

Listas ordenadas

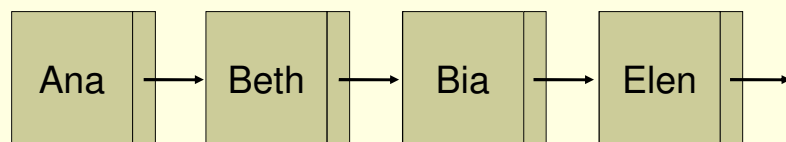
SCC-202 – Algoritmos e Estruturas de
Dados I

Prof. Thiago A. S. Pardo

Listas ordenadas

■ Definição

- *Listas em que os elementos estão ordenados por algum critério*
 - Em geral, por ordem alfabética



Listas ordenadas

- Situações
 - Cadastro de funcionários por ordem alfabética
 - Lista de passageiros em um voo
 - Lista de deputados presentes no congresso
 - Telefones da cidade

3

Listas ordenadas

- Nessas três **situações os acessos** (inserção, eliminação e consulta) são **direcionados a um elemento** específico, e não mais ao “primeiro” (ou último) a entrar na lista (fila ou pilha). Exemplos:
 - “Carla Peres” foi despedida (retire seu nome do cadastro)
 - “Edmundo” é funcionário? Verifique se seu nome consta do cadastro
 - Qual o salário do funcionário “Pedro Malan”?
 - “Sandra Bulloc” foi contratada; inclua-a no cadastro
- Nos exemplos acima, os registros (as “pastas”) de cada funcionário são (ordenados e) procurados pelo nome. O nome, neste caso, é a chave de busca, ou simplesmente **chave**.

4

Listas ordenadas

■ Operações

- cria(lista)
- isEmpty(lista)
- isFull(lista)
- esta_na_lista(lista,x)
- inserir(lista,x)
- remover(lista,x)
- imprimir_todos_da_lista(lista)
- Etc.

5

Listas ordenadas

■ Implementações

- Sequencial
- Encadeada

- Implicam mudanças na forma de manipulação da lista

6

Lista ordenada sequencial

- O que acontece se quero incluir na lista a funcionária “Alice do País das Maravilhas”?

...	...
Zorro	i+1
Zoroastro	i
Zico	i-1
...	...
Antônio	3
André	2
Ana	1

7

Lista ordenada sequencial

- O que acontece se quero incluir na lista a funcionária “Alice do País das Maravilhas”?

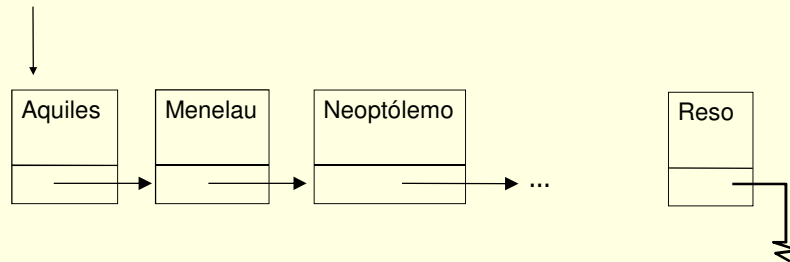
Tem que se deslocar todos os elementos da lista para inserir o novo elemento na posição correta

...	...
Zorro	i+1
Zoroastro	i
Zico	i-1
...	...
Antônio	3
André	2
Ana	1

8

Lista ordenada encadeada

- Lista de Gregos, Troianos e Tantáidas



- O que deve acontecer na lista quando
 - Nasce Filoctetes?
 - Nasce Agamêmnon?
 - Morre Menelau?

9

Lista ordenada encadeada

- Vantagem sobre a alocação sequencial
 - Os nós podem ser inseridos e eliminados na posição correta, sem realocação dos demais elementos
- **Complexidade de algoritmos**
 - O que implica?
 - A diferença é significativa?

10

Lista ordenada estática e encadeada

- **Declaração da estrutura**

```
#define TAM ...  
  
typedef struct bloco {  
    char nome[20];  
    int prox;  
} no  
  
typedef struct {  
    int ini, primeiro_vazio;  
    no v[TAM];  
} ListaOrd;
```

11

Lista ordenada dinâmica e encadeada

- **Declaração da estrutura**

```
typedef struct bloco {  
    char nome[20];  
    struct bloco *prox;  
} no  
  
typedef struct {  
    no *ini;  
} ListaOrd;
```

12

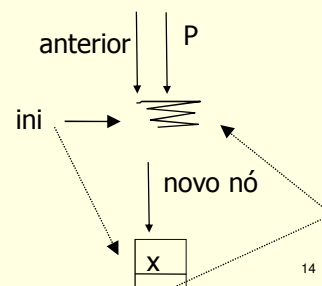
Lista ordenada dinâmica e encadeada

- Exercício: identificar os possíveis casos para a **inserção** (caso 1, caso 2, ...)
- Supondo que o elemento a ser inserido não está na lista

13

Inserção: caso 1

- `insere(L,x,erro)`
 - **Lista vazia**
 - se possível, aloca novo nó e `erro ← false`
 - `nó ← x`
 - `próximo de nó ← nada`
 - `ini ← nó`



14

Inserção: caso 2

■ `insere(L,x,erro)`

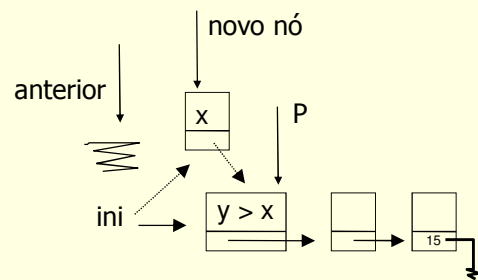
■ $x < 1^{\text{o}}$ da lista

se possível, aloca novo nó e `erro` \leftarrow false

`nó` \leftarrow x

`próximo de nó` \leftarrow P

`ini` \leftarrow nó



Inserção: caso 3

■ `insere(L,x,erro)`

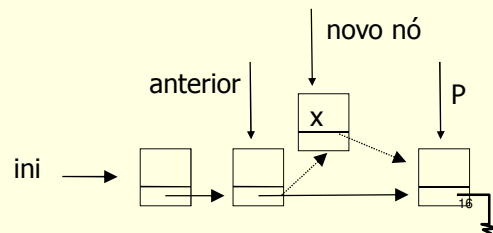
■ $x > 1^{\text{o}}$ da lista

se possível, aloca novo nó e `erro` \leftarrow false

`nó` \leftarrow x

`próximo de nó` \leftarrow P

`próximo de anterior` \leftarrow nó



Inserção

- Exercício: implementar a função de inserção

17

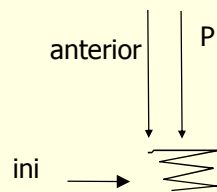
Lista ordenada dinâmica e encadeada

- Operação de **remoção** de um elemento X
 - 4 casos
 - Lista vazia
 - X não está na lista
 - X é menor do que o primeiro elemento da lista
 - X não está na lista
 - X é igual ao primeiro elemento da lista
 - X está na lista
 - X é maior do que o primeiro elemento da lista
 - X pode estar na lista

18

Remoção: caso 1

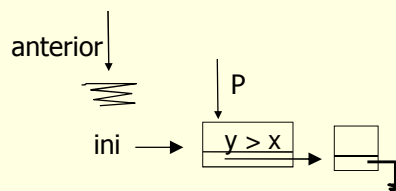
- `remove(L,x,erro)`
 - a lista é vazia
- erro \leftarrow true



19

Remoção: caso 2

- `remove(L,x,erro)`
 - $x < 1^{\circ}$ da lista
- erro \leftarrow true



20

Remoção: caso 3

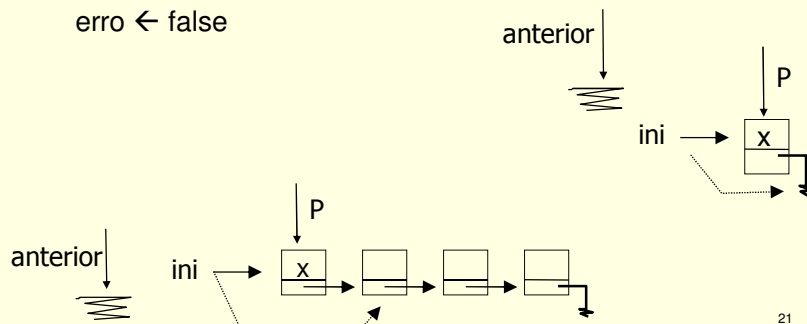
■ remove(L,x,erro)

■ x = 1º da lista

ini aponta para o próximo

libera nó indicado por p

erro ← false



21

Remoção: caso 4

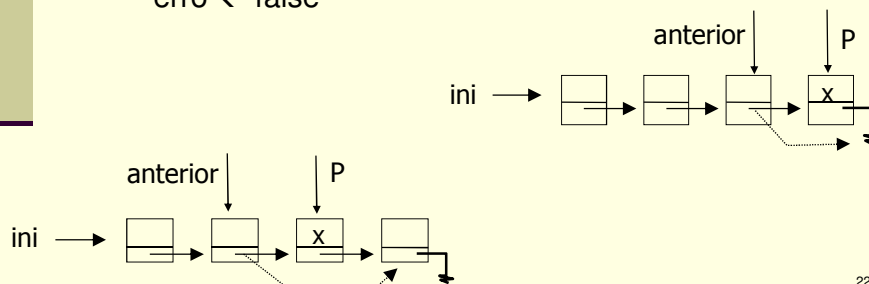
■ remove(L,x,erro)

■ x > 1º da lista e é encontrado

próximo do anterior ← próximo de P

libera nó indicado por P

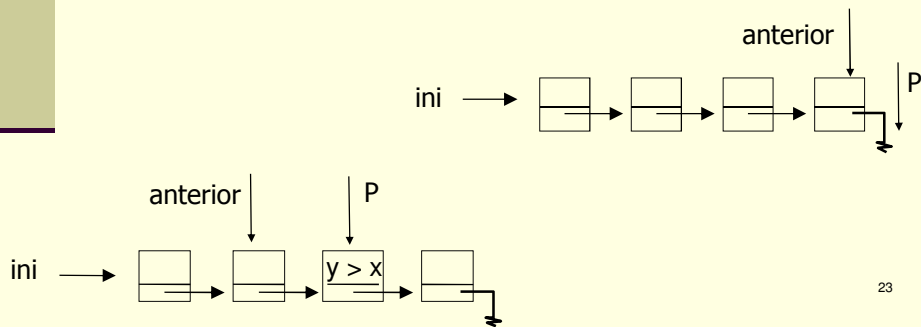
erro ← false



22

Remoção: caso 4

- `remove(L,x,erro)`
 - `x > 1º da lista e não é encontrado`
 - erro \leftarrow true



23

Remoção

- Exercício: implementar a função de remoção

24

Questão

- Como poderia ser implementada uma **lista ordenada com elementos repetidos**? O que mudaria?

25

Questão

- E se a **lista ordenada fosse circular**? O que mudaria?

26

Exercício: produzindo uma lista ordenada

- Implemente uma **função que ordene** uma lista de nomes não ordenados, trocando os nomes entre os blocos

27

Exercício: produzindo uma lista ordenada

- Implemente uma **função que ordene** uma lista de nomes não ordenados, trocando somente os ponteiros dos blocos
 - Dica: desenhe/esboce as situações possíveis

28

Exercício para entregar: união de listas ordenadas

- Situação
 - Você recebeu duas listas dinâmicas e encadeadas ordenadas, de tamanhos possivelmente diferentes
 - Você não pode se dar ao luxo de contar com mais memória do que a já usada pelas listas, pois a aplicação rodando é crítica

- Implemente uma função em C que faça a união das duas listas, mantendo a ordenação

29

Exercício extra para casa

- Implemente as operações de inserção e remoção sobre uma lista ordenada estática e encadeada

30