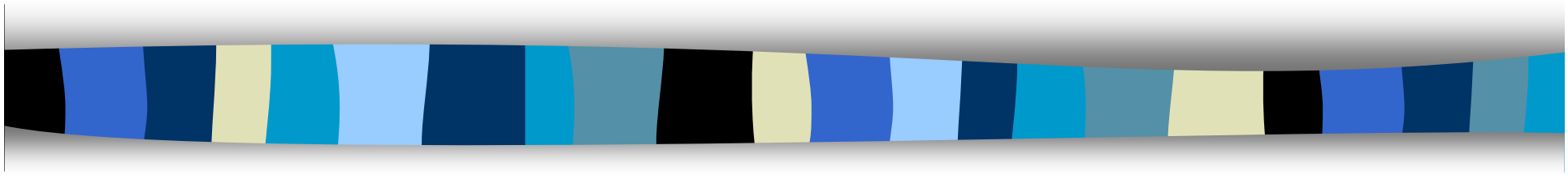
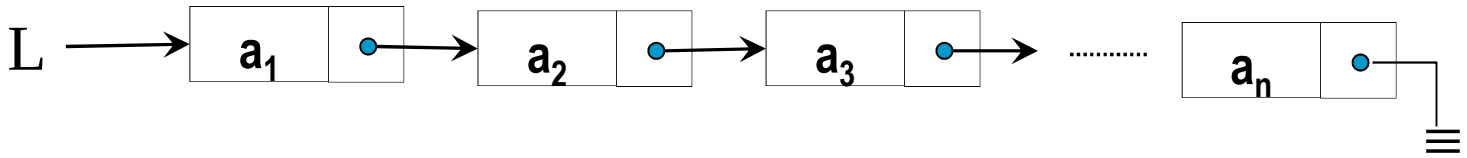
SCC-122 Estruturas de Dados



Versões de Lista Encadeada

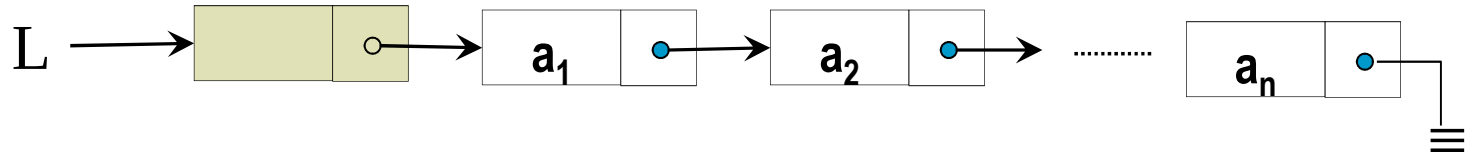
Cabeça de Lista

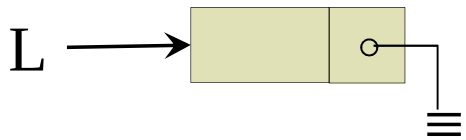


Problemas

a. Qualquer percurso requer previsão de término da lista;
b. Inserção/eliminação na 1ª posição: tratado como caso especial devido a alteração de L;

- Para contornar (b): elemento especial chamado **cabeça de lista** ou **head node** que não contém elemento lógico da lista (a_i), mas delimita a lista no seu início – podendo guardar outras informações pertinentes.





Lista Vazia

Cabeça de Lista

```
int insere(no *L, tipo_elem x){
//Inserção de x numa lista, L, ordenada – usando head node
  no *q, *qa, *j;
  int acabou;
  q = L->lig;
  qa = L;
  insere = 1;
  acabou = (q == NULL);
  while ( (!acabou) ){
    if ( q->info < x ){
        qa = q;
        q = q->lig;
        acabou = (q == NULL);
      }
    else if ( q->info > x ) acabou = 1;
          else return ( 0 );
           //retornar sem inserir pois q->info é igual a x

  return ( apos(L,qa,x) ); //inserir à direita de qa
}
```

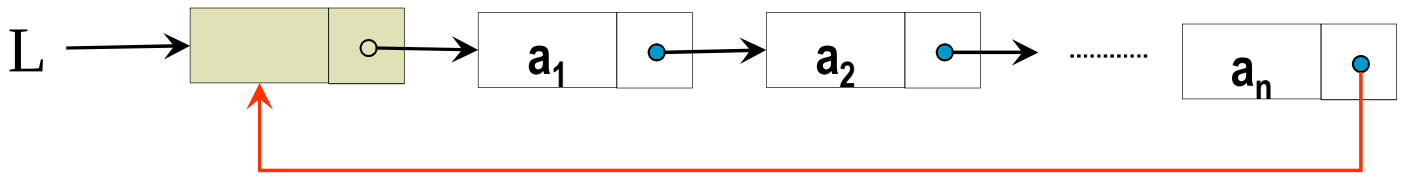
- Obs.: **L** nunca é modificado e **qa** nunca vai ser NULL

Cabeça de Lista

- O nó cabeça pode guardar informações úteis como:
  - Número de elementos da lista;
  - Se ela está ordenada ou não;
  - etc

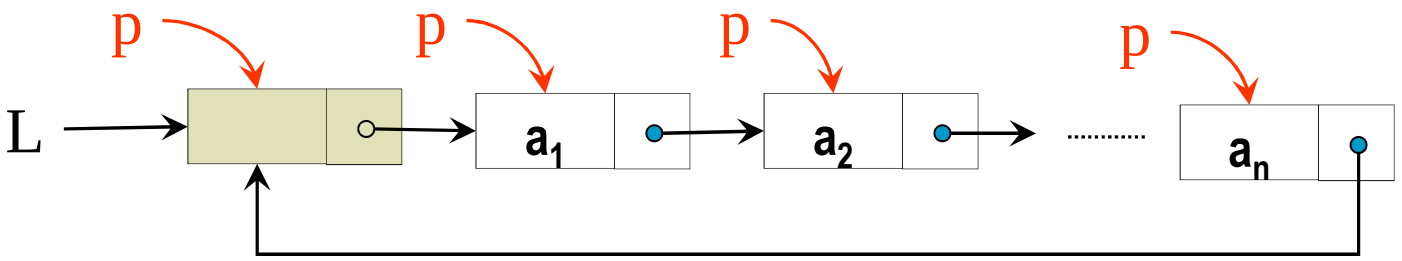
Sentinela

- Em listas encadeadas não-ordenadas, é interessante usar **sentinela** para a busca de chaves.
- O **head node** poderia ter esse papel, mas ele está no início da lista...
- Fazemos, então com que ele fique no início e no fim, tornando a Lista Circular:



Sentinela

- Dessa forma, não é necessário verificar o final de lista como (p->lig = NULL), mas sim se p, que começou em L->lig, retornou à L.



- Colocado o valor procurado, x, no head node, o algoritmo de busca, em lista não ordenada, ficaria:

Sentinela

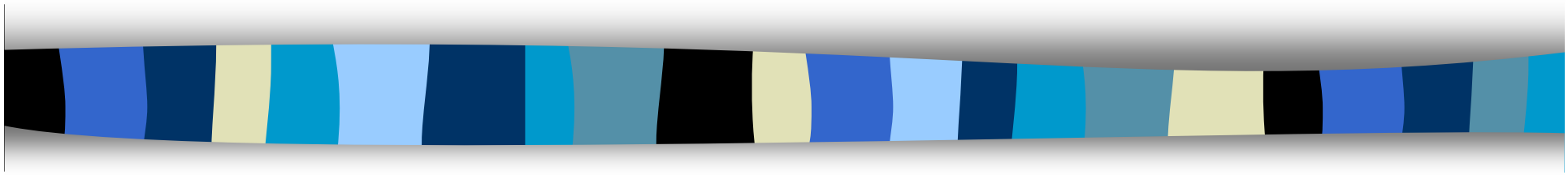
```
no *busca(no *L, tipo_elem x){
//retorna NULL ou endereço do registro que contém x
 no *p;
 L->info = x;
 p = L->lig;
 while ( p->info != x )
   p = p->lig;
 if (p == L) return ( NULL );
 else return ( p );
}
```

- E se a Lista for ordenada?
- Basta alterar o teste:

```
while (p->info < x)  p = p->lig;
```

- Reparem que essas alterações podem ser usadas na inserção e eliminação também.

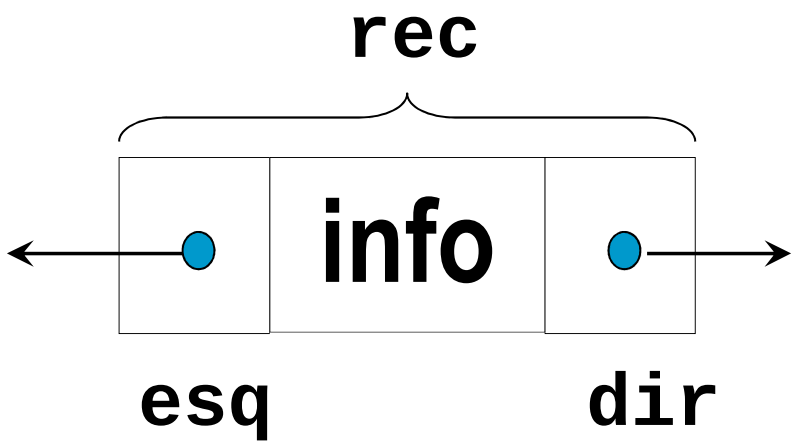
Estruturas de Dados



Lista Duplamente Encadeada

Lista Duplamente Encadeada

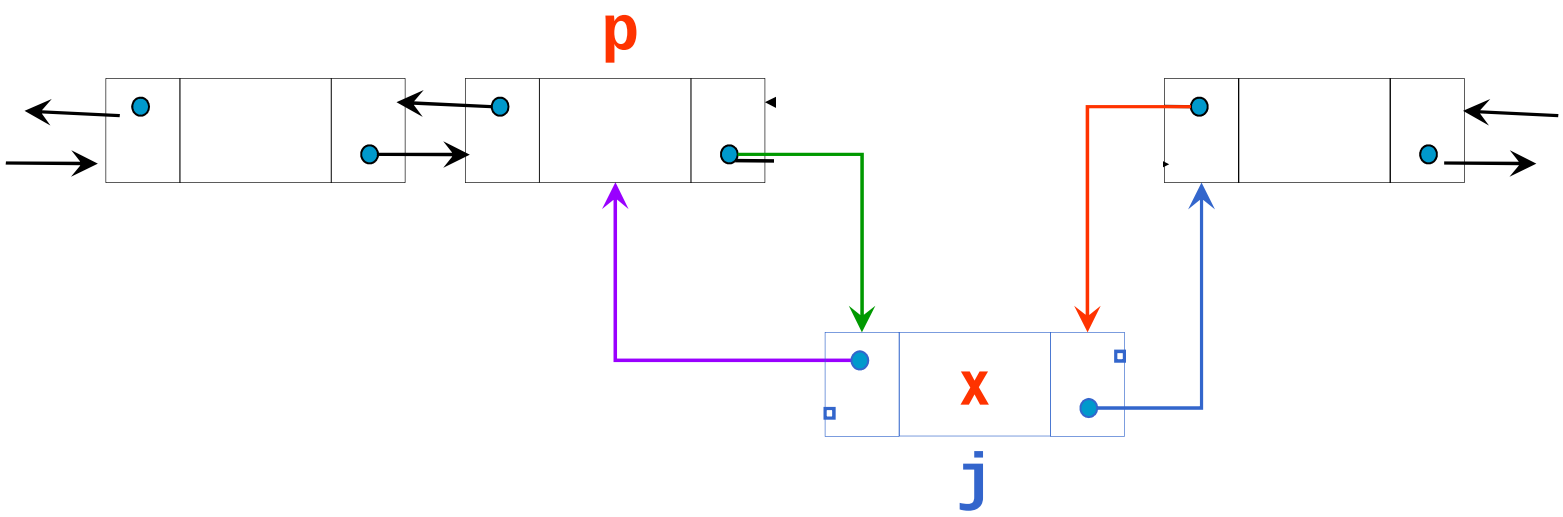
```
typedef struct _no no;

struct _no{
      tipo_elem info;
      no *esq, *dir;
};
```



- Facilita a inserção/eliminação no interior de uma lista;
- Quando a busca ocorre segundo o valor do campo info, ao encontrá-lo podemos inserir/eliminar sem a necessidade de ponteiro auxiliar.
- No encadeamento duplo pode ocorrer todo tipo de variação: Circular, com nó cabeça, sem nó cabeça, circular com nó cabeça, circular sem nó cabeça, etc.

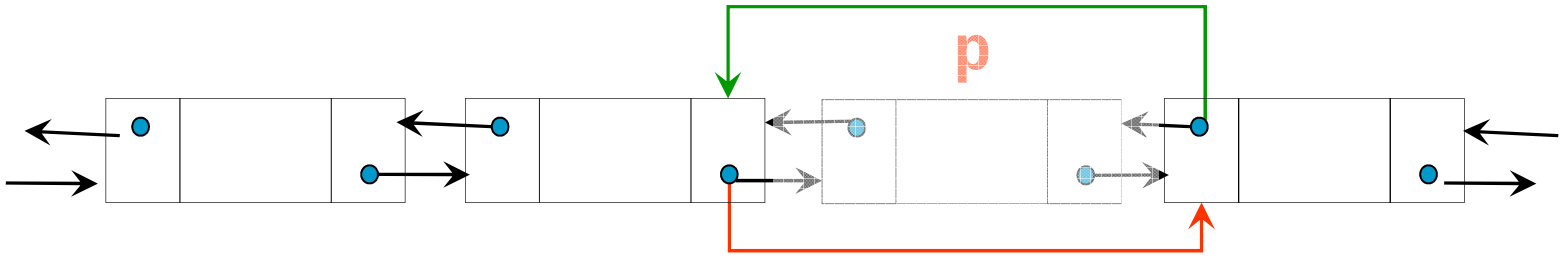
- Inserção após o registro **p**

```
j->info = x;
j->esq = p;
j->dir = p->dir;
p->dir->esq = j;
P->dir = j;
```



- Eliminação do registro **p**

```
P->dir->esq = p->esq;
P->esq->dir = p->dir;
delete(p);
```



Este material didático foi revisado e adaptado por Danilo Medeiros Eler e pelo prof. Adenilso da Silva Simão a partir do material da profa. Roseli Ap. Francelin Romero.

Foi revisado pela profa. Roseli em 2011.