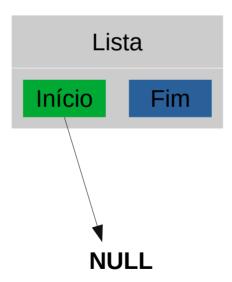
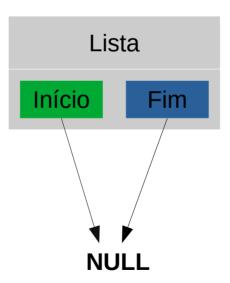
Exercício 1 - Passo 0. Criar lista

```
criarLista():
inicio ← NULO;
fim ← NULO;
tamanho ← 0;
```



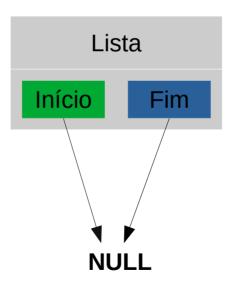
Passo 0. Criar lista

```
criarLista():
inicio ← NULO;
fim ← NULO;
tamanho ← 0;
```



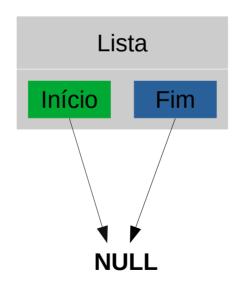
Passo 0. Criar lista

```
criarLista():
inicio ← NULO;
fim ← NULO;
tamanho ← 0;
tamanho = 0
```



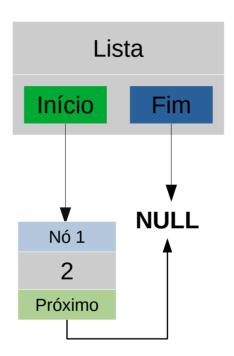
inserirEmListaVazia(valor):

```
novo ← criar_noh(valor);
inicio ← novo;
fim ← novo;
tamanho ← 1;
tamanho = 0
```

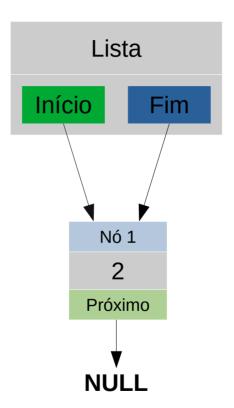




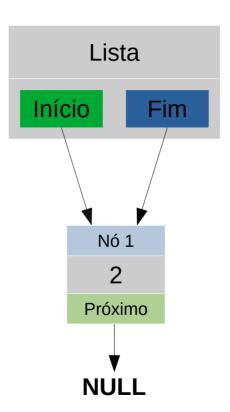
```
inserirEmListaVazia(valor):
novo ← criar_noh(valor);
inicio ← novo;
fim ← novo;
tamanho ← 1;
tamanho = 0
```



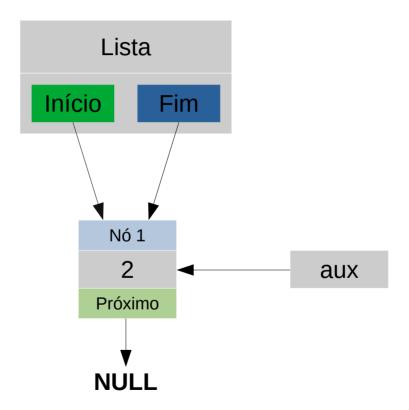
```
inserirEmListaVazia(valor):
novo ← criar_noh(valor);
inicio ← novo;
fim ← novo;
tamanho ← 1;
tamanho = 0
```



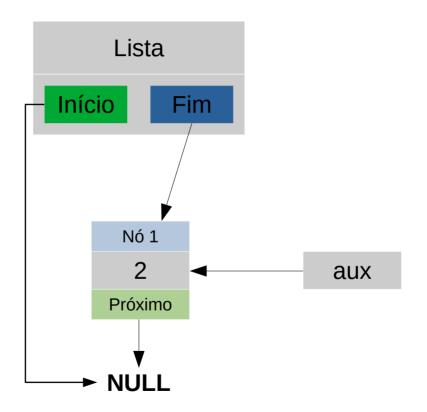
```
inserirEmListaVazia(valor):
novo ← criar_noh(valor);
inicio ← novo;
fim ← novo;
tamanho ← 1;
tamanho = 1
```



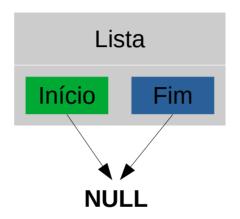
```
Se (tamanaho = 1){
    aux = inicio
    Inicio = null
    Fim = null
    delete aux
    tamanho--
}
tamanho = 1
```

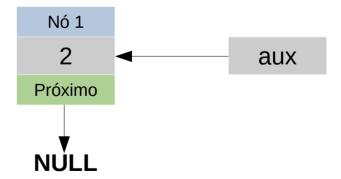


```
Se (tamanaho = 1){
    aux = inicio
    Inicio = null
    Fim = null
    delete aux
    tamanho--
}
tamanho = 1
```

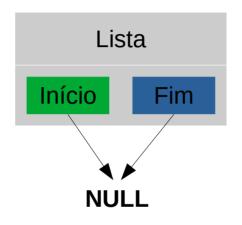


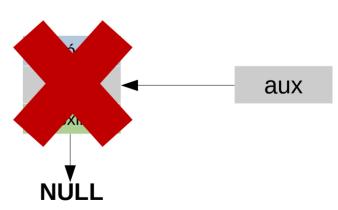
```
Se (tamanaho = 1){
    aux = inicio
    Inicio = null
    Fim = null
    delete aux
    tamanho--
}
tamanho = 1
```



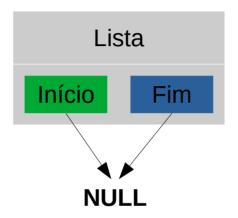


```
Se (tamanaho = 1){
    aux = inicio
    Inicio = null
    Fim = null
    delete aux
    tamanho--
}
tamanho = 1
```



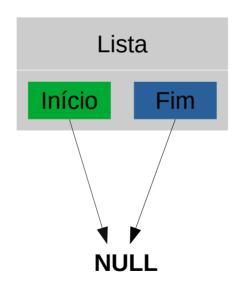


```
Se (tamanaho = 1){
    aux = inicio
    Inicio = null
    Fim = null
    delete aux
    tamanho--
}
tamanho = 0
```



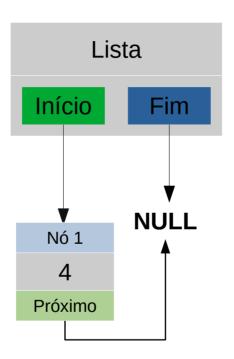
inserirEmListaVazia(valor):

```
novo ← criar_noh(valor);
inicio ← novo;
fim ← novo;
tamanho ← 1;
tamanho = 0
```

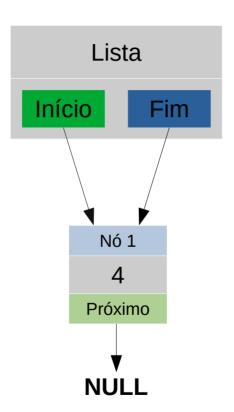




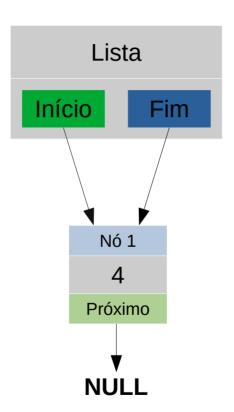
```
inserirEmListaVazia(valor):
novo ← criar_noh(valor);
inicio ← novo;
fim ← novo;
tamanho ← 1;
tamanho = 0
```



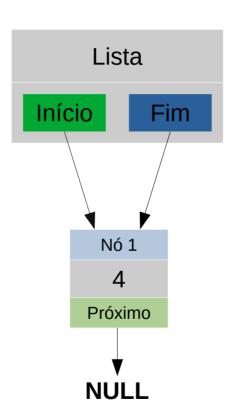
```
inserirEmListaVazia(valor):
novo ← criar_noh(valor);
inicio ← novo;
fim ← novo;
tamanho ← 1;
tamanho = 0
```

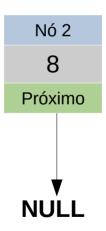


```
inserirEmListaVazia(valor):
novo ← criar_noh(valor);
inicio ← novo;
fim ← novo;
tamanho ← 1;
tamanho = 1
```



```
novo ← criar noh(valor);
aux ← inicio;
anterior ← NULL
enquanto ((auxiliar != NULL) E
(auxiliar.dado < valor)){</pre>
     anterior ← auxiliar;
     aux = aux.proximo;
novo.proximo = aux;
Se (anterior == NULL){
     inicio = novo;
}se nao{
     anteiror.proximo = novo;
tamanho++;
Se (aux = NULL){
     fim = novo;
```



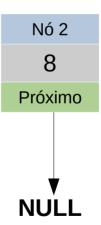


```
novo ← criar noh(valor);
aux ← inicio;
                                               Lista
anterior ← NULL
enquanto ((auxiliar != NULL) E
(auxiliar.dado < valor)){</pre>
                                         Início
                                                     Fim
     anterior ← auxiliar;
     aux = aux.proximo;
                                                                              Nó 2
novo.proximo = aux;
Se (anterior == NULL){
     inicio = novo;
                                                Nó 1
                                                                             Próximo
}se nao{
     anteiror.proximo = novo;
                                  aux
                                              Próximo
tamanho++;
Se (aux = NULL){
     fim = novo:
                                              NULL
```

```
novo ← criar noh(valor);
aux ← inicio;
                                               Lista
anterior ← NULL
enquanto ((auxiliar != NULL) E
(auxiliar.dado < valor)){</pre>
                                         Início
                                                     Fim
     anterior ← auxiliar:
     aux = aux.proximo;
                                                                              Nó 2
novo.proximo = aux;
Se (anterior == NULL){
     inicio = novo;
                                                Nó 1
                                                                             Próximo
}se nao{
     anteiror.proximo = novo;
                                  aux
                                              Próximo
tamanho++;
Se (aux = NULL){
     fim = novo:
                                                                             NULL
                                                           anterior
```

```
novo ← criar noh(valor);
aux ← inicio;
                                               Lista
anterior ← NULL
enquanto ((auxiliar != NULL) E
(auxiliar.dado < valor)){</pre>
                                         Início
                                                     Fim
     anterior ← auxiliar:
     aux = aux.proximo;
                                                                              Nó 2
novo.proximo = aux;
Se (anterior == NULL){
     inicio = novo;
                                                Nó 1
                                                                             Próximo
}se nao{
     anteiror.proximo = novo;
                                  aux
                                              Próximo
tamanho++;
                                 4 < 8
Se (aux = NULL){
     fim = novo;
                                                                             NULL
                                                           anterior
```

```
novo ← criar noh(valor);
aux ← inicio;
                                               Lista
anterior ← NULL
enquanto ((auxiliar != NULL) E
(auxiliar.dado < valor)){</pre>
                                         Início
                                                     Fim
     anterior ← auxiliar:
     aux = aux.proximo;
novo.proximo = aux;
Se (anterior == NULL){
     inicio = novo;
                                                Nó 1
}se nao{
     anteiror.proximo = novo;
                                anterior
                                              Próximo
tamanho++;
Se (aux = NULL){
     fim = novo:
                                            ►NULL
                                  aux
tamanho = 1
                                aux=NULL
                             Sai do enquanto
```

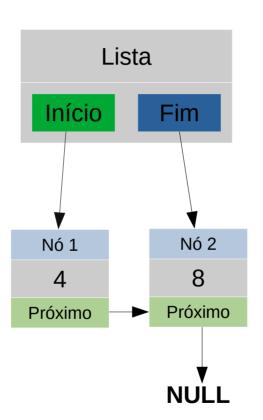


```
novo ← criar_noh(valor);
aux ← inicio;
                                              Lista
anterior ← NULL
enquanto ((auxiliar != NULL) E
(auxiliar.dado < valor)){</pre>
                                        Início
                                                     Fim
     anterior ← auxiliar:
     aux = aux.proximo;
novo.proximo = aux;
Se (anterior == NULL){
     inicio = novo;
                                               Nó 1
}se nao{
     anteiror.proximo = novo;
                               anterior
                                              Próximo
tamanho++;
Se (aux = NULL){
                                                              Nó 2
     fim = novo:
                                                               8
                                            NULL
                                  aux
tamanho = 1
                                                            Próximo
```

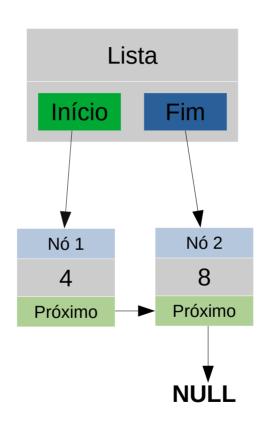
```
novo ← criar noh(valor);
aux ← inicio;
                                               Lista
anterior ← NULL
enquanto ((auxiliar != NULL) E
(auxiliar.dado < valor)){</pre>
                                         Início
                                                      Fim
     anterior ← auxiliar;
     aux = aux.proximo;
novo.proximo = aux;
Se (anterior == NULL){
     inicio = novo;
                                                               Nó 2
                                                Nó 1
}se nao{
                                                                8
     anteiror.proximo = novo;
                                                             Próximo
                                               Próximo
tamanho++;
Se (aux = NULL){
     fim = novo;
                                  aux
```

```
novo ← criar noh(valor);
aux ← inicio;
                                               Lista
anterior ← NULL
enquanto ((auxiliar != NULL) E
(auxiliar.dado < valor)){</pre>
                                         Início
                                                      Fim
     anterior ← auxiliar;
     aux = aux.proximo;
novo.proximo = aux;
Se (anterior == NULL){
     inicio = novo;
                                                               Nó 2
                                                Nó 1
}se nao{
                                                                8
     anteiror.proximo = novo;
                                                             Próximo
                                              Próximo
tamanho++;
Se (aux = NULL){
     fim = novo;
                                  aux
```

```
novo ← criar noh(valor);
aux ← inicio;
anterior ← NULL
enquanto ((auxiliar != NULL) E
(auxiliar.dado < valor)){</pre>
     anterior ← auxiliar;
     aux = aux.proximo;
novo.proximo = aux;
Se (anterior == NULL){
     inicio = novo;
}se nao{
     anteiror.proximo = novo;
tamanho++;
Se (aux = NULL){
     fim = novo:
```



```
novo ← criar noh(valor);
aux ← inicio;
anterior ← NULL
enquanto ((auxiliar != NULL) E
(auxiliar.dado < valor)){</pre>
     anterior ← auxiliar;
     aux = aux.proximo;
novo.proximo = aux;
Se (anterior == NULL){
     inicio = novo;
}se nao{
     anteiror.proximo = novo;
tamanho++;
Se (aux = NULL){
     fim = novo;
```





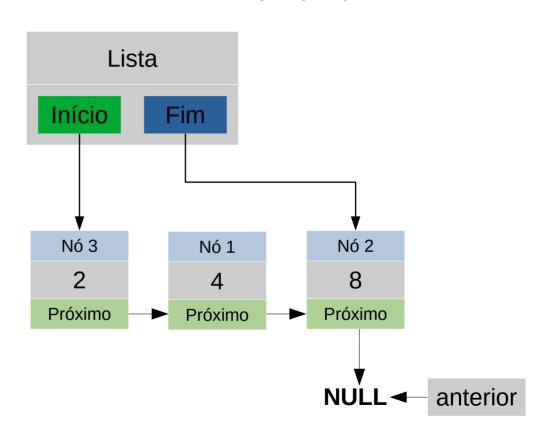
```
novo ← criar noh(valor);
aux ← inicio;
                                                      Lista
anterior ← NULL
enquanto ((auxiliar != NULL) E
(auxiliar.dado < valor)){</pre>
                                                Início
                                                             Fim
     anterior ← auxiliar;
     aux = aux.proximo;
                                                                                       Nó 3
novo.proximo = aux;
Se (anterior == NULL){
                                                                                      Próximo
     inicio = novo;
                                                               Nó 2
                                                Nó 1
}se nao{
     anteiror.proximo = novo;
                                  aux
                                                             Próximo
                                              Próximo
tamanho++;
Se (aux = NULL){
                                                                                      NULL
     fim = novo:
```

```
novo ← criar noh(valor);
aux ← inicio;
                                                      Lista
anterior ← NULL
enquanto ((auxiliar != NULL) E
(auxiliar.dado < valor)){</pre>
                                                Início
                                                            Fim
     anterior ← auxiliar:
     aux = aux.proximo;
                                                                                       Nó 3
novo.proximo = aux;
Se (anterior == NULL){
                                                                                     Próximo
     inicio = novo;
                                                              Nó 2
                                               Nó 1
}se nao{
     anteiror.proximo = novo;
                                  aux
                                                             Próximo
                                              Próximo
tamanho++;
Se (aux = NULL){
                                                                                     NULL
     fim = novo:
                                                anterior -
                                                           →NULL
tamanho = 2
```

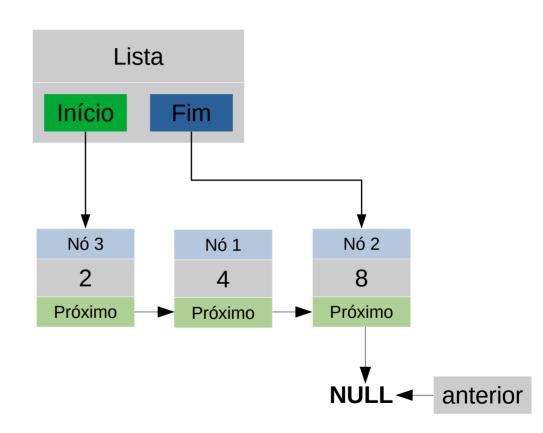
```
novo ← criar_noh(valor);
aux ← inicio;
                                                      Lista
anterior ← NULL
enquanto ((auxiliar != NULL) E
(auxiliar.dado < valor)){</pre>
                                                Início
                                                             Fim
     anterior ← auxiliar:
     aux = aux.proximo;
                                                                                        Nó 3
novo.proximo = aux;
Se (anterior == NULL){
                                                                                      Próximo
     inicio = novo;
                                                               Nó 2
                                                Nó 1
}se nao{
                                                                8
     anteiror.proximo = novo;
                                  aux
                                                             Próximo
                                              Próximo
                                  4 > 2
tamanho++;
Se (aux = NULL){
                                                                                      NULL
                               Não entra no
     fim = novo;
                                enquanto
                                                 anterior -
                                                            →NULL
tamanho = 2
```

```
novo ← criar noh(valor);
aux ← inicio;
                                                      Lista
anterior ← NULL
enquanto ((auxiliar != NULL) E
(auxiliar.dado < valor)){</pre>
                                                Início
                                                             Fim
     anterior ← auxiliar:
     aux = aux.proximo;
novo.proximo = aux;
Se (anterior == NULL){
     inicio = novo;
                                                               Nó 2
                                                Nó 1
}se nao{
     anteiror.proximo = novo;
                                  aux
                                                             Próximo
                                              Próximo
tamanho++;
Se (aux = NULL){
     fim = novo:
                                                                          anterior
                                                Nó3
tamanho = 2
                                              Próximo
```

```
novo ← criar noh(valor);
aux ← inicio;
anterior ← NULL
enquanto ((auxiliar != NULL) E
(auxiliar.dado < valor)){</pre>
     anterior ← auxiliar;
     aux = aux.proximo;
novo.proximo = aux;
Se (anterior == NULL){
     inicio = novo;
}se nao{
     anteiror.proximo = novo;
tamanho++;
Se (aux = NULL){
     fim = novo:
```



```
novo ← criar noh(valor);
aux ← inicio;
anterior ← NULL
enquanto ((auxiliar != NULL) E
(auxiliar.dado < valor)){</pre>
     anterior ← auxiliar:
     aux = aux.proximo;
novo.proximo = aux;
Se (anterior == NULL){
     inicio = novo;
}se nao{
     anteiror.proximo = novo;
tamanho++;
Se (aux = NULL){
     fim = novo:
```



```
novo ← criar noh(valor);
aux ← inicio;
anterior ← NULL
enquanto ((auxiliar != NULL) E
(auxiliar.dado < valor)){</pre>
     anterior ← auxiliar:
     aux = aux.proximo;
novo.proximo = aux;
Se (anterior == NULL){
     inicio = novo;
}se nao{
     anteiror.proximo = novo;
tamanho++;
Se (aux = NULL){
     fim = novo:
```

Inicio = novo;
e nao{
 anteiror.proximo = novo;

manho++;
e (aux = NULL){
 fim = novo;

Nó 3

Próximo

Próximo

Próximo

Nó 2

Romanho++;

Próximo

No 1

No 2

Romanho++;

Próximo

Próximo

NULL ← anterior

No 1

No 2

Autorio |

No 2

Autorio |

No 2

Autorio |

No 1

No 2

Autorio |

No 2

Autorio |

No 2

Autorio |

No 2

Autorio |

No 1

Autorio |

No 2

Autorio |

No 2

Autorio |

No 2

Autorio |

No 1

Autorio |

No 2

Autor

Início

Lista

Fim

Nó 4

Próximo

NULL

```
novo ← criar noh(valor);
                                                                                             Nó 4
aux ← inicio;
                                                       Lista
anterior ← NULL
enquanto ((auxiliar != NULL) E
                                                                                            Próximo
(auxiliar.dado < valor)){</pre>
                                                Início
                                                             Fim
     anterior ← auxiliar:
     aux = aux.proximo;
novo.proximo = aux;
                                                                                            NULL
Se (anterior == NULL){
     inicio = novo;
                                                  Nó 3
                                                                                Nó 2
                                                                 Nó 1
}se nao{
                                                                                 8
     anteiror.proximo = novo;
                                   aux
                                                Próximo
                                                                              Próximo
                                                               Próximo
tamanho++;
Se (aux = NULL){
     fim = novo:
                                                                                           anterior
```

```
novo ← criar noh(valor);
                                                                                             Nó 4
aux ← inicio;
                                                       Lista
anterior ← NULL
enquanto ((auxiliar != NULL) E
                                                                                            Próximo
(auxiliar.dado < valor)){</pre>
                                                 Início
                                                             Fim
     anterior ← auxiliar:
     aux = aux.proximo;
novo.proximo = aux;
                                                                                            NULL
Se (anterior == NULL){
     inicio = novo;
                                                  Nó 3
                                                                                Nó 2
                                                                 Nó 1
}se nao{
                                                                                 8
     anteiror.proximo = novo;
                                   aux
                                                Próximo
                                                                              Próximo
                                                               Próximo
tamanho++;
Se (aux = NULL){
     fim = novo:
                                                                                           anterior
```

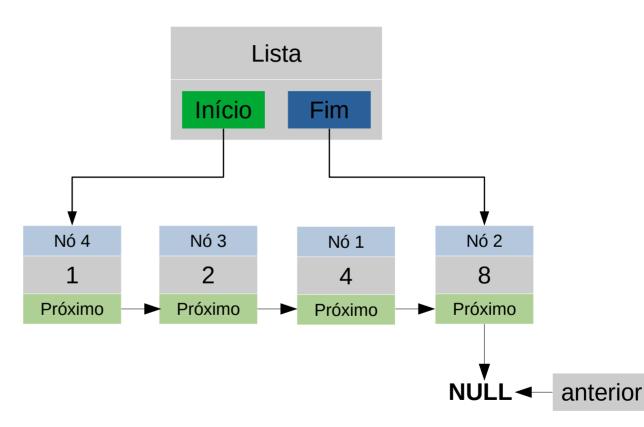
```
novo ← criar noh(valor);
                                                                                              Nó 4
aux ← inicio;
                                                       Lista
anterior ← NULL
enquanto ((auxiliar != NULL) E
                                                                                            Próximo
(auxiliar.dado < valor)){</pre>
                                                 Início
                                                              Fim
     anterior ← auxiliar:
     aux = aux.proximo;
novo.proximo = aux;
                                                                                            NULL
Se (anterior == NULL){
     inicio = novo;
                                                  Nó 3
                                                                                Nó 2
                                                                 Nó 1
}se nao{
                                                                                  8
     anteiror.proximo = novo;
                                    aux
                                   2 > 1
                                                 Próximo
                                                                               Próximo
                                                                Próximo
tamanho++;
                                Não entra no
Se (aux = NULL){
                                  enquanto
     fim = novo:
                                                                                           anterior
```

Passo 3. Inserir ordenado o elemento 4, 8, 2, 1

```
novo ← criar noh(valor);
aux ← inicio;
                                                       Lista
anterior ← NULL
enquanto ((auxiliar != NULL) E
(auxiliar.dado < valor)){</pre>
                                                 Início
                                                             Fim
     anterior ← auxiliar:
     aux = aux.proximo;
novo.proximo = aux;
Se (anterior == NULL){
     inicio = novo;
                                                  Nó 3
                                                                                Nó 2
                                                                 Nó 1
}se nao{
                                                                                 8
     anteiror.proximo = novo;
                                    aux
                                                Próximo
                                                                               Próximo
                                                               Próximo
tamanho++;
Se (aux = NULL){
     fim = novo:
                                                                                           anterior
                                                  Nó 4
tamanho = 3
                                                Próximo
```

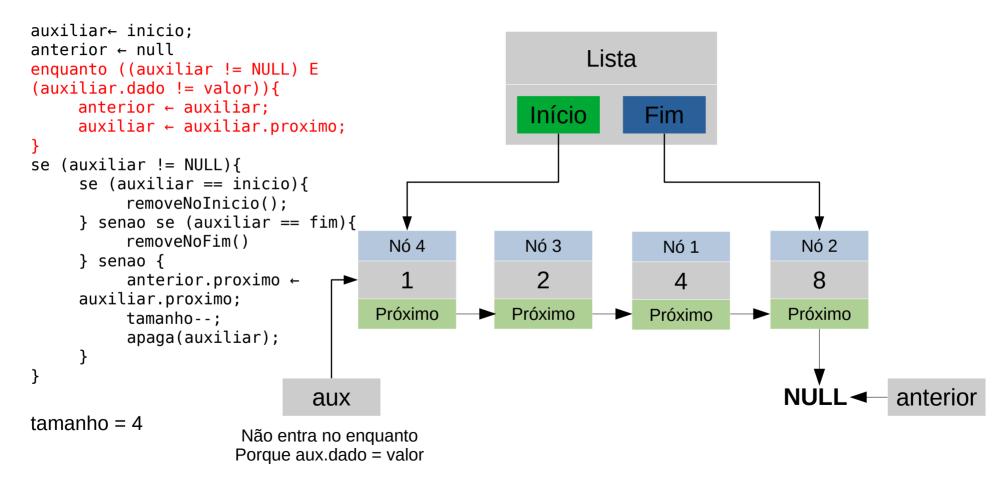
Passo 3. Inserir ordenado o elemento 4, 8, 2, 1

```
novo ← criar noh(valor);
aux ← inicio;
anterior ← NULL
enquanto ((auxiliar != NULL) E
(auxiliar.dado < valor)){</pre>
     anterior ← auxiliar:
     aux = aux.proximo;
novo.proximo = aux;
Se (anterior == NULL){
     inicio = novo;
}se nao{
     anteiror.proximo = novo;
tamanho++;
Se (aux = NULL){
     fim = novo;
```



tamanho = 4

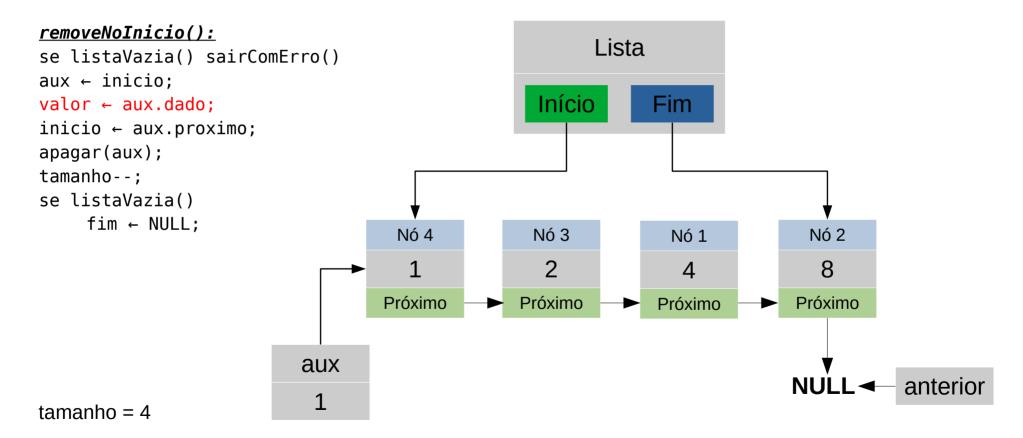
```
auxiliar← inicio;
anterior ← null
                                                           Lista
enquanto ((auxiliar != NULL) E
(auxiliar.dado != valor)){
     anterior ← auxiliar;
                                                     Início
                                                                  Fim
     auxiliar ← auxiliar.proximo;
  (auxiliar != NULL){
     se (auxiliar == inicio){
          removeNoInicio();
     } senao se (auxiliar == fim){
          removeNoFim()
                                      Nó 4
                                                     Nó3
                                                                                   Nó 2
                                                                    Nó 1
     } senao {
                                                                                    8
          anterior.proximo ←
     auxiliar.proximo;
                                     Próximo
                                                   Próximo
                                                                                 Próximo
                                                                  Próximo
          tamanho--:
          apaga(auxiliar);
                             aux
                                                                                             anterior
tamanho = 4
```

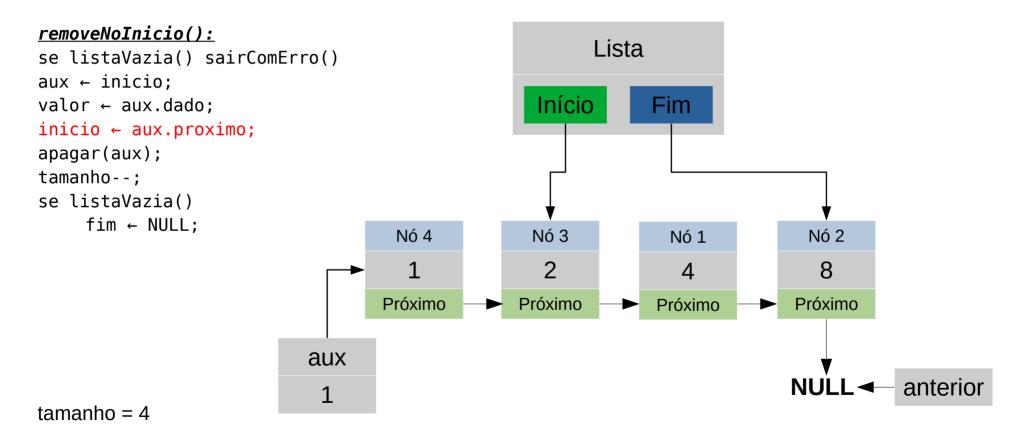


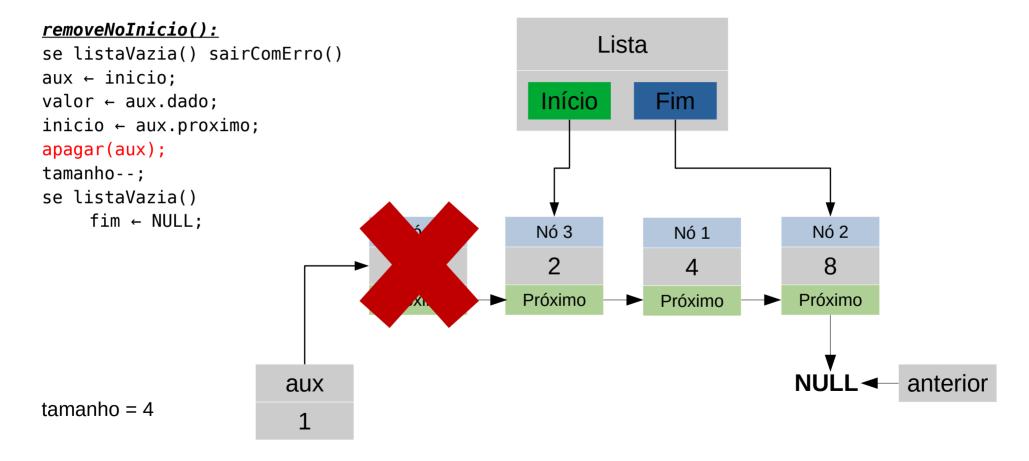
```
auxiliar← inicio;
anterior ← null
                                                            Lista
enquanto ((auxiliar != NULL) E
(auxiliar.dado != valor)){
     anterior ← auxiliar;
                                                      Início
                                                                  Fim
     auxiliar ← auxiliar.proximo;
se (auxiliar != NULL){
     se (auxiliar == inicio){
          removeNoInicio();
     } senao se (auxiliar == fim){
          removeNoFim()
                                      Nó 4
                                                     Nó 3
                                                                                   Nó 2
                                                                    Nó 1
     } senao {
                                                                                    8
          anterior.proximo ←
     auxiliar.proximo;
                                     Próximo
                                                    Próximo
                                                                                 Próximo
                                                                   Próximo
          tamanho--:
          apaga(auxiliar);
                                                                                             anterior
                              aux
tamanho = 4
```

Chama função removeNoInicio();

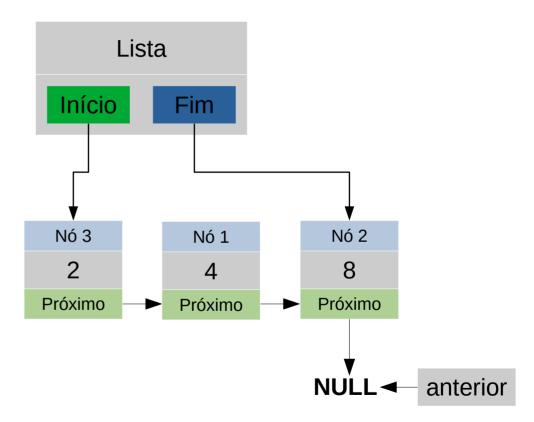
```
removeNoInicio():
                                                         Lista
se listaVazia() sairComErro()
aux ← inicio;
                                                   Início
                                                               Fim
valor ← aux.dado;
inicio ← aux.proximo;
apagar(aux);
tamanho--;
se listaVazia()
     fim ← NULL;
                                    Nó 4
                                                  Nó3
                                                                Nó 1
                                                                              Nó 2
                                                                                8
                                   Próximo
                                                                             Próximo
                                                 Próximo
                                                               Próximo
                                                                                        anterior
                            aux
tamanho = 4
```





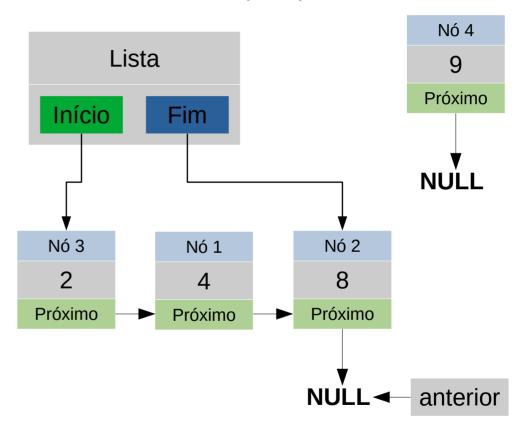


```
removeNoInicio():
se listaVazia() sairComErro()
aux ← inicio;
valor ← aux.dado;
inicio ← aux.proximo;
apagar(aux);
tamanho--;
se listaVazia()
    fim ← NULL;
```



```
novo ← criar noh(valor);
aux ← inicio;
anterior ← NULL
enquanto ((auxiliar != NULL) E
(auxiliar.dado < valor)){</pre>
     anterior ← auxiliar:
     aux = aux.proximo;
novo.proximo = aux;
Se (anterior == NULL){
     inicio = novo;
}se nao{
     anteiror.proximo = novo;
tamanho++;
Se (aux = NULL){
     fim = novo:
```

tamanho = 3



```
novo ← criar noh(valor);
                                                                                                Nó 4
aux ← inicio;
                                                            Lista
anterior ← NULL
enquanto ((auxiliar != NULL) E
                                                                                               Próximo
(auxiliar.dado < valor)){</pre>
                                                      Início
                                                                   Fim
     anterior ← auxiliar:
     aux = aux.proximo;
                                                                                              NULL
novo.proximo = aux;
Se (anterior == NULL){
     inicio = novo;
                                                      Nó3
                                                                                    Nó 2
                                                                    Nó 1
}se nao{
                                                                                     8
     anteiror.proximo = novo;
                                      aux
                                                    Próximo
                                                                                  Próximo
                                                                   Próximo
tamanho++;
Se (aux = NULL){
     fim = novo:
                                                                                              anterior
tamanho = 3
```

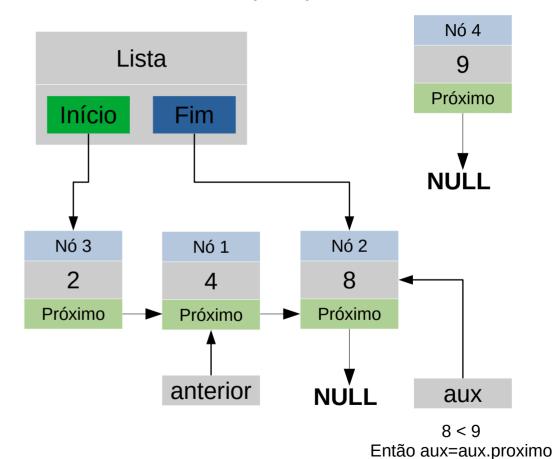
```
novo ← criar noh(valor);
                                                                                                Nó 4
aux ← inicio;
                                                            Lista
anterior ← NULL
enquanto ((auxiliar != NULL) E
                                                                                               Próximo
(auxiliar.dado < valor)){</pre>
                                                      Início
                                                                   Fim
     anterior ← auxiliar:
     aux = aux.proximo;
                                                                                              NULL
novo.proximo = aux;
Se (anterior == NULL){
     inicio = novo;
                                                      Nó3
                                                                                    Nó 2
                                                                     Nó 1
}se nao{
                                                                                     8
     anteiror.proximo = novo;
                                      aux
                                                    Próximo
                                                                                  Próximo
                                                                   Próximo
tamanho++;
Se (aux = NULL){
     fim = novo:
                                                                                              anterior
tamanho = 3
```

tamanho = 3

```
novo ← criar_noh(valor);
                                                                                                  Nó 4
aux ← inicio;
                                                             Lista
anterior ← NULL
enquanto ((auxiliar != NULL) E
                                                                                                Próximo
(auxiliar.dado < valor)){</pre>
                                                       Início
                                                                    Fim
     anterior ← auxiliar:
     aux = aux.proximo;
novo.proximo = aux;
Se (anterior == NULL){
     inicio = novo;
                                                      Nó3
                                                                                     Nó 2
                                                                      Nó 1
}se nao{
                                                                                      8
     anteiror.proximo = novo;
                                       aux
                                                     Próximo
                                                                                   Próximo
                                                                    Próximo
                                       2 < 9
tamanho++;
Se (aux = NULL){
                               Então aux=aux.proximo
                                   E anterior=aux
     fim = novo;
                                                                                               anterior
```

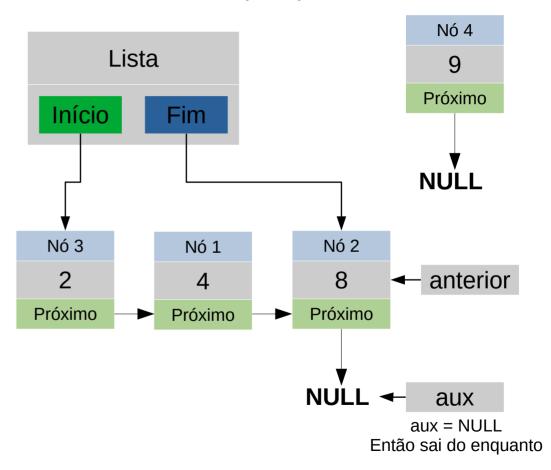
```
novo ← criar noh(valor);
                                                                                                 Nó 4
aux ← inicio;
                                                             Lista
anterior ← NULL
enquanto ((auxiliar != NULL) E
                                                                                               Próximo
(auxiliar.dado < valor)){</pre>
                                                      Início
                                                                   Fim
     anterior ← auxiliar:
     aux = aux.proximo;
                                                                                               NULL
novo.proximo = aux;
Se (anterior == NULL){
     inicio = novo;
                                                      Nó3
                                                                                    Nó 2
                                                                     Nó 1
}se nao{
                                                                                     8
     anteiror.proximo = novo;
                                    anterior
                                                    Próximo
                                                                                   Próximo
                                                                   Próximo
tamanho++;
Se (aux = NULL){
     fim = novo:
                                                                     aux
                                                                                  NULL
                                                                     4 < 9
tamanho = 3
                                                             Então aux=aux.proximo
                                                                 E anterior=aux
```

```
novo ← criar noh(valor);
aux ← inicio;
anterior ← NULL
enquanto ((auxiliar != NULL) E
(auxiliar.dado < valor)){</pre>
     anterior ← auxiliar:
     aux = aux.proximo;
novo.proximo = aux;
Se (anterior == NULL){
     inicio = novo;
}se nao{
     anteiror.proximo = novo;
tamanho++;
Se (aux = NULL){
     fim = novo:
tamanho = 3
```

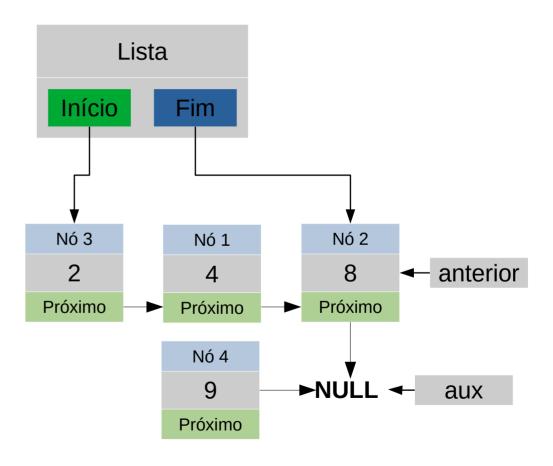


E anterior=aux

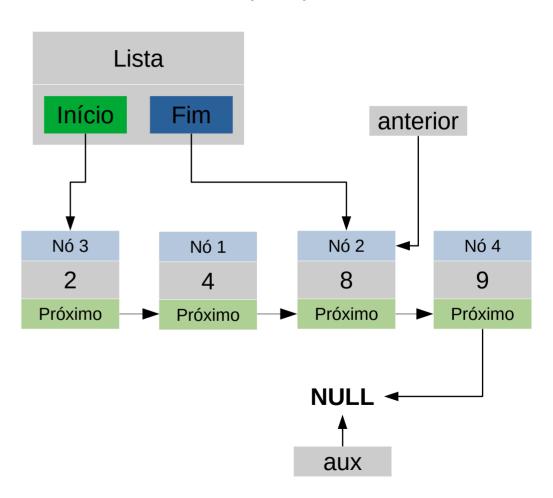
```
novo ← criar noh(valor);
aux ← inicio;
anterior ← NULL
enquanto ((auxiliar != NULL) E
(auxiliar.dado < valor)){</pre>
     anterior ← auxiliar:
     aux = aux.proximo;
novo.proximo = aux;
Se (anterior == NULL){
     inicio = novo;
}se nao{
     anteiror.proximo = novo;
tamanho++;
Se (aux = NULL){
     fim = novo:
tamanho = 3
```



```
novo ← criar noh(valor);
aux ← inicio;
anterior ← NULL
enquanto ((auxiliar != NULL) E
(auxiliar.dado < valor)){</pre>
     anterior ← auxiliar:
     aux = aux.proximo;
novo.proximo = aux;
Se (anterior == NULL){
     inicio = novo;
}se nao{
     anteiror.proximo = novo;
tamanho++;
Se (aux = NULL){
     fim = novo:
tamanho = 3
```



```
novo ← criar noh(valor);
aux ← inicio;
anterior ← NULL
enquanto ((auxiliar != NULL) E
(auxiliar.dado < valor)){</pre>
     anterior ← auxiliar:
     aux = aux.proximo;
novo.proximo = aux;
Se (anterior == NULL){
     inicio = novo;
}se nao{
     anteiror.proximo = novo;
tamanho++;
Se (aux = NULL){
     fim = novo:
tamanho = 3
```



```
novo ← criar noh(valor);
aux ← inicio;
anterior ← NULL
enquanto ((auxiliar != NULL) E
(auxiliar.dado < valor)){</pre>
     anterior ← auxiliar:
     aux = aux.proximo;
novo.proximo = aux;
Se (anterior == NULL){
     inicio = novo;
}se nao{
     anteiror.proximo = novo;
tamanho++;
Se (aux = NULL){
     fim = novo:
tamanho = 4
```

manho++;
(aux = NULL) {
fim = novo;

manho = 4

Lista

Fim

Nó 1

anterior

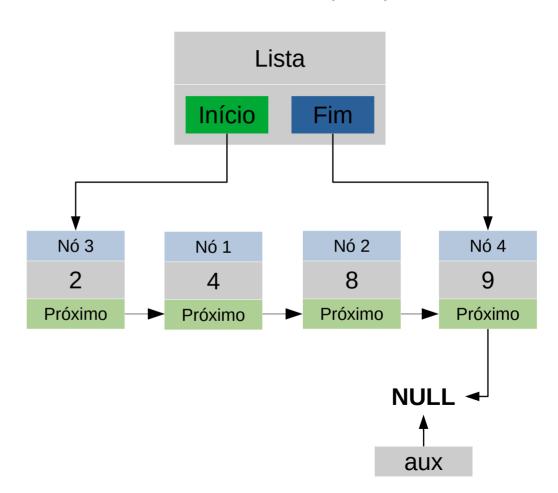
Nó 4

Nó 2

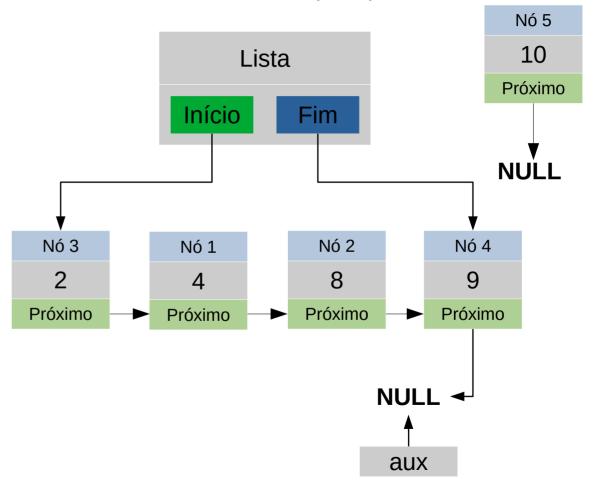
aux

Início

```
novo ← criar noh(valor);
aux ← inicio;
anterior ← NULL
enquanto ((auxiliar != NULL) E
(auxiliar.dado < valor)){</pre>
     anterior ← auxiliar:
     aux = aux.proximo;
novo.proximo = aux;
Se (anterior == NULL){
     inicio = novo;
}se nao{
     anteiror.proximo = novo;
tamanho++;
Se (aux = NULL){
     fim = novo:
tamanho = 4
```



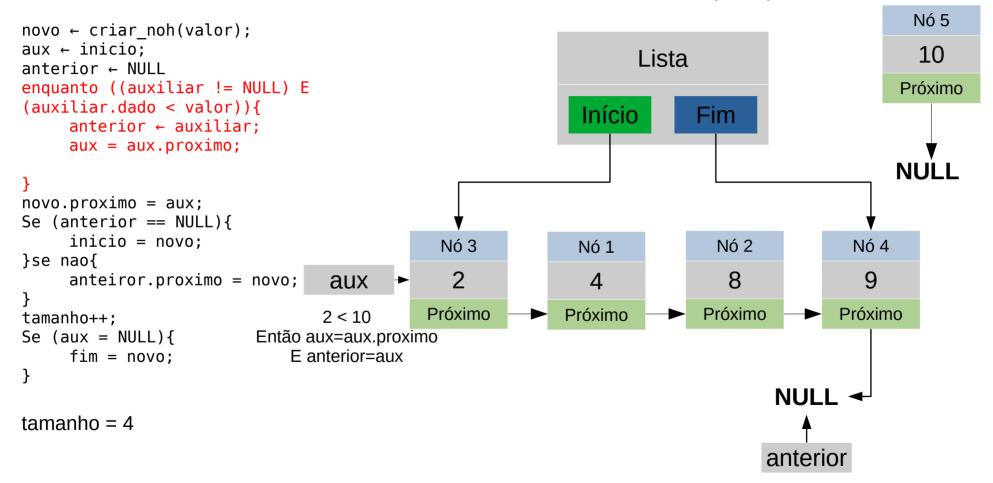
```
novo ← criar noh(valor);
aux ← inicio;
anterior ← NULL
enquanto ((auxiliar != NULL) E
(auxiliar.dado < valor)){</pre>
     anterior ← auxiliar:
     aux = aux.proximo;
novo.proximo = aux;
Se (anterior == NULL){
     inicio = novo;
}se nao{
     anteiror.proximo = novo;
tamanho++;
Se (aux = NULL){
     fim = novo:
tamanho = 4
```



tamanho = 4

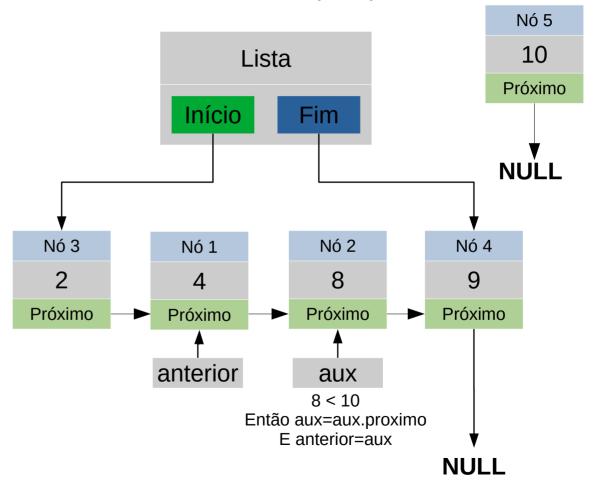
```
Nó 5
novo ← criar noh(valor);
aux ← inicio;
                                                                                                 10
                                                                   Lista
anterior ← NULL
enquanto ((auxiliar != NULL) E
                                                                                               Próximo
(auxiliar.dado < valor)){</pre>
                                                            Início
                                                                         Fim
     anterior ← auxiliar:
     aux = aux.proximo;
                                                                                               NULL
novo.proximo = aux;
Se (anterior == NULL){
     inicio = novo;
                                             Nó3
                                                                           Nó 2
                                                                                          Nó 4
                                                            Nó 1
}se nao{
     anteiror.proximo = novo;
                                 aux
                                            Próximo
                                                                          Próximo
                                                                                         Próximo
                                                           Próximo
tamanho++;
Se (aux = NULL){
     fim = novo:
```

```
Nó 5
novo ← criar_noh(valor);
aux ← inicio;
                                                                                                10
                                                                  Lista
anterior ← NULL
enquanto ((auxiliar != NULL) E
                                                                                              Próximo
(auxiliar.dado < valor)){</pre>
                                                            Início
                                                                         Fim
     anterior ← auxiliar:
     aux = aux.proximo;
                                                                                              NULL
novo.proximo = aux;
Se (anterior == NULL){
     inicio = novo;
                                             Nó3
                                                                           Nó 2
                                                                                         Nó 4
                                                            Nó 1
}se nao{
     anteiror.proximo = novo;
                                 aux
                                           Próximo
                                                                         Próximo
                                                                                        Próximo
                                                          Próximo
tamanho++;
Se (aux = NULL){
     fim = novo:
                                                                                 NULL
tamanho = 4
```

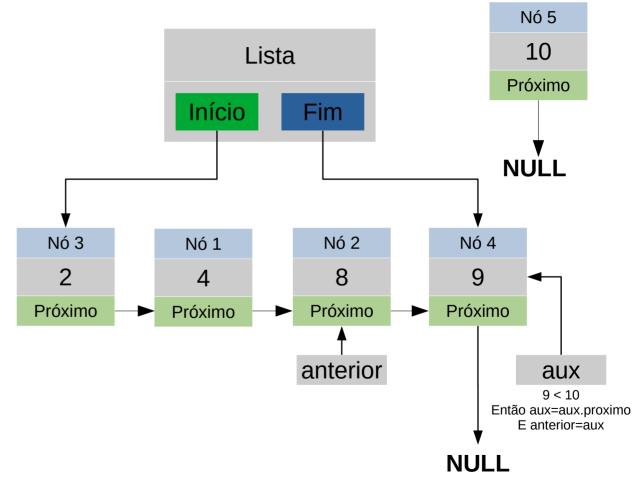


```
Nó 5
novo ← criar_noh(valor);
aux ← inicio;
                                                                                                  10
                                                                   Lista
anterior ← NULL
enquanto ((auxiliar != NULL) E
                                                                                                Próximo
(auxiliar.dado < valor)){</pre>
                                                             Início
                                                                          Fim
     anterior ← auxiliar:
     aux = aux.proximo;
                                                                                               NULL
novo.proximo = aux;
Se (anterior == NULL){
     inicio = novo;
                                             Nó3
                                                            Nó 1
                                                                            Nó 2
                                                                                           Nó 4
}se nao{
     anteiror.proximo = novo; anterior ►
                                            Próximo
                                                                          Próximo
                                                                                         Próximo
                                                           Próximo
tamanho++;
Se (aux = NULL){
     fim = novo:
                                                            aux
                                                                                  NULL
                                                            4 < 10
                                                    Então aux=aux.proximo
tamanho = 4
                                                        F anterior=aux
```

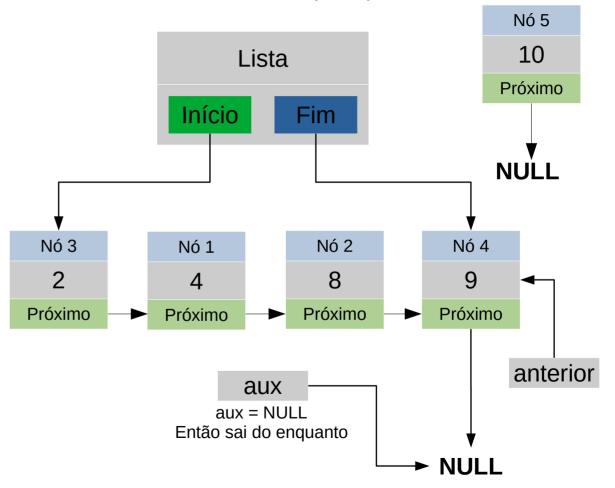
```
novo ← criar noh(valor);
aux ← inicio;
anterior ← NULL
enquanto ((auxiliar != NULL) E
(auxiliar.dado < valor)){</pre>
     anterior ← auxiliar:
     aux = aux.proximo;
novo.proximo = aux;
Se (anterior == NULL){
     inicio = novo;
}se nao{
     anteiror.proximo = novo;
tamanho++;
Se (aux = NULL){
     fim = novo:
tamanho = 4
```



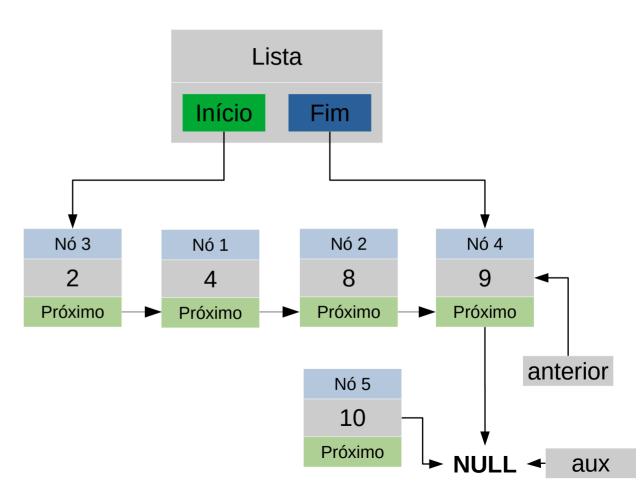
```
novo ← criar noh(valor);
aux ← inicio;
anterior ← NULL
enquanto ((auxiliar != NULL) E
(auxiliar.dado < valor)){</pre>
     anterior ← auxiliar:
     aux = aux.proximo;
novo.proximo = aux;
Se (anterior == NULL){
     inicio = novo;
}se nao{
     anteiror.proximo = novo;
tamanho++;
Se (aux = NULL){
     fim = novo:
tamanho = 4
```



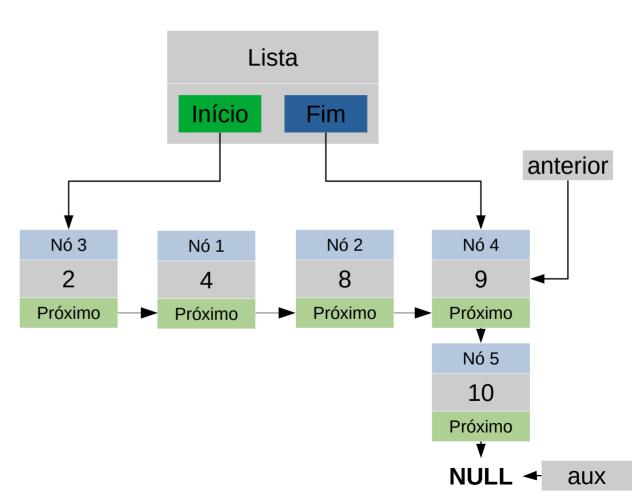
```
novo ← criar noh(valor);
aux ← inicio;
anterior ← NULL
enquanto ((auxiliar != NULL) E
(auxiliar.dado < valor)){</pre>
     anterior ← auxiliar:
     aux = aux.proximo;
novo.proximo = aux;
Se (anterior == NULL){
     inicio = novo;
}se nao{
     anteiror.proximo = novo;
tamanho++;
Se (aux = NULL){
     fim = novo:
tamanho = 4
```



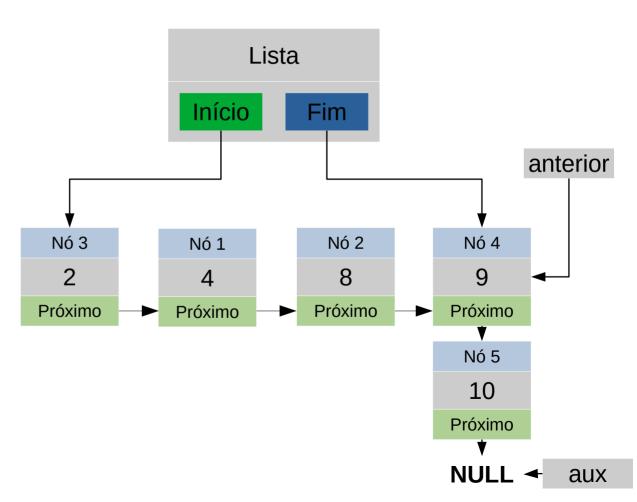
```
novo ← criar_noh(valor);
aux ← inicio;
anterior ← NULL
enquanto ((auxiliar != NULL) E
(auxiliar.dado < valor)){</pre>
     anterior ← auxiliar:
     aux = aux.proximo;
novo.proximo = aux;
Se (anterior == NULL){
     inicio = novo;
}se nao{
     anteiror.proximo = novo;
tamanho++;
Se (aux = NULL){
     fim = novo:
tamanho = 4
```



```
novo ← criar_noh(valor);
aux ← inicio;
anterior ← NULL
enquanto ((auxiliar != NULL) E
(auxiliar.dado < valor)){</pre>
     anterior ← auxiliar:
     aux = aux.proximo;
novo.proximo = aux;
Se (anterior == NULL){
     inicio = novo;
}se nao{
     anteiror.proximo = novo;
tamanho++;
Se (aux = NULL){
     fim = novo:
tamanho = 4
```



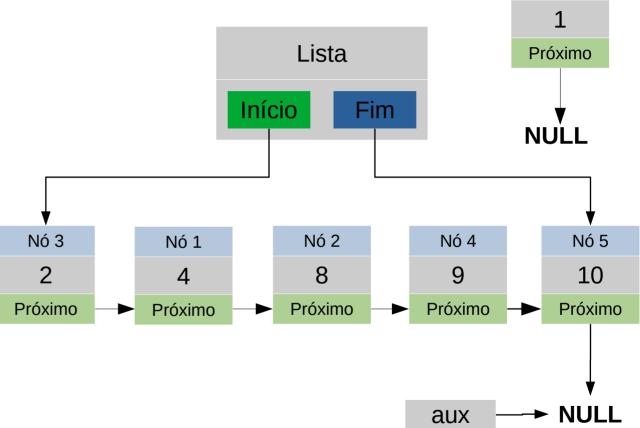
```
novo ← criar_noh(valor);
aux ← inicio;
anterior ← NULL
enquanto ((auxiliar != NULL) E
(auxiliar.dado < valor)){</pre>
     anterior ← auxiliar:
     aux = aux.proximo;
novo.proximo = aux;
Se (anterior == NULL){
     inicio = novo;
}se nao{
     anteiror.proximo = novo;
tamanho++;
Se (aux = NULL){
     fim = novo:
tamanho = 5
```



```
novo ← criar_noh(valor);
aux ← inicio;
                                                                       Lista
anterior ← NULL
enquanto ((auxiliar != NULL) E
(auxiliar.dado < valor)){</pre>
                                                                 Início
                                                                              Fim
     anterior ← auxiliar:
     aux = aux.proximo;
novo.proximo = aux;
Se (anterior == NULL){
     inicio = novo;
                                         Nó 3
                                                                        Nó 2
                                                                                       Nó 4
                                                                                                     Nó 5
                                                         Nó 1
}se nao{
                                                                         8
     anteiror.proximo = novo;
                                                                                                      10
                                        Próximo
                                                                      Próximo
                                                                                     Próximo
                                                                                                    Próximo
                                                       Próximo
tamanho++;
Se (aux = NULL){
     fim = novo:
                                                                                      aux
tamanho = 5
```

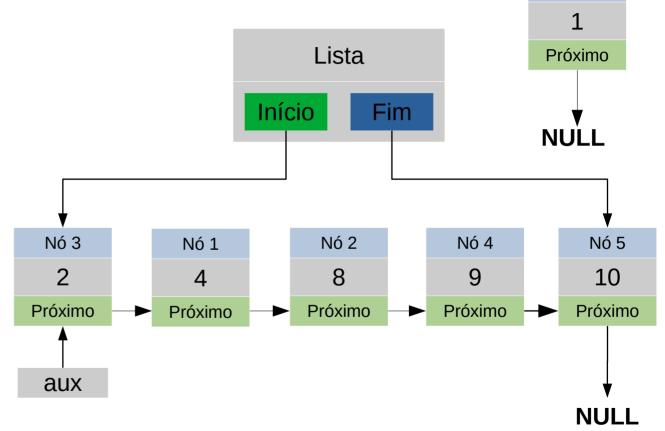
```
novo ← criar noh(valor);
aux ← inicio;
anterior ← NULL
enquanto ((auxiliar != NULL) E
(auxiliar.dado < valor)){</pre>
     anterior ← auxiliar:
     aux = aux.proximo;
novo.proximo = aux;
Se (anterior == NULL){
     inicio = novo;
}se nao{
     anteiror.proximo = novo;
tamanho++;
Se (aux = NULL){
     fim = novo;
```

tamanho = 5



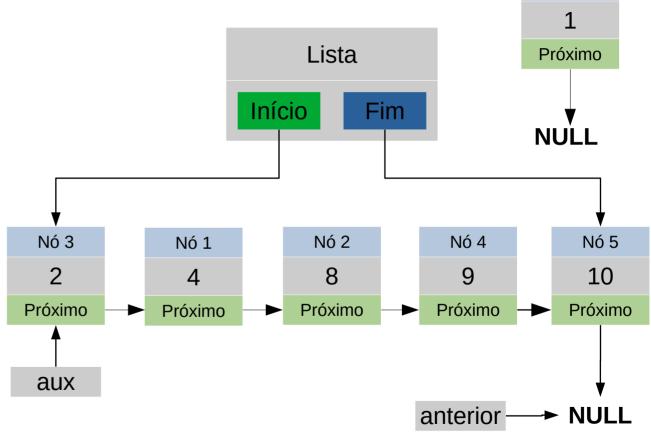
```
novo ← criar noh(valor);
aux ← inicio;
anterior ← NULL
enquanto ((auxiliar != NULL) E
(auxiliar.dado < valor)){</pre>
     anterior ← auxiliar:
     aux = aux.proximo;
novo.proximo = aux;
Se (anterior == NULL){
     inicio = novo;
}se nao{
     anteiror.proximo = novo;
tamanho++;
Se (aux = NULL){
     fim = novo:
```

tamanho = 5



```
novo ← criar noh(valor);
aux ← inicio;
anterior ← NULL
enquanto ((auxiliar != NULL) E
(auxiliar.dado < valor)){</pre>
     anterior ← auxiliar:
     aux = aux.proximo;
novo.proximo = aux;
Se (anterior == NULL){
     inicio = novo;
}se nao{
     anteiror.proximo = novo;
tamanho++;
Se (aux = NULL){
     fim = novo:
```

tamanho = 5



Passo 5. Inserir ordenado os elementos 9,10,1 Nó 6 novo ← criar noh(valor); aux ← inicio; Lista Próximo anterior ← NULL enquanto ((auxiliar != NULL) E (auxiliar.dado < valor)){</pre> Início Fim anterior ← auxiliar: NULL aux = aux.proximo; novo.proximo = aux; Se (anterior == NULL){ inicio = novo; Nó 3 Nó 2 Nó 4 Nó 5 Nó 1 }se nao{ 8 anteiror.proximo = novo; 10 Próximo Próximo Próximo Próximo Próximo tamanho++; Se (aux = NULL){ fim = novo:aux

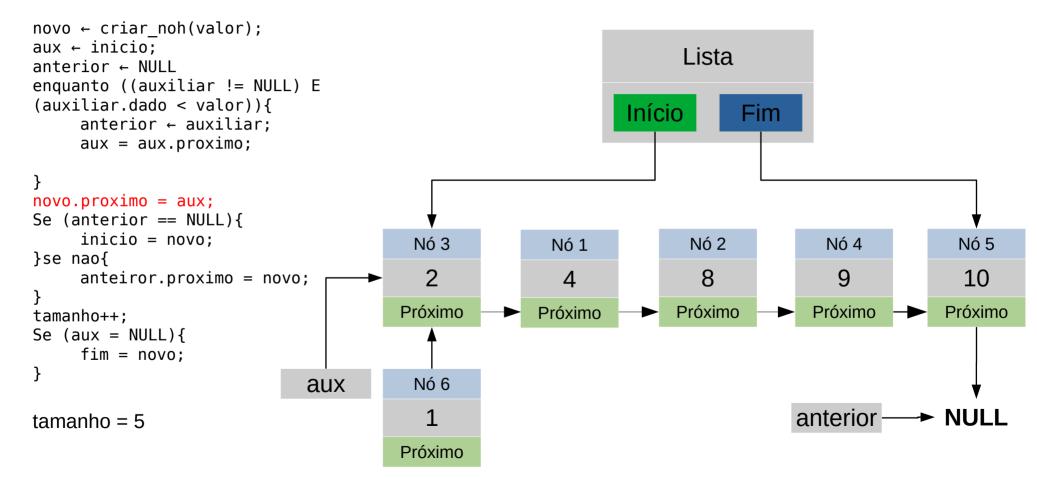
anterior

2 > 1

Então não entra no enquanto

tamanho = 5

Passo 5. Inserir ordenado os elementos 9,10,1

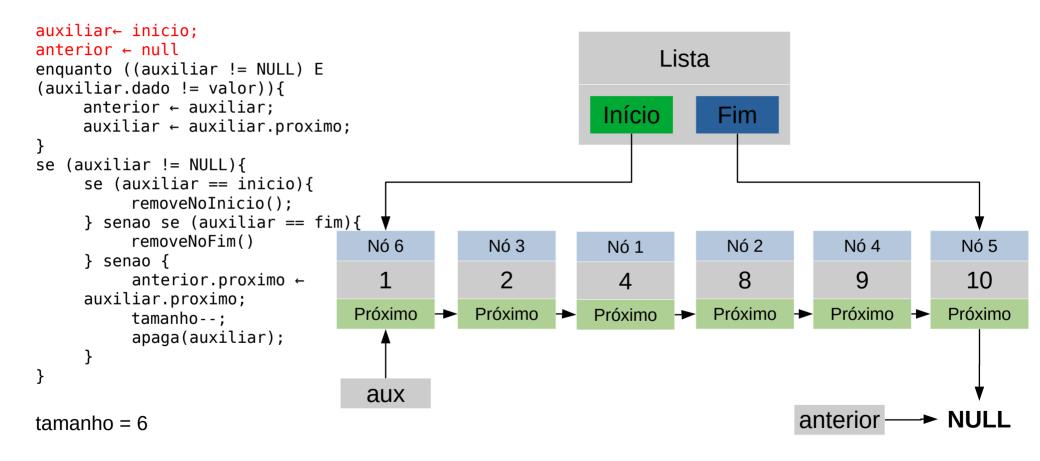


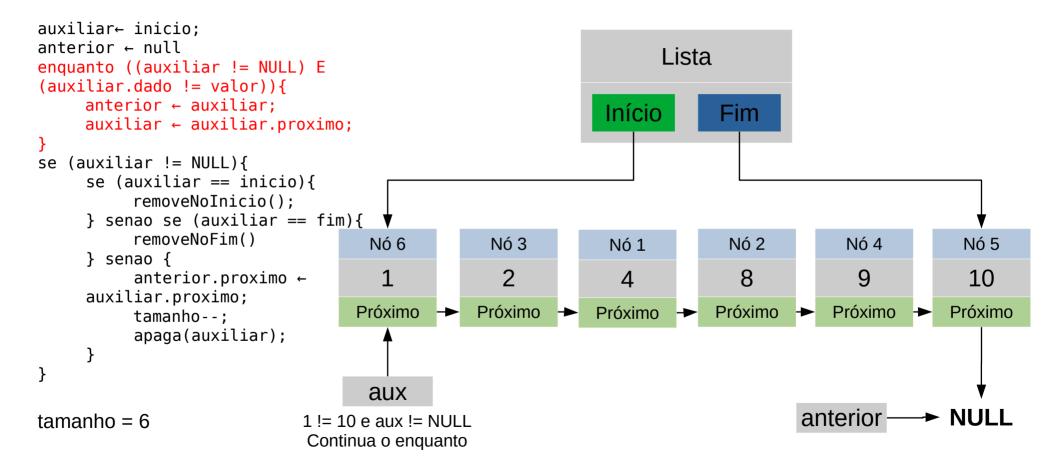
Passo 5. Inserir ordenado os elementos 9,10,1

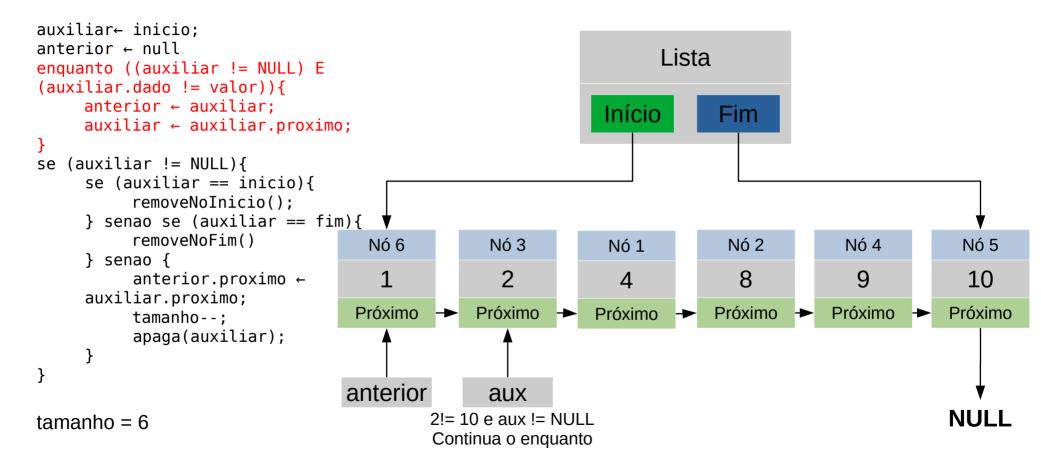
```
novo ← criar noh(valor);
aux ← inicio;
                                                                   Lista
anterior ← NULL
enquanto ((auxiliar != NULL) E
(auxiliar.dado < valor)){</pre>
                                                             Início
                                                                         Fim
     anterior ← auxiliar:
     aux = aux.proximo;
novo.proximo = aux;
Se (anterior == NULL){
    inicio = novo;
                                   Nó 6
                                                Nó3
                                                                          Nó 2
                                                                                       Nó 4
                                                                                                   Nó 5
                                                             Nó 1
}se nao{
     anteiror.proximo = novo;
                                                                                                    10
                                  Próximo -
                                               Próximo -
                                                                         Próximo →
                                                                                     Próximo →
                                                                                                  Próximo
                                                            Próximo
tamanho++;
Se (aux = NULL){
     fim = novo:
                                                aux
                                                                                  anterior
tamanho = 5
```

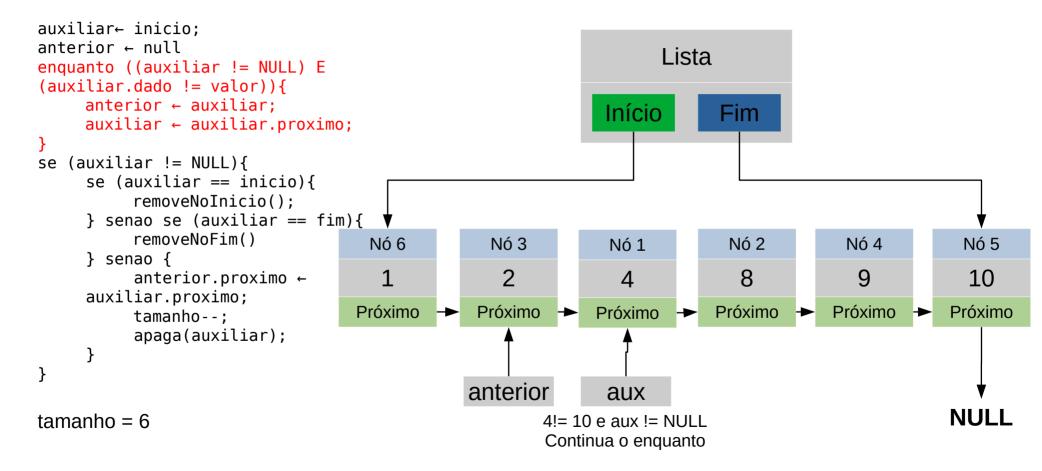
Passo 5. Inserir ordenado os elementos 9,10,1

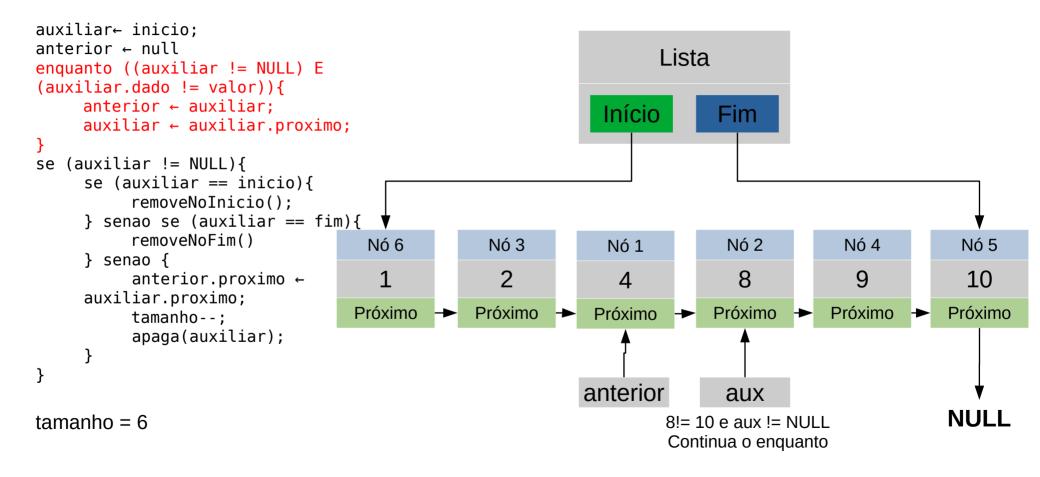
```
novo ← criar noh(valor);
aux ← inicio;
                                                                   Lista
anterior ← NULL
enquanto ((auxiliar != NULL) E
(auxiliar.dado < valor)){</pre>
                                                             Início
                                                                         Fim
     anterior ← auxiliar:
     aux = aux.proximo;
novo.proximo = aux;
Se (anterior == NULL){
     inicio = novo;
                                   Nó 6
                                                 Nó3
                                                                          Nó 2
                                                                                       Nó 4
                                                                                                    Nó 5
                                                             Nó 1
}se nao{
     anteiror.proximo = novo;
                                                                                                    10
                                  Próximo -
                                               Próximo -
                                                                         Próximo →
                                                                                     Próximo →
                                                                                                  Próximo
                                                            Próximo
tamanho++;
Se (aux = NULL){
     fim = novo:
                                                aux
                                                                                  anterior
tamanho = 6
```

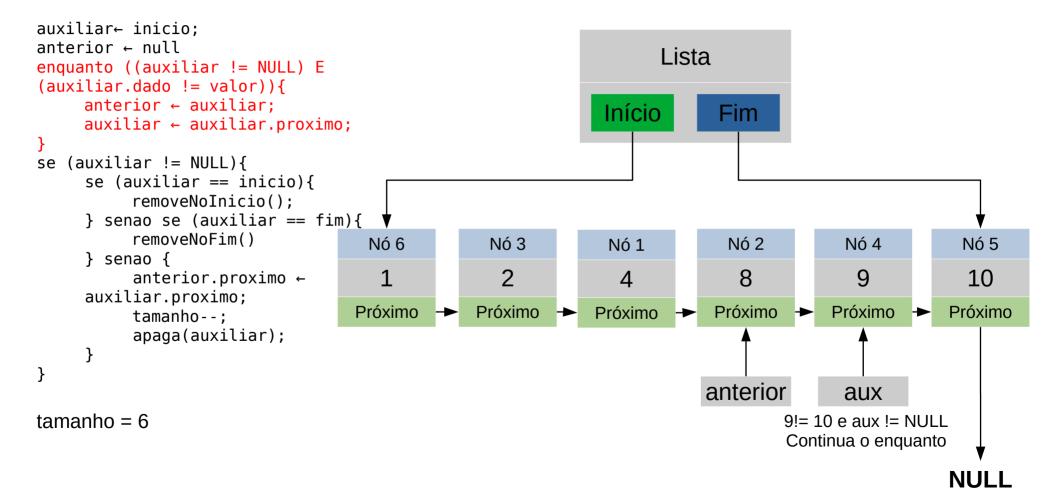




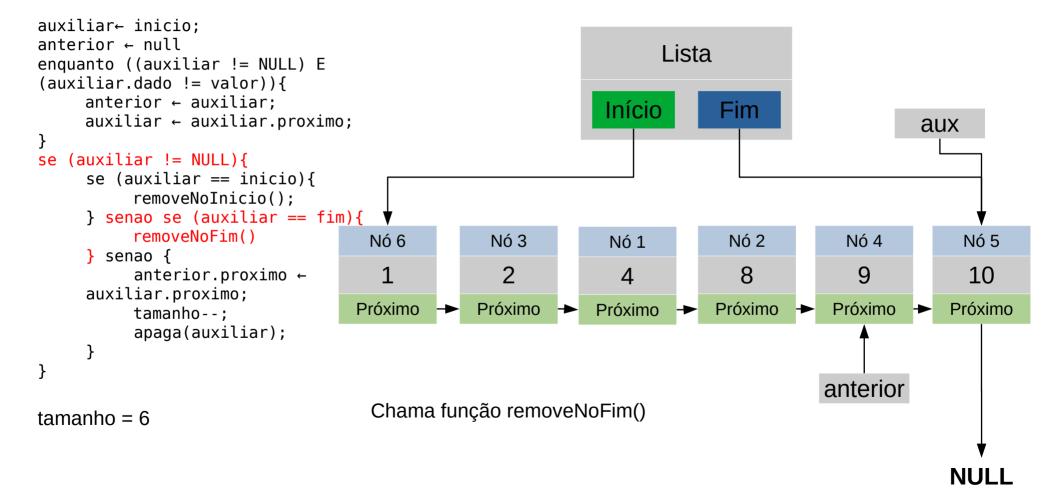


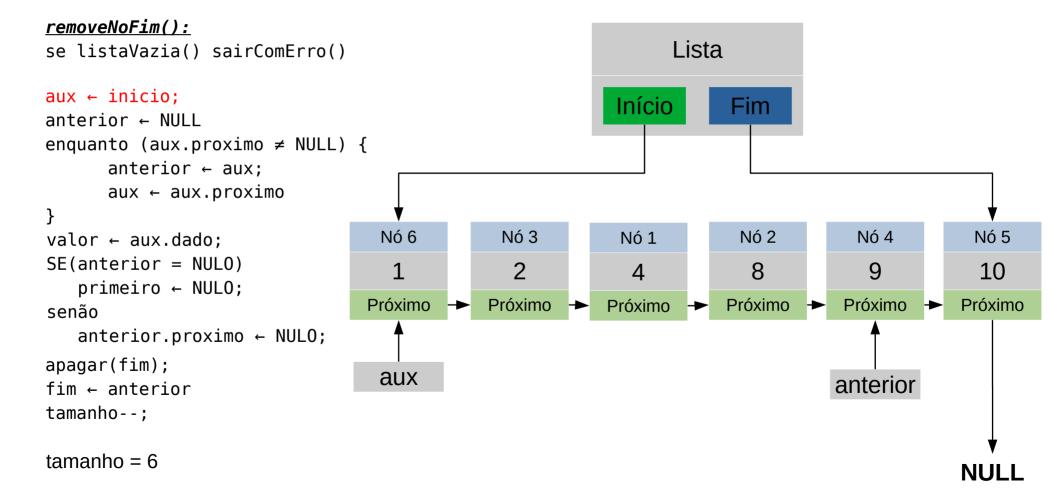


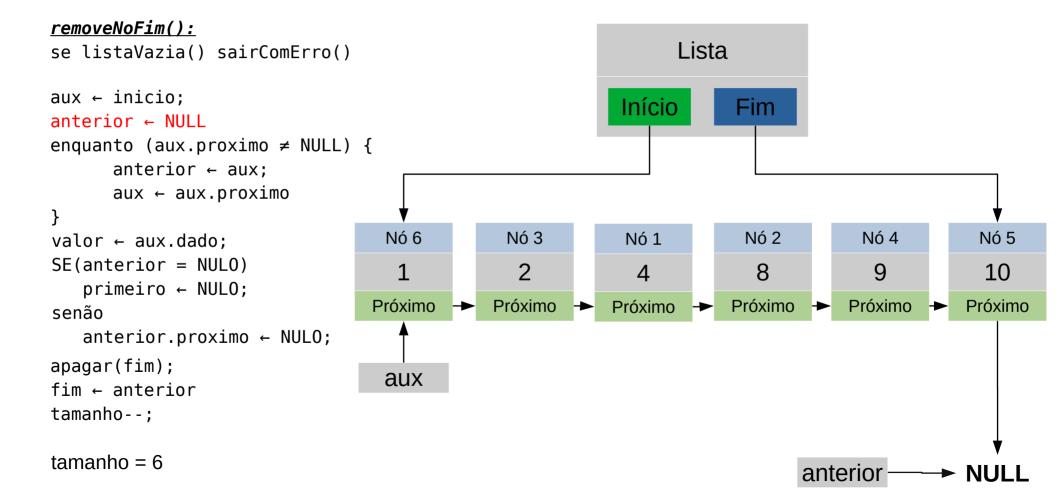


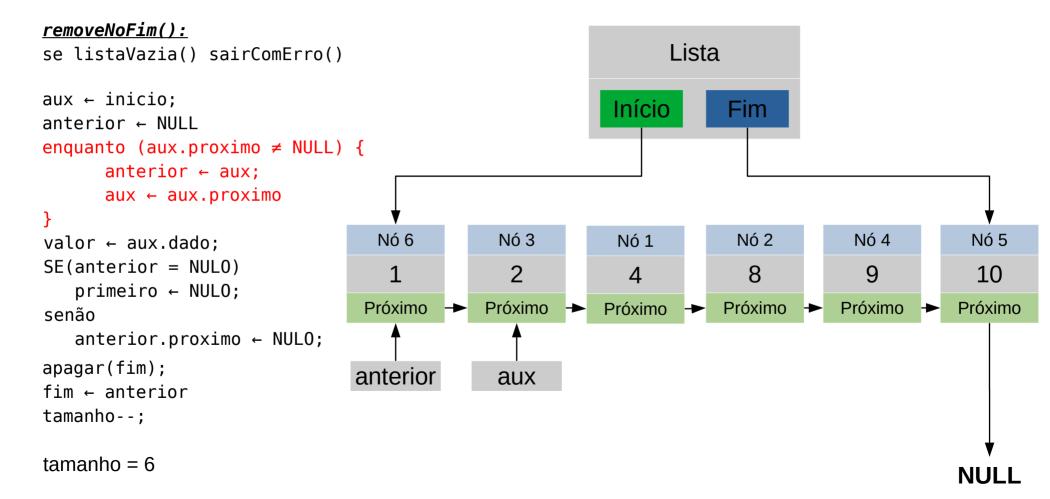


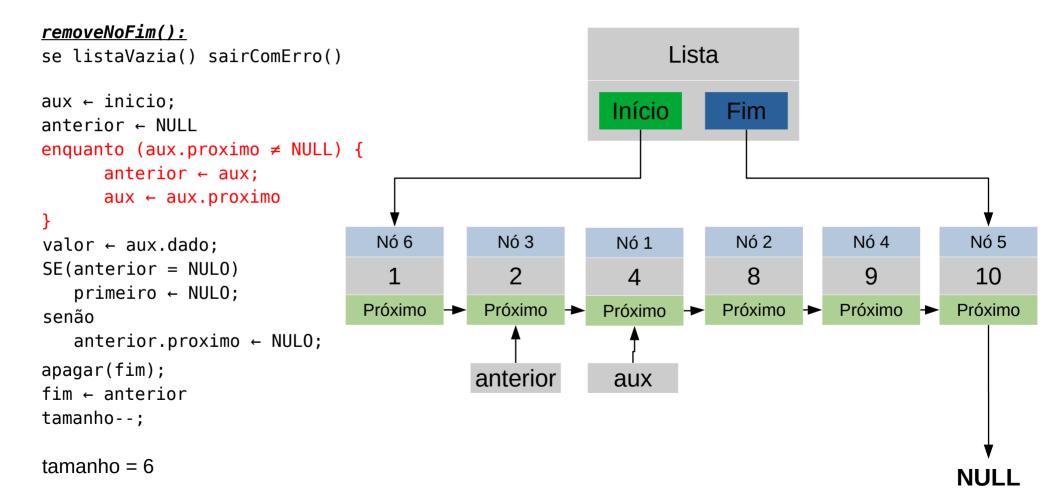
```
auxiliar← inicio;
anterior ← null
                                                                   Lista
enquanto ((auxiliar != NULL) E
                                                                                              10 = 10
(auxiliar.dado != valor)){
                                                                                          Sai do enquanto
     anterior ← auxiliar;
                                                             Início
                                                                         Fim
     auxiliar ← auxiliar.proximo;
                                                                                               aux
se (auxiliar != NULL){
     se (auxiliar == inicio){
          removeNoInicio();
     } senao se (auxiliar == fim){
          removeNoFim()
                                   Nó 6
                                                 Nó3
                                                                          Nó 2
                                                                                       Nó 4
                                                                                                    Nó 5
                                                             Nó 1
     } senao {
                                                                            8
          anterior.proximo ←
                                                                                                    10
     auxiliar.proximo;
                                  Próximo
                                               Próximo
                                                                         Próximo
                                                                                     Próximo →
                                                                                                  Próximo
                                                            Próximo
          tamanho--:
          apaga(auxiliar);
                                                                                    anterior
tamanho = 6
```

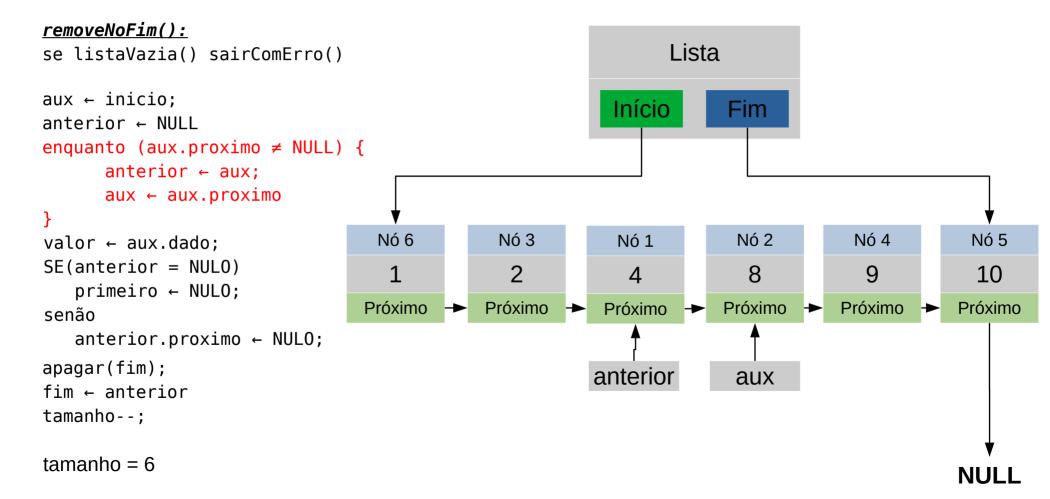


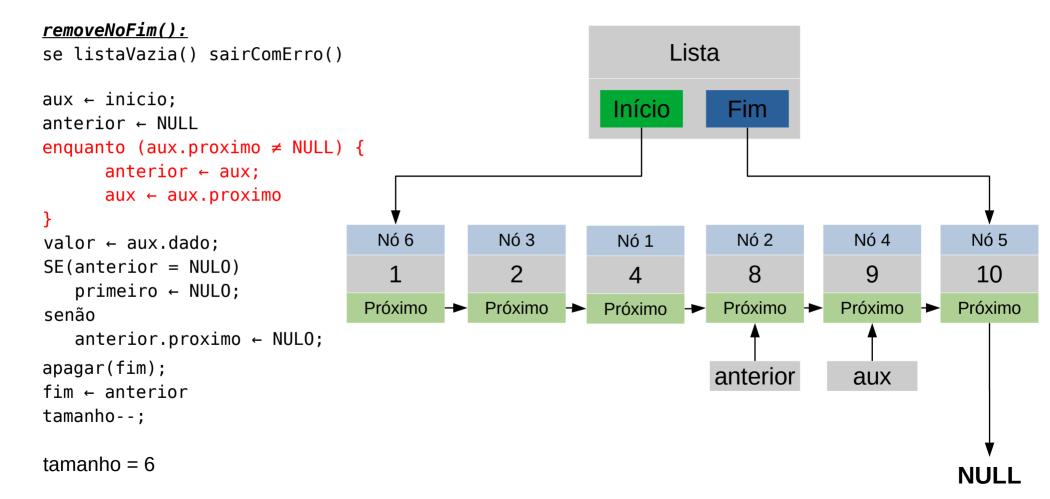


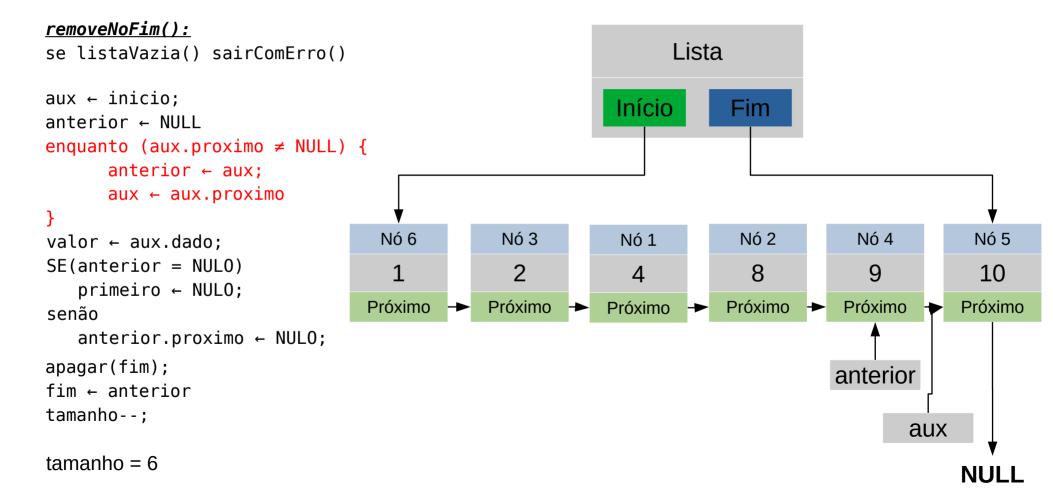


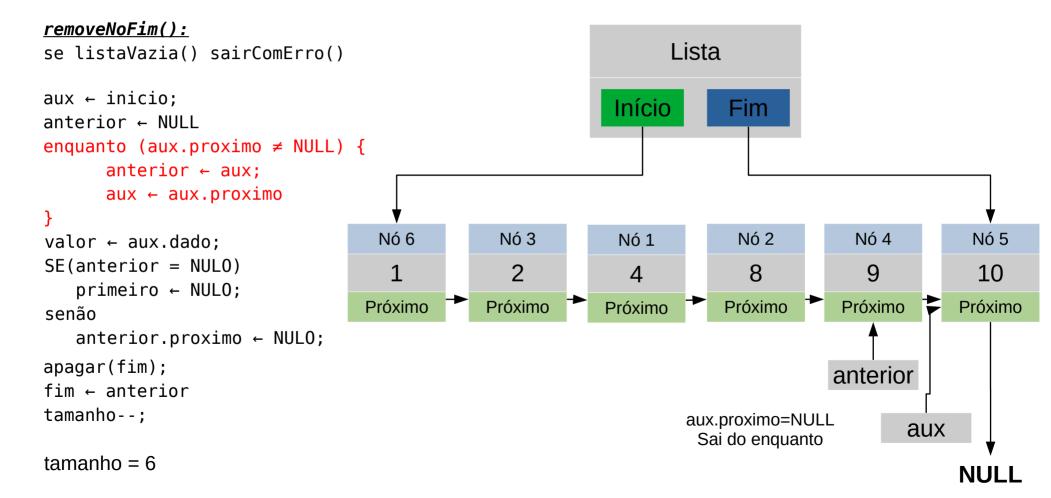


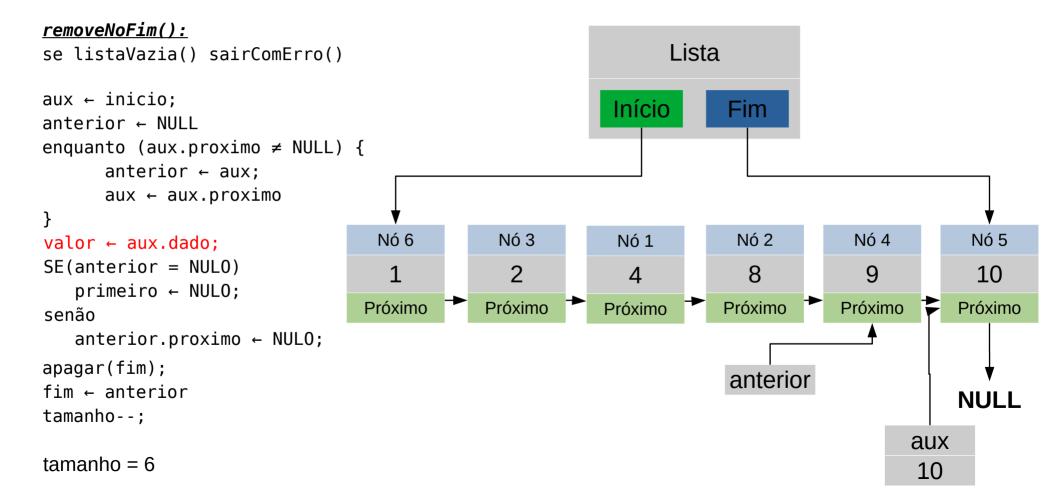


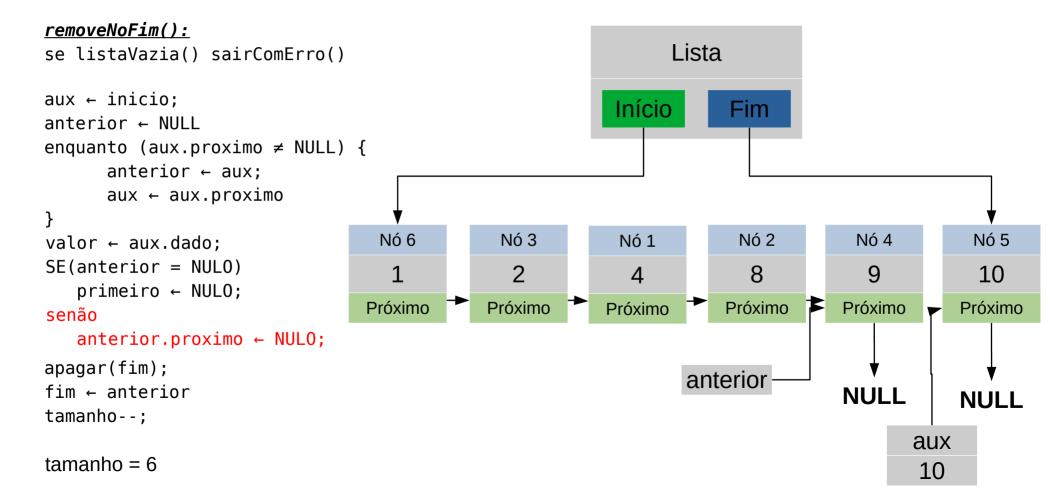


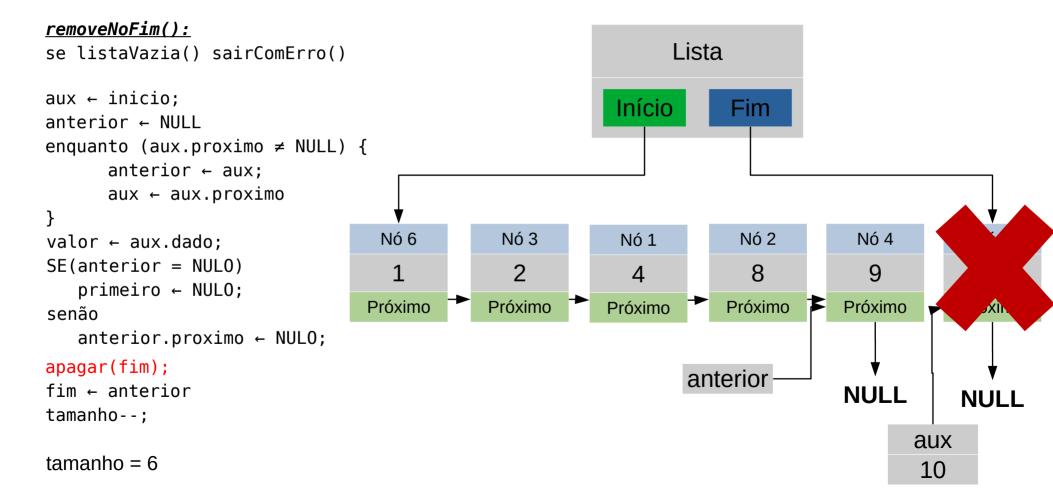












tamanho = 6

```
removeNoFim():
                                                              Lista
se listaVazia() sairComErro()
aux ← inicio;
                                                        Início
                                                                    Fim
anterior ← NULL
enquanto (aux.proximo ≠ NULL) {
      anterior ← aux;
      aux ← aux.proximo
                                      Nó 6
                                                  Nó3
                                                              Nó 1
                                                                          Nó 2
                                                                                      Nó 4
valor ← aux.dado;
SE(anterior = NUL0)
                                                                                       9
   primeiro ← NULO;
                                                 Próximo
                                                             Próximo
                                                                         Próximo
                                     Próximo
                                                                                     Próximo
senão
   anterior.proximo ← NULO;
apagar(fim);
                                                                     anterior
fim ← anterior
                                                                                     NULL
tamanho--;
```

```
removeNoFim():
                                                             Lista
se listaVazia() sairComErro()
aux ← inicio;
                                                        Início
                                                                    Fim
anterior ← NULL
enquanto (aux.proximo ≠ NULL) {
      anterior ← aux;
      aux ← aux.proximo
                                      Nó 6
                                                  Nó3
                                                              Nó 1
                                                                          Nó 2
                                                                                      Nó 4
valor ← aux.dado;
SE(anterior = NUL0)
                                                                                       9
   primeiro ← NULO;
                                                 Próximo
                                                             Próximo
                                                                         Próximo
                                     Próximo
                                                                                     Próximo
senão
   anterior.proximo ← NULO;
apagar(fim);
                                                                     anterior
fim ← anterior
                                                                                     NULL
tamanho--;
```

tamanho = 5

```
auxiliar← inicio;
anterior ← null
                                                                Lista
enquanto ((auxiliar != NULL) E
(auxiliar.dado != valor)){
     anterior ← auxiliar;
                                                           Início
                                                                       Fim
     auxiliar ← auxiliar.proximo;
  (auxiliar != NULL){
     se (auxiliar == inicio){
          removeNoInicio();
     } senao se (auxiliar == fim){
          removeNoFim()
                                        Nó 6
                                                     Nó 3
                                                                              Nó 2
                                                                                           Nó 4
                                                                 Nó 1
     } senao {
                                                                                            9
          anterior.proximo ←
     auxiliar.proximo;
                                      Próximo
                                                   Próximo →
                                                                             Próximo -
                                                                                         Próximo
                                                                Próximo
          tamanho--:
          apaga(auxiliar);
                               aux
                                                                         anterior
                                                                                         NULL
```

```
auxiliar← inicio;
anterior ← null
                                                                 Lista
enquanto ((auxiliar != NULL) E
(auxiliar.dado != valor)){
     anterior ← auxiliar;
                                                           Início
                                                                        Fim
     auxiliar ← auxiliar.proximo;
se (auxiliar != NULL){
     se (auxiliar == inicio){
          removeNoInicio();
     } senao se (auxiliar == fim){
          removeNoFim()
                                        Nó 6
                                                     Nó 3
                                                                               Nó 2
                                                                                            Nó 4
                                                                  Nó 1
     } senao {
                                                                                             9
          anterior.proximo ←
     auxiliar.proximo;
                                       Próximo
                                                    Próximo →
                                                                              Próximo -
                                                                                          Próximo
                                                                 Próximo
          tamanho--:
          apaga(auxiliar);
                                aux
                                                                          anterior
                                                                                          NULL
                         1 != 2 e aux != NULL
                         Continua o enquanto
```

```
auxiliar← inicio;
anterior ← null
                                                                 Lista
enquanto ((auxiliar != NULL) E
(auxiliar.dado != valor)){
     anterior ← auxiliar;
                                                           Início
                                                                        Fim
     auxiliar ← auxiliar.proximo;
se (auxiliar != NULL){
     se (auxiliar == inicio){
          removeNoInicio();
     } senao se (auxiliar == fim){
          removeNoFim()
                                        Nó 6
                                                     Nó 3
                                                                               Nó 2
                                                                                            Nó 4
                                                                  Nó 1
     } senao {
                                                                                             9
          anterior.proximo ←
     auxiliar.proximo;
                                       Próximo
                                                    Próximo →
                                                                             Próximo
                                                                                          Próximo
                                                                 Próximo
          tamanho--:
          apaga(auxiliar);
                             anterior
                                                                                          NULL
                                                     aux
                                                     2 = 2
                                                 Sai do enquanto
```

```
auxiliar← inicio;
anterior ← null
                                                                 Lista
enquanto ((auxiliar != NULL) E
(auxiliar.dado != valor)){
     anterior ← auxiliar;
                                                           Início
                                                                        Fim
     auxiliar ← auxiliar.proximo;
se (auxiliar != NULL){
     se (auxiliar == inicio){
          removeNoInicio();
     } senao se (auxiliar == fim){
          removeNoFim()
                                        Nó 6
                                                     Nó3
                                                                               Nó 2
                                                                                           Nó 4
                                                                  Nó 1
     } senao {
                                                                                             9
          anterior.proximo ←
     auxiliar.proximo;
                                       Próximo
                                                    Próximo →
                                                                Próximo
                                                                             Próximo
                                                                                          Próximo
          tamanho--:
          apaga(auxiliar);
                             anterior
                                                                                          NULL
                                                     aux
                                                     2 = 2
                                                Sai do enquanto
```

```
auxiliar← inicio;
anterior ← null
                                                                 Lista
enquanto ((auxiliar != NULL) E
(auxiliar.dado != valor)){
     anterior ← auxiliar;
                                                           Início
                                                                        Fim
     auxiliar ← auxiliar.proximo;
se (auxiliar != NULL){
     se (auxiliar == inicio){
          removeNoInicio();
     } senao se (auxiliar == fim){
          removeNoFim()
                                        Nó 6
                                                     Nó3
                                                                               Nó 2
                                                                                           Nó 4
                                                                  Nó 1
     } senao {
                                                                                             9
          anterior.proximo ←
     auxiliar.proximo;
                                       Próximo
                                                    Próximo →
                                                                Próximo
                                                                             Próximo
                                                                                          Próximo
          tamanho--:
          apaga(auxiliar);
                             anterior
                                                                                          NULL
                                                     aux
                                                     2 = 2
                                                Sai do enquanto
```

```
auxiliar← inicio;
anterior ← null
                                                                 Lista
enquanto ((auxiliar != NULL) E
(auxiliar.dado != valor)){
     anterior ← auxiliar;
                                                           Início
                                                                        Fim
     auxiliar ← auxiliar.proximo;
se (auxiliar != NULL){
     se (auxiliar == inicio){
          removeNoInicio();
     } senao se (auxiliar == fim){
          removeNoFim()
                                        Nó 6
                                                                  Nó 1
                                                                               Nó 2
                                                                                            Nó 4
     } senao {
                                                                                             9
          anterior.proximo ←
     auxiliar.proximo;
                                       Próximo
                                                                 Próximo
                                                                              Próximo -
                                                                                          Próximo
          tamanho--:
          apaga(auxiliar);
                             anterior
                                                                                          NULL
                                                     aux
                                                     2 = 2
                                                 Sai do enquanto
```

Resultado

tamanho = 4

