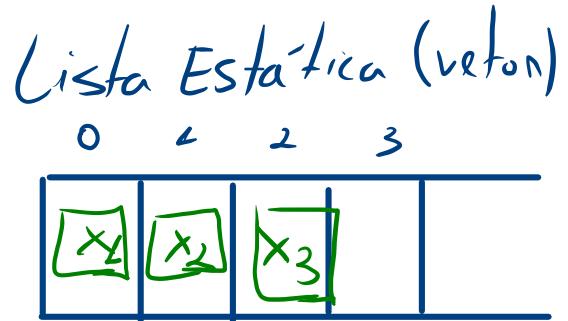


Listas Encadeadas



Listas Encadeadas

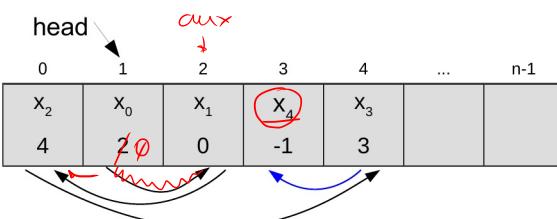
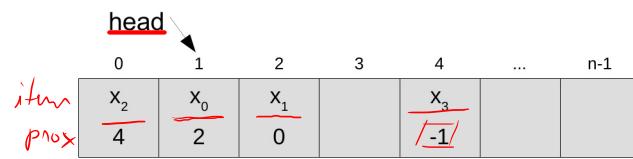
- Elementos consecutivos na lista não implica em elementos consecutivos na representação. A ordem é lógica.
- Cada item da lista contém a informação que é necessária para alcançar o próximo item
- Na implementação é necessário armazenar separadamente a informação do primeiro elemento da lista



Listas Encadeadas

- É possível inserir e retirar elementos sem necessidade de deslocar os itens seguintes da lista
- Existem duas formas de representar listas encadeadas, através de:
 - arrays, denominada lista encadeada estática
 - por referências (ponteiros), denominada lista encadeada dinâmica

Listas Encadeadas Estáticas



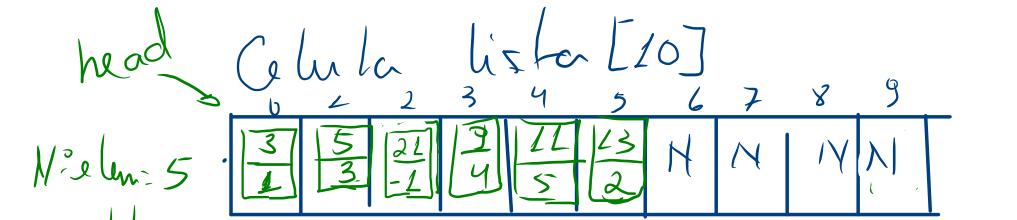
ultimo = x₁

head.prox = aux.prox;
aux=null;

4

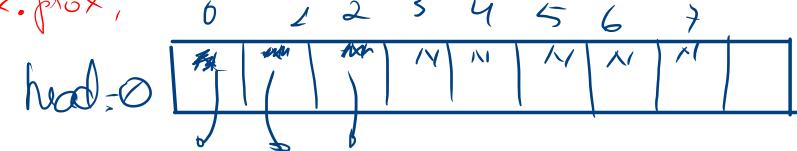
```
struct Celula {
    Elemento e;
    int prox;
};
```

linguagem
int prox;
C
Prox armaz pos.vetor



ultimo = x₁

ultimo = 6



Listas Encadeadas

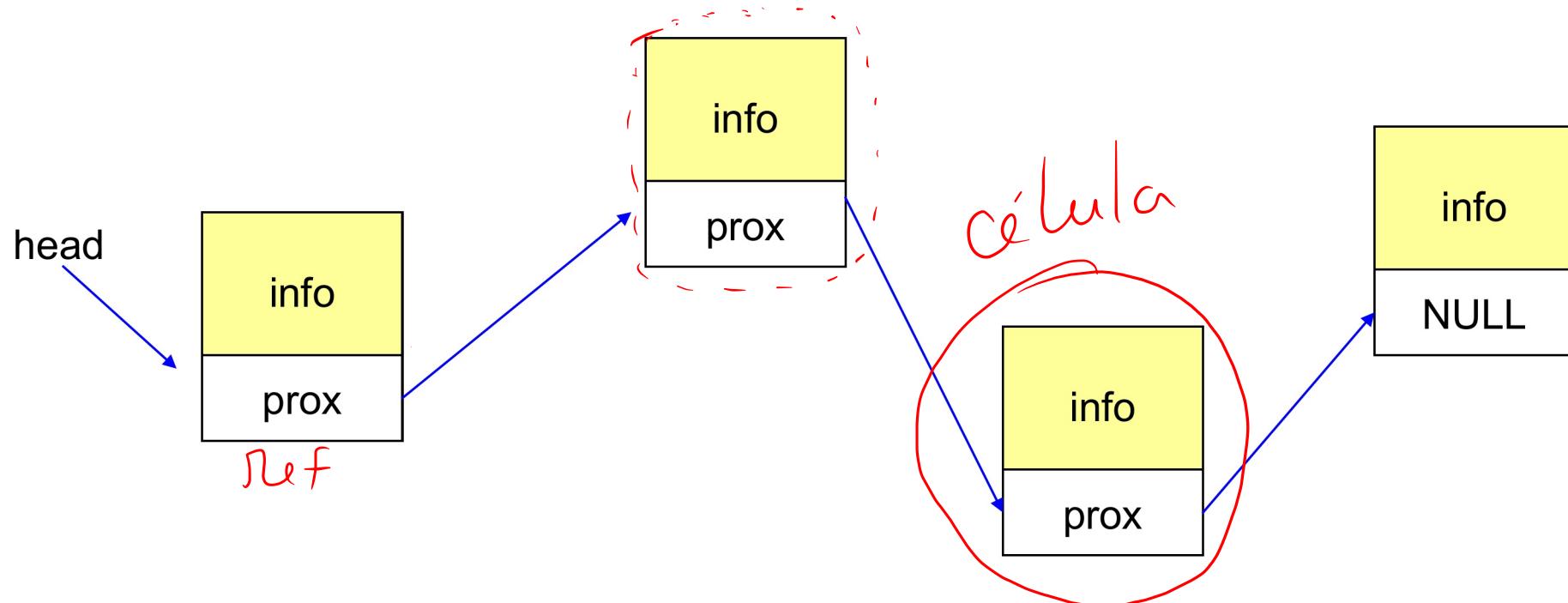
- Vantagens
 - Elimina o problema dos deslocamentos de células
 - Não é necessário saber previamente o número de elementos a serem armazenados (lista dinâmica)
- Desvantagens
 - Não se consegue acessar os elementos da lista em tempo constante
 - Mais operações para manter integridade dos dados

Listas Encadeadas Dinâmicas

- Cada item da lista é encadeado com o seguinte através de uma variável de referência (ponteiro)
- Esta implementação permite utilizar posições não contíguas de memória, sendo possível inserir e retirar elementos sem deslocar os itens da lista
- A lista é constituída de células, onde cada célula contém um item da lista e um apontador para a célula seguinte
Referência
- É conveniente utilizar listas encadeadas dinâmicas em aplicações em que não existe previsão sobre o crescimento da lista, por não ser necessário definir o tamanho da lista a priori

Listas Encadeadas Dinâmicas

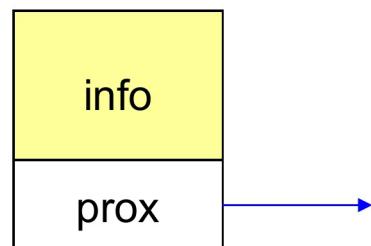
- Características:
 - Células não estão contíguas na memória



Desvantagem: utilização de memória extra para as referências

Sobre as Célula da Lista

- Célula: guarda as informações sobre cada elemento.
- Para isso define-se cada célula como uma estrutura que possui:
 - Campos de informações (*dado que será armazenado*)
 - Ponteiro (referência) para a próxima célula



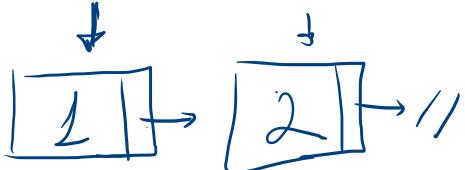
- Lister Estática (ArrayList)

↳ lista dentro do vetor

- lista Encadeada Estática

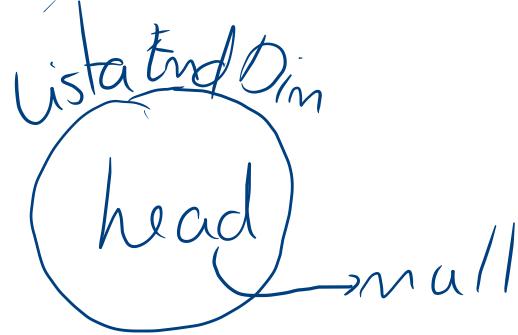
- lista Encadeada Dinâmica (Simplesmente) (LinkedList)

head

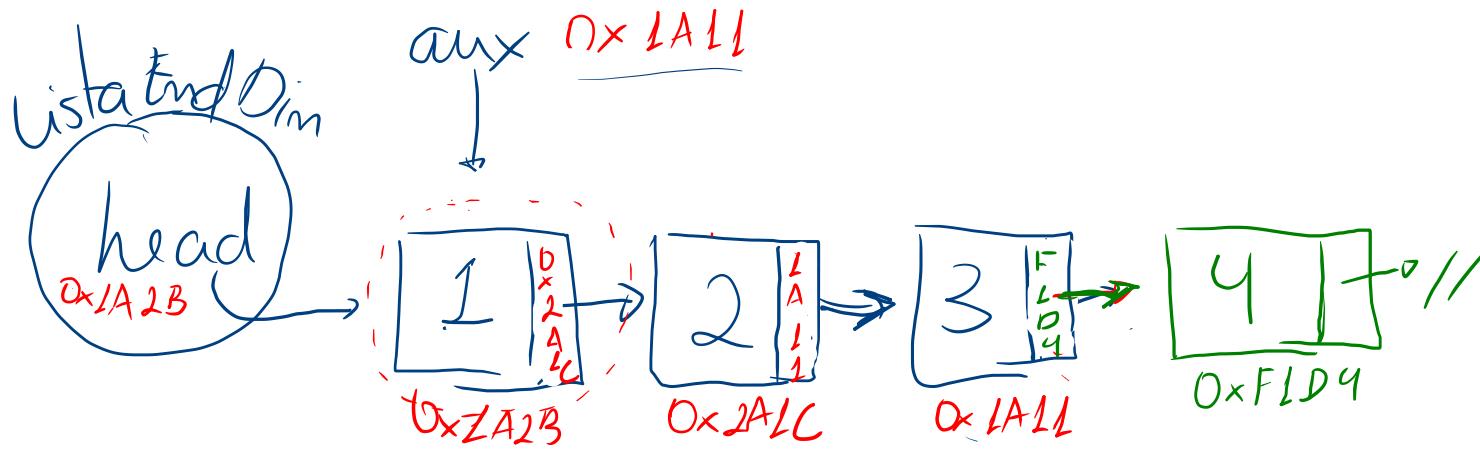
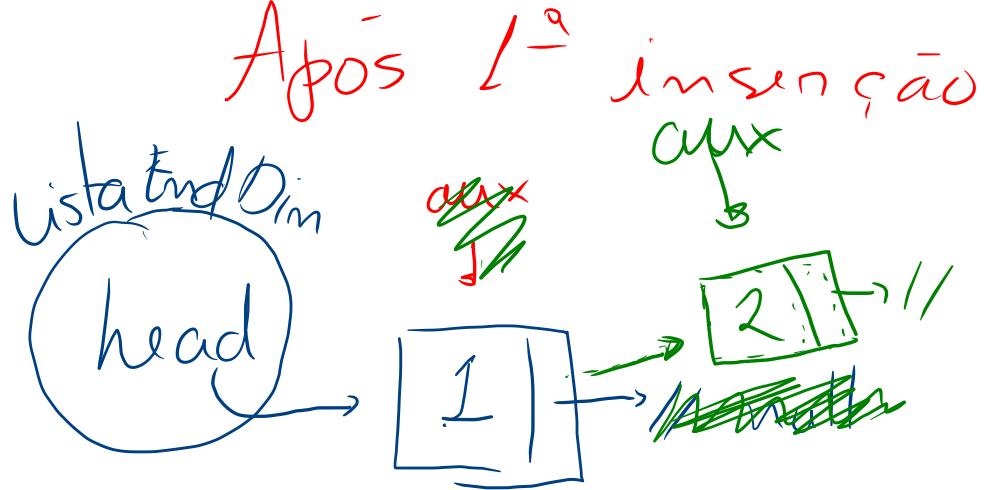


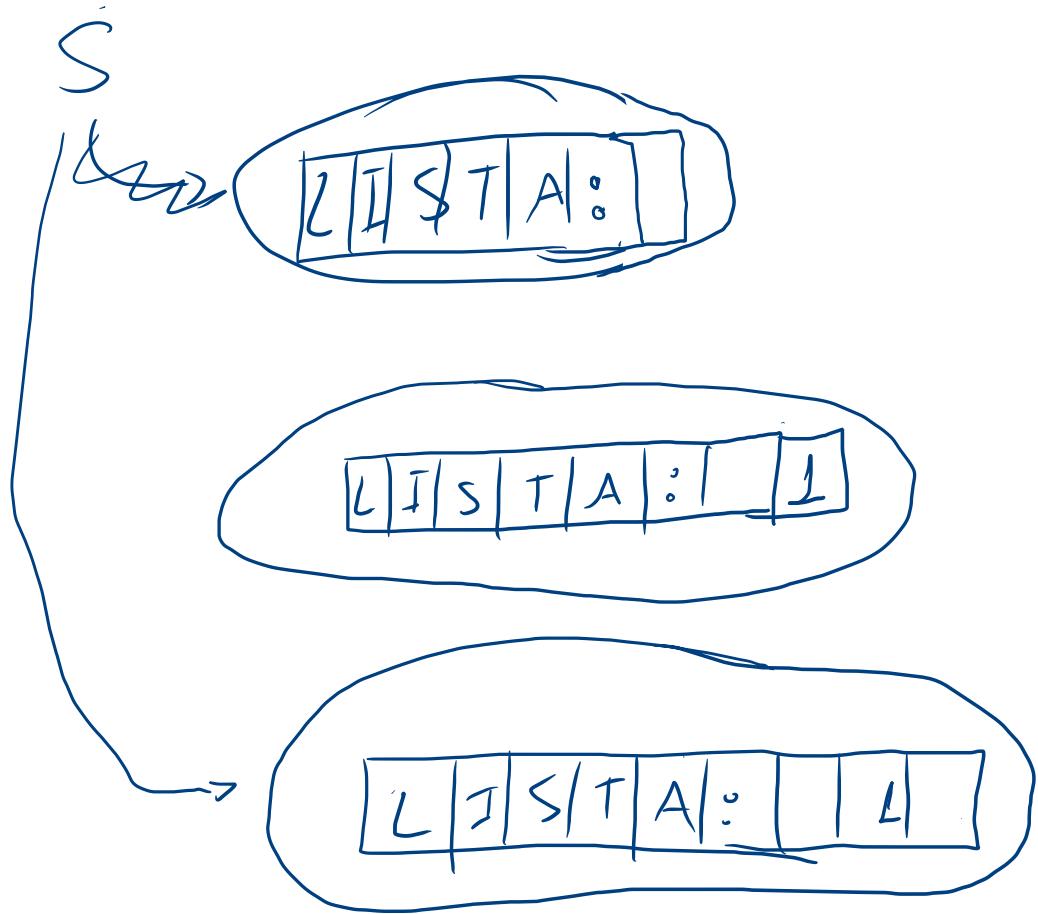
0	1	2	3
$\frac{x_1}{2}$	$\frac{x_3}{3}$	$\frac{x_2}{1}$	$\frac{x_4}{-1}$





Vazio

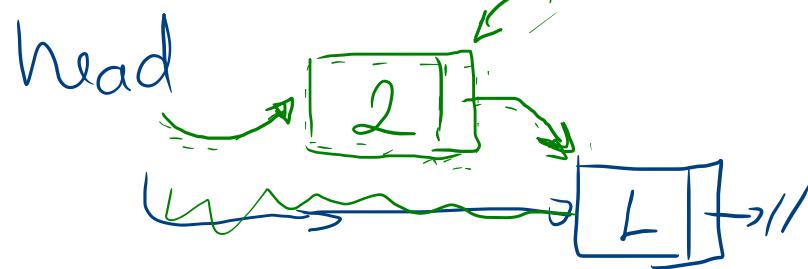




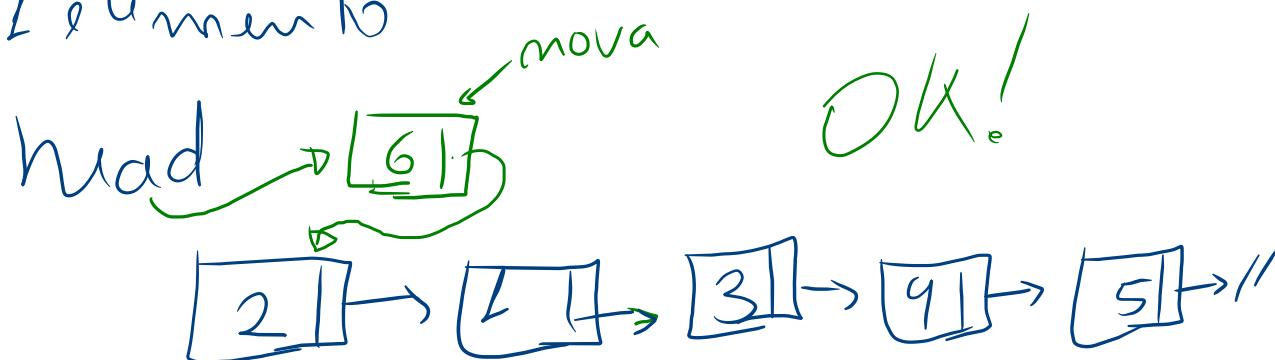
• Vazia



• L elementos



• + L elementos

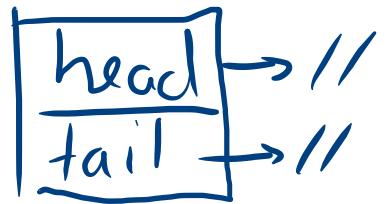


Lista Enc. Dinâmica com tail (nabo)

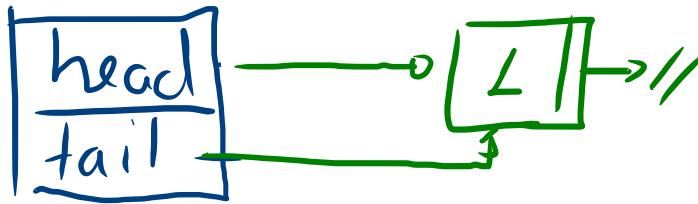
- Inserção no Final em $O(1)$

X

- Vazia

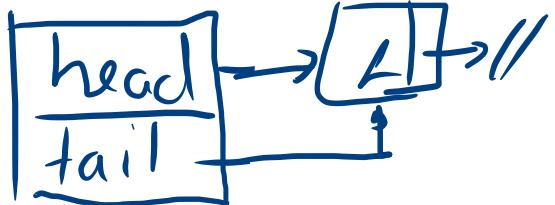


1^{a} inserção
⇒

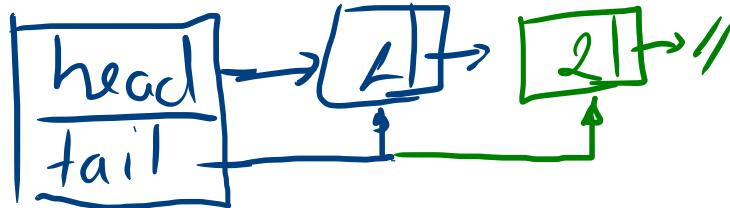


ADD

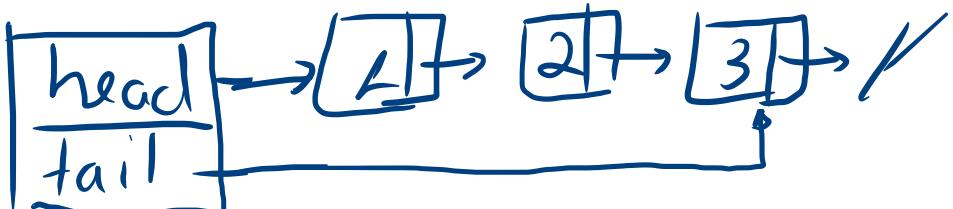
- 1 elemento



⇒

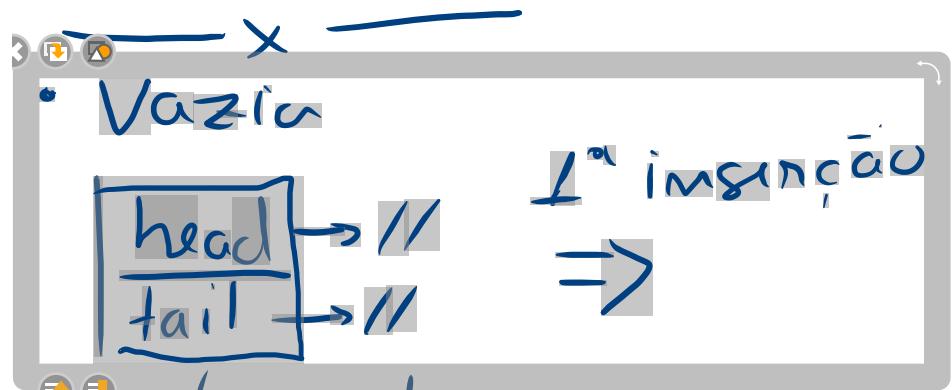


- +1 elemento

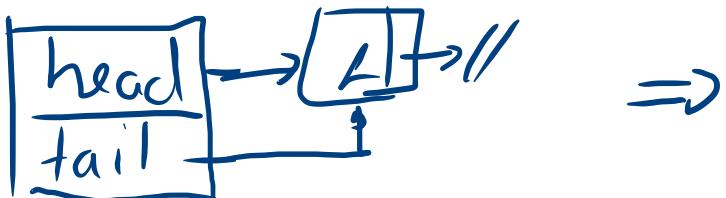


Lista Enc. Dinâmica com tail (rabo)

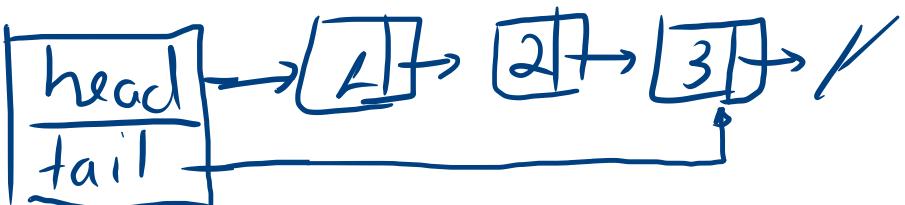
- Inscrição no Final em $O(1)$



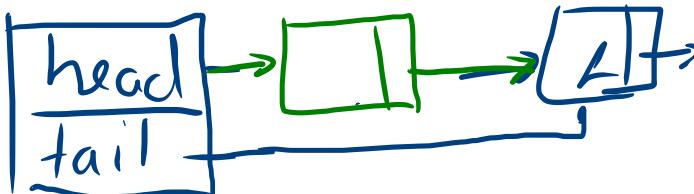
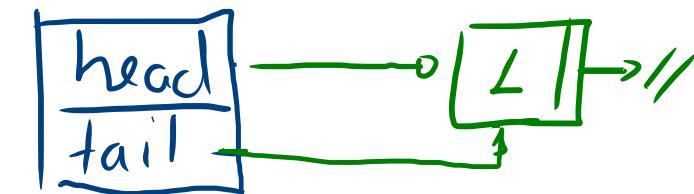
- 1º elemento



- +1 elemento



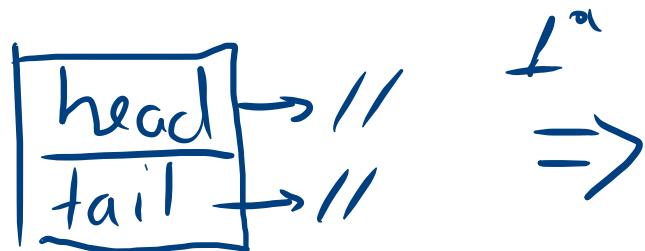
INSERT



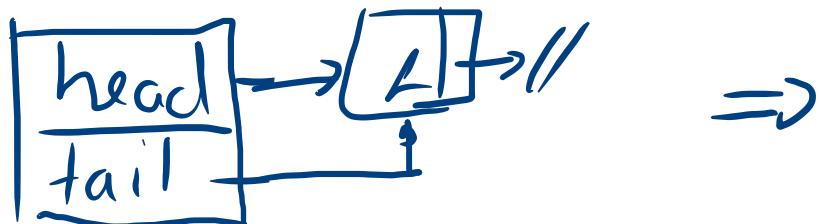
Lista Enc. Dinâmica com tail (rabo)

- Exclusão

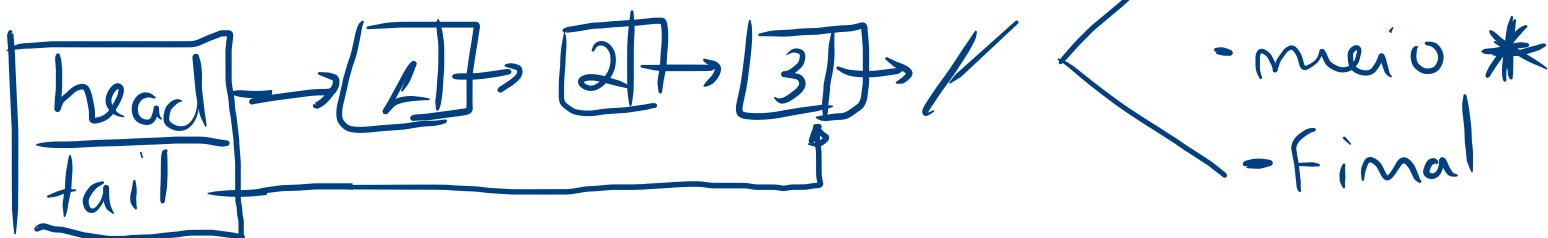
- Vazia



- 1 elemento



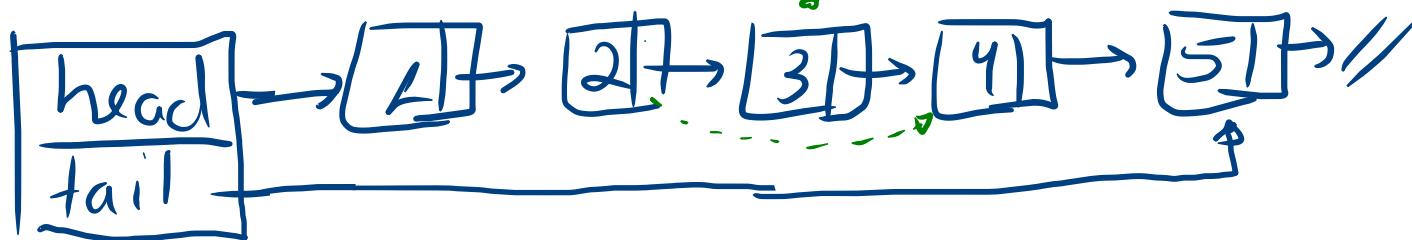
- +1 elemento



Lista Enc. Dinâmica com tail (nabo)

- Exclusão

- +1 elemento ^{ant aux}



- Vazia

