Conteúdo

1. Introdução
2. Listas
3. Pilhas e Filas
4. Árvores
5. Árvores de Pesquisa
   - Árvore Binária e Árvore AVL
   - Árvore N-ária e Árvore B
6. Tabelas de Dispersão (Hashing)
7. Métodos de Acesso a Arquivos
8. Métodos de Ordenação de Dados

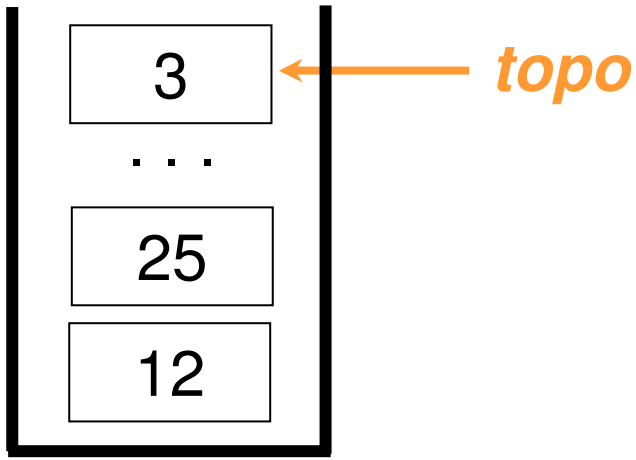
Pilhas

Pilha

PILHA: lista linear cujas inserções e exclusões se realizam em uma única extremidade (topo da lista).

O último elemento colocado na pilha fica no topo.

O elemento retirado da pilha é sempre o elemento do topo da pilha.

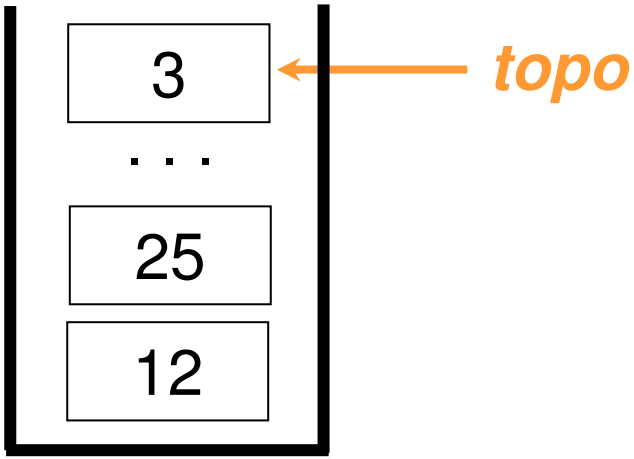


Pilha

➡ Uma pilha contém:

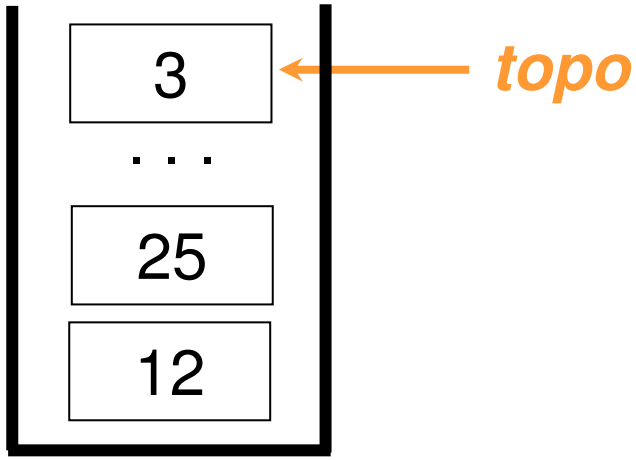
- um método para inserir elementos na pilha (empilha)
- um método para retirar elementos da pilha (desempilha)
- um método para acessar o topo da pilha (retornaTopo)

➡ As filas são conhecidas como listas LIFO (Last In First Out) - último que entra, primeiro que sai.

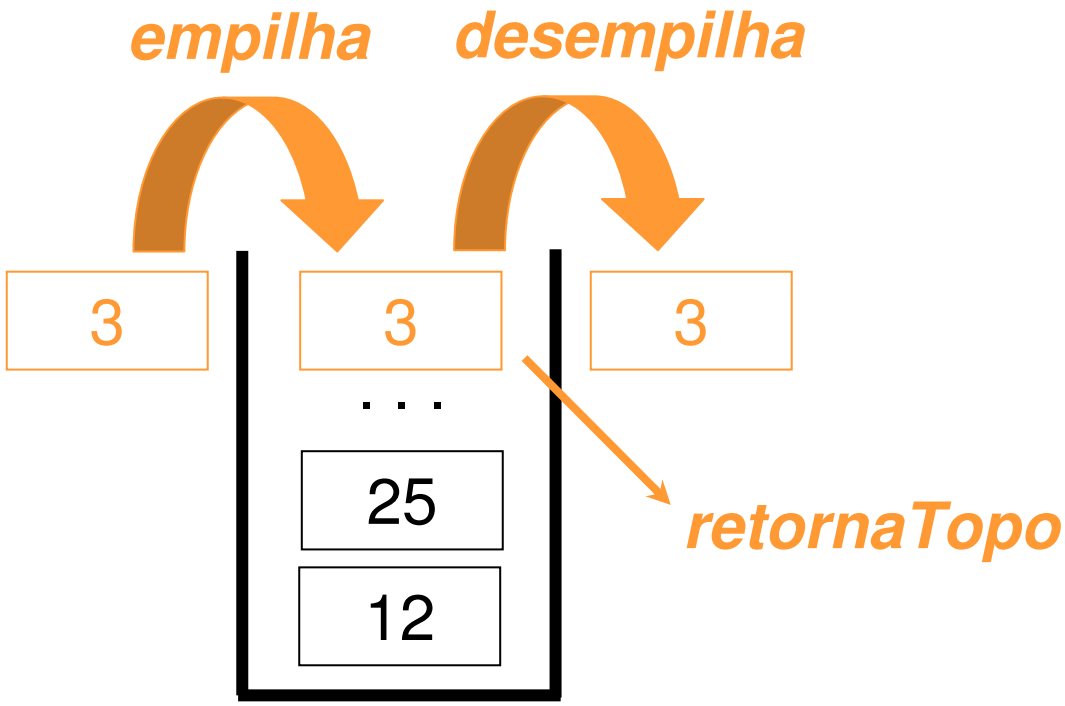


Exemplos de Pilhas

- Pilha de livros
- Pilha de caixas
- Pilha de “objetos” quaisquer, …



Operações sobre uma Pilha



LIFO: “last-in; first-out”

Interface da Pilha

```
public interface Pilha<E>  {

      ???

}
```

Interface da Pilha

```
public interface Pilha<E> extends EstruturaDeDados {

    public void empilha (E elemento);
    public E desempilha () throws ExcecaoEstruturaVazia;
    public E topo () throws ExcecaoEstruturaVazia;
    public boolean contem (E elemento);
    public int retornaPosicao (E elemento);
}
```

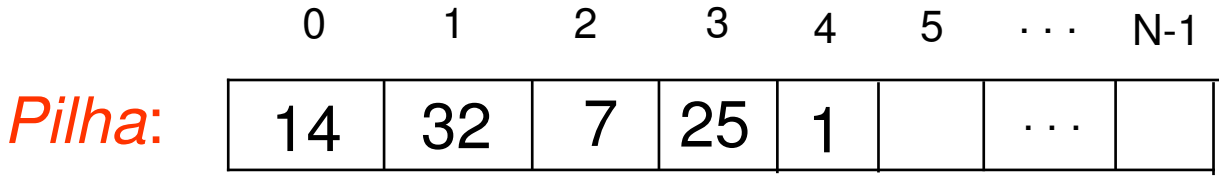
Alternativas de Implementação

- Pilha como Array
- Pilha como Lista Encadeada

Pilha como Array

Considere a pilha como uma estrutura linear representada através de um array com N posições.

Uma pilha tem um atributo topo que indica a posição do topo da pilha no array.

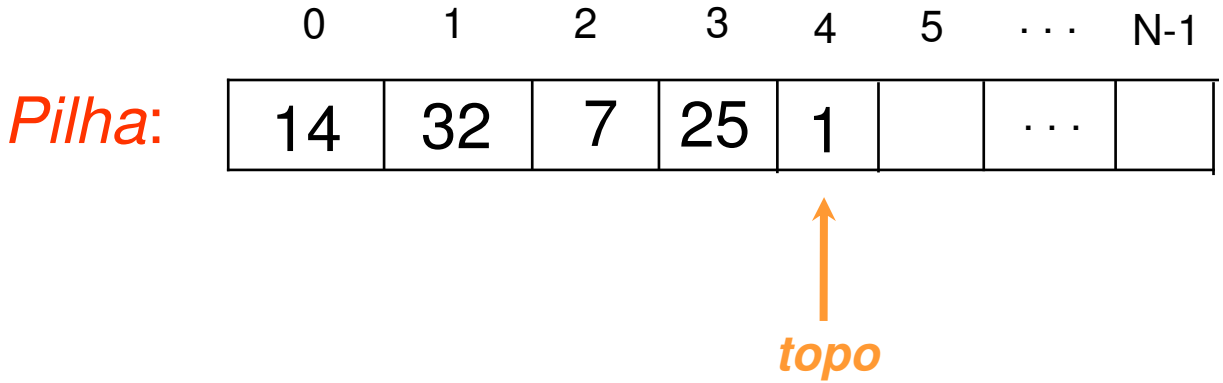


Onde deve ficar o topo da pilha?

Pilha como Array

Considere a pilha como uma estrutura linear representada através de um array com N posições.

Uma pilha tem um atributo topo que indica a posição do topo da pilha no array.



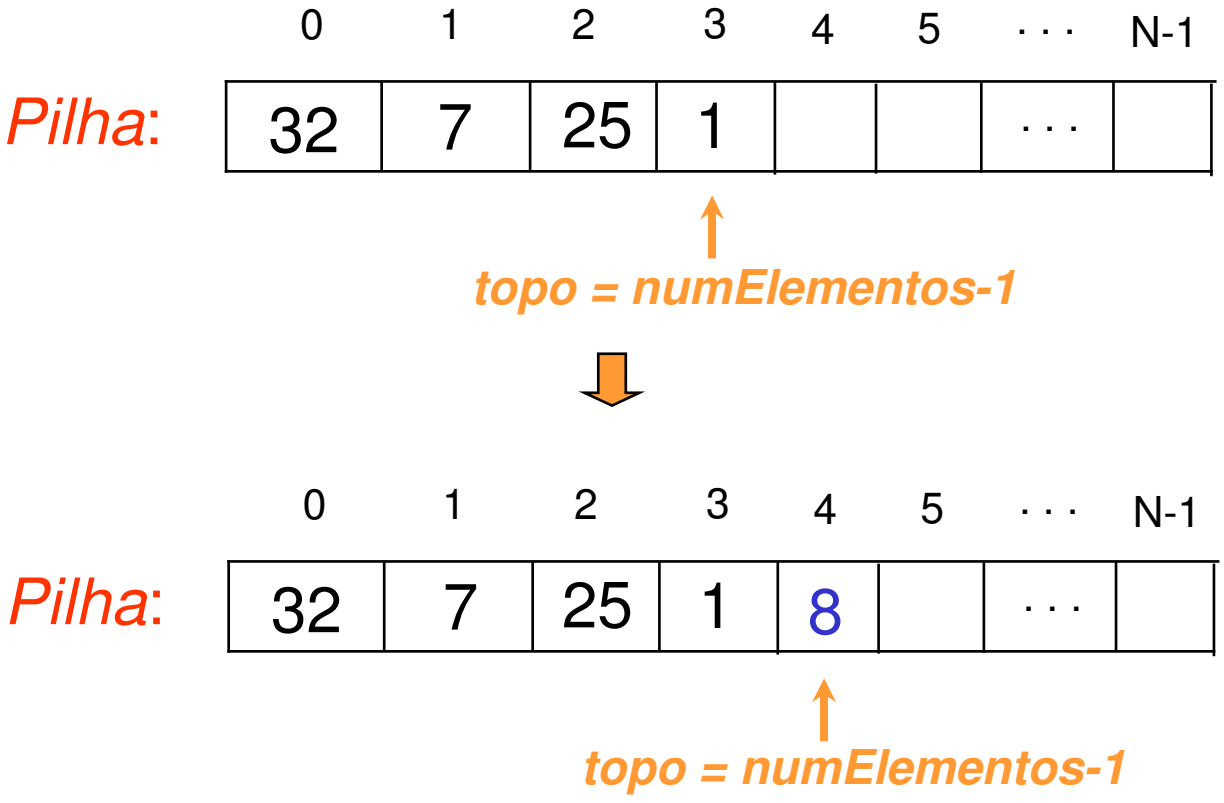
Pilha como Array - Construtor

```
public class PilhaArray<E> implements Pilha<E> {
   private E[] elementos;
   private int numElementos;

public PilhaArray (int tamanho) {
   this.elementos = (E[]) new Object[tamanho];    }

public PilhaArray () {
   this.elementos = (E[]) new Object[10];    }

…
}
```

a partir do numElementos sabemos qual é o topo

Complexidade: O(?)

Pilha como Array - Empilha



Pilha como Array - Empilha

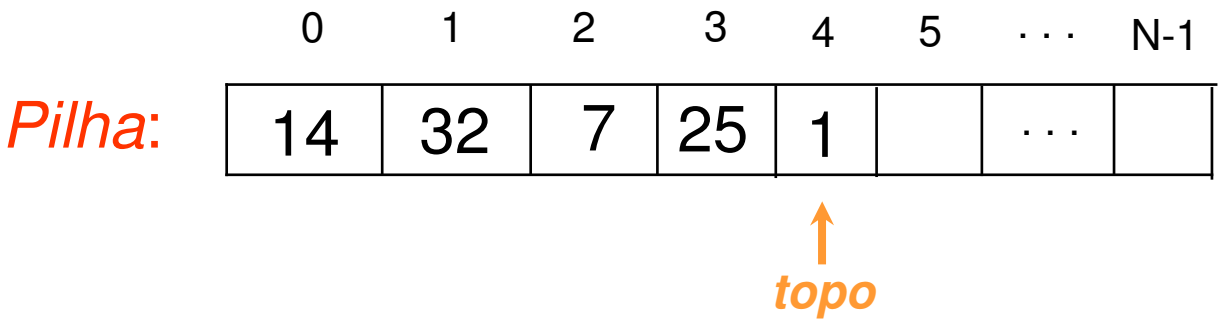
```
public void empilha (E elemento) {

???

}
```

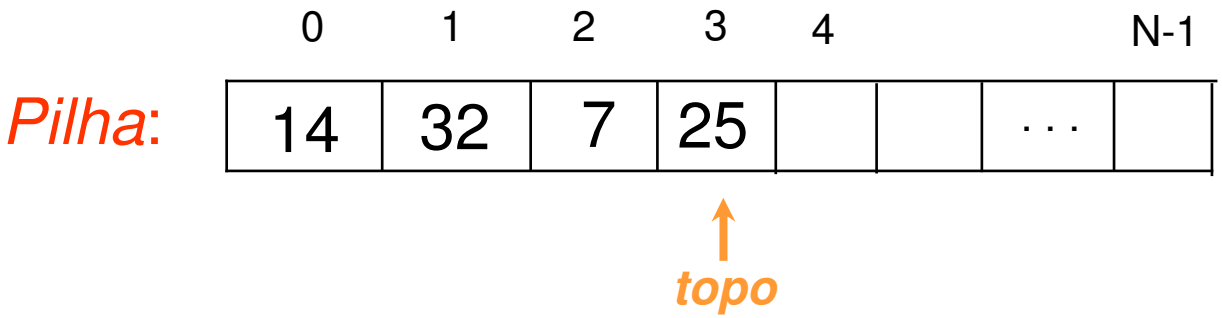
Pilha como Array - Empilha

```
public void empilha (E elemento) {
   if (this.numElementos == this.elementos.length) {
      E[] novoArray = (E[]) new Object[this.elementos.length * 2];
      System.arraycopy(elementos,0,novoArray,0,this.elementos.length);
      this.elementos = novoArray;
   }
   this.elementos[this.numElementos] = elemento;
   this.numElementos++;
}
```

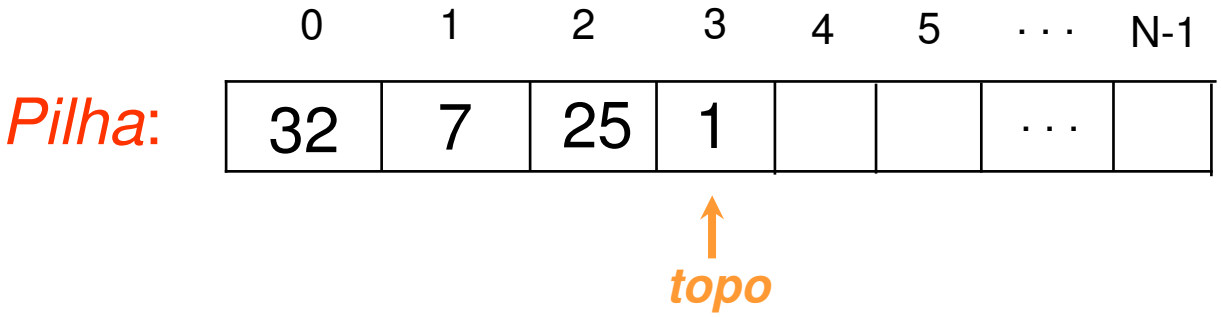
Complexidade: O(?)

Pilha como Array - Desempilha



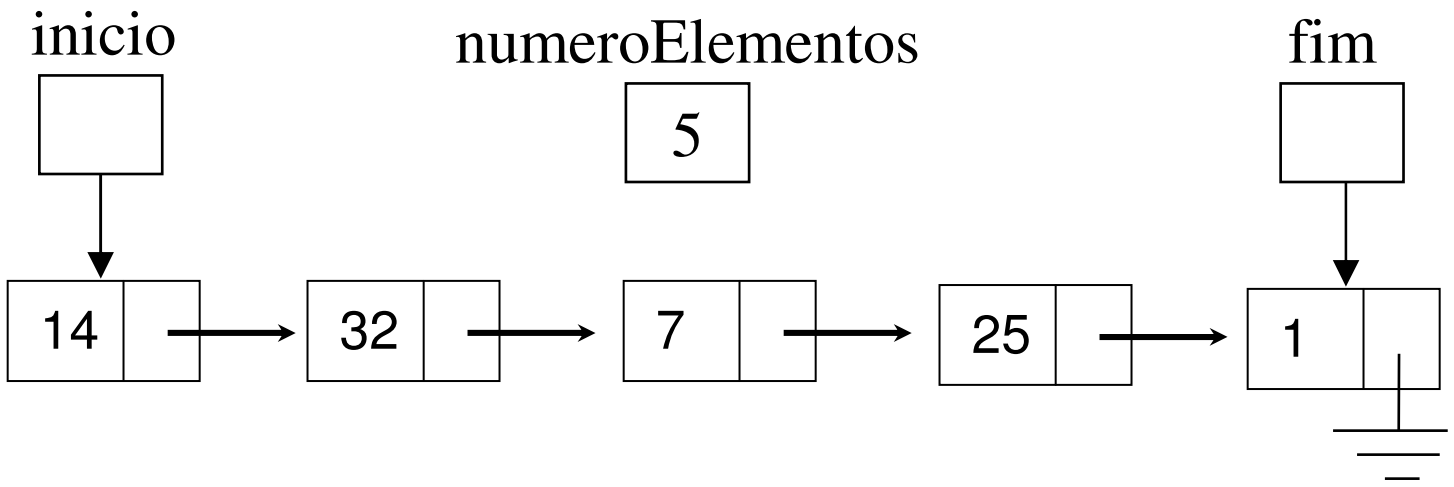


Pilha como Array - Topo



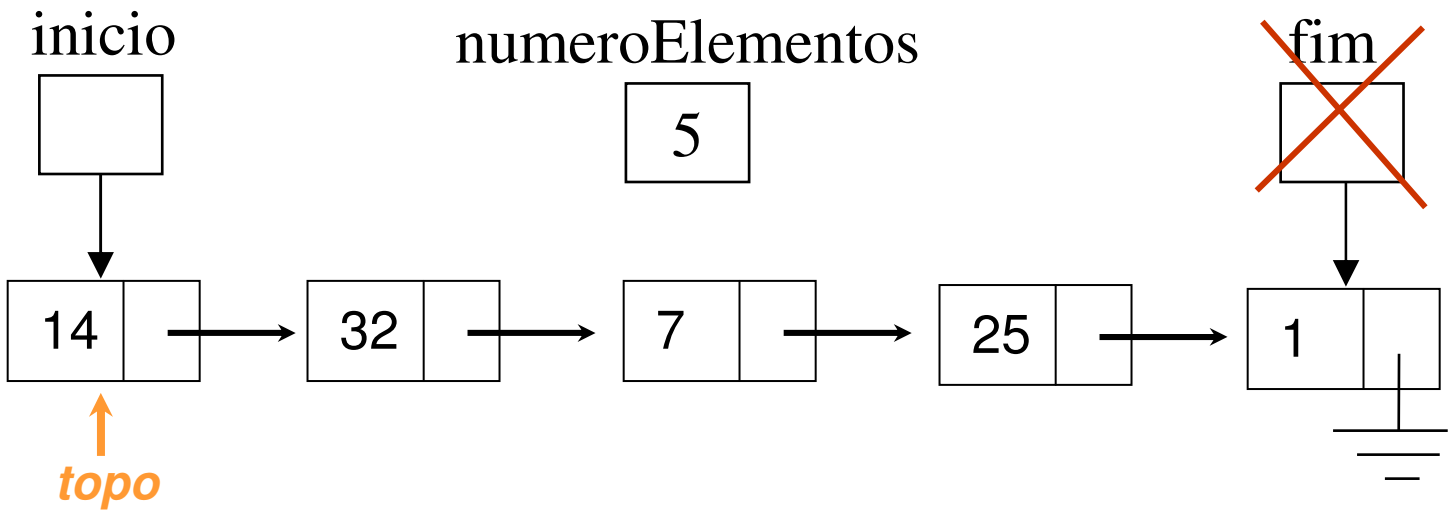
➡ ***Topo = 3***

Pilha Encadeada



Onde deve ficar o topo da pilha?

Pilha Encadeada



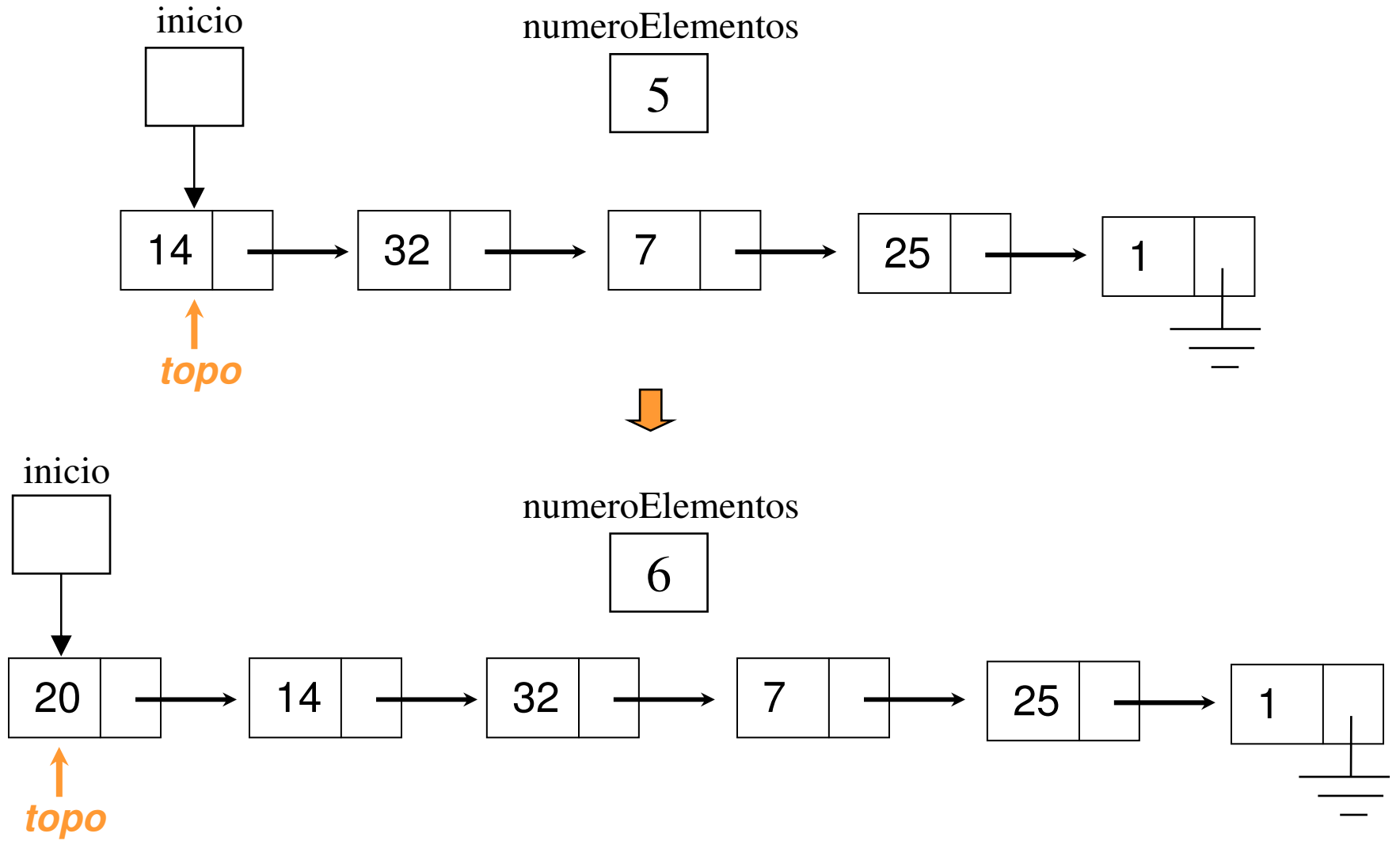
Onde deve ficar o topo da pilha?

Pilha Encadeada - Construtor

```
public class PilhaEncadeada<E> implements Pilha<E> {
   private Nodo inicio;
   private int numElementos;

public PilhaEncadeada() {
   this.inicio = null;
   this.numElementos = 0;
}

…
}
```

Complexidade: O(1)

Pilha Encadeada - Empilha



Pilha Encadeada - Empilha

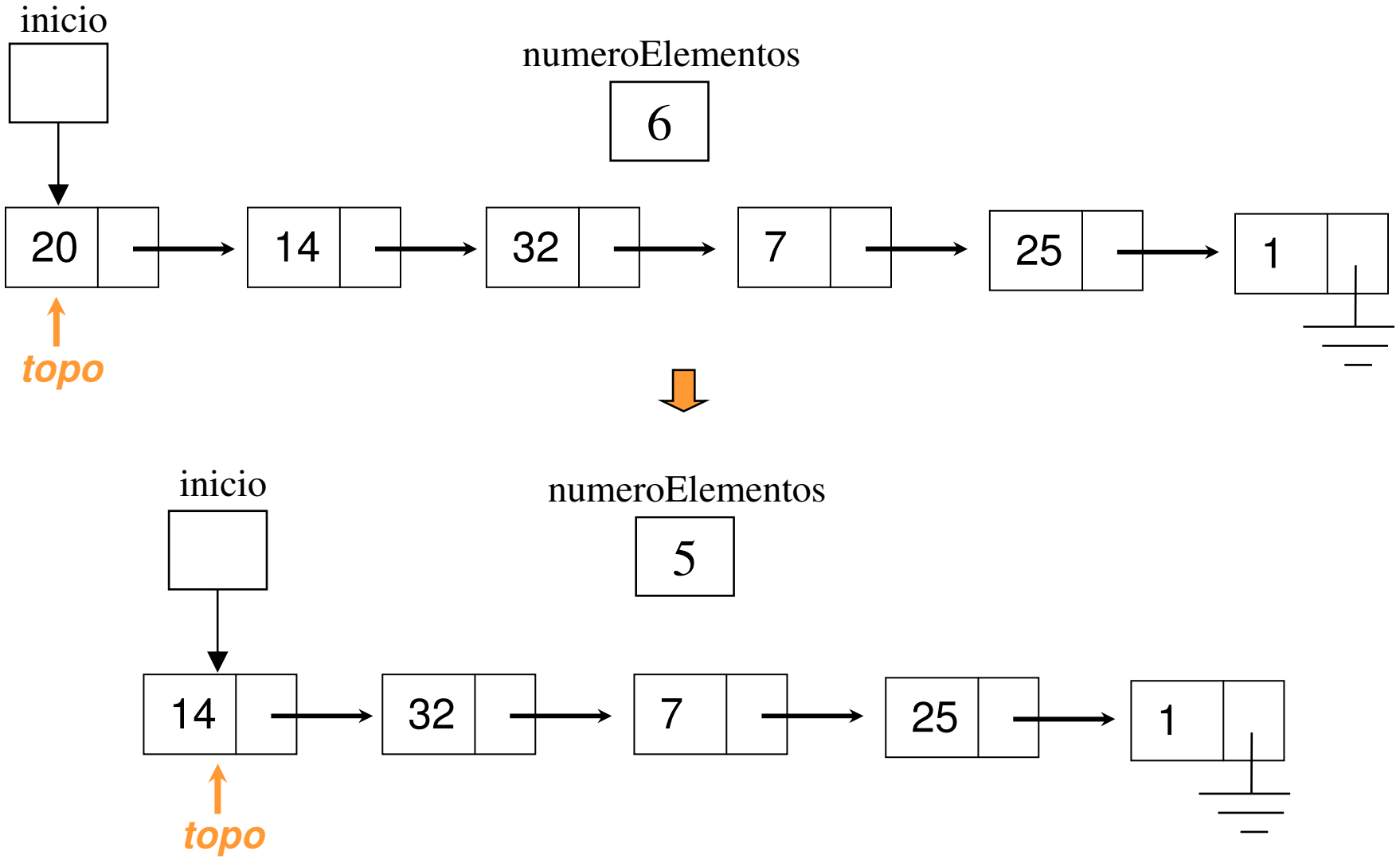
```
public void empilha (E elemento) {
        ???
}
```

Pilha Encadeada - Empilha

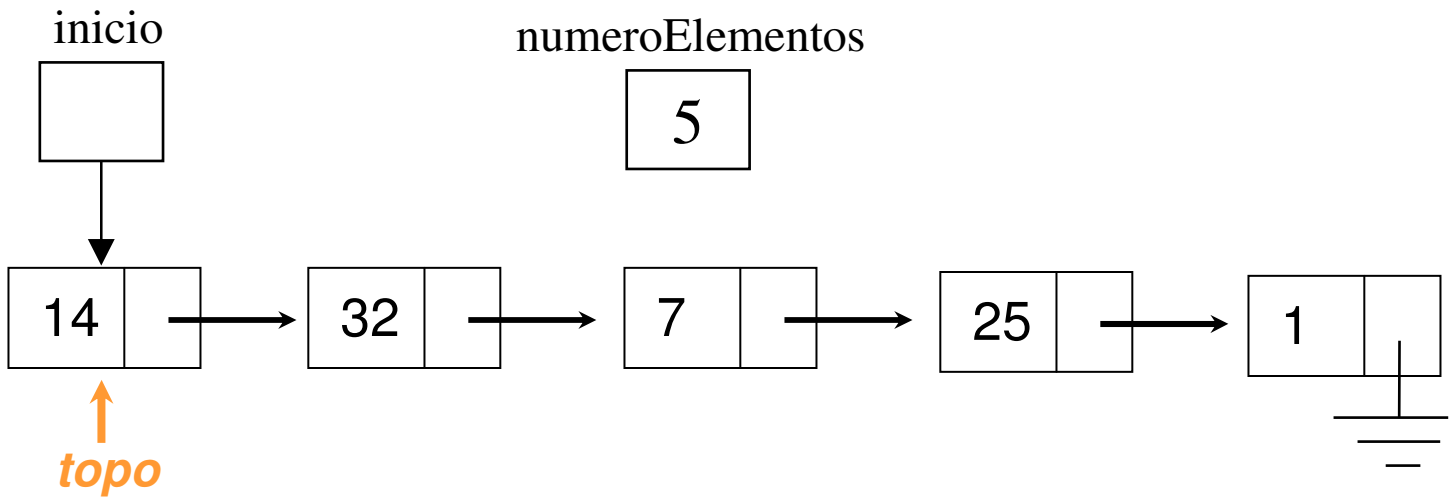
```
public void empilha (E elemento) {
   Nodo nodo = new Nodo (elemento);
   nodo.atribuiProximo (this.inicio);
   this.inicio = nodo;
   this.numElementos++;
}
```

Complexidade: O(?)

Pilha Encadeada - Desempilha



Pilha Encadeada - Topo



➡ ***Topo = 14***