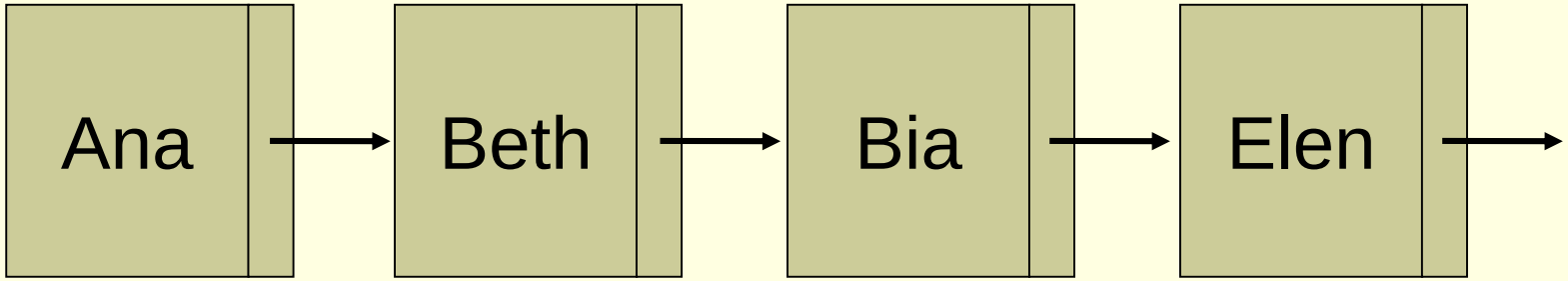
Listas ordenadas

SCC-502 – Algoritmos e Estruturas de Dados I

Prof. Bruno Matheus

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 vôo
  - Lista de deputados presentes no congresso
  - Telefones da cidade

Listas ordenadas

- Nessas 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:
  - “Faustão” foi despedido (retire seu nome do cadastro)
  - “Edmundo” é funcionário? Verifique se seu nome consta no 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**.

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.

Listas ordenadas

- Implementações
  - Sequencial
  - Encadeada
    - Implicam mudanças na forma de manipulação da lista

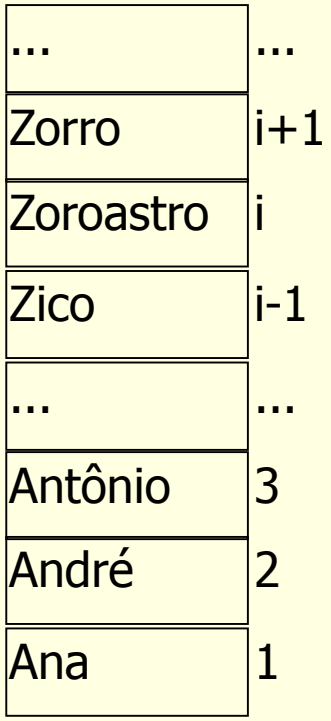
Lista ordenada **sequencial**

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

Lista ordenada **sequencial**

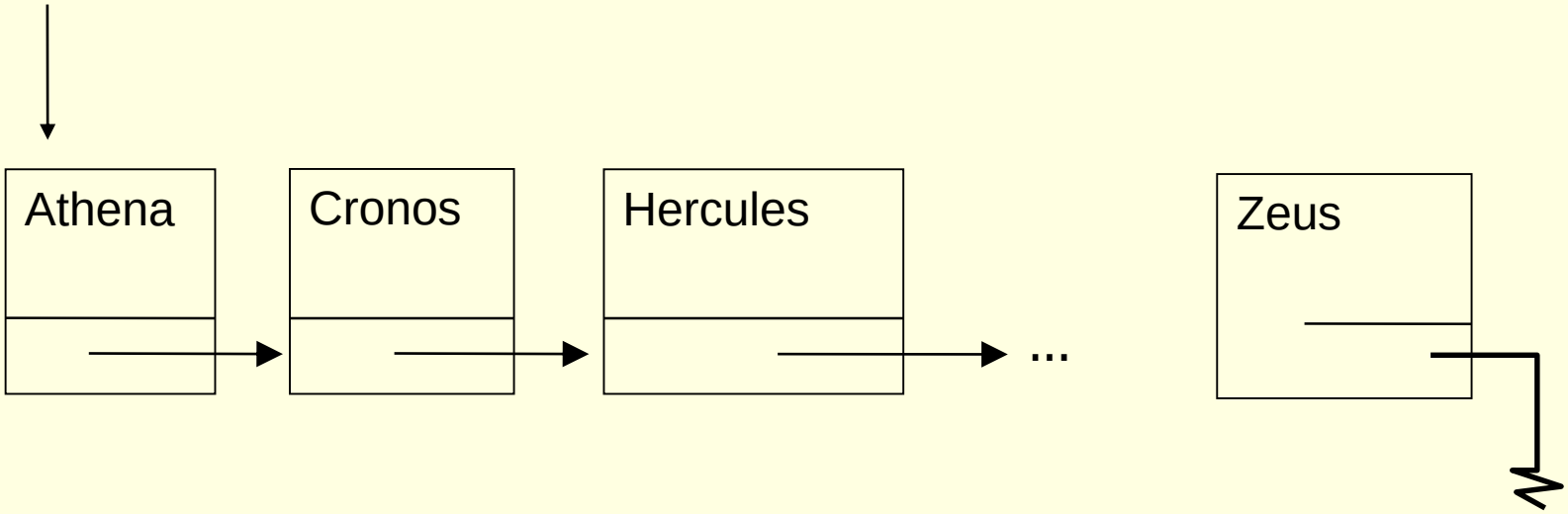
- O que acontece se quero incluir na lista a funcionária “Alice do País das Maravilhas”?

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



Lista ordenada **encadeada**

- Lista de Gregos



- O que deve acontecer na lista quando
  - Nasce Hades?
  - Nasce Aquiles?
  - Morre Cronos?

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?

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;
```

Lista ordenada dinâmica e encadeada

- **Declaração da estrutura**

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

typedef struct {
    no *ini;
} ListaOrd;
```

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

Inserção: caso 1

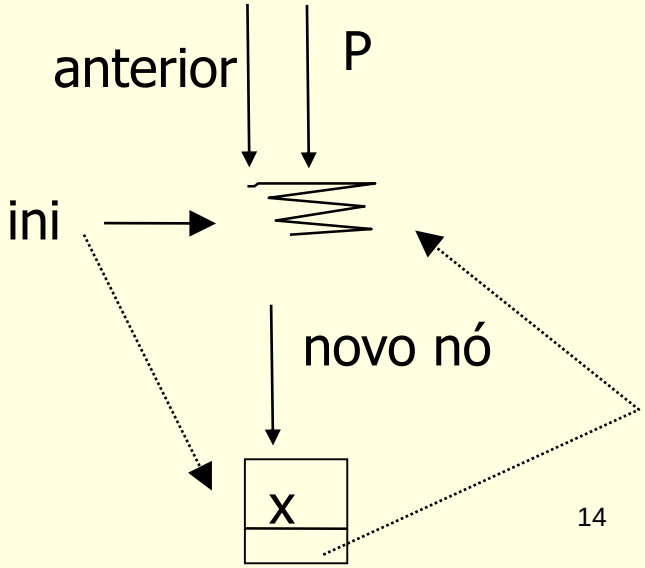
- insere(L,x,erro)
  - Lista vazia

    se possível, aloca novo nó e erro ← false

    nó ← x

    próximo de nó ← nada

    ini ← nó



Inserção: caso 2

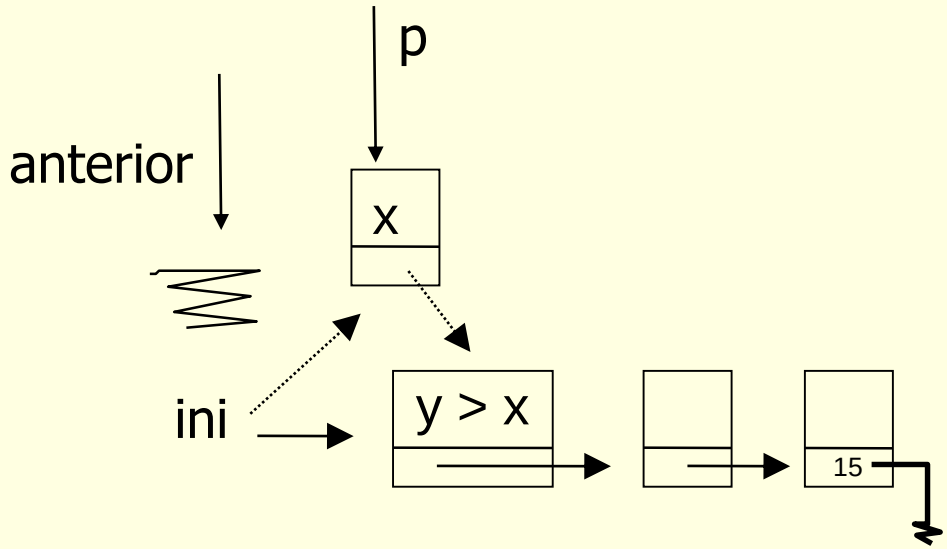
- insere(L,x,erro)
  - x < 1º da lista

    se possível, aloca novo nó e erro ← false

    nó ← x

    próximo de nó ← ini

    ini ← p



Inserção: caso 3

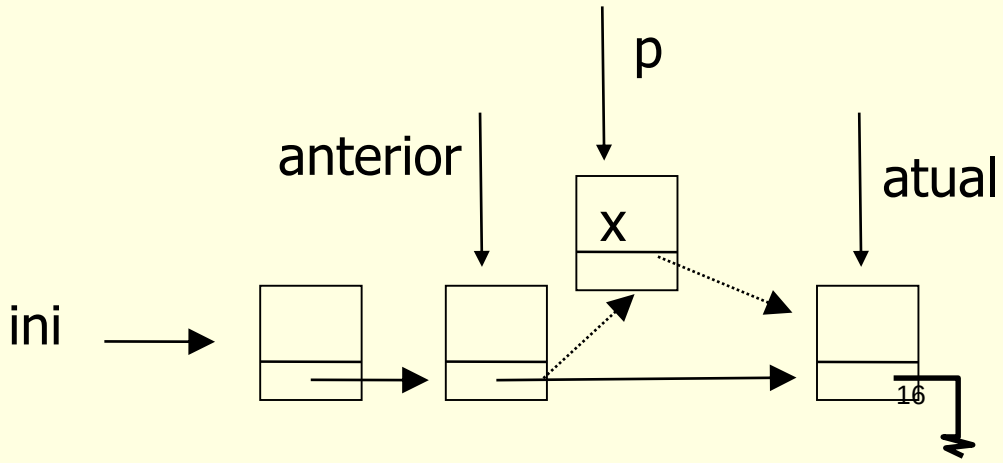
- insere(L,x,erro)
  - x > 1º da lista

    se possível, aloca novo nó e erro ← false

    p ← x

    próximo de nó ← atual

    próximo de anterior ← p



Inserção

- Exercício: implementar a função de inserção

Questão

- O que é melhor: inserir em uma lista desordenada ou em uma lista ordenada?

Remoção

- Quais seriam os casos?

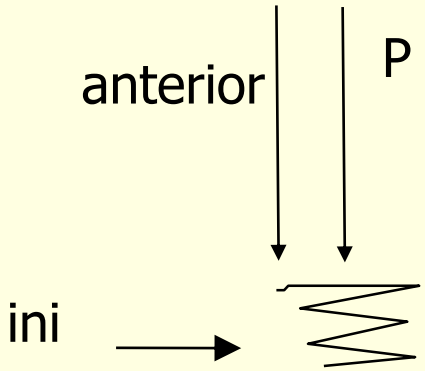
Lista ordenada dinâmica e encadeada

- 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

Remoção: caso 1

- remove(L,x,erro)
  - a lista é vazia

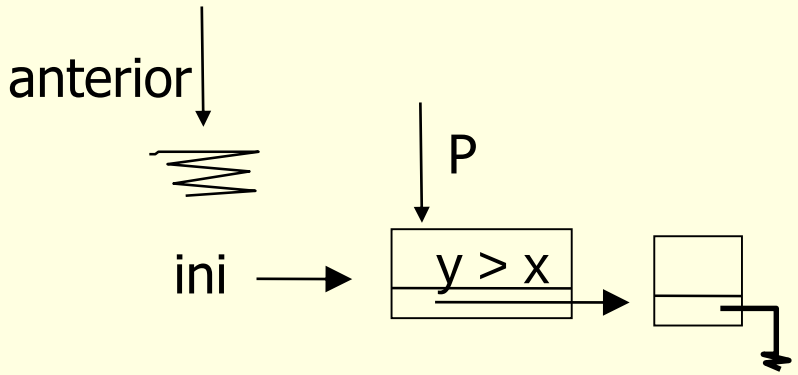
    erro ← true



Remoção: caso 2

- remove(L,x,erro)
  - x < 1º da lista

    erro ← true



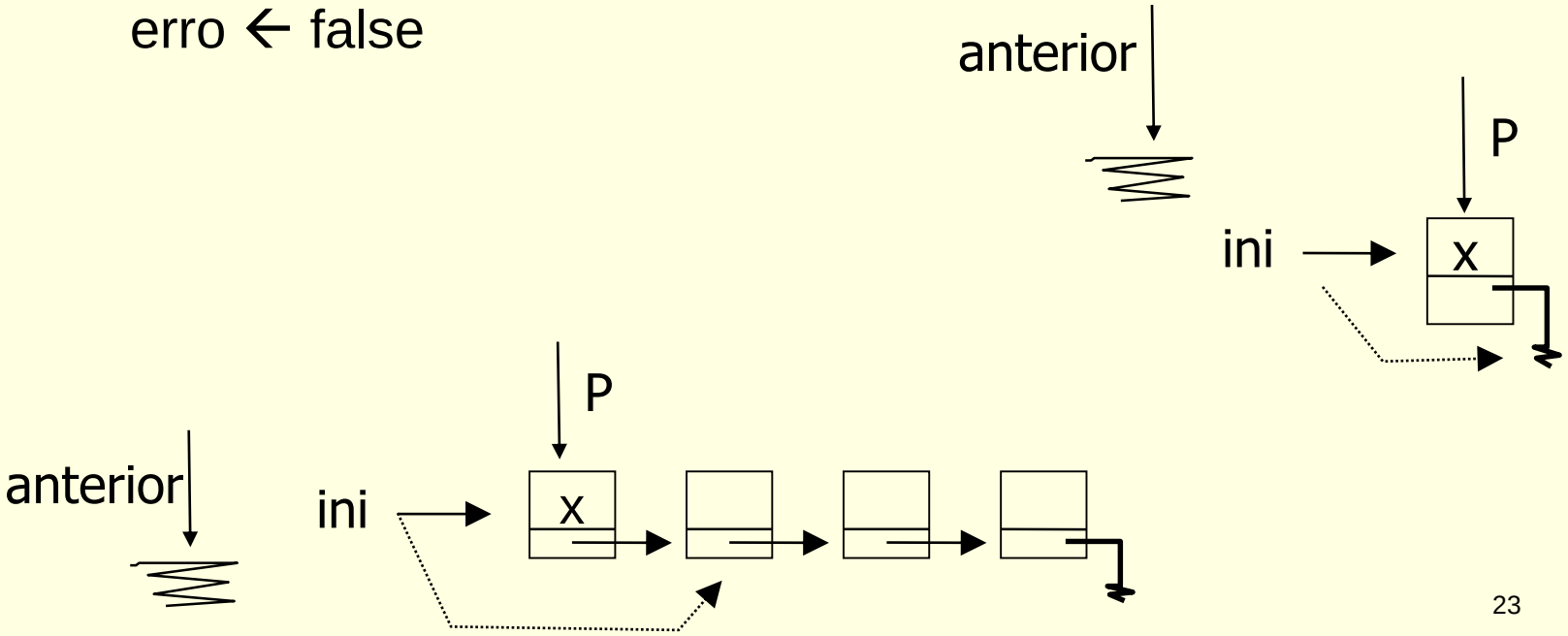
Remoção: caso 3

- remove(L,x,erro)
  - x = 1º da lista

    ini aponta para o próximo

    libera nó indicado por p

    erro ← false



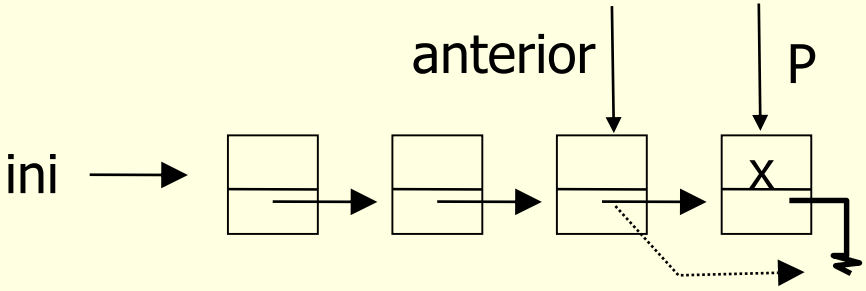
Remoção: caso 4

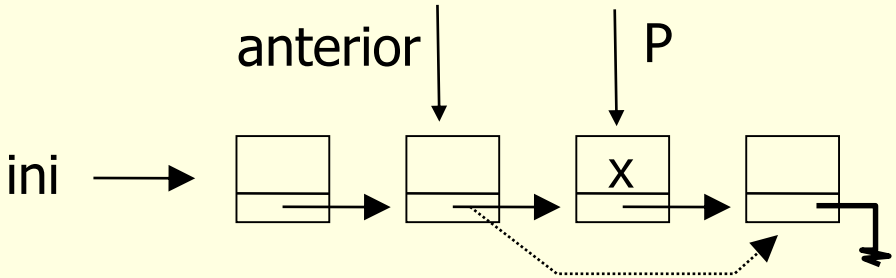
- remove(L,x,erro)
  - x > 1º da lista e é encontrado

    próximo do anterior ← próximo de P

    libera nó indicado por P

    erro ← false

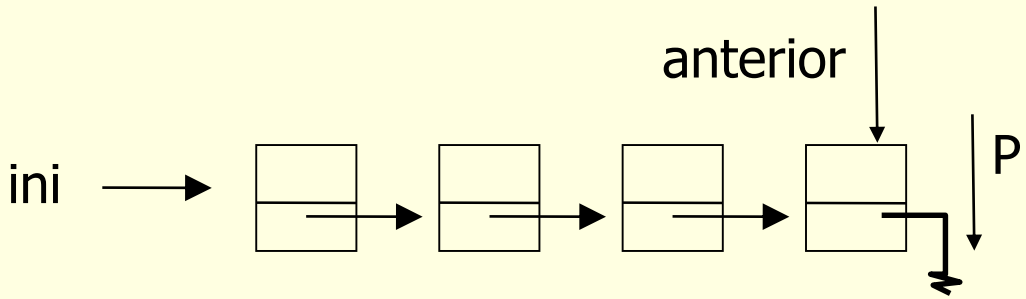


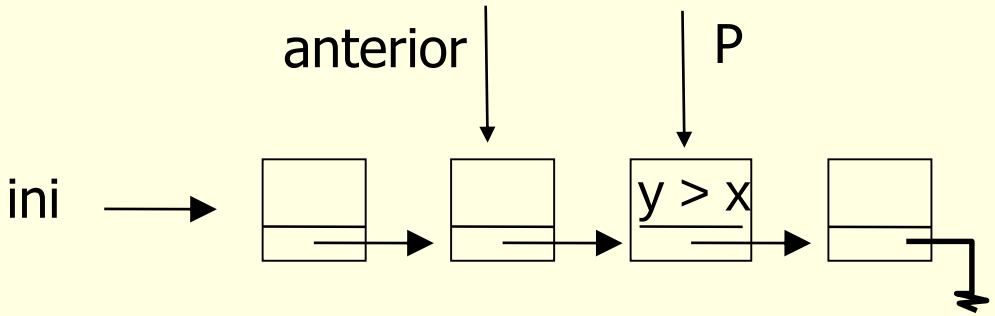


Remoção: caso 4

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

    erro ← true





Remoção

- Exercício: implementar a função de remoção

Remoção

- Melhor ou pior do que na lista desordenada?

Busca

- Exercício: implementar a função de busca por um elemento

Busca

- Melhor ou pior do que na lista desordenada?

Moral da história

- Lista ordenada vs. não ordenada

Operação	Lista ordenada	Lista não ordenada
Inserção	☹	☺
Remoção	☺	☹
Busca	☺	☹

Questão

- E se a lista foi construída de forma não ordenada? O que fazer?

Exercício para casa

Implemente uma função que ordene uma lista (dinâmica e encadeada) de nomes não ordenados, trocando os nomes entre os blocos

Exercício para entregar

Implemente uma função que ordene uma lista (dinâmica e encadeada) de números não ordenados, trocando somente os ponteiros dos blocos

- Dica: desenhe/esboce as situações possíveis

Questões para pensar

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

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

Exercício extra para casa

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