Aula 16: Manipulação de listas duplamente encadeadas

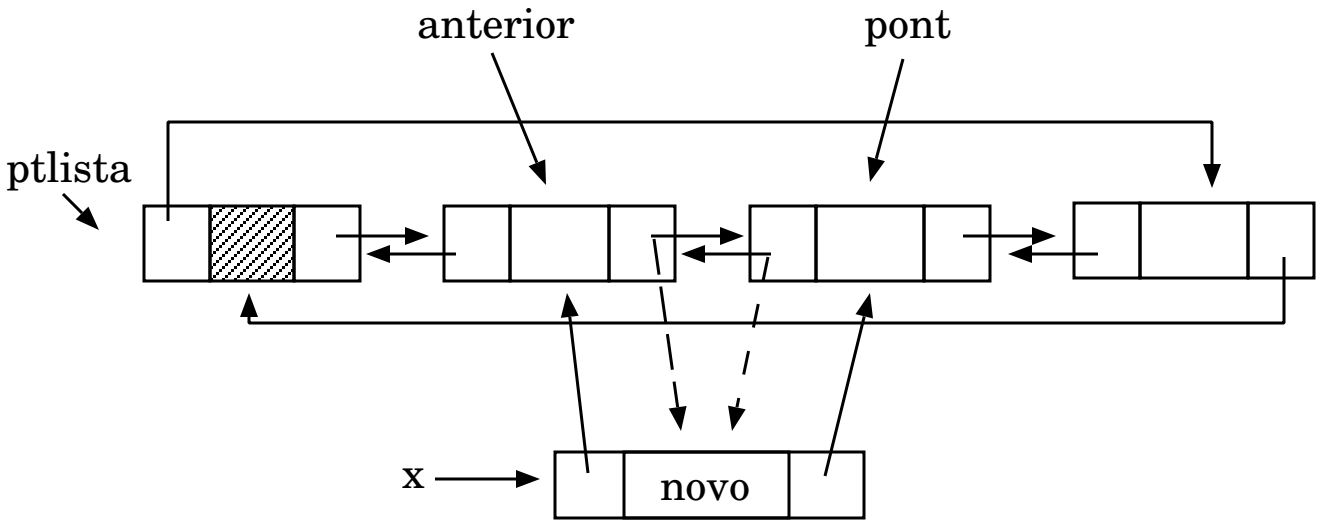
- Algoritmo de inserção
- Algoritmo de remoção
- Complexidade dos algoritmos

Inserção de um nó em uma lista duplamente encadeada

- A inserção em uma lista duplamente encadeada é feita da mesma forma que a inserção em lista simplesmente encadeada, sendo realizada em três fases:
  - Solicitação à LED de um novo nó
  - Inicialização do nó
  - Inserção do nó na lista, com o acerto dos ponteiros na estrutura
- Porém, a fase 3 requer mais cuidados, pois há mais ponteiros a serem acertados

Inserção de um nó em lista duplamente encadeada

- O novo nó é inserido entre os nós apontados por anterior e pont



Descrição do algoritmo de inserção de um nó em lista duplamente encadeada

Algoritmo: Inserção de um nó em lista duplamente encadeada entre os nós anterior e pont

```
pont := busca-dup( x )
se pont = ptlista ou pont↑.chave ≠ x
    então
        anterior := pont↑.ant
        ocupar( pt )                    % solicitar nó
        pt↑.info  := novo_valor         % inicializar nó
        pt↑.chave := x
        pt↑.ant   := anterior
        pt↑.post  := pont
        anterior↑.post := pt            % acertar lista
        pont↑.ant := pt
    senão
        "elemento já se encontra na lista"
```

Complexidade do algoritmo de inserção em lista duplamente encadeada

- A complexidade da inserção depende da complexidade da busca, já que as três fases da inserção podem ser executadas em tempo constante

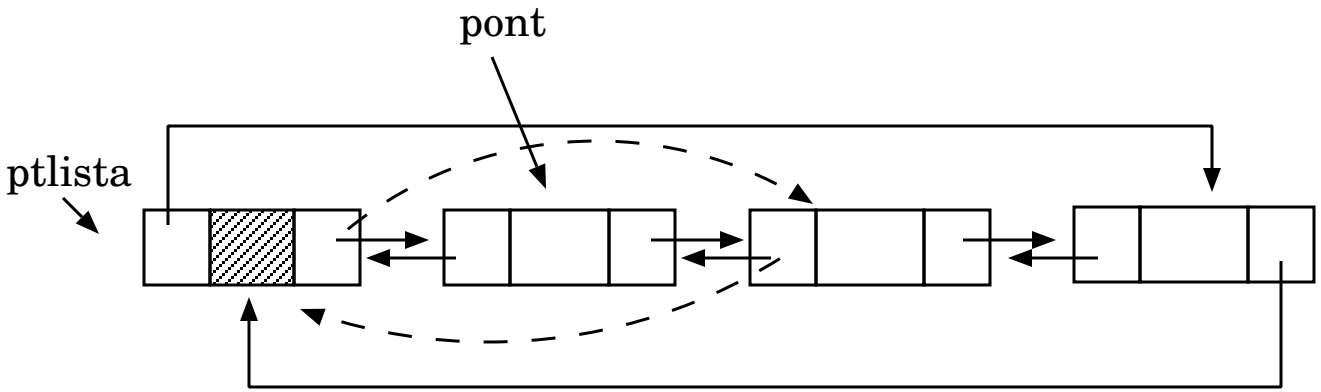
- Portanto, a complexidade é O(n), onde n é o número de nós da lista

Remoção de um nó de uma lista duplamente encadeada

- Como na inserção, a remoção é realizada em três fases:
  - Remoção do nó da lista, com o acerto dos ponteiros na estrutura
  - Utilização da informação contida no nó
  - Devolução do nó removido à LED

Remoção de um nó de uma lista duplamente encadeada

- Remove-se o nó apontado por pont, como na figura abaixo



Descrição do algoritmo de remoção de um nó de uma lista duplamente encadeada

Algoritmo: Remoção do nó apontado por pont de uma lista duplamente encadeada

```
pont := busca-dup( x )
se pont ≠ ptlista e pont↑.chave = x
    então
        anterior  := pont↑.ant
        posterior := pont↑.post
        anterior↑.post := posterior      % acertar lista
        posterior↑.ant := anterior
        valor_recuperado := pont↑.info   % utilizar nó
        desocupar( pont )                % devolver nó
    senão
        "elemento não se encontra na lista"
```

Complexidade da remoção de uma lista duplamente encadeada

- Como no caso da inserção, a complexidade da remoção depende da complexidade da busca, já que as três fases da remoção podem ser executadas em tempo constante

- Portanto, a complexidade é O(n), onde n é o número de nós da lista

Exercício final

Descreva um algoritmo que imprima os elementos de uma lista duplamente encadeada com n elementos da seguinte forma:

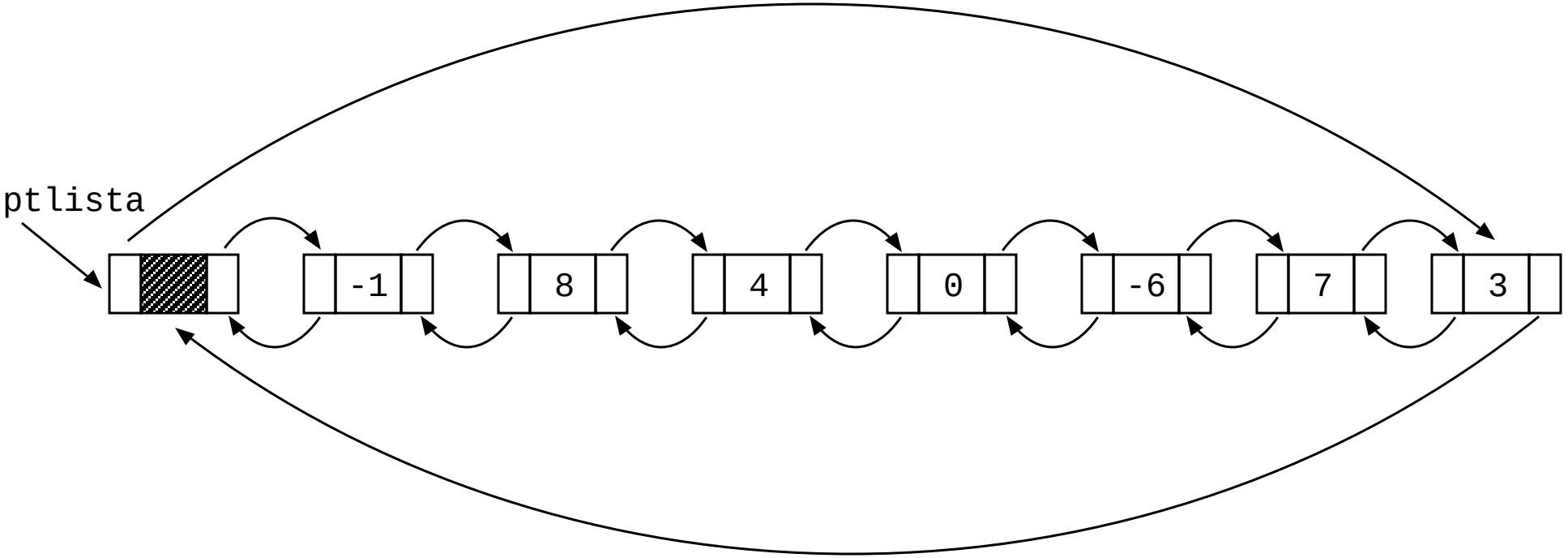
- Imprimir primeiro elemento
- Imprimir n-ésimo elemento
- Imprimir segundo elemento
- Imprimir (n-1)-ésimo elemento

...

- E assim sucessivamente

Exercício final

- Exemplo: Considere a lista



- Impressão: -1, 3, 8, 7, 4, -6, 0