Estruturas de Dados:

Listas Encadeadas Simples remoção

Helena Graziottin Ribeiro
hgrib@ucs.br

UCS
UNIVERSIDADE
DE CAXIAS DO SUL

Listas encadeadas simples

Operações em listas:

- **remoção:**
  - **do “meio”**
  - **do último**
  - **do primeiro**
  - **do único**
- **em C, não esquecer de utilizar a função free( ) para desalocar a área de memória**

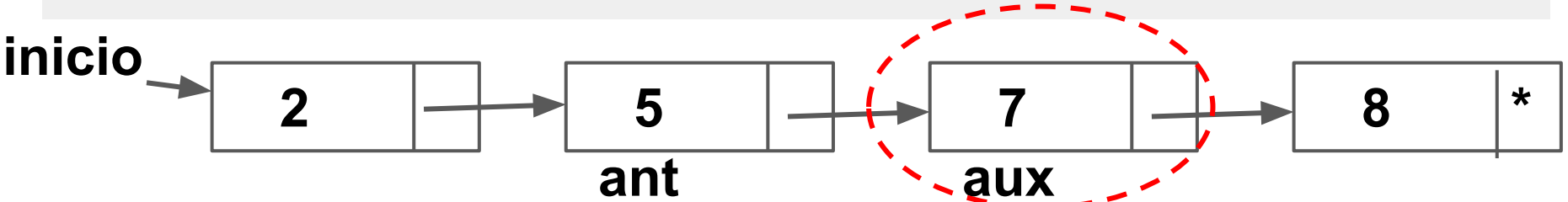
Listas encadeadas: remoção (no meio)

- Deve haver algum critério para caracterizar a remoção no meio, por exemplo:
  - por valor (pode haver elementos repetidos, ou não)
  - remoção de posição determinada

Exemplo: remover o nodo com valor 7

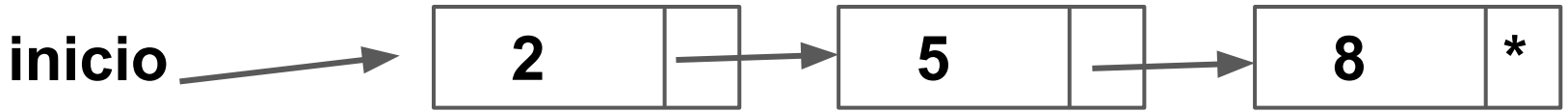
Listas encadeadas: remoção (no meio)

```
Elemento *aux=inicio, *ant=inicio;
int valor = 7;
while (aux != NULL && aux->info != valor ){
      ant = aux;
      aux = aux->prox;}
if (aux!=NULL){
   ant->prox = aux->prox;
   free(aux);}
```



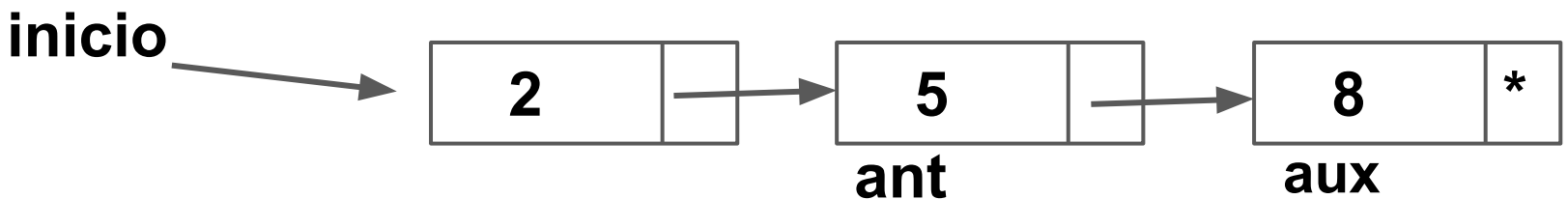
Listas encadeadas: inserção (no meio)

```
Elemento *aux=inicio, *ant=inicio;
int valor = 7;
while (aux != NULL && aux->info != valor ){
      ant = aux;
      aux = aux->prox;}
if (aux!=NULL){
   ant->prox = aux->prox;
   free(aux);}
```



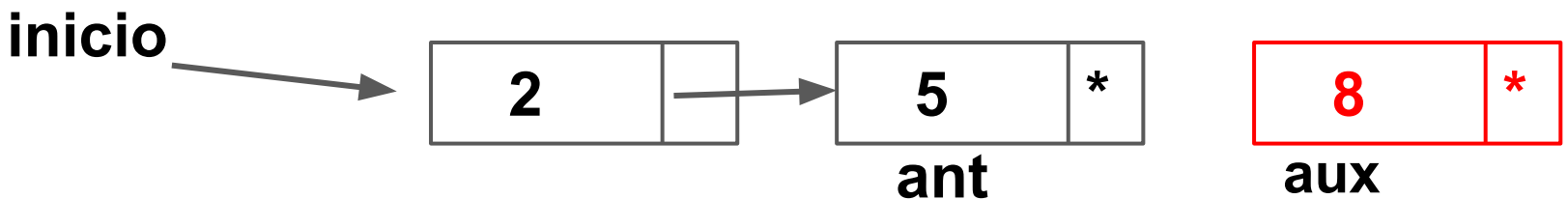
Listas encadeadas: remoção (último)

```
Elemento *aux=inicio, *ant=inicio;
while (aux->prox != NULL){
      ant = aux;
      aux = aux->prox;}
ant->prox = NULL;
free(aux);}
```



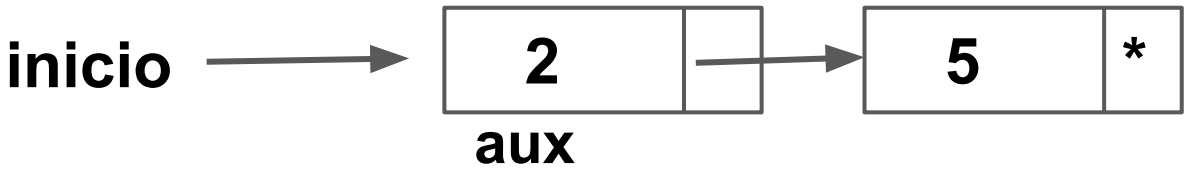
Listas encadeadas: remoção (último)

```
Elemento *aux=inicio, *ant=inicio;
while (aux->prox != NULL){
      ant = aux;
      aux = aux->prox;}
ant->prox = NULL;
free(aux);}
```



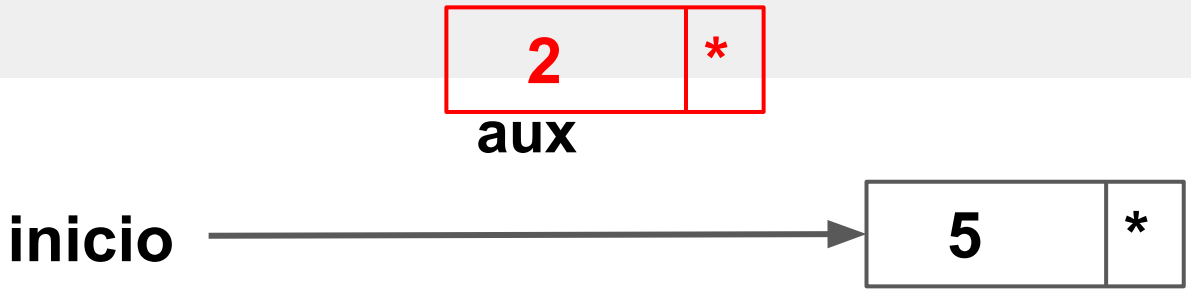
Listas encadeadas: remoção (primeiro)

```
Elemento *aux=inicio;
inicio = inicio->prox;}
free(aux);}
```



Listas encadeadas: remoção (primeiro)

```
Elemento *aux=inicio;
inicio = inicio->prox;
free(aux);}
```



Listas encadeadas: remoção (1º e único)

```
free(inicio);
inicio = NULL;
```