



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 Array





Implementação com Array

➡ Os elementos ficam justapostos na memória através da utilização de um array unidimensional.

0	1	2	3	4	5	6	7	8	9	...	N
e_1	e_2	e_3	e_4	e_5	e_6	e_7	...	e_m			





Características

- Posições contíguas de memória.
- Acesso direto a um elemento (posições são indexadas)
- Lista possui tamanho máximo predefinido (a não ser que, dependendo da linguagem, um novo array seja criado em tempo de execução).

0	1	2	3	4	5	6	7	8	9	...	N
6	5	9	35	5	15	18	...	20			

Exemplo: `Array[4] = 5`





Desvantagens

- Necessidade de deslocar os elementos da lista sempre que houver uma inserção ou exclusão, para preservar a ordenação lógica ou seqüencial.
- Não otimização dos recursos de memória porque o número de posições do array é pré-determinado (a não ser que, dependendo da linguagem, um novo array seja criado em tempo de execução).

0	1	2	3	4	5	6	7	8	9	...	N
6	5	9	35	5	15	18	...	20			





Operações sobre a Lista

Exemplo:

0	1	2	3	4	5	6	7	8	9	10	11
6	5	9	35	5	15	18					

totalElementos = 7

tamanhoArray = 12





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 18
- retorna o elemento da posição 3
- retorna a posição do elemento 5





Implementação

⇒ Classe **ListaArray** implementa Lista

Atributos

- **elementos** (array que contém os elementos da lista)
- **numeroElementos**

elementos

0	1	2	3	4	5	6	7	8	9	...	N
6	5	9	35	5	15	18					

numeroElementos

7





Implementação

⇒ Classe **ListaArray** implementa Lista

Atributos

- **elementos** (array que contém os elementos da lista)
- **numeroElementos**

Métodos

- construtor ()
+
• métodos especificados na interface Lista





Implementação

⇒ Método construtor da ListaArray

```
public class ListaArray<E> implements Lista<E> {  
  
    private E[] elementos;  
    private int numElementos;  
  
    public ListaArray (int tamanho) {  
        this.elementos = (E[]) new Object[tamanho];  
    }  
  
    public ListaArray ( ) {  
        this.elementos = (E[]) new Object[10];  
    }  
}
```





Implementação

⇒ Métodos especificados na interface Lista

- `public void insere (E elemento, int posicao) throws ExcecaoPosicaoInvalida`

`/**`

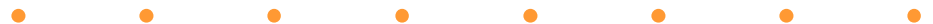
`* Insere o elemento na posição indicada pelo parâmetro.`

`* @param elemento Elemento a ser inserido na lista`

`* @param posicao Posição em que o elemento será inserido na lista. A lista começa na posição 0.`

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

`*/`





Implementação

⇒ Métodos especificados na interface Lista

- public int retornaPosicao (E elemento)

/**

* Retorna a posição na lista do elemento indicado pelo parâmetro.

* Para os elementos repetidos, retorna a posição da primeira ocorrência.

* @param elemento Elemento da lista cuja posição será retornada.

* @return A posição do elemento. Caso o elemento não exista na lista, retorna -1. A lista começa na posição 0.

*/

