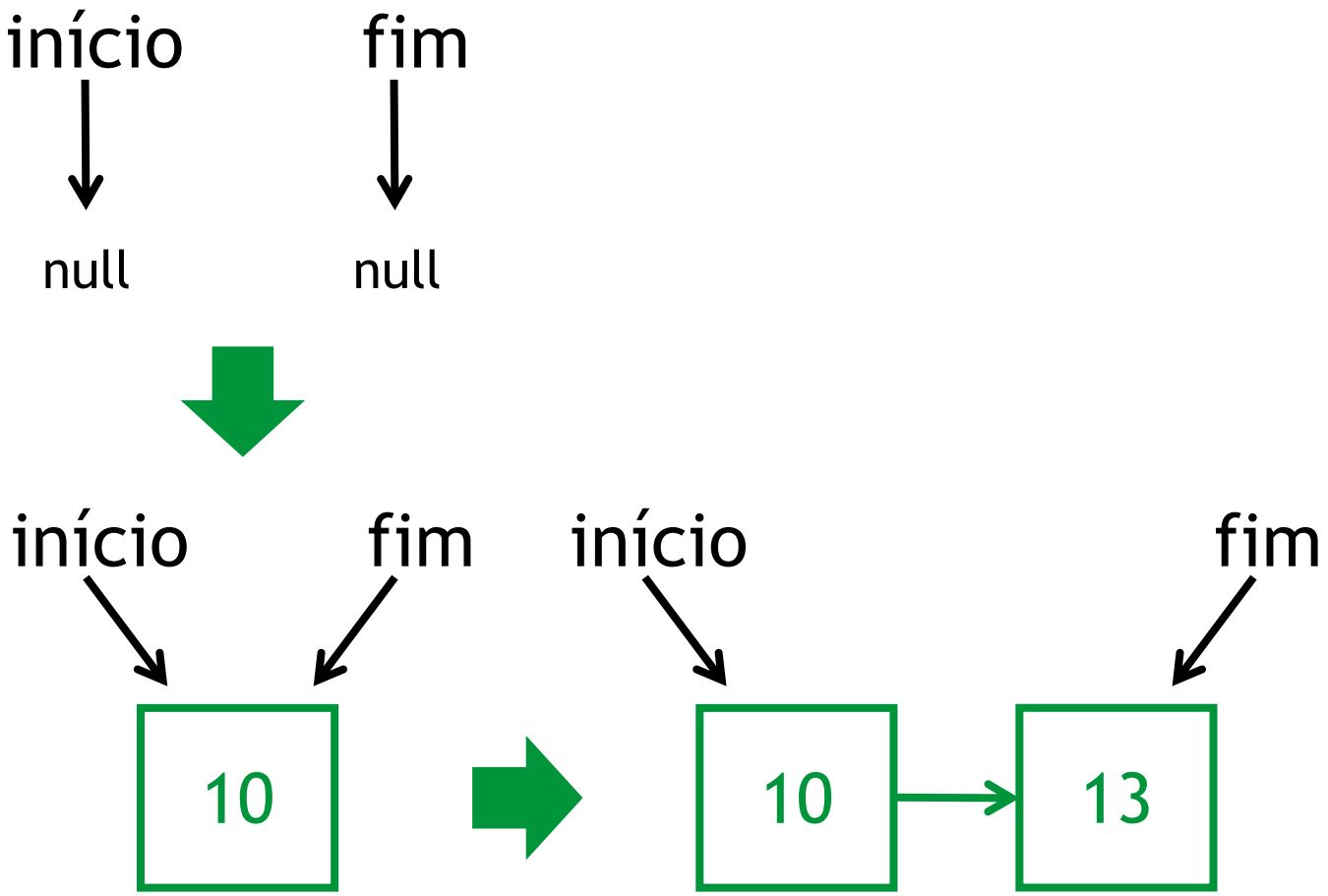
Filas

- Operação enfileirar (enqueue):



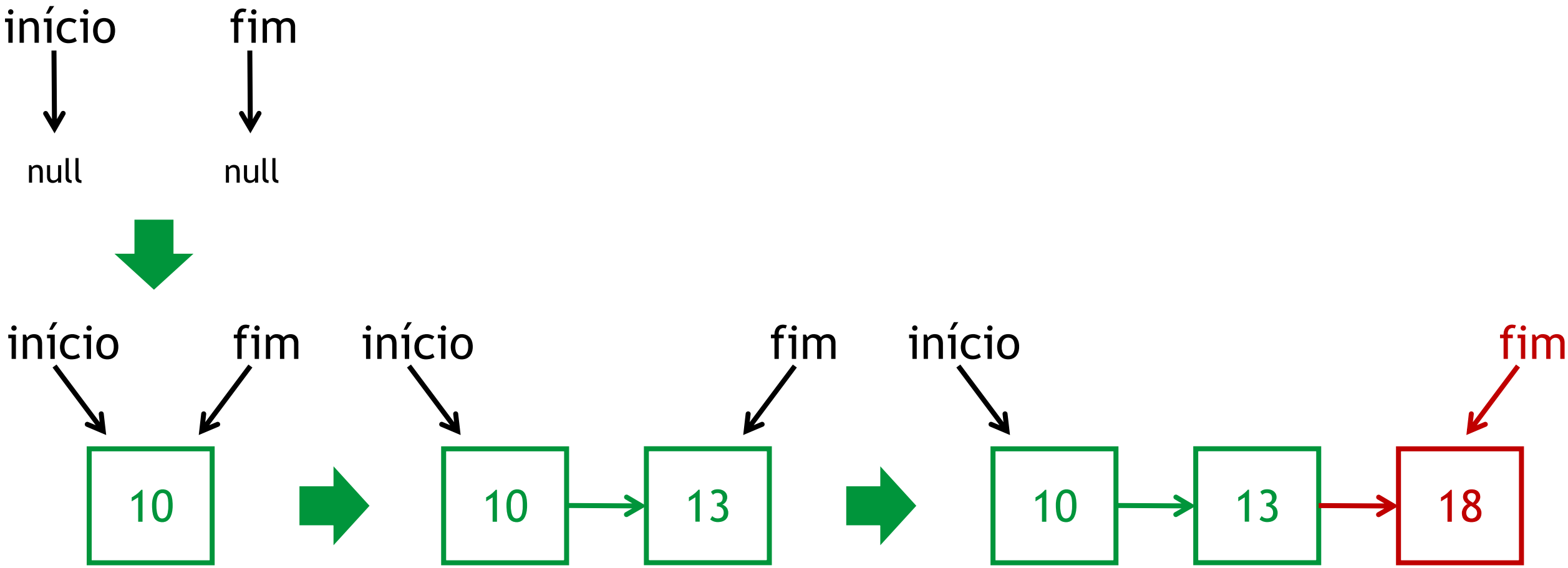
Filas

- Operação enfileirar (enqueue):



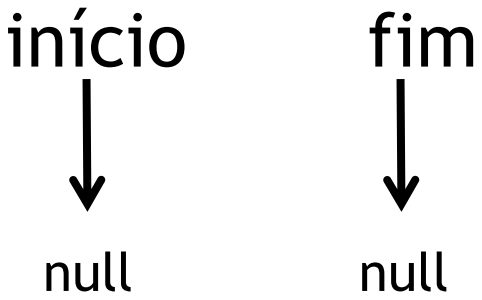
Filas

- Operação enfileirar (enqueue):

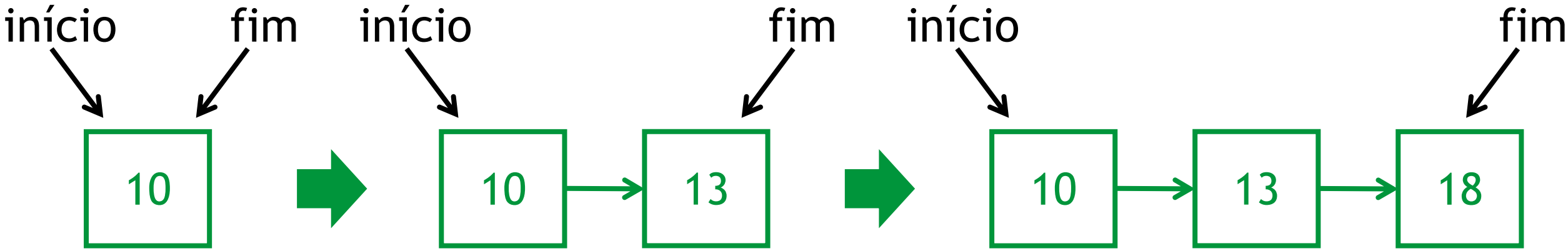


Filas

- Operação enfileirar (enqueue):

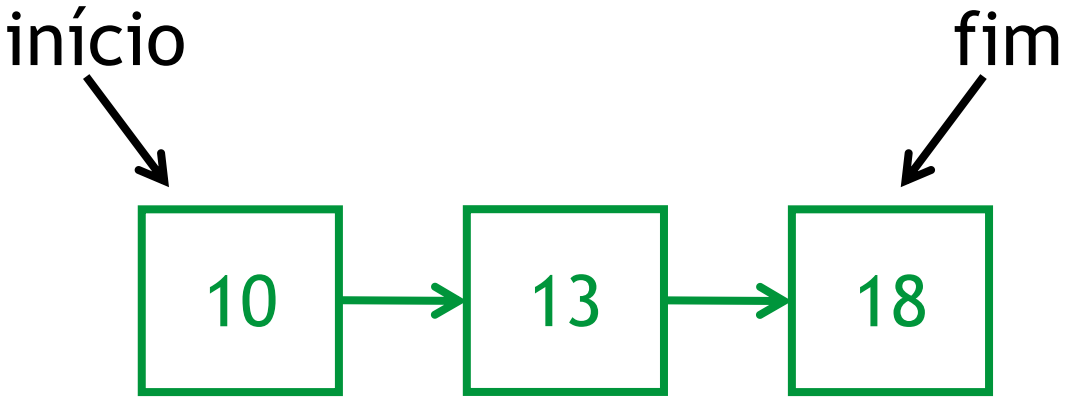



```
void enqueue(Fila *f, T elemento) {
    //insere o elemento no final da fila
}
```



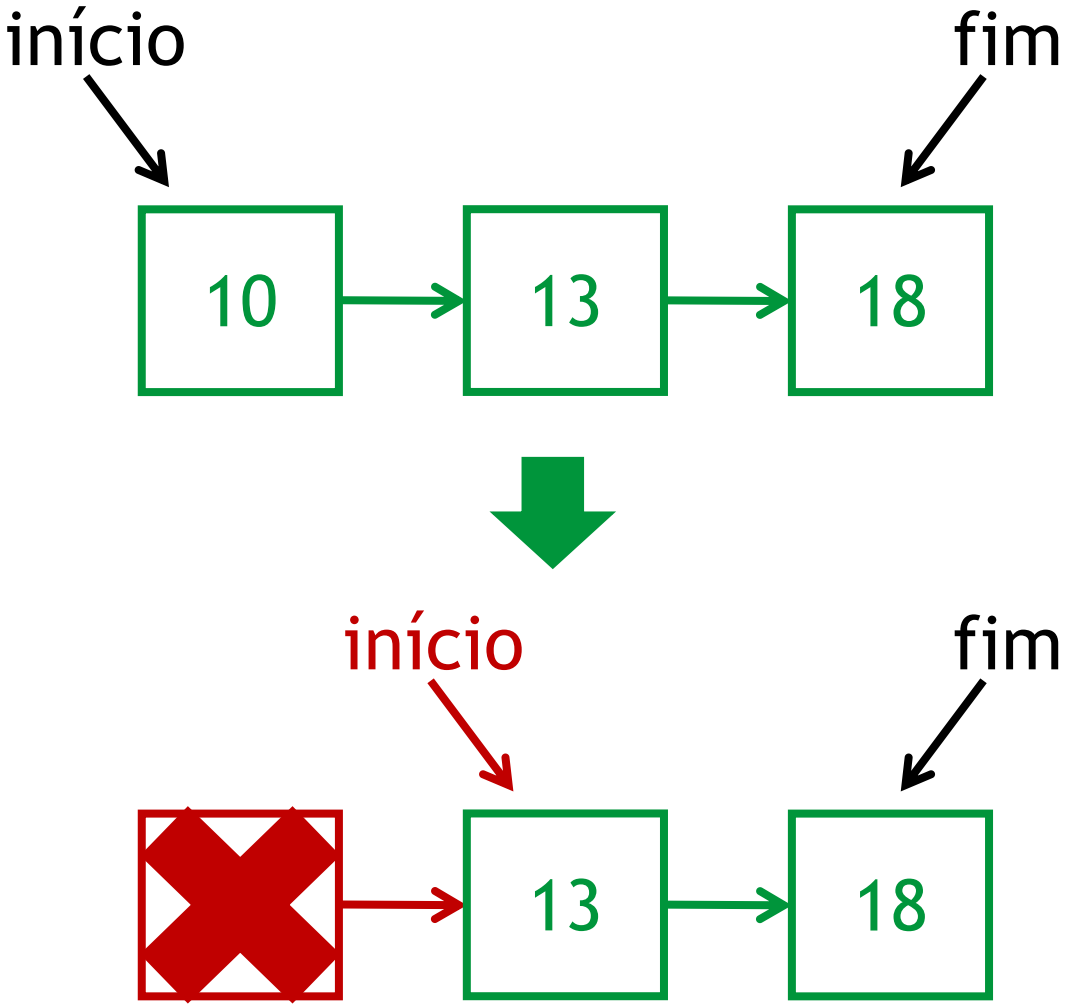
Filas

- Operação desenfileirar (dequeue):



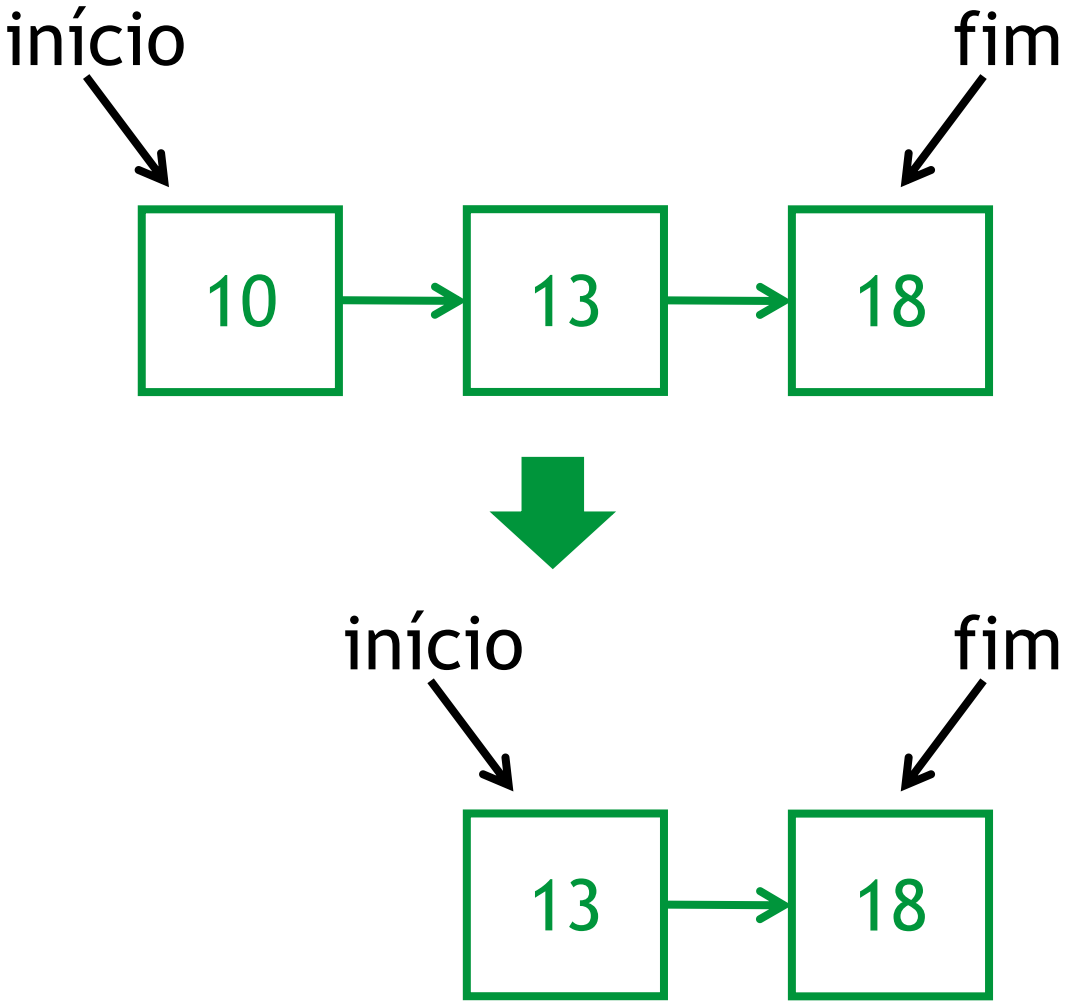
Filas

- Operação desenfileirar (dequeue):



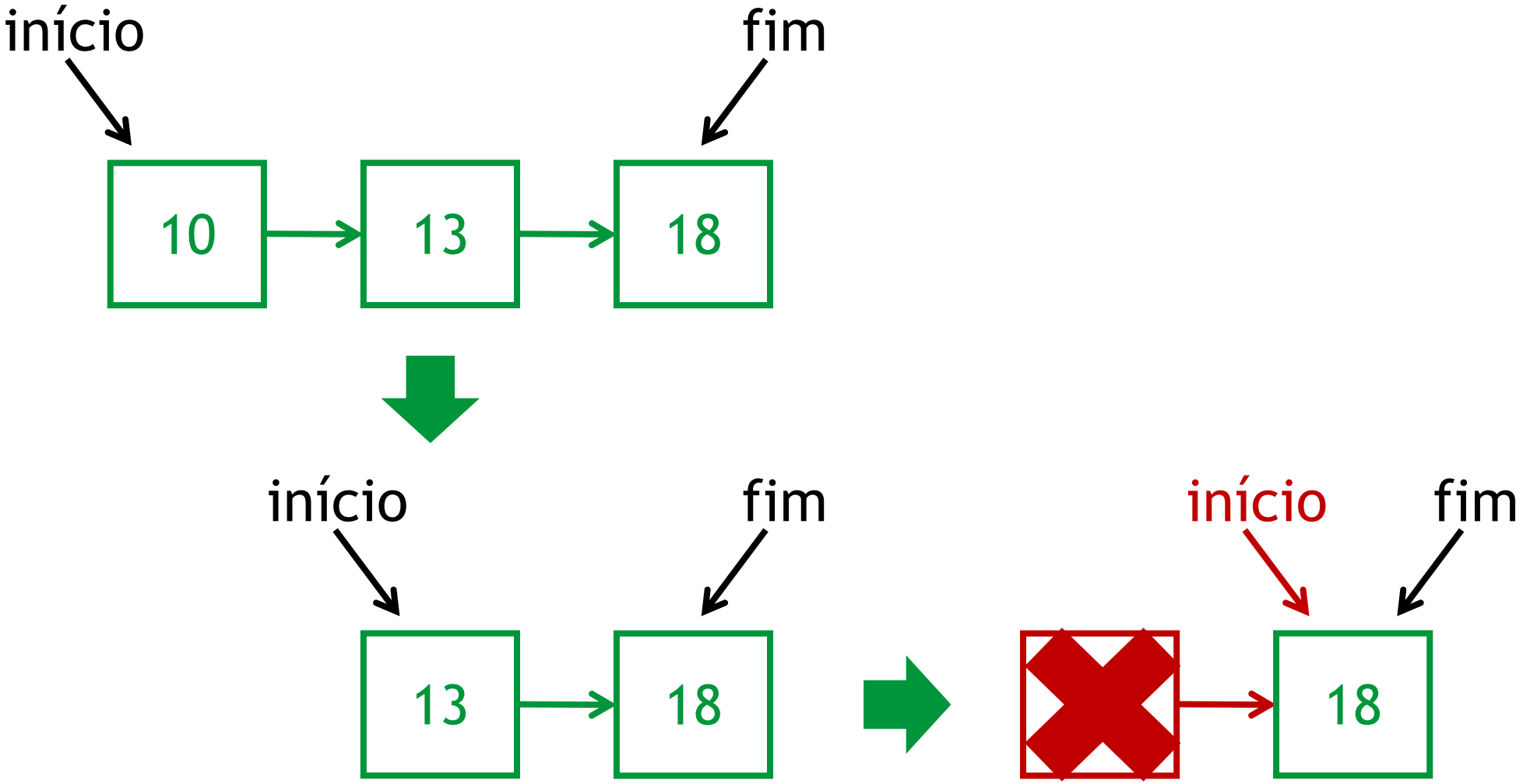
Filas

- Operação desenfileirar (dequeue):



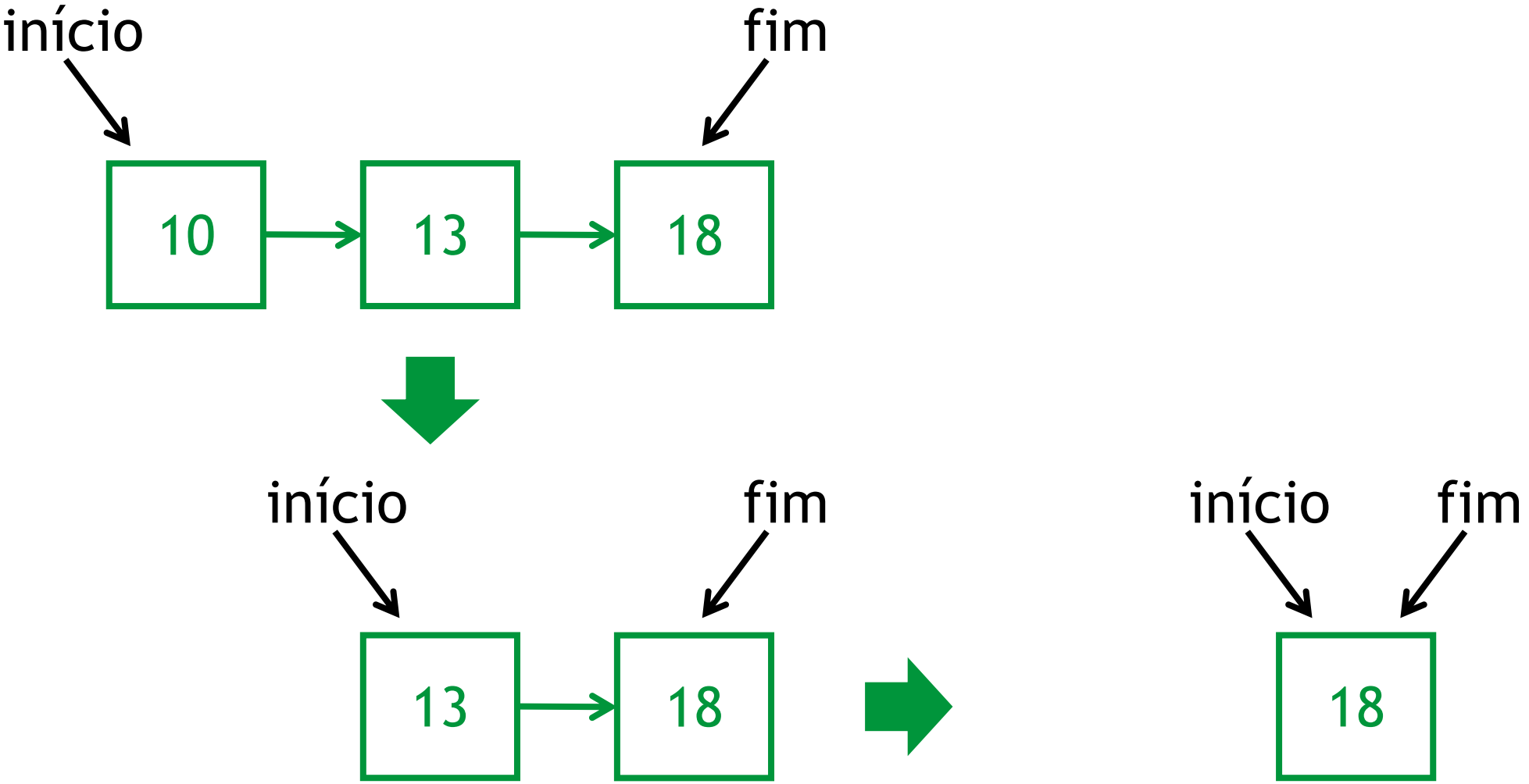
Filas

- Operação desenfileirar (dequeue):



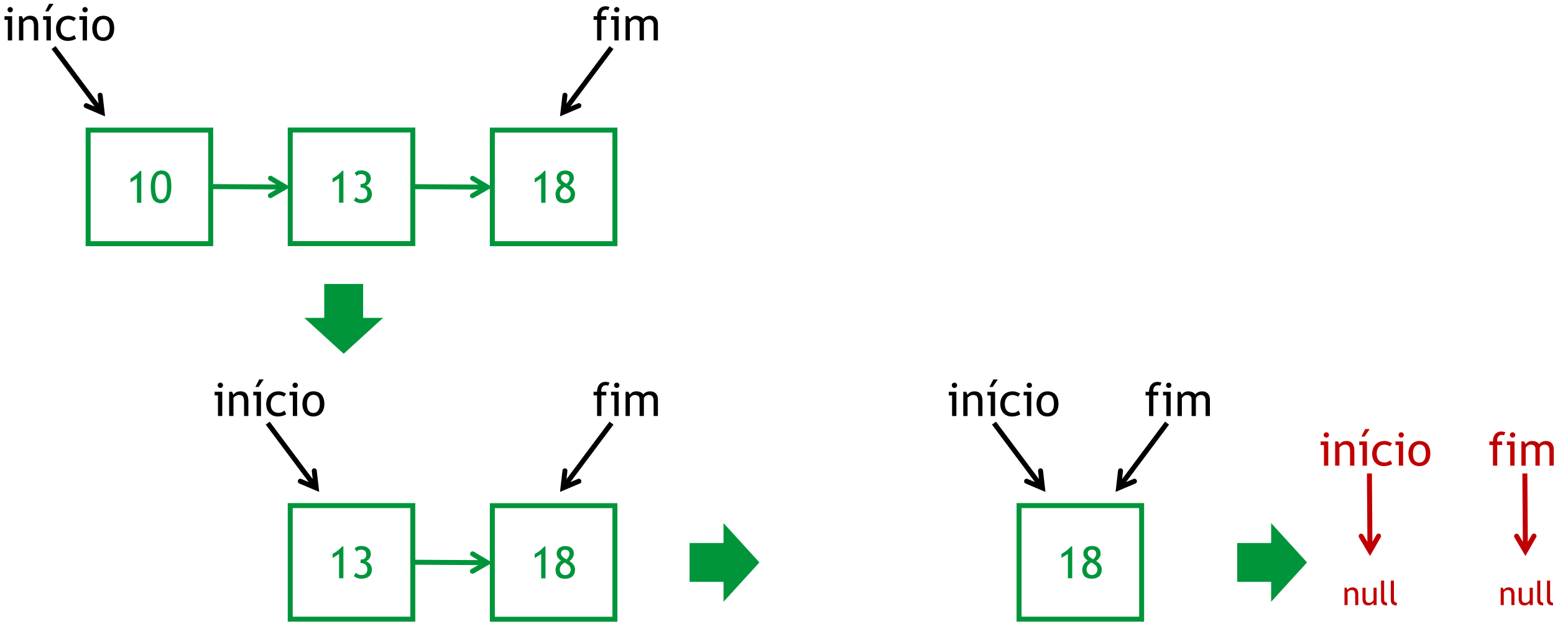
Filas

- Operação desenfileirar (dequeue):



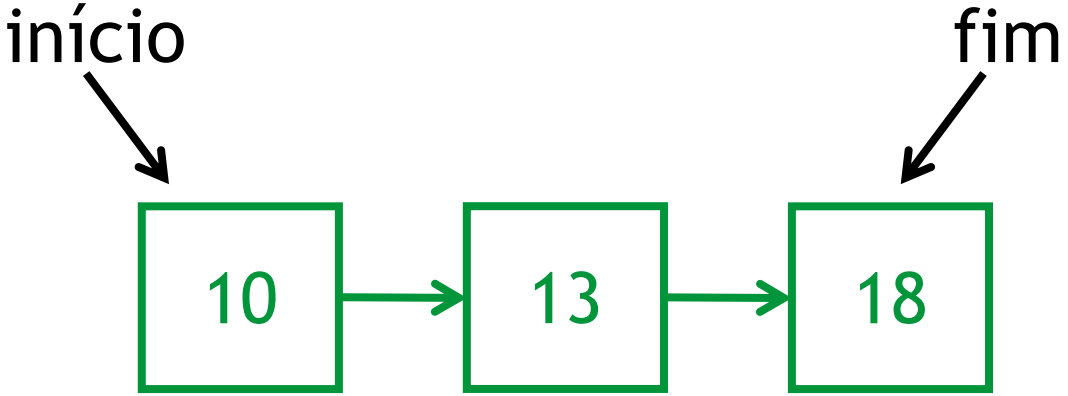
Filas

- Operação desenfileirar (dequeue):

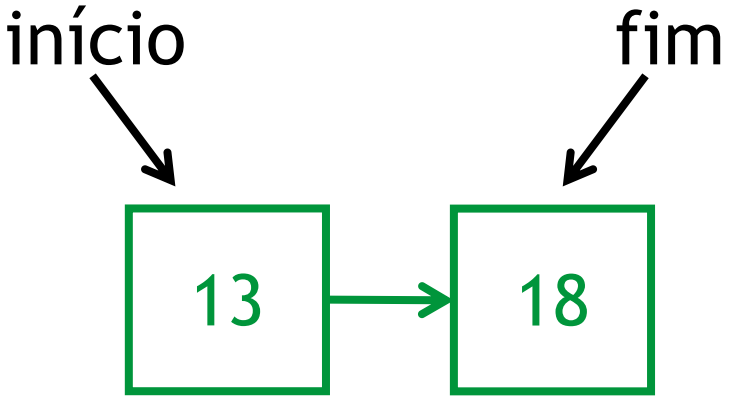


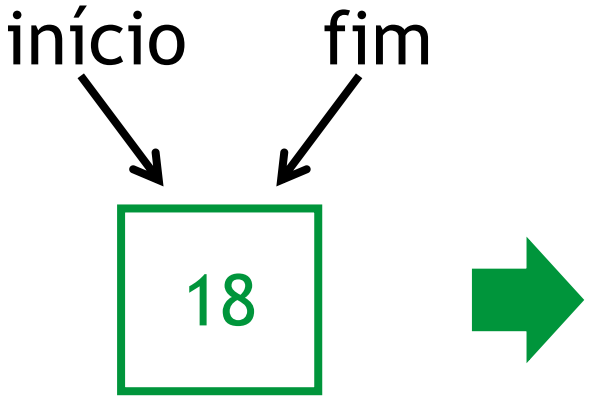
Filas

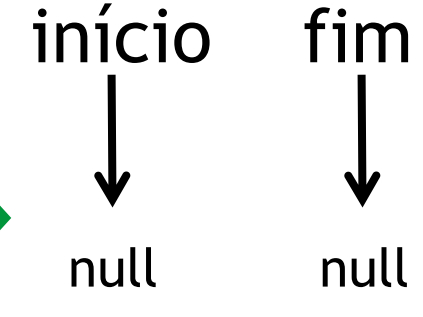
- Operação desenfileirar (dequeue):




```
T dequeue(Fila *f) {
    /*remove e retorna o elemento
      do início da fila*/
}
```

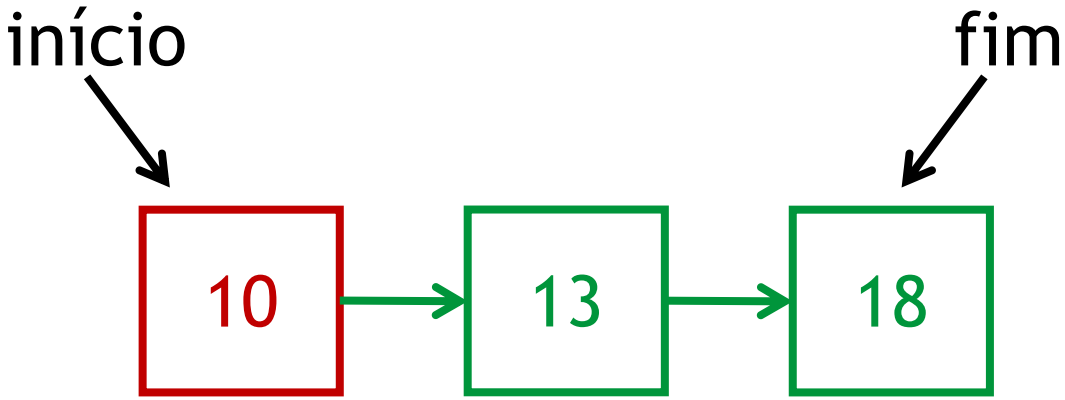






Filas

- Operação de consulta ao primeiro elemento (front):



Filas

- Verificar se a fila está vazia (isEmpty)
- Verificar o tamanho da fila (size)
- Limpar a fila (clear)

Já discutimos a implementação dessas funcionalidades como listas!!!

Filas

INSTITUTO FEDERAL
Pernambuco

- Aplicações de filas:
  - Gerenciamento de processos em sistemas operacionais, onde processos aguardam na fila para execução;
  - Controle de tarefas em filas de impressão;
  - Sistemas de atendimento, como filas de chamadas em call centers;
  - Processamento de requisições em servidores e aplicações web;
  - Gerenciamento de clientes em sistemas bancários e de atendimento presencial.

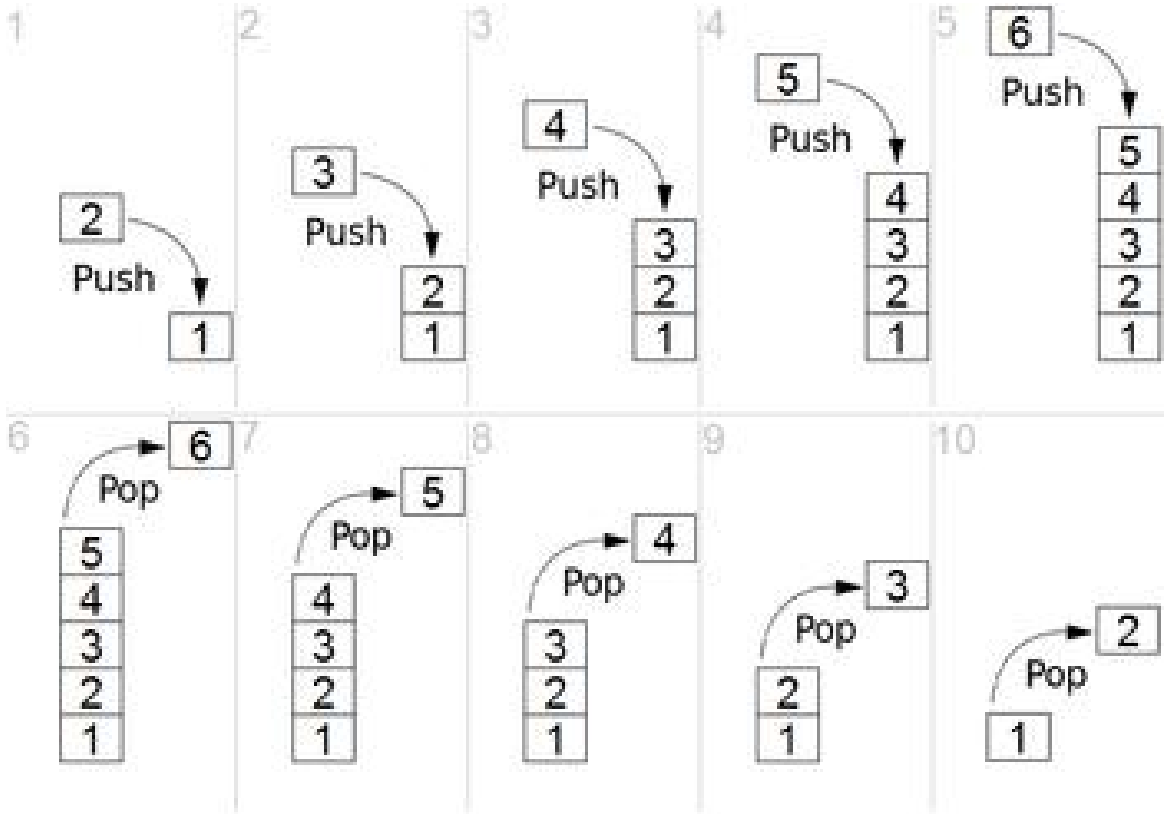
Pilhas

INSTITUTO FEDERAL
Pernambuco

- Pilhas são estruturas de dados que seguem a política LIFO (Last In, First Out), na qual o último elemento inserido é o primeiro a ser removido.
- As operações de inserção e remoção ocorrem sempre no topo da pilha, denominado topo (top).

Pilhas

INSTITUTO FEDERAL
Pernambuco



Pilhas

INSTITUTO FEDERAL
Pernambuco

- Operações comuns em uma pilha:
  - void push(Pilha *p, T elemento); → insere um elemento no topo da pilha;
  - T pop(Pilha *p); → remove e retorna o elemento do topo da pilha;
  - T peak(Pilha *p); → retorna o elemento do topo sem removê-lo;
  - int isEmpty(Pilha *p); → verifica se a pilha está vazia;
  - int size(Pilha *p); → retorna a quantidade de elementos armazenados;
  - void clear(Pilha *p); → remove todos os elementos da pilha.
- Implementações:
  - Baseada em vetores;
  - Baseada em listas encadeadas.

Pilhas

INSTITUTO FEDERAL
Pernambuco

- Definindo uma pilha:

```c
typedef struct No{
    T elemento;
    struct No *prox;
} No;

typedef struct {
    No *topo;
    int tamanho;
} Pilha;

void inicializar(Pilha *p) {
    p->topo = NULL;
    p->tamanho = 0;
}
```

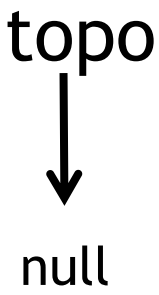
```c
typedef struct {
    T dados[MAX];
    int tamanho;
} Pilha;

void inicializar(Pilha *p) {
    p->tamanho = 0;
}
```

Qual a diferença entre as definições?

Pilhas

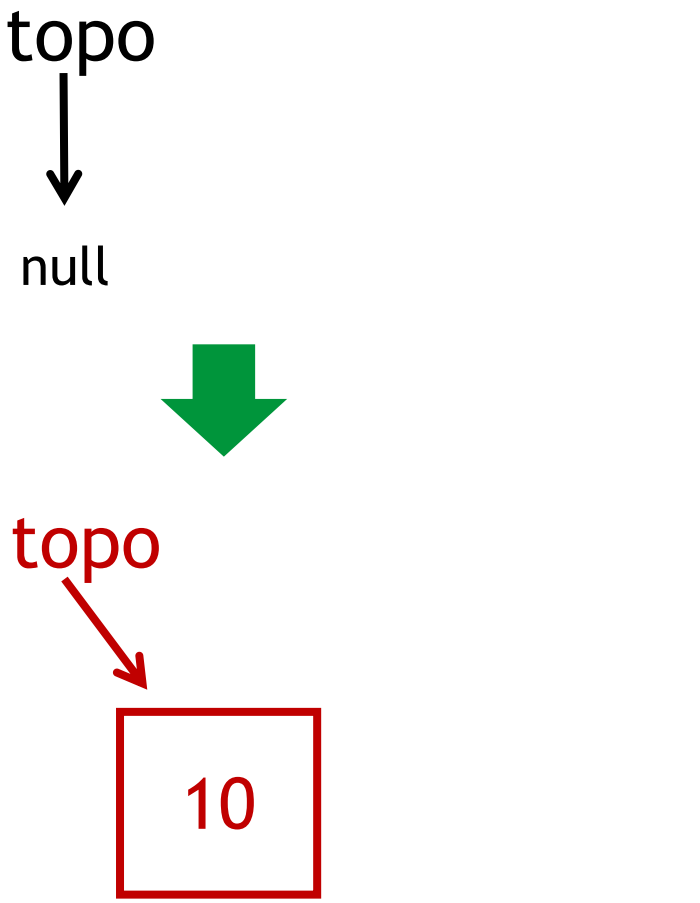
- Operação empilhar (push):



INSTITUTO FEDERAL
Pernambuco

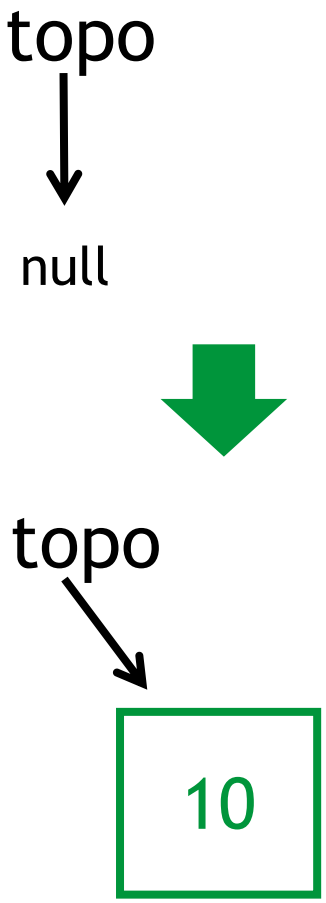
Pilhas

- Operação empilhar (push):



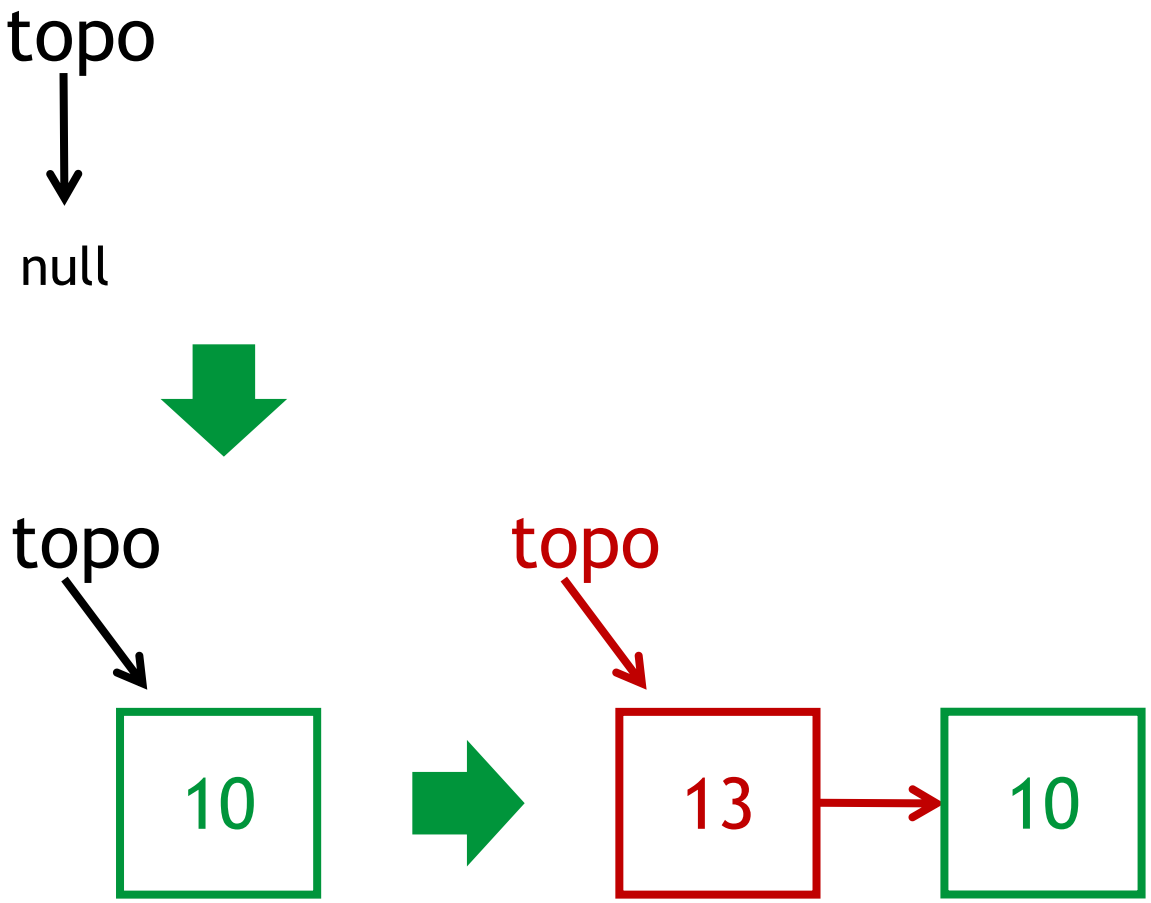
Pilhas

- Operação empilhar (push):



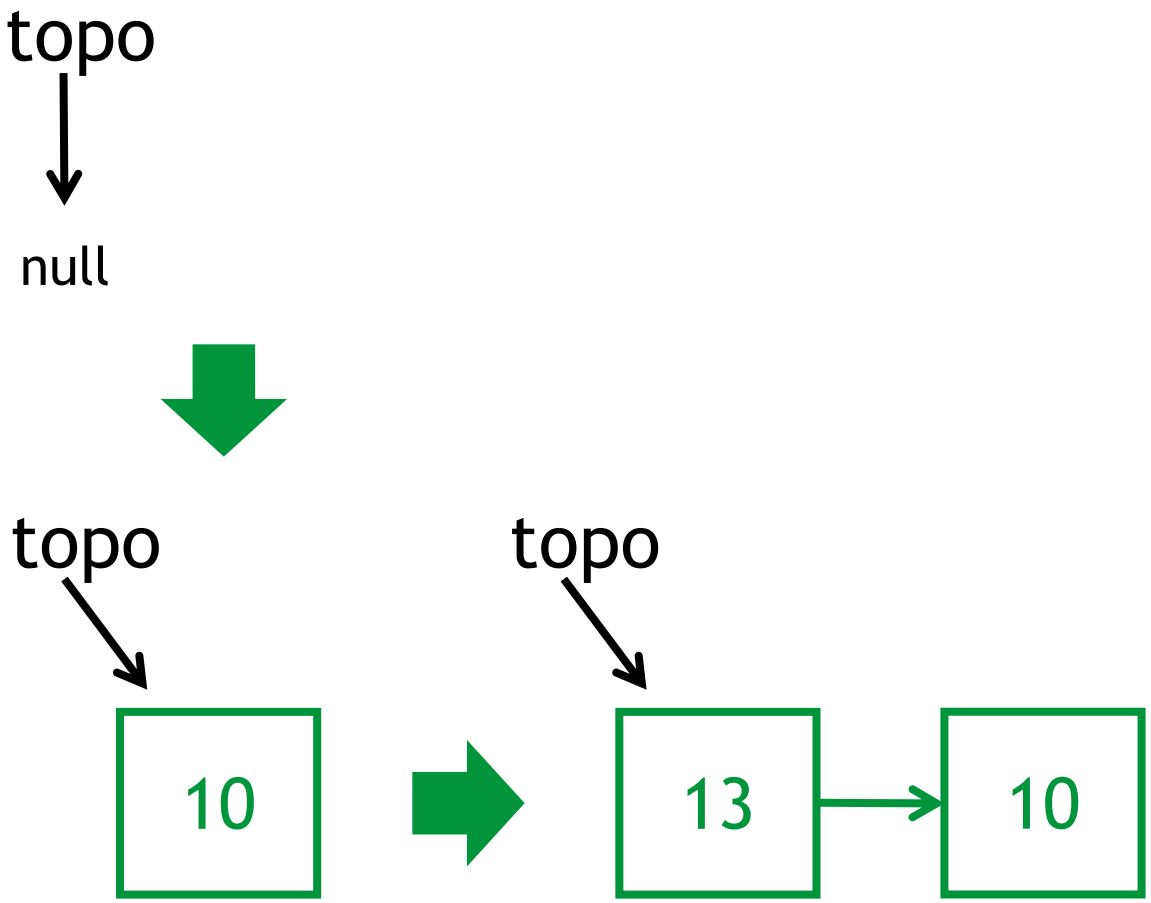
Pilhas

- Operação empilhar (push):



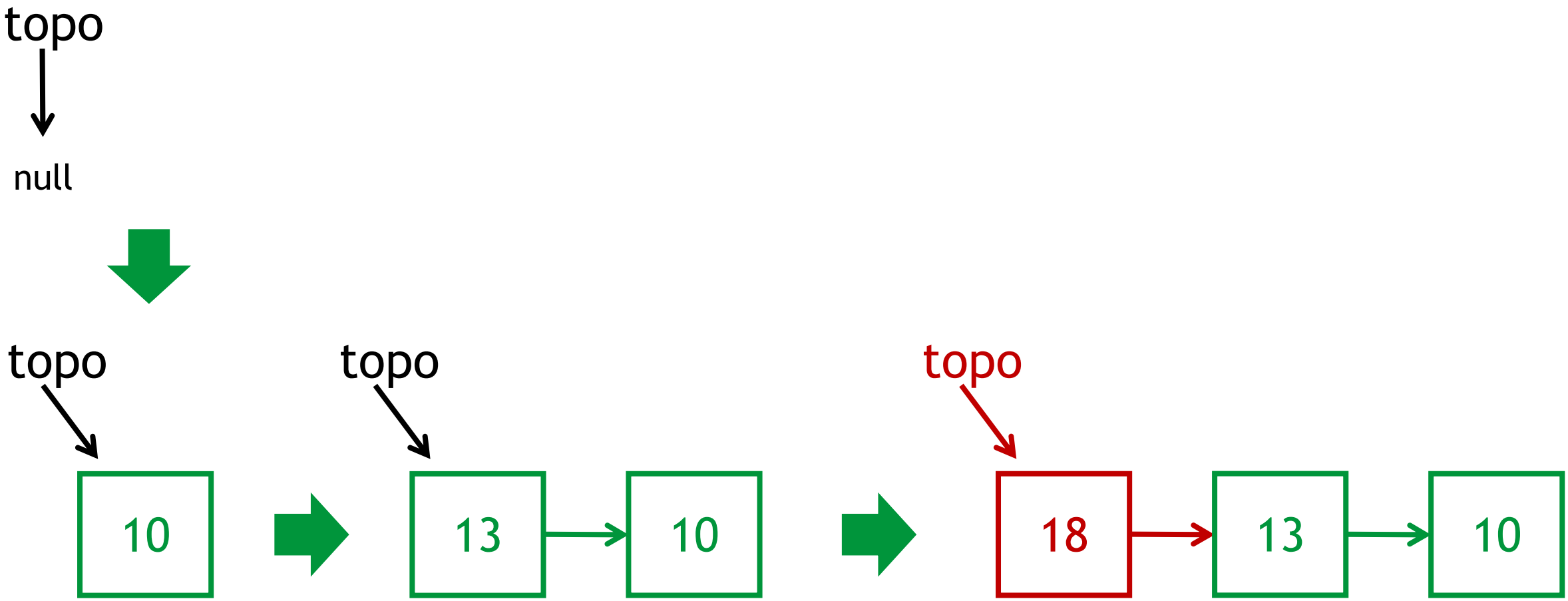
Pilhas

- Operação empilhar (push):



Pilhas

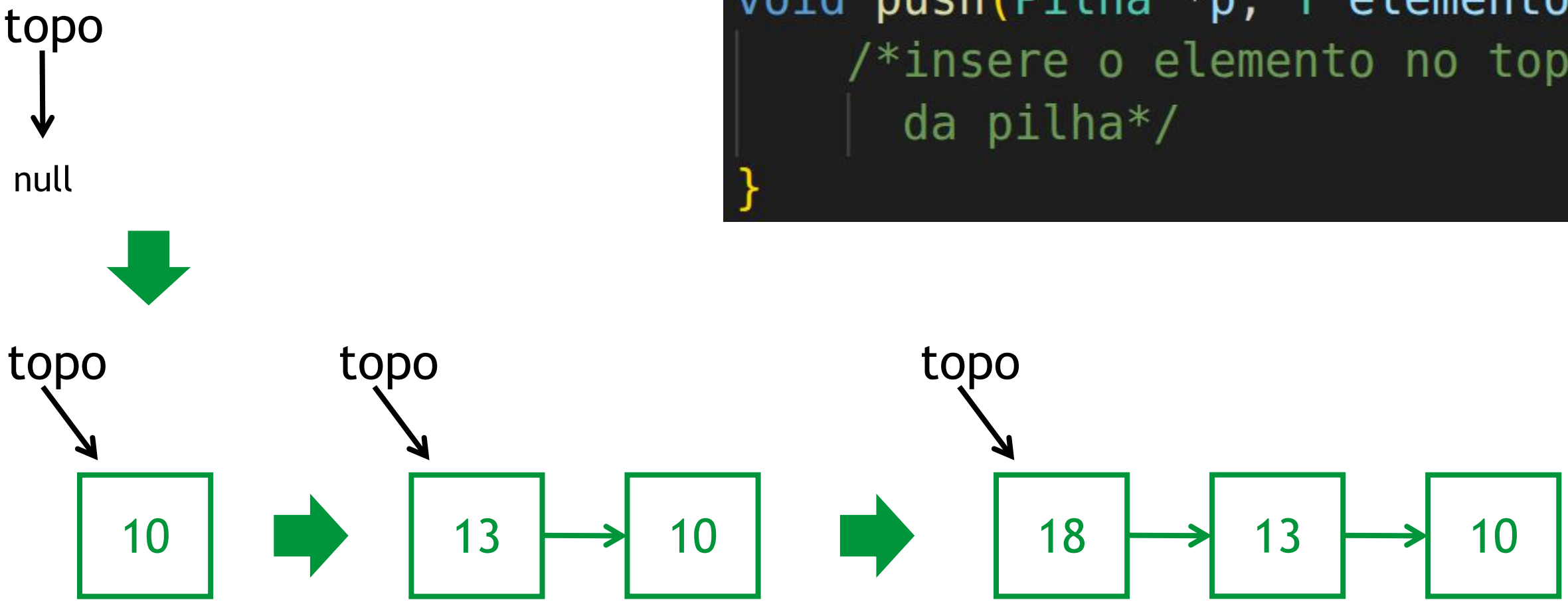
- Operação empilhar (push):



Pilhas

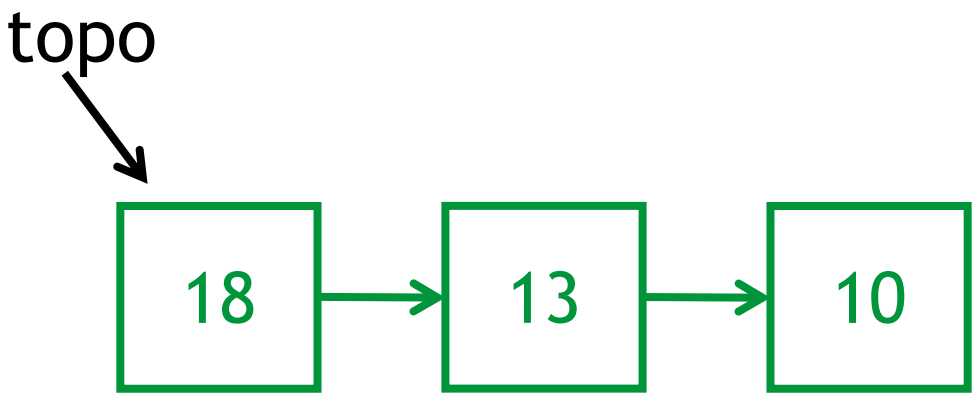
- Operação empilhar (push):

```
void push(Pilha *p, T elemento) {
    /*insere o elemento no topo
      da pilha*/
}
```



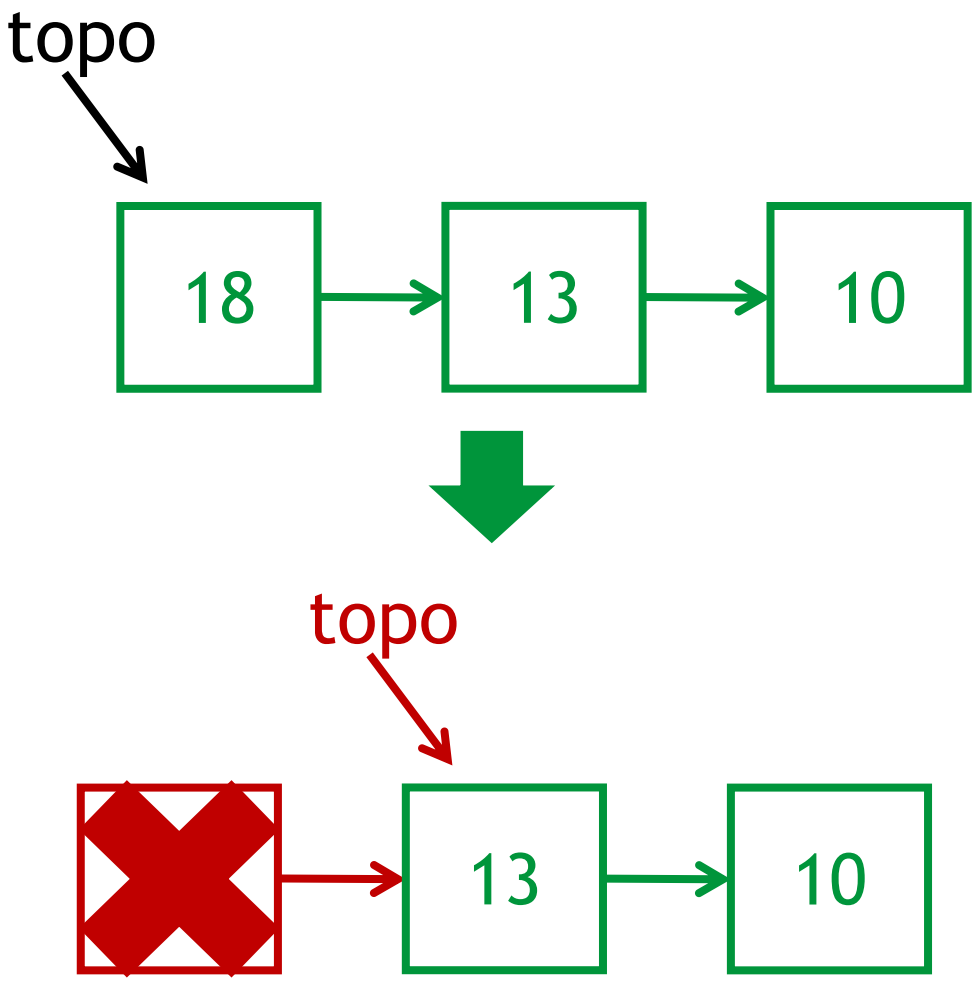
Pilhas

- Operação desempilhar (pop):



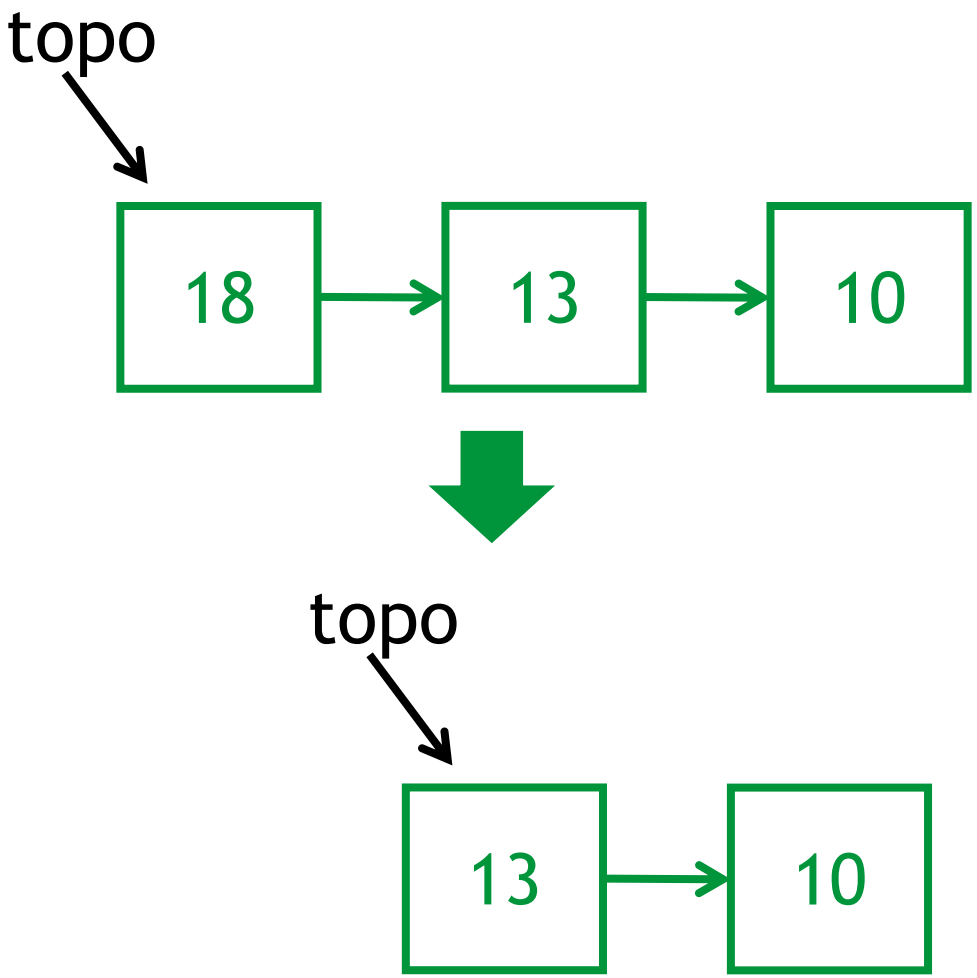
Pilhas

- Operação desempilhar (pop):



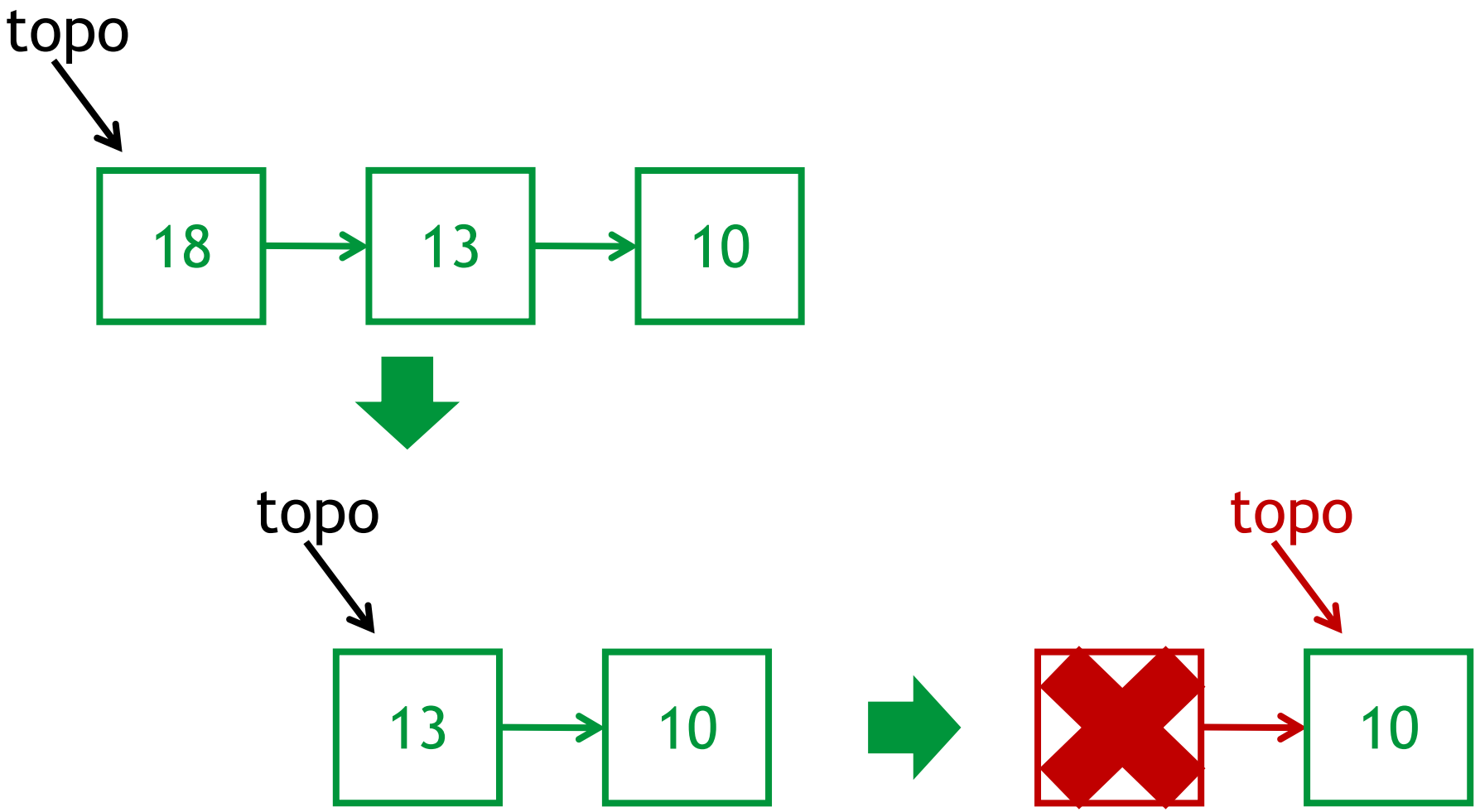
Pilhas

- Operação desempilhar (pop):



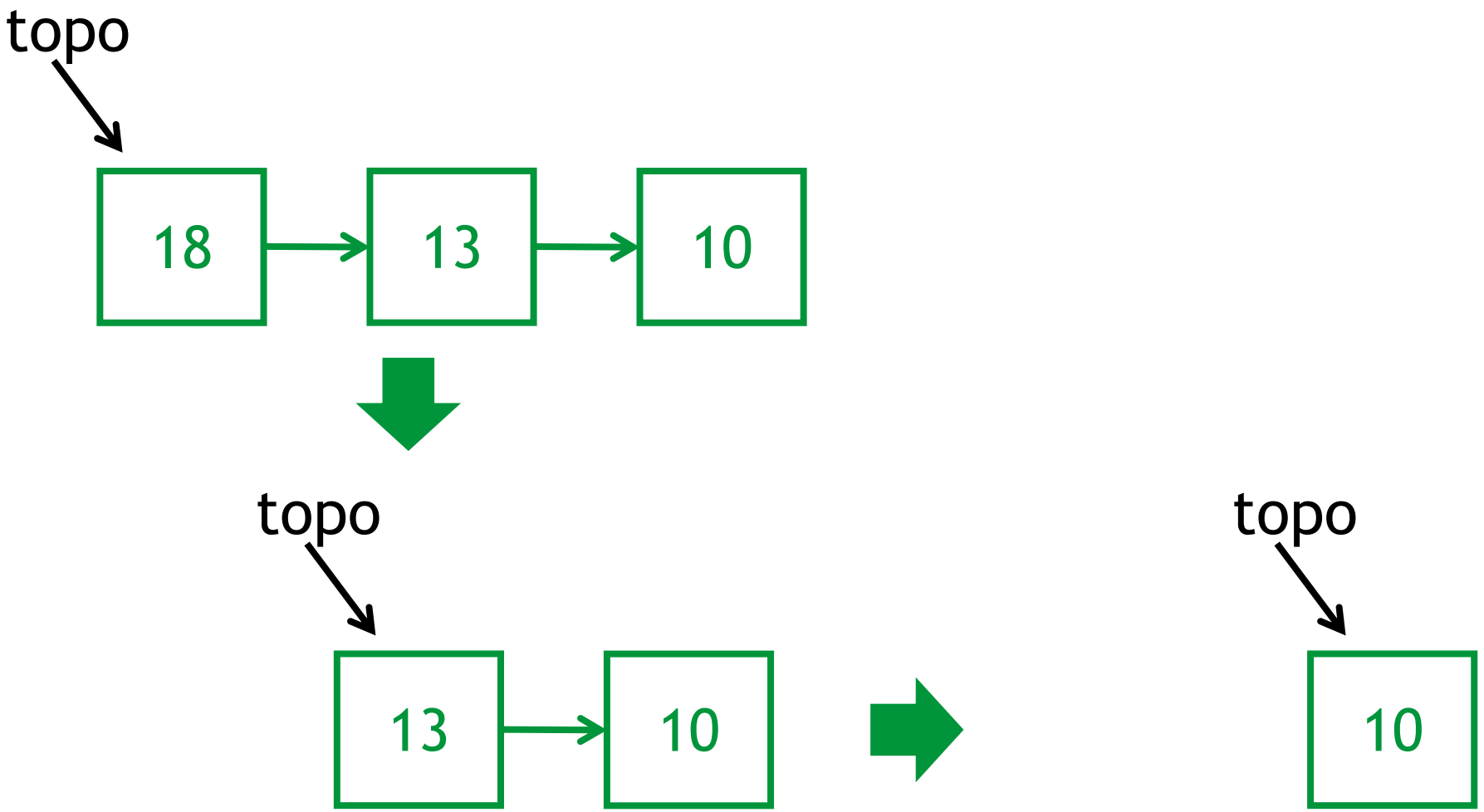
Pilhas

- Operação desempilhar (pop):



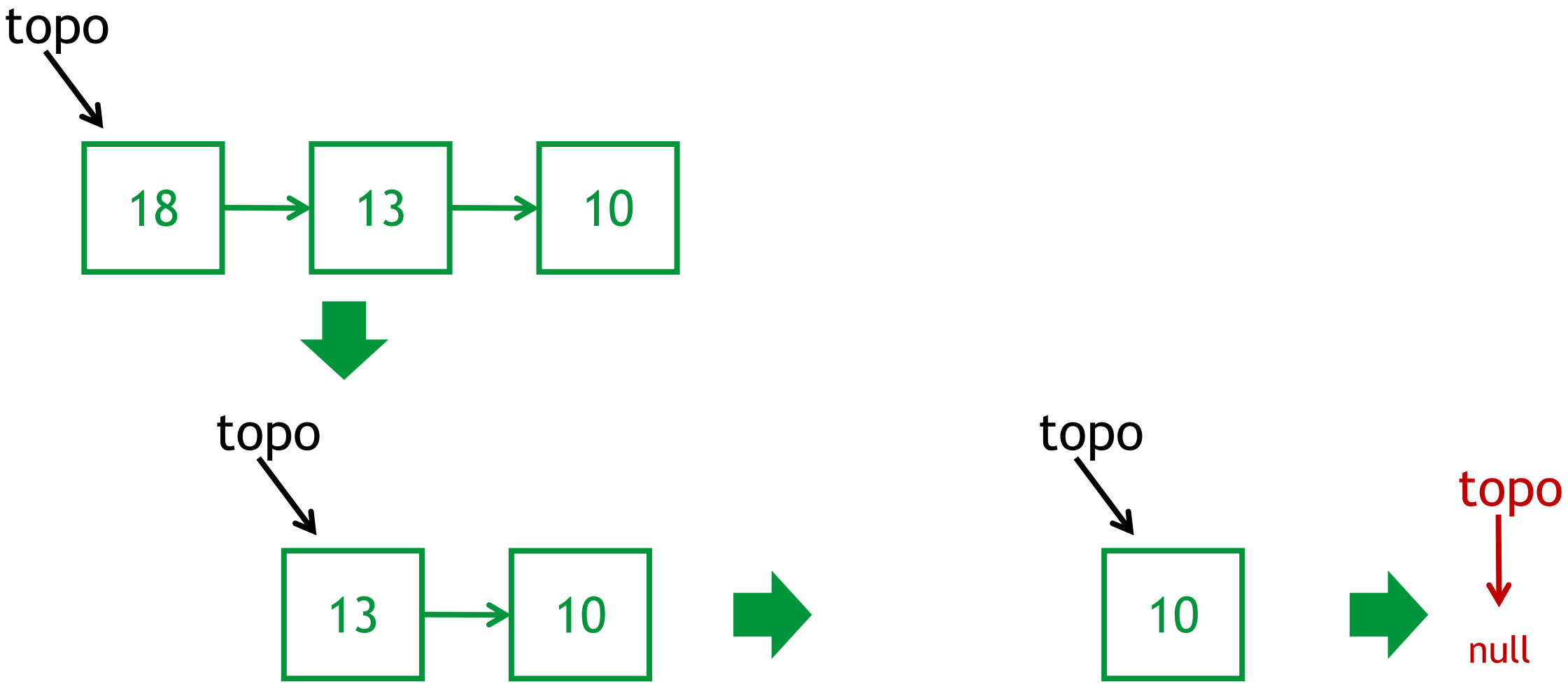
Pilhas

- Operação desempilhar (pop):



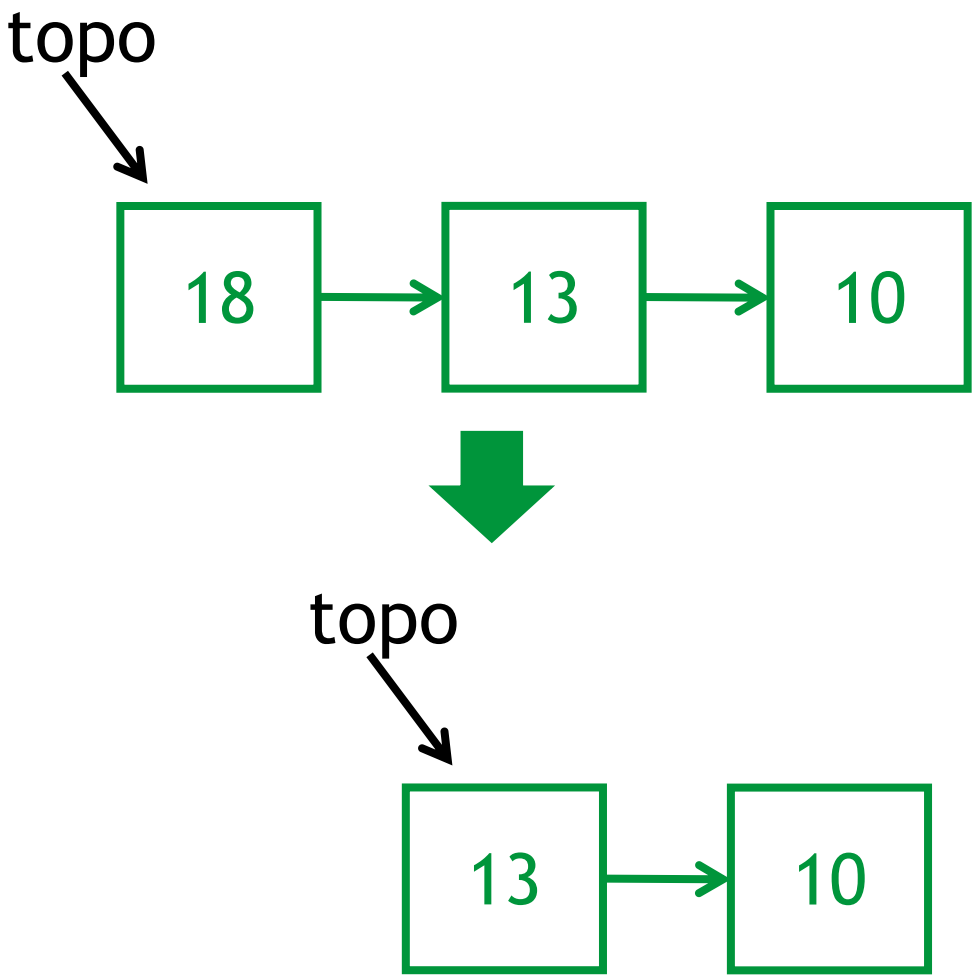
Pilhas

- Operação desempilhar (pop):



Pilhas

- Operação desempilhar (pop):

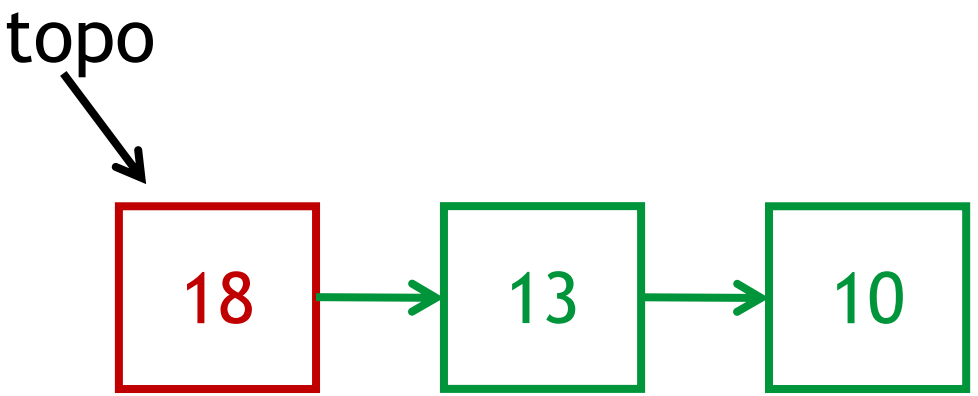



```
T pop(Pilha *p) {
    /*remove e retorna o
    elemento do topo da pilha*/
}
```

Pilhas

INSTITUTO FEDERAL
Pernambuco

- Operação de consulta do elemento do topo (peak):



Pilhas

- Verificar se a pilha está vazia (isEmpty)
- Verificar o tamanho da pilha (size)
- Limpar a pilha (clear)

Já discutimos a implementação dessas funcionalidades como listas!!!

Pilhas

INSTITUTO FEDERAL
Pernambuco

- Aplicações de pilhas:
  - Implementação de operações de “desfazer” e “refazer” em editores de texto;
  - Gerenciamento da pilha de chamadas de funções durante a execução de programas;
  - Avaliação e conversão de expressões aritméticas em compiladores e interpretadores;
  - Verificação de balanceamento de símbolos, como parênteses e chaves;
  - Navegação entre páginas em navegadores web (histórico de páginas visitadas).

Exercícios

INSTITUTO FEDERAL
Pernambuco

1. Implemente um Tipo Abstrato de Dados (TAD) Fila em C. A fila deve seguir a política FIFO (First In, First Out), em que o primeiro elemento inserido é o primeiro a ser removido.

- Funcionalidades:
  - void enqueue(Fila *f, T elemento); → insere um elemento no final da fila;
  - T dequeue(Fila *f); → remove e retorna o elemento do início da fila;
  - T front(Fila *f); → retorna o elemento do início sem removê-lo;
  - int isEmpty(Fila *f); → verifica se a fila está vazia;
  - int size(Fila *f); → retorna a quantidade de elementos armazenados;
  - void clear(Fila *f); → remove todos os elementos da fila.
- A fila deve ser implemente utilizando vetor e de forma encadeada simples.

Exercícios

INSTITUTO FEDERAL
Pernambuco

2. Implemente um Tipo Abstrato de Dados (TAD) Pilha em C. A pilha deve seguir a política LIFO (Last In, First Out), em que o último elemento inserido é o primeiro a ser removido.

- Funcionalidades:
  - void push(Pilha *p, T elemento); → insere um elemento no topo da pilha;
  - T pop(Pilha *p); → remove e retorna o elemento do topo da pilha;
  - T peek(Pilha *p); → retorna o elemento do topo sem removê-lo;
  - int isEmpty(Pilha *p); → verifica se a pilha está vazia;
  - int size(Pilha *p); → retorna a quantidade de elementos armazenados;
  - void clear(Pilha *p); → remove todos os elementos da pilha.
- A pilha deve ser implemente utilizando vetor e de forma encadeada simples.

Dúvidas

INSTITUTO FEDERAL
Pernambuco

ALGORITMOS E ESTRUTURA DE DADOS

Curso de Engenharia de Software
Lucas Sampaio Leite