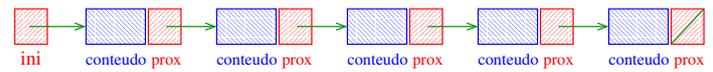


## Melhores momentos

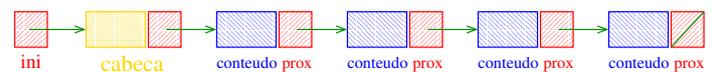
## Listas

Ilustração de uma lista encadeada “sem cabeça”



## AULA 9

Ilustração de uma lista encadeada “com cabeça”



### Estrutura de uma lista

```
struct celula {
    int conteudo;
    struct celula *prox;
};
typedef struct celula Celula;

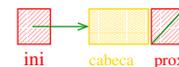
Celula *ini;
/* inicialmente a lista esta vazia */
ini = NULL;
```



### Estrutura de uma lista com cabeça

```
struct celula {
    int conteudo;
    struct celula *prox;
};
typedef struct celula Celula;

Celula *ini, cabeca;
/* inicialmente a lista esta vazia */
cabeca.prox = NULL;
ini = &cabeca;
```



### Estrutura de uma lista com cabeça

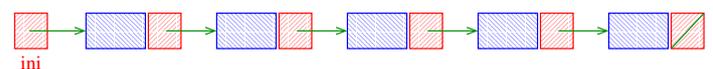
```
struct celula {
    int conteudo;
    struct celula *prox;
};
typedef struct celula Celula;

Celula *ini;
/* inicialmente a lista esta vazia */
ini = malloc(sizeof(Celula));
ini->prox = NULL;
```



### Imprime uma lista

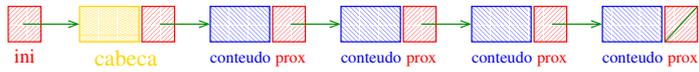
Esta função `imprime` o `conteudo` de cada célula de uma lista encadeada `ini`.



```
void imprima (Celula *ini)
{
    Celula *p;
    for (p=ini; p != NULL; p=p->prox)
        printf("%d\n", p->conteudo);
}
```

## Imprime uma lista com cabeça

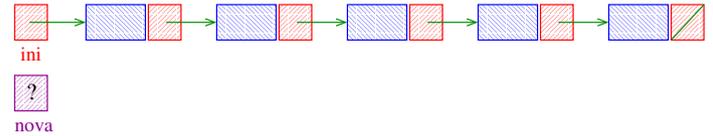
Esta função `imprime` o `conteudo` de cada célula de uma lista encadeada com cabeça `ini`.



```
void imprima (Celula *ini)
{
    Celula *p;
    for (p=ini->prox; p != NULL; p=p->prox)
        printf("%d\n", p->conteudo);
}
```

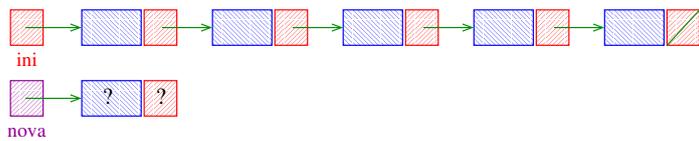
< > < > < > < > < > < >

## Inserção no início de uma lista



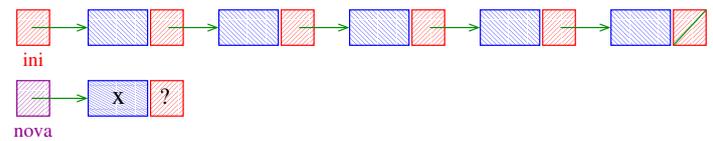
< > < > < > < > < > < >

## Inserção no início de uma lista



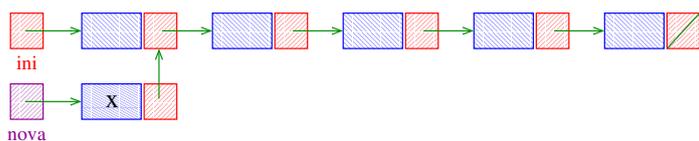
< > < > < > < > < > < >

## Inserção no início de uma lista



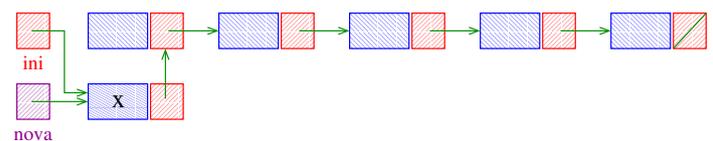
< > < > < > < > < > < >

## Inserção no início de uma lista



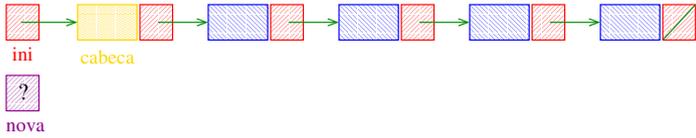
< > < > < > < > < > < >

## Inserção no início de uma lista

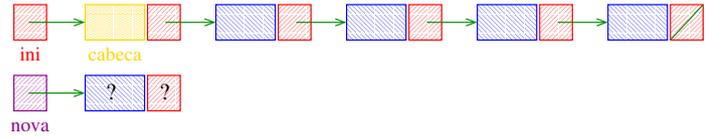


< > < > < > < > < > < >

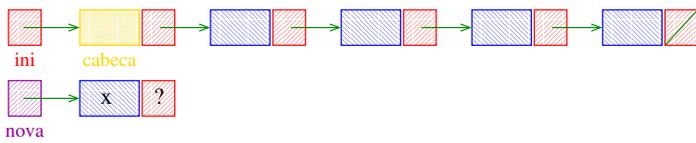
### Inserção no início de uma lista com cabeça



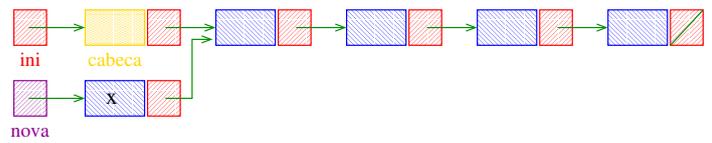
### Inserção no início de uma lista com cabeça



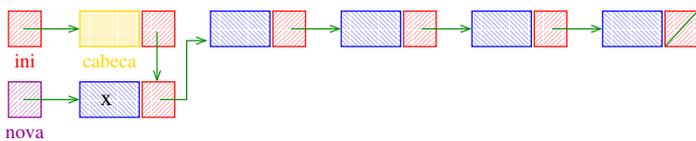
### Inserção no início de uma lista com cabeça



### Inserção no início de uma lista com cabeça



### Inserção no início de uma lista com cabeça



AULA 10

## Mais listas encadeadas ainda

PF 4, S 3.3

<http://www.ime.usp.br/~pf/algoritmos/aulas/lista.html>

## Busca e Remoção em uma lista

**Remove**, caso exista, a primeira célula da lista **ini** que contém o elemento **x**.

```
Celula *buscaRemove(int x, Celula *ini) {  
    Celula *p, *q;  
    if (ini == NULL) return ini;  
    if (ini->conteudo == x) {  
        q = ini;  
        ini = q->prox;  
        free(q);  
    }  
}
```

Navigation icons

Navigation icons

## Busca e Remoção em uma lista

```
else {  
    p = ini;  
    q = p->prox;  
    while (q!=NULL && q->conteudo!=x){  
        p = q;  
        q = p->prox;  
    }  
    if (q != NULL) {  
        p->prox = q->prox;  
        free(q);  
    }  
}  
return ini;  
}
```

Navigation icons

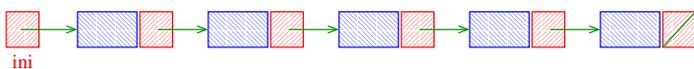
## Exemplos de chamadas de buscaRemove

```
Celula *ini, *ini2;  
ini = ini2 = NULL;  
  
[...manipulação das listas ...]  
  
ini = buscaRemove(22, ini);  
ini2 = buscaRemove(x+1, ini2);  
ini2 = buscaRemove(x+y, ini2);  
ini = buscaRemove(valor, ini);
```

Navigation icons

## Busca e Remoção em uma lista

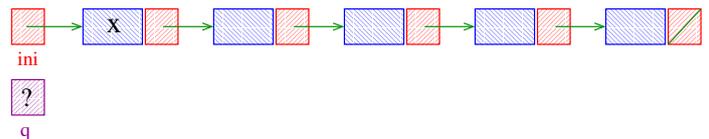
**Remove**, caso exista, a primeira célula da lista **ini** que contém o elemento **x**.



Navigation icons

## Busca e Remoção em uma lista

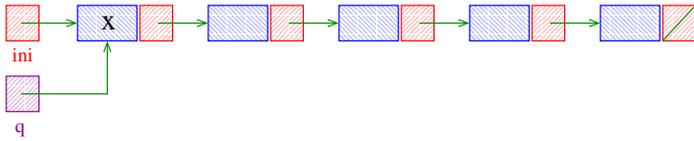
**Remove**, caso exista, a primeira célula da lista **ini** que contém o elemento **x**.



Navigation icons

## Busca e Remoção em uma lista

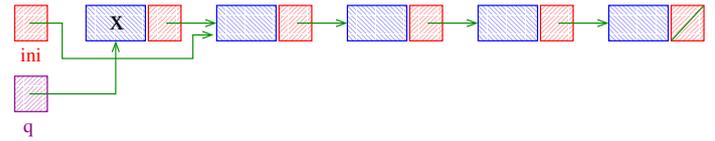
Remove, caso exista, a primeira célula da lista *ini* que contém o elemento *x*.



< > < > < > < > < > < >

## Busca e Remoção em uma lista

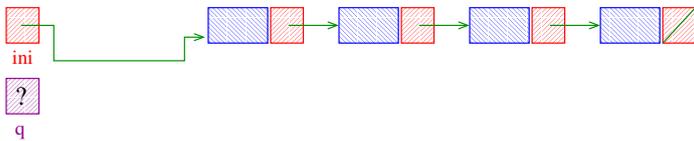
Remove, caso exista, a primeira célula da lista *ini* que contém o elemento *x*.



< > < > < > < > < > < >

## Busca e Remoção em uma lista

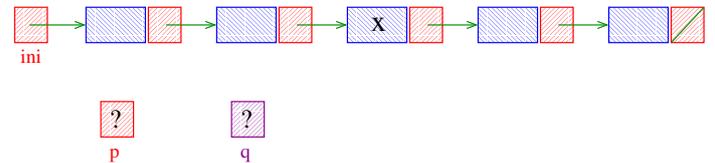
Remove, caso exista, a primeira célula da lista *ini* que contém o elemento *x*.



< > < > < > < > < > < >

## Busca e Remoção em uma lista

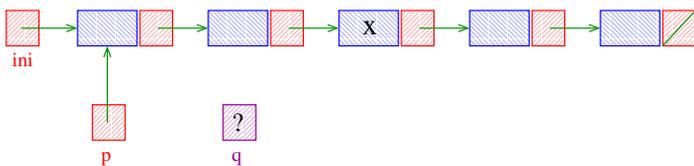
Remove, caso exista, a primeira célula da lista *ini* que contém o elemento *x*.



< > < > < > < > < > < >

## Busca e Remoção em uma lista

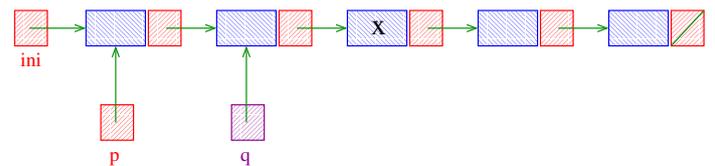
Remove, caso exista, a primeira célula da lista *ini* que contém o elemento *x*.



< > < > < > < > < > < >

## Busca e Remoção em uma lista

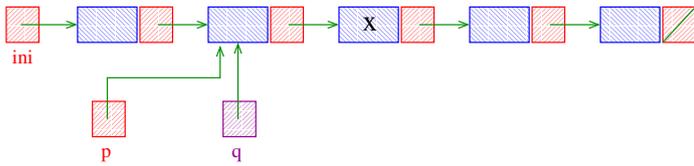
Remove, caso exista, a primeira célula da lista *ini* que contém o elemento *x*.



< > < > < > < > < > < >

## Busca e Remoção em uma lista

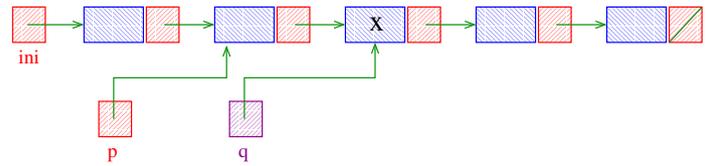
**Remove**, caso exista, a primeira célula da lista **ini** que contém o elemento **x**.



< > < > < > < > < > < >

## Busca e Remoção em uma lista

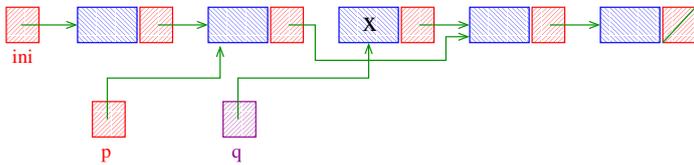
**Remove**, caso exista, a primeira célula da lista **ini** que contém o elemento **x**.



< > < > < > < > < > < >

## Busca e Remoção em uma lista

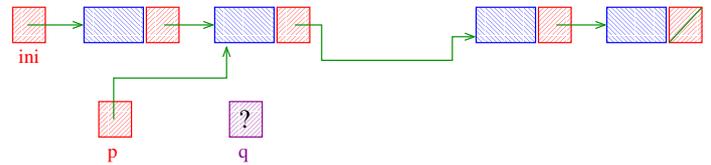
**Remove**, caso exista, a primeira célula da lista **ini** que contém o elemento **x**.



< > < > < > < > < > < >

## Busca e Remoção em uma lista

**Remove**, caso exista, a primeira célula da lista **ini** que contém o elemento **x**.



< > < > < > < > < > < >

## Busca e Remoção em uma lista com cabeça

**Remove**, caso exista, a primeira célula da lista com cabeça **ini** que contém o elemento **x**.

```
Celula *buscaRemove (int x, Celula *ini){
    Celula *p, *q;
    if (ini == NULL) return ini;
    if (ini->conteudo == x) {
        q = ini;
        ini = q->prox;
        free(q);
    }
}
```

< > < > < > < > < > < >

## Busca e Remoção em uma lista com cabeça

**Remove**, caso exista, a primeira célula da lista com cabeça **ini** que contém o elemento **x**.

```
Celula *buscaRemove (int x, Celula *ini){
    Celula *p, *q;
    if (ini == NULL) return ini;
    if (ini->conteudo == x) {
        q = ini;
        ini = q->prox;
        free(q);
    }
}
```

< > < > < > < > < > < >

## Busca e Remoção em uma lista com cabeça

```
else {
    p = ini;
    q = p->prox;
    while (q!=NULL && q->conteudo!=x){
        p = q;
        q = p->prox;
    }
    if (q != NULL) {
        p->prox = q->prox;
        free(q);
    }
}
return ini;
}
```

## Busca e Remoção em uma lista com cabeça

```
else {
    p = ini;
    q = p->prox;
    while (q!=NULL && q->conteudo!=x){
        p = q;
        q = p->prox;
    }
    if (q != NULL) {
        p->prox = q->prox;
        free(q);
    }
}
return ini;
}
```

## Busca e Remoção em uma lista com cabeça

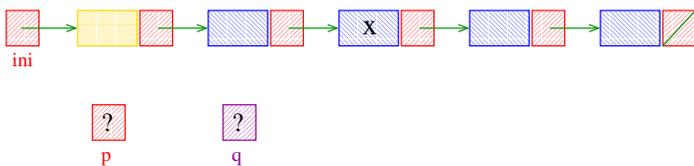
```
void buscaRemove (int x, Celula *ini) {
    Celula *p, *q;
    p = ini;
    q = p->prox;
    while (q!=NULL && q->conteudo!=x){
        p = q;
        q = p->prox;
    }
    if (q != NULL) {
        p->prox = q->prox;
        free(q);
    }
}
```

## Exemplos de chamadas de buscaRemove

```
Celula *ini, *ini2;
Celula cabeca;
ini = &cabeca;
cabeca.prox = NULL;
ini2 = malloc(sizeof(Celula));
ini2->prox = NULL;
[...manipulação das listas ...]
buscaRemove(22,&cabeca);
buscaRemove(33,ini);
buscaRemove(x+1,ini2);
buscaRemove(x+y,ini2);
buscaRemove(valor,ini);
```

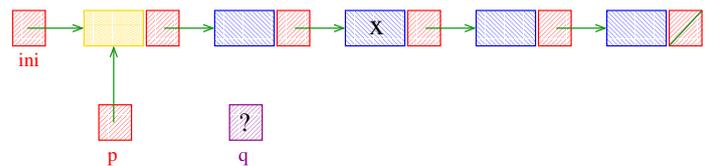
## Busca e Remoção em uma lista com cabeça

Remove, caso exista, a primeira célula da lista com cabeça ini que contém o elemento x.



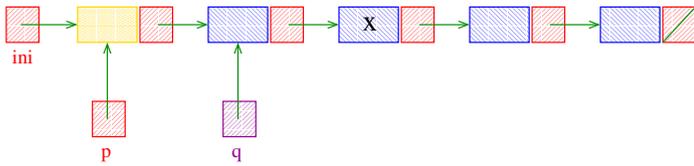
## Busca e Remoção em uma lista com cabeça

Remove, caso exista, a primeira célula da lista com cabeça ini que contém o elemento x.



## Busca e Remoção em uma lista com cabeça

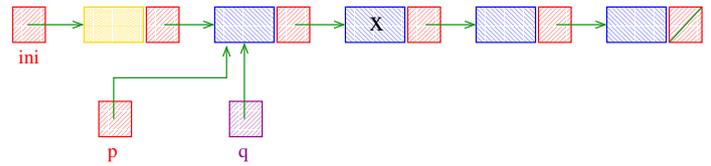
**Remove**, caso exista, a primeira célula da lista com cabeça *ini* que contém o elemento *x*.



< > < > < > < > < > < >

## Busca e Remoção em uma lista com cabeça

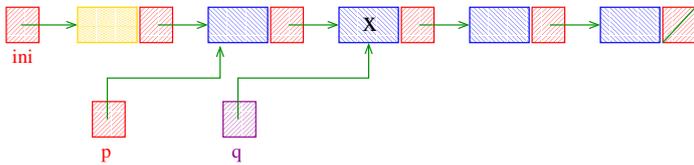
**Remove**, caso exista, a primeira célula da lista com cabeça *ini* que contém o elemento *x*.



< > < > < > < > < > < >

## Busca e Remoção em uma lista com cabeça

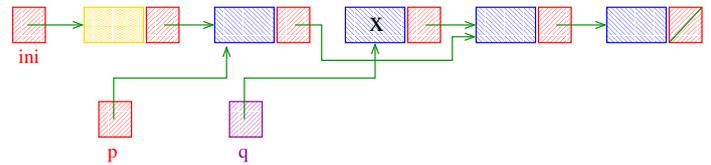
**Remove**, caso exista, a primeira célula da lista com cabeça *ini* que contém o elemento *x*.



< > < > < > < > < > < >

## Busca e Remoção em uma lista com cabeça

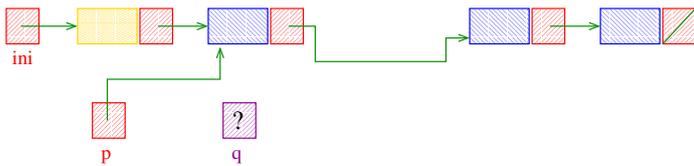
**Remove**, caso exista, a primeira célula da lista com cabeça *ini* que contém o elemento *x*.



< > < > < > < > < > < >

## Busca e Remoção em uma lista com cabeça

**Remove**, caso exista, a primeira célula da lista com cabeça *ini* que contém o elemento *x*.



< > < > < > < > < > < >

## Busca e Inserção em uma lista

**Recebe** uma lista *ini* e **insere** uma célula de conteúdo *x* antes da primeira célula de conteúdo *y*. Se nenhuma célula contém *y*, insere a célula com *y* no final da lista.

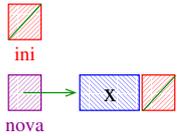
```
Celula *buscaInsere(int x, Celula *ini) {
    Celula *p, *q, *nova;
    nova = malloc(sizeof(Celula));
    nova->conteudo = x;
    if (ini == NULL || ini->conteudo == y){
        nova->prox = ini;
        ini = nova;
    }
}
```

< > < > < > < > < > < >



## Busca e Inseção em uma lista

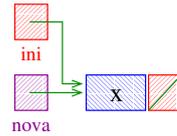
Inserir uma célula de conteúdo  $x$  antes da primeira célula de conteúdo  $y$ . Se nenhuma célula contém  $y$ , inserir a célula com  $y$  no final da lista.



< > < > < > < > < > < >

## Busca e Inseção em uma lista

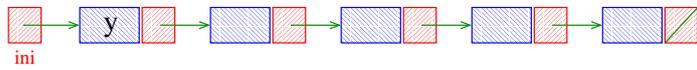
Inserir uma célula de conteúdo  $x$  antes da primeira célula de conteúdo  $y$ . Se nenhuma célula contém  $y$ , inserir a célula com  $y$  no final da lista.



< > < > < > < > < > < >

## Busca e Inseção em uma lista

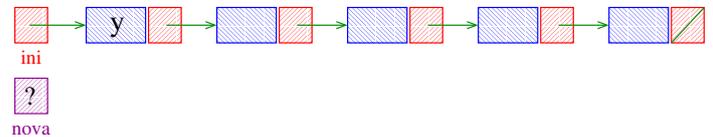
Inserir uma célula de conteúdo  $x$  antes da primeira célula de conteúdo  $y$ . Se nenhuma célula contém  $y$ , inserir a célula com  $y$  no final da lista.



< > < > < > < > < > < >

## Busca e Inseção em uma lista

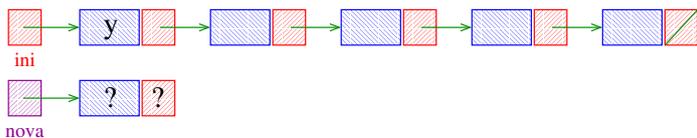
Inserir uma célula de conteúdo  $x$  antes da primeira célula de conteúdo  $y$ . Se nenhuma célula contém  $y$ , inserir a célula com  $y$  no final da lista.



< > < > < > < > < > < >

## Busca e Inseção em uma lista

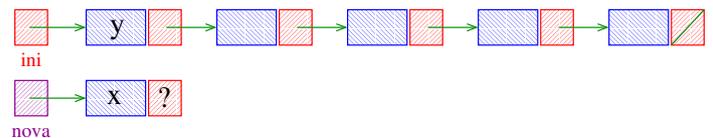
Inserir uma célula de conteúdo  $x$  antes da primeira célula de conteúdo  $y$ . Se nenhuma célula contém  $y$ , inserir a célula com  $y$  no final da lista.



< > < > < > < > < > < >

## Busca e Inseção em uma lista

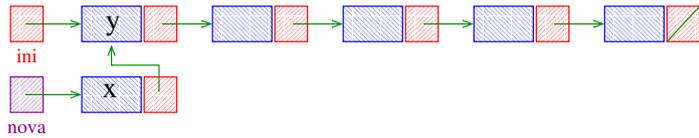
Inserir uma célula de conteúdo  $x$  antes da primeira célula de conteúdo  $y$ . Se nenhuma célula contém  $y$ , inserir a célula com  $y$  no final da lista.



< > < > < > < > < > < >

## Busca e Inseção em uma lista

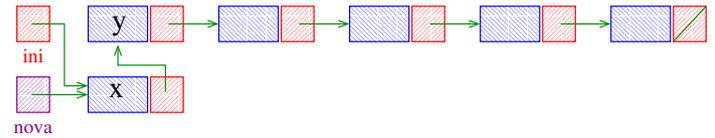
Inserir uma célula de conteúdo **x** antes da primeira célula de conteúdo **y**. Se nenhuma célula contém **y**, inserir a célula com **y** no final da lista.



< > < > < > < > < > < >

## Busca e Inseção em uma lista

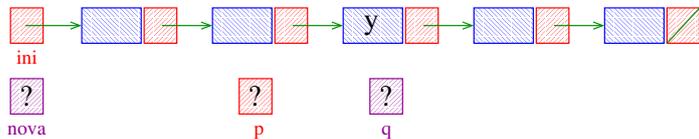
Inserir uma célula de conteúdo **x** antes da primeira célula de conteúdo **y**. Se nenhuma célula contém **y**, inserir a célula com **y** no final da lista.



< > < > < > < > < > < >

## Busca e Inseção em uma lista

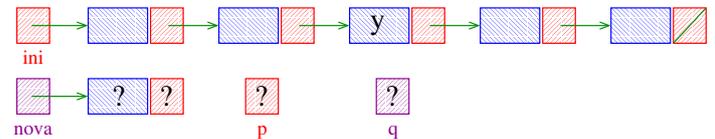
Inserir uma célula de conteúdo **x** antes da primeira célula de conteúdo **y**. Se nenhuma célula contém **y**, inserir a célula com **y** no final da lista.



< > < > < > < > < > < >

## Busca e Inseção em uma lista

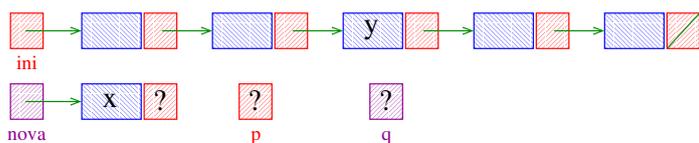
Inserir uma célula de conteúdo **x** antes da primeira célula de conteúdo **y**. Se nenhuma célula contém **y**, inserir a célula com **y** no final da lista.



< > < > < > < > < > < >

## Busca e Inseção em uma lista

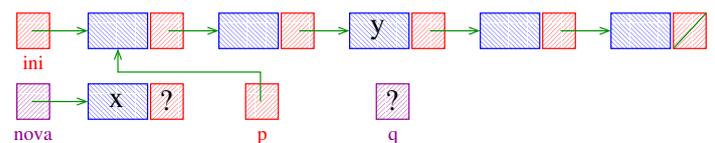
Inserir uma célula de conteúdo **x** antes da primeira célula de conteúdo **y**. Se nenhuma célula contém **y**, inserir a célula com **y** no final da lista.



< > < > < > < > < > < >

## Busca e Inseção em uma lista

Inserir uma célula de conteúdo **x** antes da primeira célula de conteúdo **y**. Se nenhuma célula contém **y**, inserir a célula com **y** no final da lista.



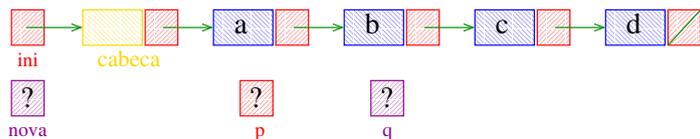
< > < > < > < > < > < >





## Busca e Inserção em uma lista com cabeça

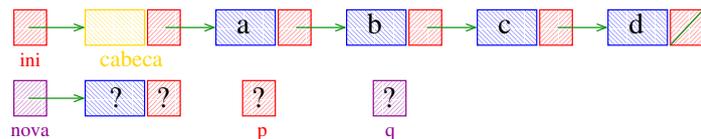
Recebe uma lista com cabeça *ini* e insere uma célula de conteúdo *x* antes da primeira célula de conteúdo *y*. Se nenhuma célula contém *y*, insere a célula com *y* no final da lista.



Navigation icons: back, forward, search, etc.

## Busca e Inserção em uma lista com cabeça

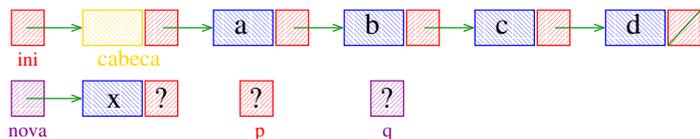
Recebe uma lista com cabeça *ini* e insere uma célula de conteúdo *x* antes da primeira célula de conteúdo *y*. Se nenhuma célula contém *y*, insere a célula com *y* no final da lista.



Navigation icons: back, forward, search, etc.

## Busca e Inserção em uma lista com cabeça

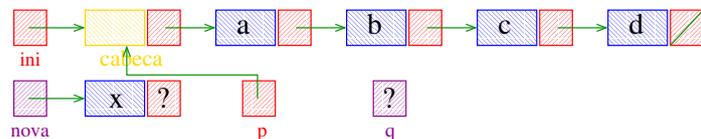
Recebe uma lista com cabeça *ini* e insere uma célula de conteúdo *x* antes da primeira célula de conteúdo *y*. Se nenhuma célula contém *y*, insere a célula com *y* no final da lista.



Navigation icons: back, forward, search, etc.

## Busca e Inserção em uma lista com cabeça

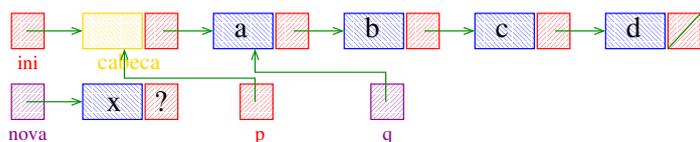
Recebe uma lista com cabeça *ini* e insere uma célula de conteúdo *x* antes da primeira célula de conteúdo *y*. Se nenhuma célula contém *y*, insere a célula com *y* no final da lista.



Navigation icons: back, forward, search, etc.

## Busca e Inserção em uma lista com cabeça

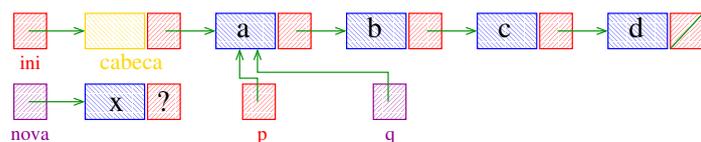
Recebe uma lista com cabeça *ini* e insere uma célula de conteúdo *x* antes da primeira célula de conteúdo *y*. Se nenhuma célula contém *y*, insere a célula com *y* no final da lista.



Navigation icons: back, forward, search, etc.

## Busca e Inserção em uma lista com cabeça

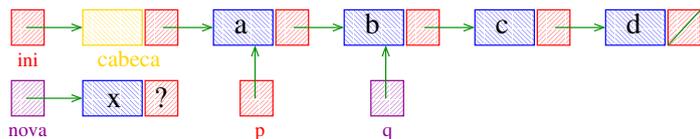
Recebe uma lista com cabeça *ini* e insere uma célula de conteúdo *x* antes da primeira célula de conteúdo *y*. Se nenhuma célula contém *y*, insere a célula com *y* no final da lista.



Navigation icons: back, forward, search, etc.

## Busca e Inserção em uma lista com cabeça

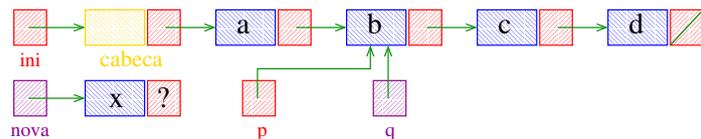
Recebe uma lista com cabeça *ini* e insere uma célula de conteúdo *x* antes da primeira célula de conteúdo *y*. Se nenhuma célula contém *y*, insere a célula com *y* no final da lista.



Navigation icons: back, forward, search, etc.

## Busca e Inserção em uma lista com cabeça

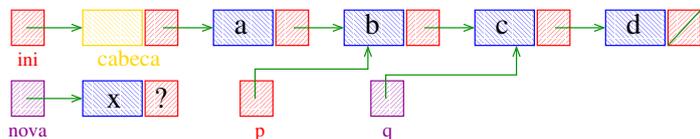
Recebe uma lista com cabeça *ini* e insere uma célula de conteúdo *x* antes da primeira célula de conteúdo *y*. Se nenhuma célula contém *y*, insere a célula com *y* no final da lista.



Navigation icons: back, forward, search, etc.

## Busca e Inserção em uma lista com cabeça

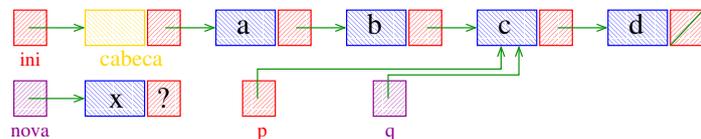
Recebe uma lista com cabeça *ini* e insere uma célula de conteúdo *x* antes da primeira célula de conteúdo *y*. Se nenhuma célula contém *y*, insere a célula com *y* no final da lista.



Navigation icons: back, forward, search, etc.

## Busca e Inserção em uma lista com cabeça

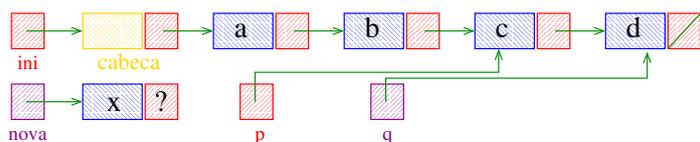
Recebe uma lista com cabeça *ini* e insere uma célula de conteúdo *x* antes da primeira célula de conteúdo *y*. Se nenhuma célula contém *y*, insere a célula com *y* no final da lista.



Navigation icons: back, forward, search, etc.

## Busca e Inserção em uma lista com cabeça

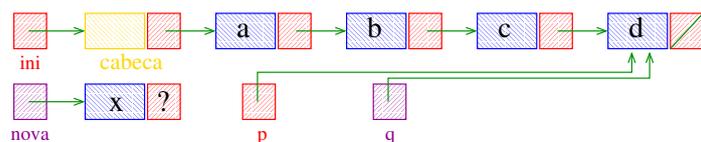
Recebe uma lista com cabeça *ini* e insere uma célula de conteúdo *x* antes da primeira célula de conteúdo *y*. Se nenhuma célula contém *y*, insere a célula com *y* no final da lista.



Navigation icons: back, forward, search, etc.

## Busca e Inserção em uma lista com cabeça

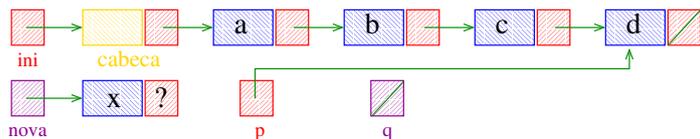
Recebe uma lista com cabeça *ini* e insere uma célula de conteúdo *x* antes da primeira célula de conteúdo *y*. Se nenhuma célula contém *y*, insere a célula com *y* no final da lista.



Navigation icons: back, forward, search, etc.

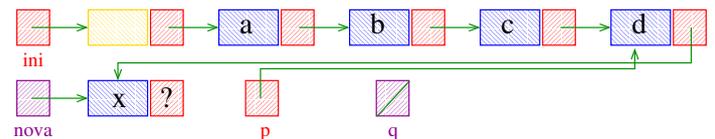
## Busca e Inserção em uma lista com cabeça

Recebe uma lista com cabeça *ini* e insere uma célula de conteúdo *x* antes da primeira célula de conteúdo *y*. Se nenhuma célula contém *y*, insere a célula com *y* no final da lista.



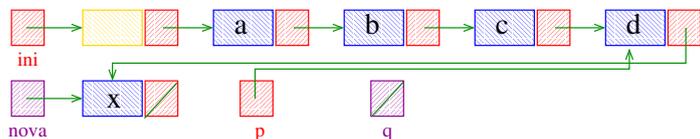
## Busca e Inserção em uma lista com cabeça

Recebe uma lista com cabeça *ini* e insere uma célula de conteúdo *x* antes da primeira célula de conteúdo *y*. Se nenhuma célula contém *y*, insere a célula com *y* no final da lista.



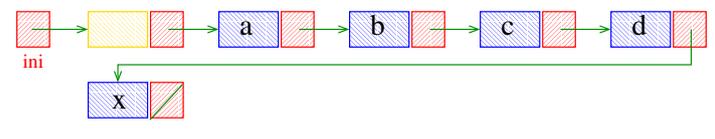
## Busca e Inserção em uma lista com cabeça

Recebe uma lista com cabeça *ini* e insere uma célula de conteúdo *x* antes da primeira célula de conteúdo *y*. Se nenhuma célula contém *y*, insere a célula com *y* no final da lista.



## Busca e Inserção em uma lista com cabeça

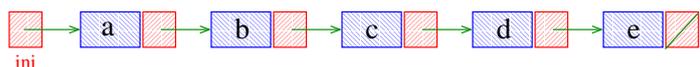
Recebe uma lista com cabeça *ini* e insere uma célula de conteúdo *x* antes da primeira célula de conteúdo *y*. Se nenhuma célula contém *y*, insere a célula com *y* no final da lista.



## Inversão de uma lista

Recebe uma lista *ini* e inverte a ordem de suas células alterando apenas os ponteiros.

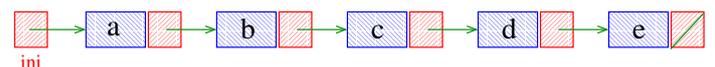
Lista antes da inversão:



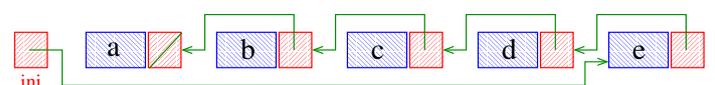
## Inversão de uma lista

Recebe uma lista *ini* e inverte a ordem de suas células alterando apenas os ponteiros.

Lista antes da inversão:



Lista depois da inversão:



## Inversão de uma lista

Recebe uma lista `ini` e `inverte` a ordem de suas células alterando apenas os ponteiros.

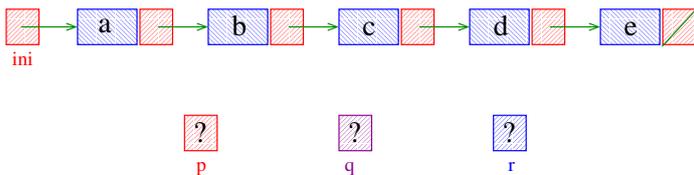
```
Celula *inverte(Celula *ini) {
    Celula *p, *q, *r;
    p = NULL; q = ini;
    while (q != NULL) {
        r = q->prox;
        q->prox = p;
        p = q;
        q = r;
    }
    return p;
}
```

## Exemplos de chamadas

```
Celula *ini, *ini2;
Celula cabeca;
ini = &cabeca;
cabeca.prox = NULL;
ini2 = malloc(sizeof(Celula));
ini2->prox = NULL;
[...manipulação das listas ...]
ini->prox = inverte(ini->prox);
cabeca.prox = inverte(cabeca.prox);
ini->prox = inverte(cabeca.prox);
cabeca.prox = inverte(ini->prox);
ini2->prox = inverte(ini2->prox);
```

## Inversão de uma lista

Recebe uma lista `ini` e `inverte` a ordem de suas células alterando apenas os ponteiros.

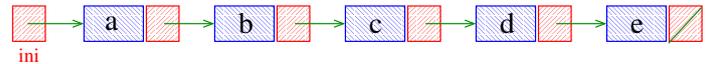


## Exemplos de chamadas

```
Celula *ini, *ini2;
ini = ini2 = NULL;
[...manipulação da lista ...]
ini = inverte(ini);
ini2 = inverte(ini2);
```

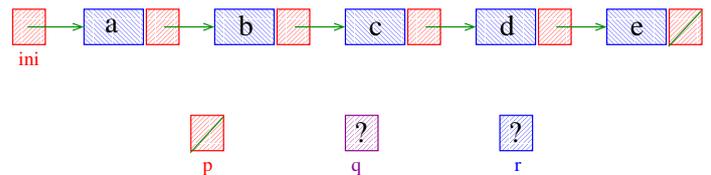
## Inversão de uma lista

Recebe uma lista `ini` e `inverte` a ordem de suas células alterando apenas os ponteiros.



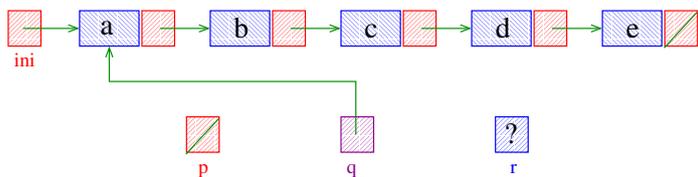
## Inversão de uma lista

Recebe uma lista `ini` e `inverte` a ordem de suas células alterando apenas os ponteiros.



## Inversão de uma lista

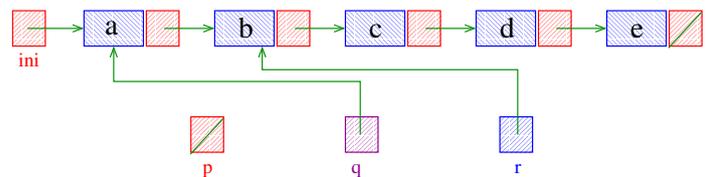
Recebe uma lista **ini** e **inverte** a ordem de suas células alterando apenas os ponteiros.



< > < > < > < > < > < >

## Inversão de uma lista

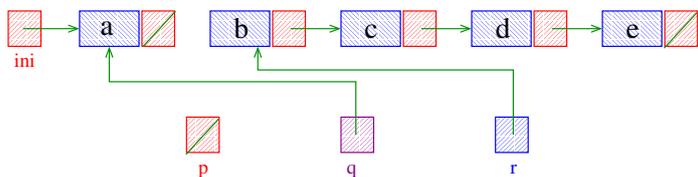
Recebe uma lista **ini** e **inverte** a ordem de suas células alterando apenas os ponteiros.



< > < > < > < > < > < >

## Inversão de uma lista

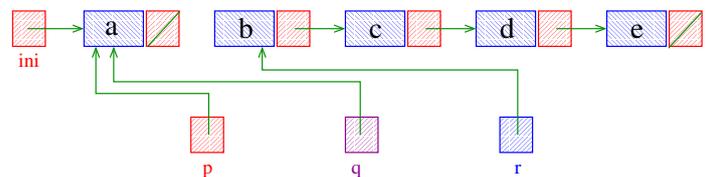
Recebe uma lista **ini** e **inverte** a ordem de suas células alterando apenas os ponteiros.



< > < > < > < > < > < >

## Inversão de uma lista

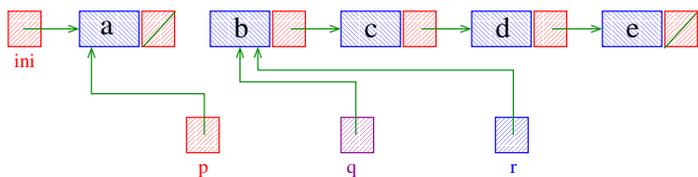
Recebe uma lista **ini** e **inverte** a ordem de suas células alterando apenas os ponteiros.



< > < > < > < > < > < >

## Inversão de uma lista

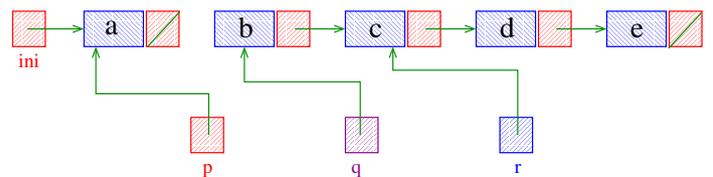
Recebe uma lista **ini** e **inverte** a ordem de suas células alterando apenas os ponteiros.



< > < > < > < > < > < >

## Inversão de uma lista

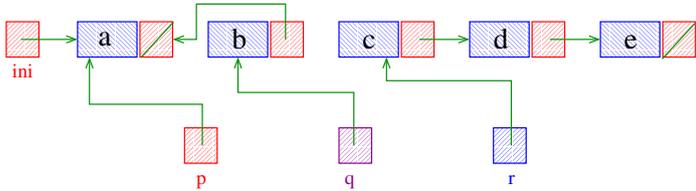
Recebe uma lista **ini** e **inverte** a ordem de suas células alterando apenas os ponteiros.



< > < > < > < > < > < >

## Inversão de uma lista

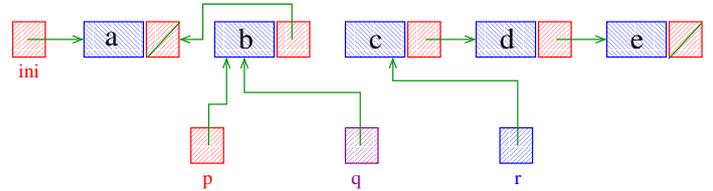
Recebe uma lista *ini* e *inverte* a ordem de suas células alterando apenas os ponteiros.



< > < > < > < > < > < >

## Inversão de uma lista

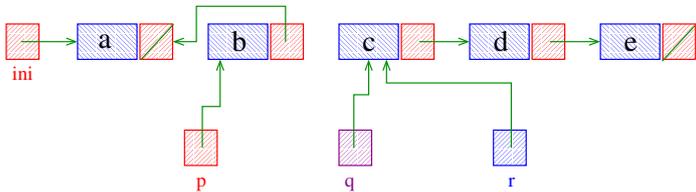
Recebe uma lista *ini* e *inverte* a ordem de suas células alterando apenas os ponteiros.



< > < > < > < > < > < >

## Inversão de uma lista

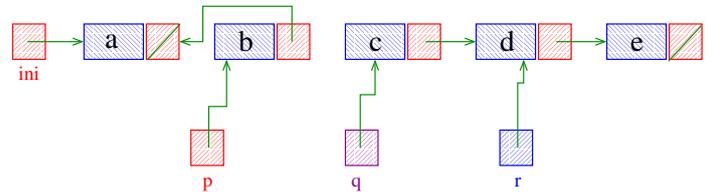
Recebe uma lista *ini* e *inverte* a ordem de suas células alterando apenas os ponteiros.



< > < > < > < > < > < >

## Inversão de uma lista

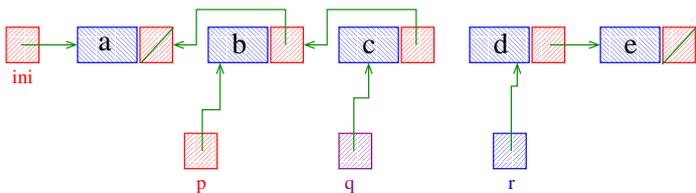
Recebe uma lista *ini* e *inverte* a ordem de suas células alterando apenas os ponteiros.



< > < > < > < > < > < >

## Inversão de uma lista

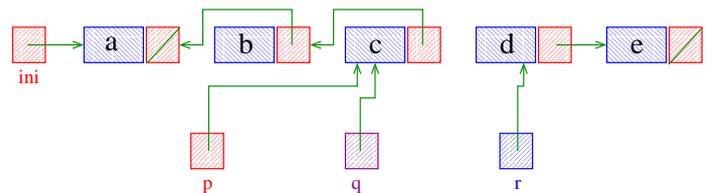
Recebe uma lista *ini* e *inverte* a ordem de suas células alterando apenas os ponteiros.



< > < > < > < > < > < >

## Inversão de uma lista

Recebe uma lista *ini* e *inverte* a ordem de suas células alterando apenas os ponteiros.

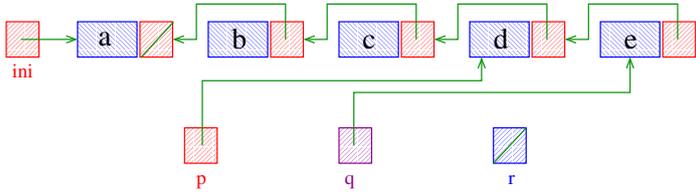


< > < > < > < > < > < >



## Inversão de uma lista

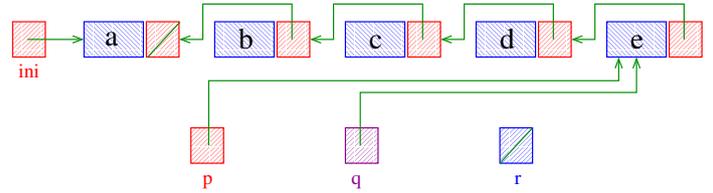
Recebe uma lista *ini* e *inverte* a ordem de suas células alterando apenas os ponteiros.



< > < > < > < > < > < >

## Inversão de uma lista

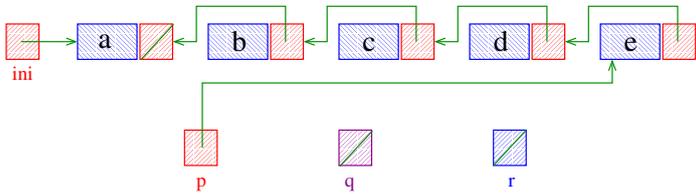
Recebe uma lista *ini* e *inverte* a ordem de suas células alterando apenas os ponteiros.



< > < > < > < > < > < >

## Inversão de uma lista

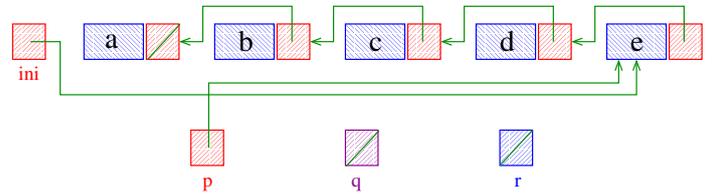
Recebe uma lista *ini* e *inverte* a ordem de suas células alterando apenas os ponteiros.



< > < > < > < > < > < >

## Inversão de uma lista

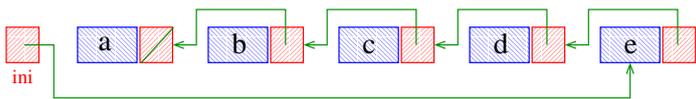
Recebe uma lista *ini* e *inverte* a ordem de suas células alterando apenas os ponteiros.



< > < > < > < > < > < >

## Inversão de uma lista

Recebe uma lista *ini* e *inverte* a ordem de suas células alterando apenas os ponteiros.



< > < > < > < > < > < >