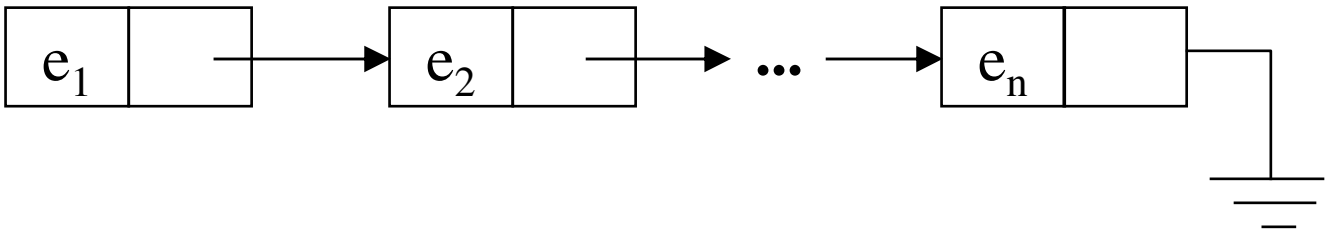
Conteúdo

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

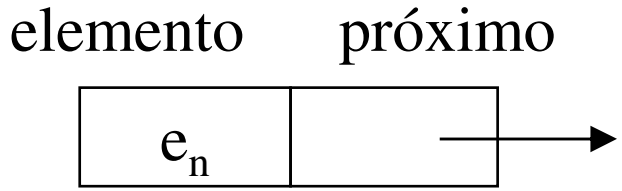
Listas Implementadas Utilizando Encadeamento

Listas Encadeadas

- Os elementos estão associados entre si através de referências.

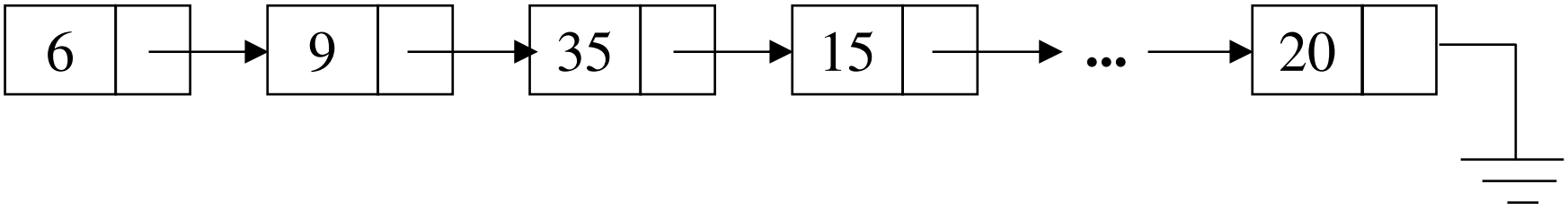


- Cada nodo da lista mantém um elemento e uma referência para o próximo nodo.



Listas Encadeadas

Exemplo



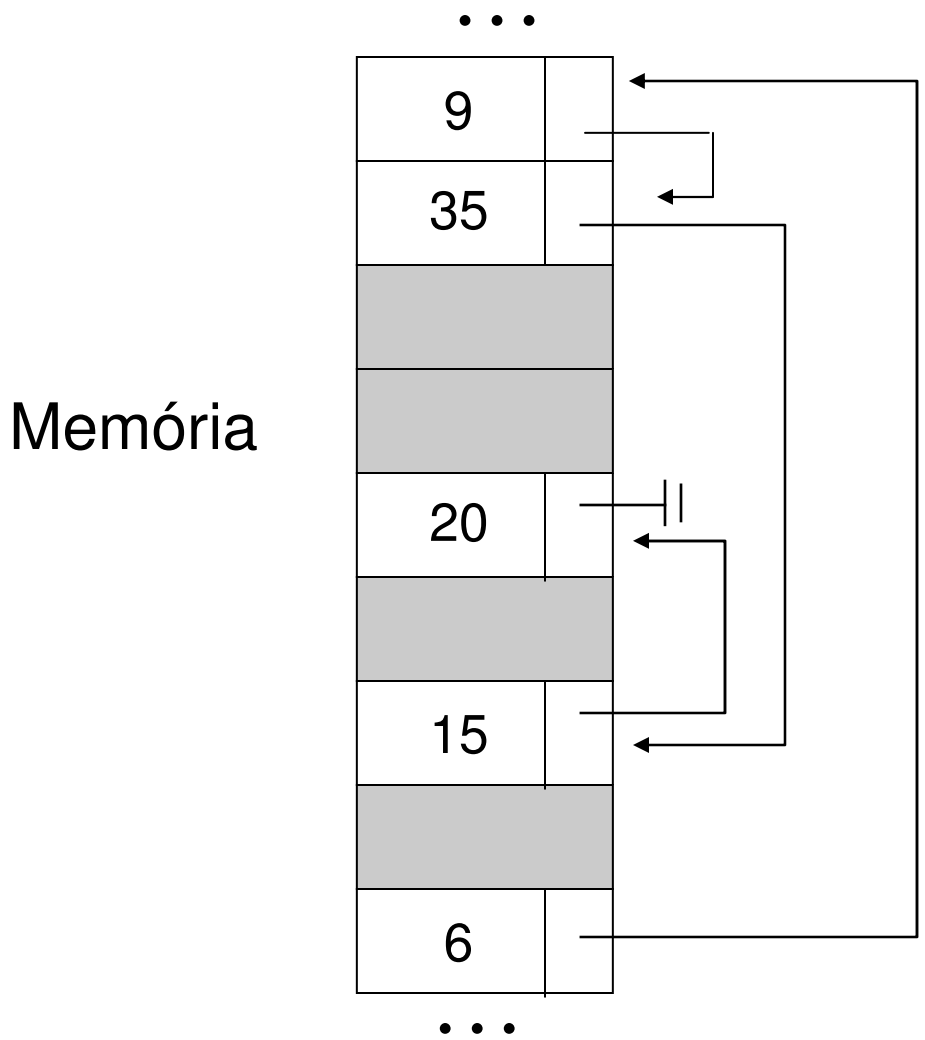
⇨ Não há limite máximo para o número de elementos na lista

- O limite é a capacidade da memória!

⇨ Elementos não estão necessariamente em posições contíguas da memória

- Alocação de novos elementos em tempo de execução, conforme a lista cresce.

Características



Vantagens e Desvantagens

Vantagens:

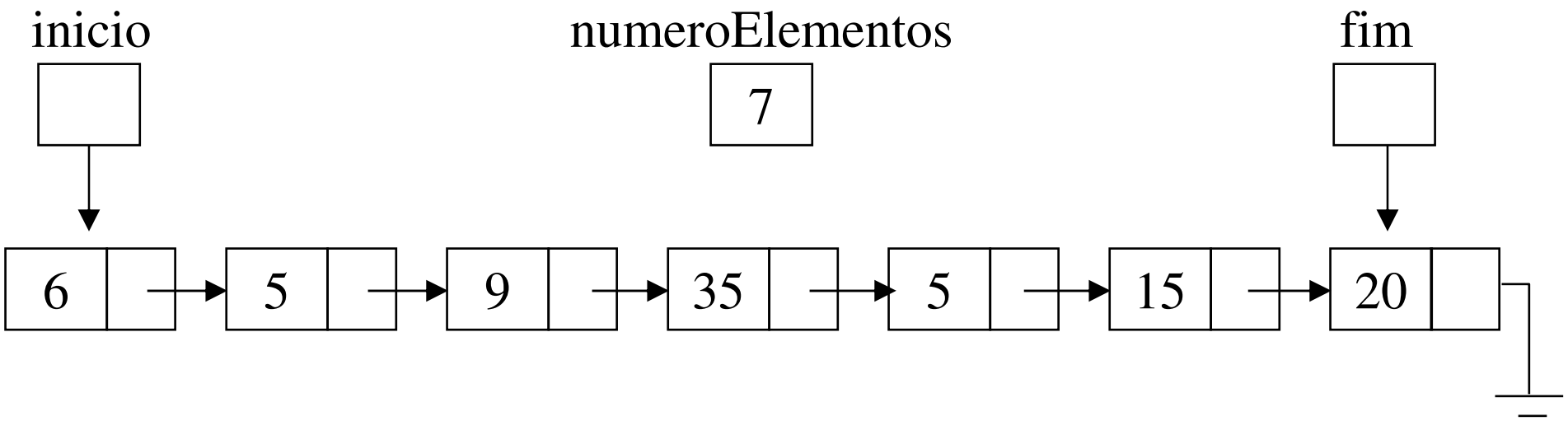
- As operações de inserção e remoção são mais simples.
- Melhor aproveitamento da memória, pois não é alocado previamente o espaço de toda a estrutura.

Desvantagem:

- A operação de acesso a um elemento em determinada posição é mais trabalhosa, pois requer que todos os nodos anteriores sejam percorridos.

Operações sobre a Lista

Exemplo:



Operações sobre a Lista

Operações:

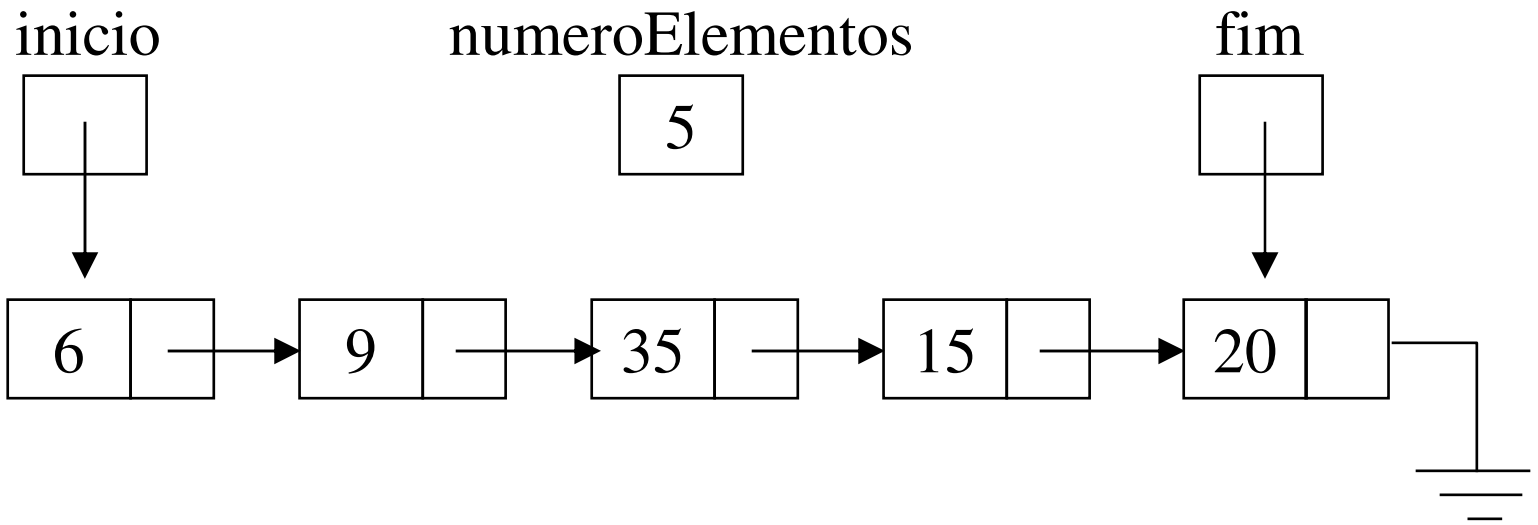
- inserir o elemento 30 na posição 7
- inserir o elemento 15 na posição 3
- inserir o elemento 16 na posição 0
- excluir da posição 9
- excluir da posição 5
- excluir da posição 0
- excluir o elemento 20
- retorna o elemento da posição 3
- retorna a posição do elemento 5

Implementação

- Classe **ListaEncadeada** implementa Lista

Atributos

- **inicio** (referência a um objeto da classe Nodo)
- **fim** (referência a um objeto da classe Nodo)
- **numeroElementos**

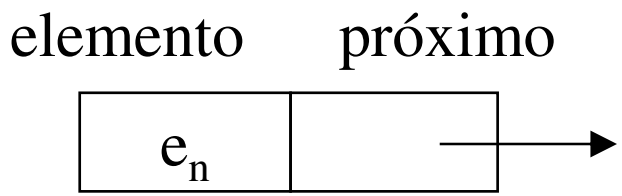


Implementação

- Classe **Nodo**

Atributos:

- **elemento**
- **proximo** (referência a um outro objeto da classe Nodo)



Implementação

- Classe **Nodo**

Atributos:

- private E elemento
- private Nodo proximo

Métodos:

- public Nodo (E elemento)
- public void atribuiElemento (E elemento)
- public void atribuiProximo (Nodo nodo)
- public E retornaElemento ( )
- public Nodo retornaProximo ( )

Implementação

⇨ Classe **ListaEncadeada** implementa Lista

Atributos

- **inicio** (referência a um objeto da classe Nodo)
- **fim** (referência a um objeto da classe Nodo)
- **numeroElementos**

Métodos

- construtor ()

  +

- métodos especificados na interface Lista

Implementação

- Método construtor da Lista Encadeada

```
public class ListaEncadeada<E> implements Lista<E> {

private Nodo inicio;
private Nodo fim;
private int numElementos;

public ListaEncadeada() {
        this.inicio = null;
        this.fim = null;
        this.numElementos = 0;
}
```

Implementação

Métodos especificados na interface Lista

- public void insere (E elemento);

/** Insere o elemento no final da lista.

* @param elemento Elemento a ser inserido no final da lista. */

- public E remove (int posicao) throws ExcecaoPosicaoInvalida;

/** Remove o elemento que se encontra na posição indicada pelo argumento e retorna o elemento.

* @param posicao Posicão do elemento que será removido da lista. A lista começa na posição 0.

* @return O elemento removido.

* @throws ExcecaoPosicaoInvalida se a posição for menor do que zero, ou igual ou maior do que o número de elementos. */