

LISTAS

Profa. Me. Andréa Zotovici

**São Paulo
2018**

INTRODUÇÃO

Tipo Abstrato de Dados.

É descrito por propriedades e operações.

Quando utilizado não é necessário conhecer detalhes da sua implementação, apenas as suas operações.

Encapsula os dados necessários e as operações que podem ser realizadas sobre os dados.

Referências Complementares:

ICMC-USP. Disponível em <http://www.lcad.icmc.usp.br/~nonato/ED/node6.html>.

Acesso em: 05/03/2018.

INTRODUÇÃO

Quais informações manipulamos diariamente como uma coleção?

INTRODUÇÃO

Quais informações manipulamos diariamente como uma coleção?

- Alunos
- Cinemas
- Contatos Pessoais
- Contatos Profissionais
- Escolas
- Livros
- Presentes

INTRODUÇÃO

Quais operações podem ser realizadas sobre as coleções citadas?

INTRODUÇÃO

Operações:

- Busca;
- Combinação de duas listas em uma;
- Intercalação;
- Inserção;
- Ordenação;
- Remoção;
- etc.....

INTRODUÇÃO

Qual é a relação entre listas e tipo abstrato de dados?

INTRODUÇÃO

Lista é um tipo abstrato de dados.

A Lista pode possuir elementos:

- com tipo primitivo de dados, como uma lista de números inteiros, uma lista de letras ou uma lista de números reais;
- com tipo abstrato de dados, como uma lista de alunos, lista de contatos, lista de cinemas e etc...

LISTA COM ELEMENTOS DO TIPO PRIMITIVO DE DADOS

ListaDeInteiros
- dados: int[] - tamanho: int
+ ListaDeInteiros(c: int) + vazia(): boolean + cheia(): boolean + adicionaInicio(e: int): void + adicionaFinal(e: int): void + adiciona(int e, int posicao): void + removeInicio(): int + removeFinal(): int + remove(int posicao): int + obtemPrimeiro(): int + obtemUltimo(): int

L1:ListaDeInteiros				
	0	1	2	3
dados	10	8	12	
tamanho	3			

LISTA COM ELEMENTOS DO TIPO ABSTRATO DE DADOS

Dada a lista de alunos

listaAlunos: ListaDeAlunos			
	0	1	2
dados	Nome: “Adriana” RA: 12135567	Nome: “Adriano” RA: 13100314	
tamanho	2		

LISTA COM ELEMENTOS DO TIPO ABSTRATO DE DADOS

ListaDeAlunos
<ul style="list-style-type: none">- dados: Aluno[]- tamanho: int
<ul style="list-style-type: none">+ ListaDeAlunos(c: int)+ vazia(): boolean+ cheia(): boolean+ adicionaInicio(e: Aluno): void+ adicionaFinal(e: Aluno): void+ adiciona(Aluno e, int posicao): void+ removeInicio(): Aluno+ removeFinal(): Aluno+ remove(int posicao): Aluno+ obtemPrimeiro(): Aluno+ obtemUltimo(): Aluno

ESTÁTICA

X DINÂMICA

Alocação Estática

Quantidade constante de elementos.

Aloca espaço de acordo com a quantidade de elementos.

Usa *arrays*.

Alocação Dinâmica

Não há quantidade máxima de elementos (o limite é a memória do computador).

Utiliza somente o espaço de memória suficiente para determinado elemento.

Ponteiros ou Referências.

ALOCAÇÃO ESTÁTICA

Dada a Lista Vazia

L1:ListaDeInteiros			
dados	0	1	2
			tamanho: 0

Adicione o elemento 15 no início da lista, processo:

- Lista está cheia?

- Não:

- guarde 15 no vetor denominado dados, índice 0
- some 1 em tamanho

L1:ListaDeInteiros			
dados	0	1	2
	15		
			tamanho: 1

ALOCAÇÃO ESTÁTICA

Dada a Lista

L1:ListaDeInteiros			
	0	1	2
dados	15		
tamanho: 1			

Adicione o elemento 30 no início da lista, processo:

- Lista está cheia?

- Não:

- passe, a partir do último elemento, todos os elementos uma posição à frente
- guarde 30 no vetor denominado dados, índice 0
- some 1 em tamanho

L1:ListaDeInteiros			
	0	1	2
dados	30	15	
tamanho:2			

ALOCAÇÃO ESTÁTICA

Dada a Lista

L1:ListaDeInteiros			
	0	1	2
dados	30	15	
			tamanho: 2

Adicione o elemento 53 no início da lista, processo:

- Lista está cheia?

- Não:

- passe, a partir do último elemento, todos os elementos uma posição à frente
- guarde 53 no vetor denominado dados, índice 0
- some 1 em tamanho

L1:ListaDeInteiros			
	0	1	2
dados	53	30	15
			tamanho:3

ALOCAÇÃO ESTÁTICA

Dada a Lista

L1:ListaDeInteiros				
	0	1	2	
dados	53	30	15	tamanho: 3

Adicione o elemento 47 no início da lista, processo:

- Lista está cheia?

- Sim:

- Mostre a mensagem "Lista cheia"

L1:ListaDeInteiros				
	0	1	2	
dados	53	30	15	tamanho:3

ALOCAÇÃO ESTÁTICA

Dada a Lista

L1:ListaDeInteiros				
	0	1	2	
dados	53	30	15	tamanho: 3

Remova o elemento do início da lista, processo:

- Lista está vazia?
- Não
 - guarde o elemento do início da lista em uma variável
 - passe cada elemento um índice à esquerda, a partir do índice 1
 - decremente um de tamanho

L1:ListaDeInteiros				
	0	1	2	
dados	30	15		tamanho:2

retorna 53

ALOCAÇÃO ESTÁTICA

L1:ListaDeInteiros			
dados	0	1	2
	30	15	
tamanho:2			

Os elementos da lista são os que estão a partir do índice 0 (zero) até o índice equivalente tamanho – 1, portanto: 30 e 15

ALOCAÇÃO ESTÁTICA

Dada a Lista

L1:ListaDeInteiros			
	0	1	2
dados	30	15	
tamanho: 2			

Remova o elemento do início da lista, processo:

- Lista está vazia?
- Não
 - guarde o elemento do início da lista em uma variável
 - passe cada elemento um índice à esquerda, a partir do índice 1
 - decremente um de tamanho

L1:ListaDeInteiros			
	0	1	2
dados	15		
tamanho: 1			

retorna 30

ALOCAÇÃO ESTÁTICA

Dada a Lista

L1:ListaDeInteiros			
dados	0	1	2
	15		
tamanho: 1			

Remova o elemento do início da lista, processo:

- Lista está vazia?
- Não
 - guarde o elemento do início da lista em uma variável
 - passe cada elemento um índice à esquerda, a partir do índice 1
 - decremente um de tamanho

L1:ListaDeInteiros			
dados	0	1	2
tamanho: 0			

retorna 15

ALOCAÇÃO ESTÁTICA

Dada a Lista

L1:ListaDeInteiros			
dados	0	1	2
			tamanho: 0

Remova o elemento do início da lista, processo:

- Lista está vazia?

- Sim

- mostre a mensagem “ERRO! Lista Vazia”

ALOCAÇÃO ESTÁTICA

Exemplo de Lista com RA e Nome de Alunos:

```
public class ListaDeAlunos {  
    private Aluno[] dados;  
    private int tamanho;  
  
    public ListaDeAlunos(int capacidade) {  
        dados = new Aluno[capacidade];  
        tamanho = 0;  
    }  
  
    public boolean vazia() {  
        return tamanho == 0;  
    }  
  
    public boolean cheia() {  
        return tamanho == dados.length;  
    }  
}
```

ALOCAÇÃO ESTÁTICA

```
public void adicionaFinal(Aluno aluno) {  
    if (!cheia()) {  
        dados[tamanho] = aluno;  
        tamanho++;  
    } else  
    {  
        System.out.println("Lista Cheia!");  
    }  
}
```

```
public Aluno removeFinal ( ) {  
    Aluno aluno=null;  
    if (!vazia()) {  
        tamanho--;  
        aluno = dados[tamanho];  
    }  
    return aluno;  
}
```

```
}
```

EXERCÍCIO

1) Considere a lista de nomes por alocação estática de memória:

0	1	2	3
"Ana"	"Rosa"	"Raul"	

tamanho=3

a) Qual será a situação da lista e do total de alunos da lista (ou mensagem impressa) se o método `adicionaFinal("Sandra")`, for invocado?

0	1	2	3
"Ana"	"Rosa"	"Raul"	

tamanho=_____

EXERCÍCIO

1) Continuação - preencha todas as posições:

b) Qual será a situação da lista e do total de alunos da lista (ou mensagem impressa) se o método `adicionaFinal("Tania")`, for invocado?

0 1 2 3

--	--	--	--

tamanho=_____

c) Qual será a situação da lista e do total de alunos da lista (ou mensagem impressa) se o método `removeFinal()`, for invocado?

0 1 2 3

--	--	--	--

tamanho=_____

EXERCÍCIO

2) Mostre detalhadamente o processo de cada método invocado a seguir para uma Lista com alocação estática de memória, com capacidade máxima 4, que inicia vazia. Adicione todas as mensagens geradas pelos métodos e informe todos os valores devolvidos:

- a) l1.adicionaInicio(74)
- b) l1.adicionaInicio(73)
- c) l1.adicionaInicio(72)
- d) l1.adicionaInicio(71)
- e) l1.adicionaInicio(70)
- f) l1.removeInicio()
- g) l1.removeInicio()
- h) l1.removeInicio()
- i) l1.removeInicio()
- j) l1.removeInicio()

REFERÊNCIAS BIBLIOGRÁFICAS

EDELWEISS, N; GALANTE, R. **Estruturas de Dados**. Livros Didáticos UFRGS, V.18. Bookman, 2009.

GUIMARÃES, A. de M.; LAGES, N. A. **Algoritmos e Estruturas de Dados**. Livros Técnicos e Científicos, 1994.

LAFORE, Robert. **Estrutura de Dados & Algoritmos em Java**. Rio de Janeiro: Ed. Ciência Moderna, 2004.

KOFFMANN, E. B. **Objetos, abstração, estrutura de dados e projeto**. LTC, 2008.

SHILDT, Hebert. **C Completo e Total**. McGraw Hil, 1991.

TENENBAUM, Aaron; LANGSAM, Yedidyah; AUGENSTEIN, Moshe J. **Estruturas de Dados Usando C**. 1a. Ed. São Paulo: Makron Books, 1995.