Estruturas de Dados e Algoritmos

Lista Ligada (Abordagem Recursiva)

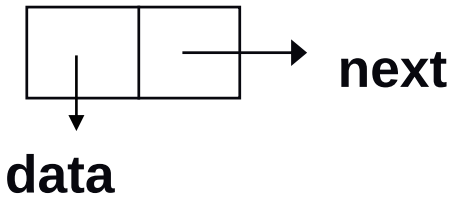
Adalberto Cajueiro

Departamento de Sistemas e Computação

Universidade Federal de Campina Grande

Questões de Implementação

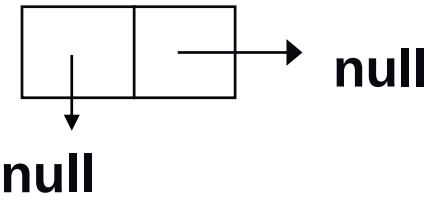
- Implementacao dinâmica – através de duas abordagens:
  - Estrutura recursiva com métodos iterativos
    - Cada elemento (nó) da lista contém um dado armazenado e um apontador para o próximo elemento da lista
  - Estrutura recursiva com métodos recursivos
    - Cada elemento (nó) da lista contém um dado armazenado e um apontador para o próximo elemento da lista
    - Não existe mais a idéia de modificar o head. Precisa-se apenas da referencia para o primeiro nó da lista. Toda a manipulação (e percurso) é resolvida de forma recursiva.
    - Todo método é invocado no primeiro nó da lista. Se precisar ir para os outros nós isso é feito de forma recursiva.



Questões de Implementação

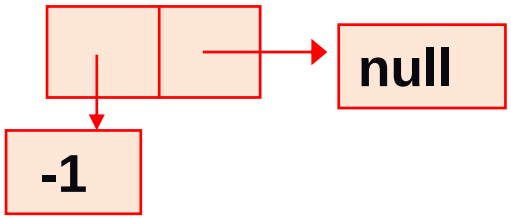
- Lista vazia
  - Representada por um nó especial (NIL) sem dado
  - Pode também ser representado por NULL
- Nós sentinela
  - Representam a lista vazia e são acrescentados nas extremidades das listas
  - Simplificam e aceleram alguns algoritmos sobre lista e dão garantia de que uma lista sempre contém algum nó (mesmo que sjea vazio)

Nó sentilena NIL

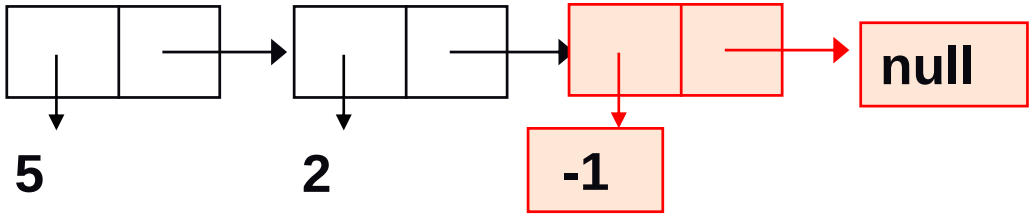


Implementação

- Exemplo: lista de inteiros não-negativos



Lista vazia=NIL
(caso base)



Lista não vazia
(caso indutivo)

Como implementar em Java?

O fato da lista ser implementada de forma toda recursiva muda sua interface ?

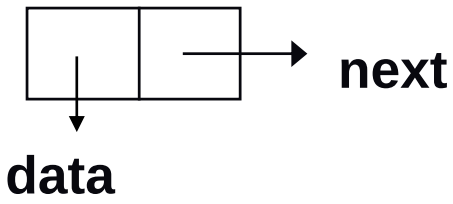
LISTA (IMPLEMENTACAO)

```
public interface LinkedList<T> {
 public boolean isEmpty();
 public int size();
 public T search(T element);
 public void insert(T element);
 public void remove(T element);
 public T[] toArray();
}
```

Lista (Implementacao)

- Na abordagem anterior qual era a estrutura recursiva?
- Qual a relação que existe entre um nó da lista e a lista encadeada agora?
- Uma lista agora precisa encapsular um nó?
  - Precisa de head?

LISTA (IMPLEMENTACAO)




```
public class RecursiveSingleLinkedListImpl<T>
 implements LinkedList<T>{

  protected T data;
  protected RecursiveSingleLinkedListImpl<T> next;

}
```

O que o construtor default origina?

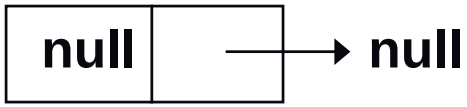
LISTA (IMPLEMENTACAO)

```
public class RecursiveSingleLinkedListImpl<T>
 implements LinkedList<T>{

  protected T data;
  protected RecursiveSingleLinkedListImpl<T> next;

}
```

- As implementações completamente recursivas estabelecem duas formas de resolver o problema:
  - Caso base: lista vazia
  - Caso indutivo: lista não vazia
- TODOS os métodos devem seguir esse template!

LISTA (ISEMPTY)

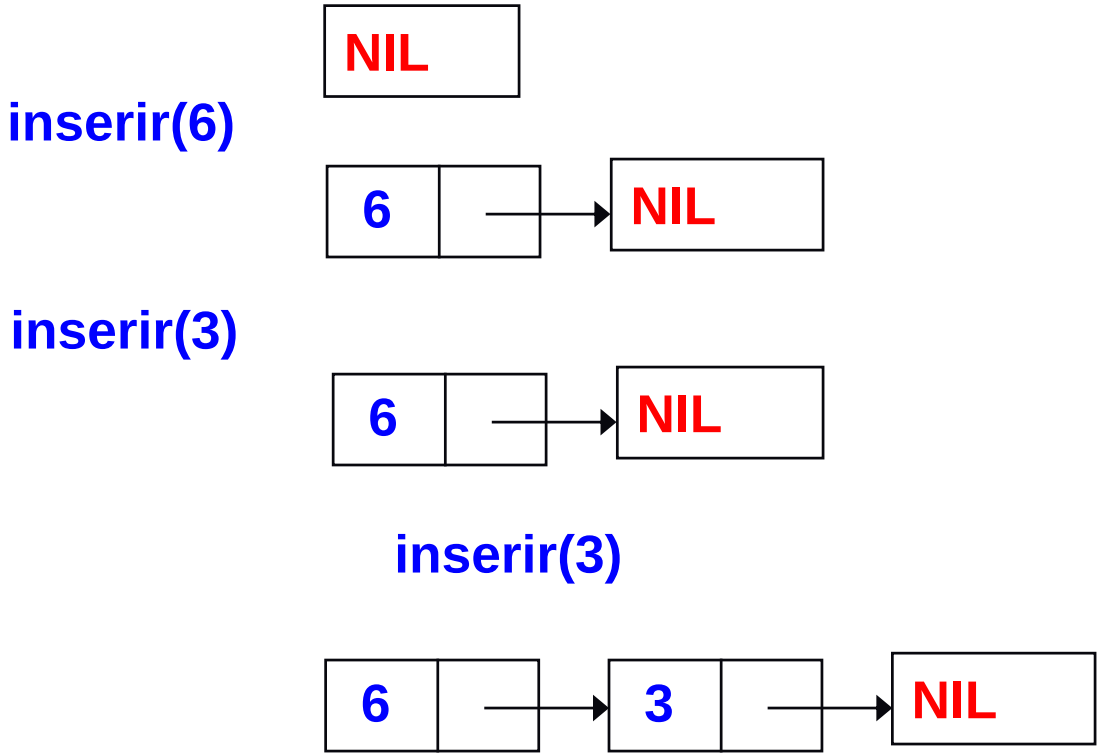
- Como seria o isEmpty?




```
isEmpty(){
  if data == null //caso base
      return true
  else //caso indutivo
     return false
}
```

LISTA (INSERIR)

- Admitindo que a lista não é ordenada, as insercoes acontecem **sempre** ao final da lista



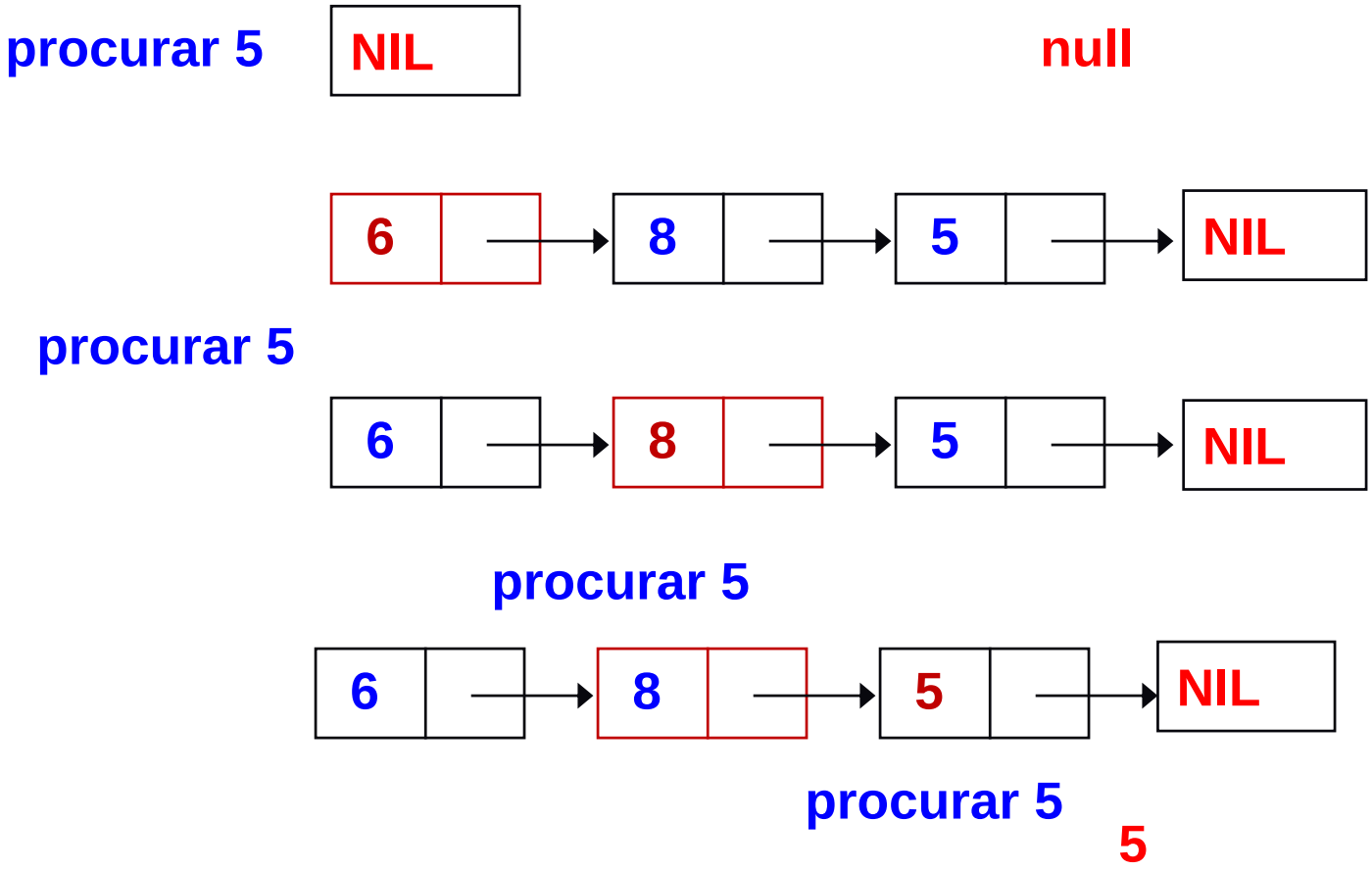
Lista (Inserir)

- Como seria a inserção em uma lista (considerando o template)?

```
list-insert(item){
   if(isEmpty) //caso base
      data = item
      next = new RecursiveSingleLinkedListImpl<T>( )
   else //caso indutivo
      next.list-insert(item)
}
```

LISTA (SEARCH)

- Procura sequencial pelo valor armazenado



Lista (Search)

- Como seria a procura em uma lista?

```
list-search(item){
   if(isEmpty) //caso base

   else //caso indutivo




}
```

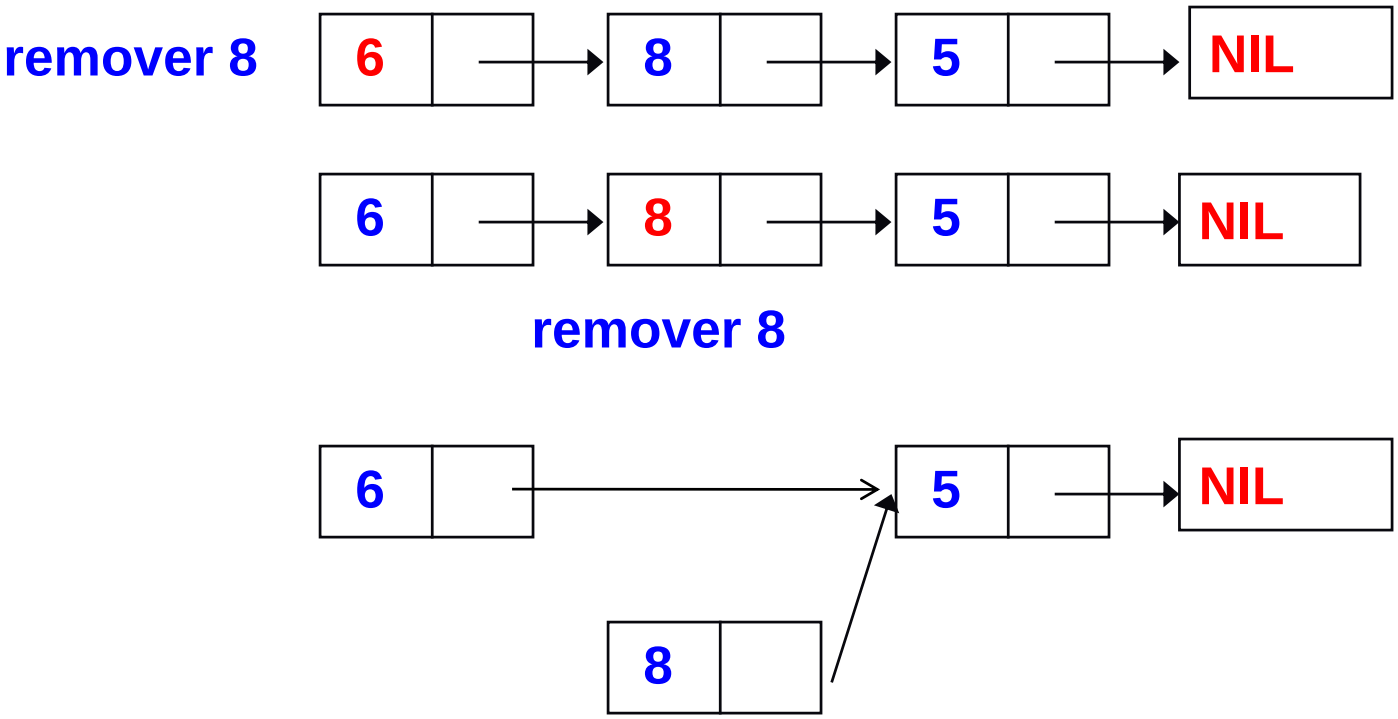
Lista (Search)

- Como seria a procura em uma lista?

```
list-search(item){
   if(isEmpty) //caso base
      return null
   else //caso indutivo
      if data==item
         return data
      else
         return next.list-search(item)
}
```

LISTA (REMOVER)

- As remocoes se dão pelo valor armazenado



Como seria o algoritmo de remoção na lista?

Lista (Remover)

- Como seria a remoção em uma lista?

```
list-remove(item){
   if(isEmpty) //caso base
      //nao faz nada porque nao contem o elemento
   else //caso indutivo




}
```

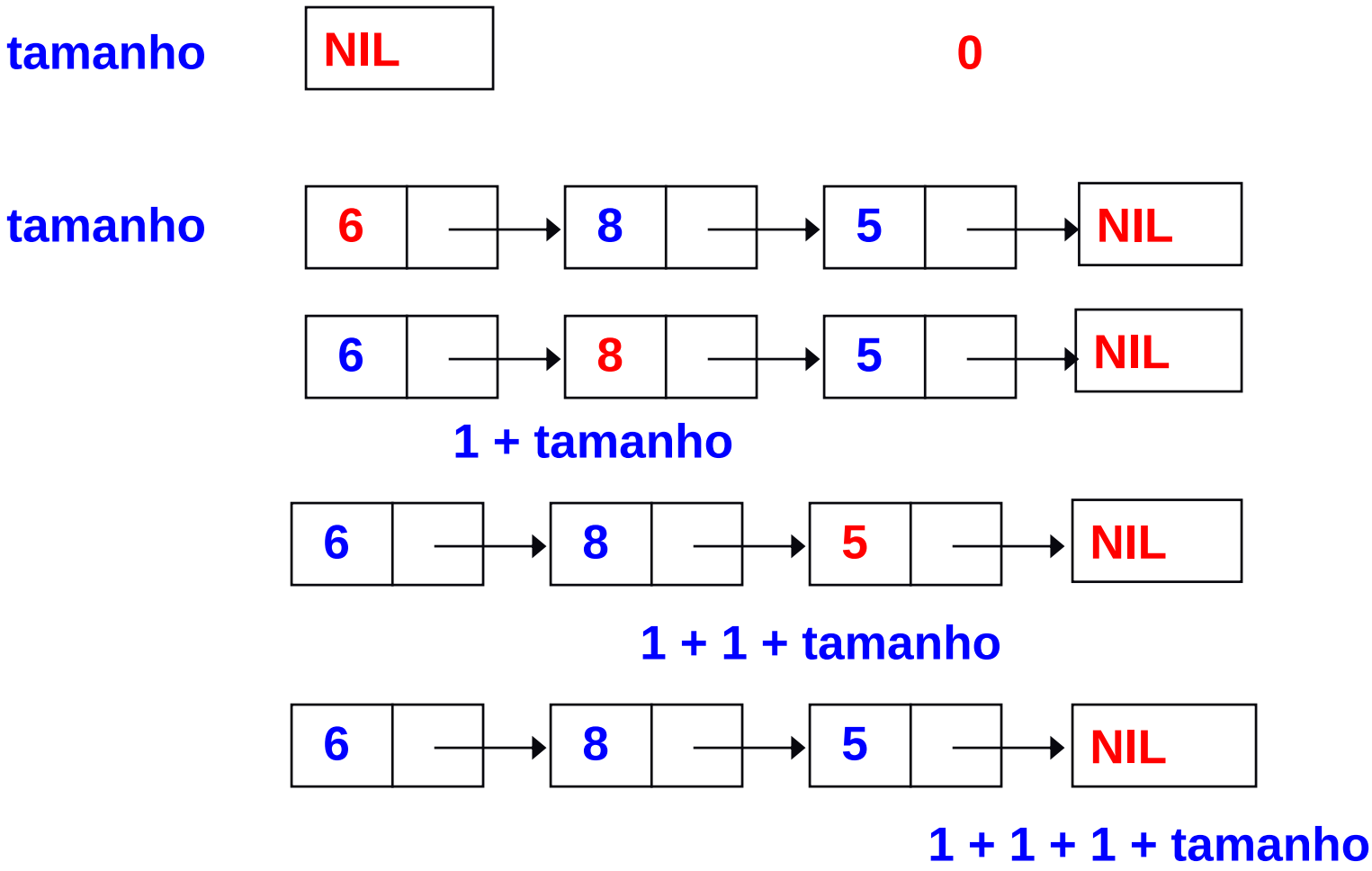
Lista (Remover)

- Como seria a remoção em uma lista?

```
list-remove(item){
   if(isEmpty) //caso base
      //nao faz nada porque nao contem o elemento
   else //caso indutivo
      if data==item
         data = next.data
         next = next.next
      else
         next.list-remove(item)
}
```

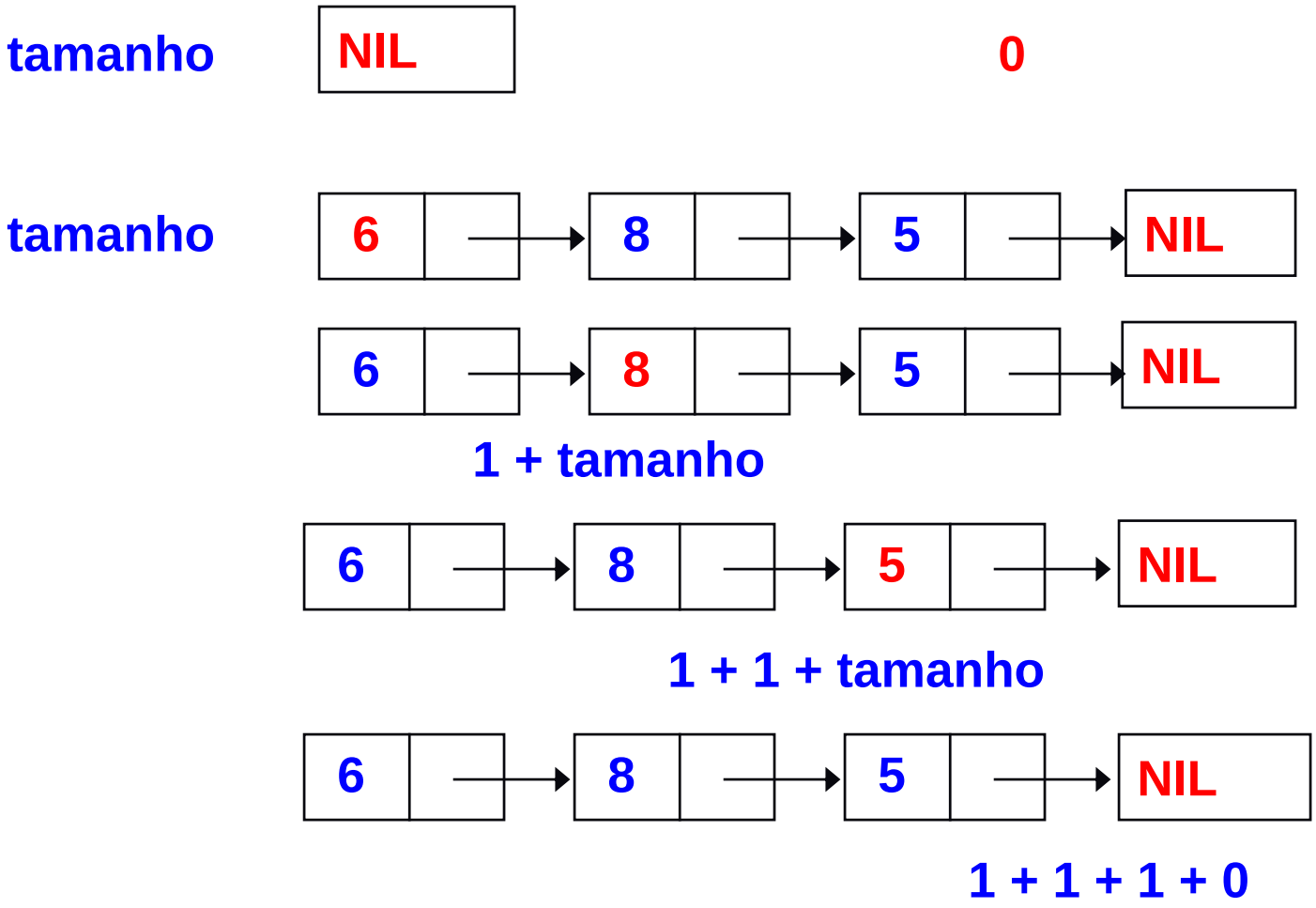
LISTA (TAMANHO)

- Como calcular o tamanho de uma lista?



LISTA (TAMANHO)

- Como calcular o tamanho de uma lista?



Lista (Tamanho)

- Como seria calcular o tamanho de uma lista?

```
list-size(){
   if(isEmpty) //caso base

   else //caso indutivo

}
```

Lista (Tamanho)

- Como seria calcular o tamanho de uma lista?

```
list-size(){
   if(isEmpty) //caso base
      return 0
   else //caso indutivo
      return 1 + next.list-size()
}
```

LISTA (TOARRAY)

- Como seria para transformar a lista em um array?

```
T[] list-toArray(){
   T[] result = new T[]

   return result
}
```

LISTA (TOARRAY)

- Como seria para transformar a lista em um array?

```
T[] list-toArray(){
   T[] result = new T[]
   toArray(result,this)
   return result
}

toArray (T[] array, RecursiveSingleLinkedListImpl node){
   if (!node.isEmpty){
      array.add(node.data)
      toArray(array,node.next)
   }
}
```

Exercício

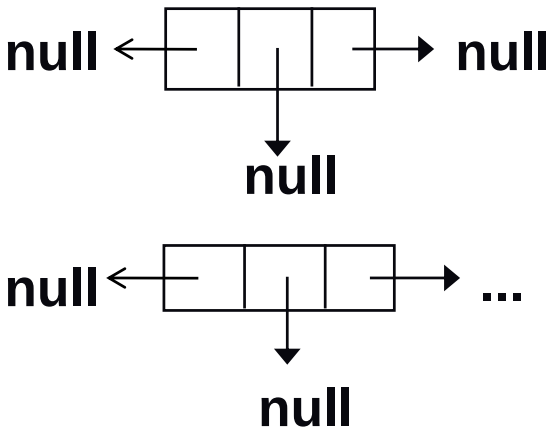
- Implemente um método que inverta uma lista ligada (abordagem recursiva).
- Implemente um método que encontre o maior elemento de uma lista ligada (abordagem recursiva).

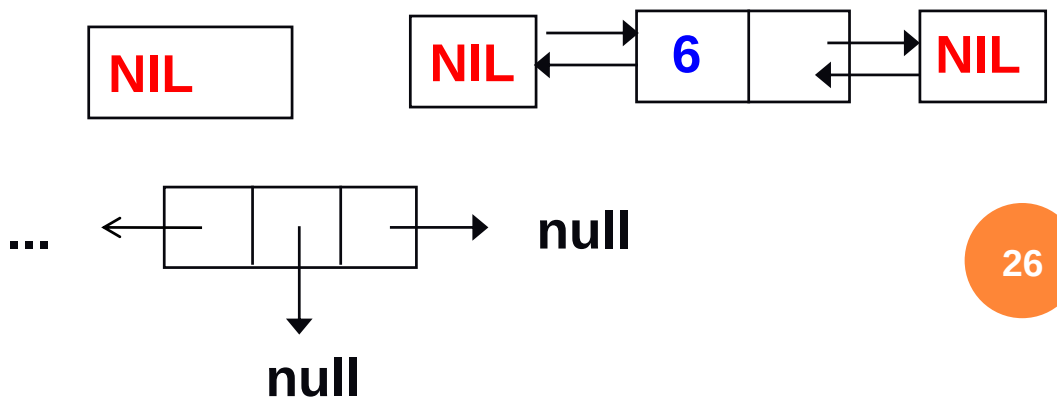
Lista Duplamente Ligada

Abordagem Recursiva

Lista Duplamente Ligada

- Estrutura semelhante a abordagem com métodos iterativos
- Interface de serviços é a mesma
- Existência de nós sentinela (início e fim da lista)
- Inexistência de head e last
  - Precisa-se apenas do primeiro nó da lista
  - O início de uma lista **não-vazia** NÃO é o sentinela





Lista Duplamente Ligada

```
public interface LinkedList<T> {
 public boolean isEmpty();
 public int size();
 public T search(T element);
 public void insert(T element);
 public void remove(T element);
 public T[] toArray();
 public void insertFirst(T element);
 public void removeFirst();
 public void removeLast();
}
```

Lista Duplamente Ligada

```
public interface LinkedList<T> {
 public boolean isEmpty();
 public int size();
 public T search(T element);
 public void insert(T element);
 public void remove(T element);
 public T[] toArray();
 public void insertFirst(T element);
 public void removeFirst();
 public void removeLast();
}
```

Esses métodos já existem em algum lugar?

Lista Duplamente Ligada(Implementacao)

```
public class RecursiveDoubleLinkedListImpl<T>
   extends RecursiveSingleLinkedListImpl<T>
     implements DoubleLinkedList<T>{

  protected RecursiveDoubleLinkedListImpl<T> previous;

}
```

O que o construtor default origina?

Algum método herdado precisa ser sobrescrito?

LISTA DUPLAMENTE LIGADA(IMPLEMENTACAO)

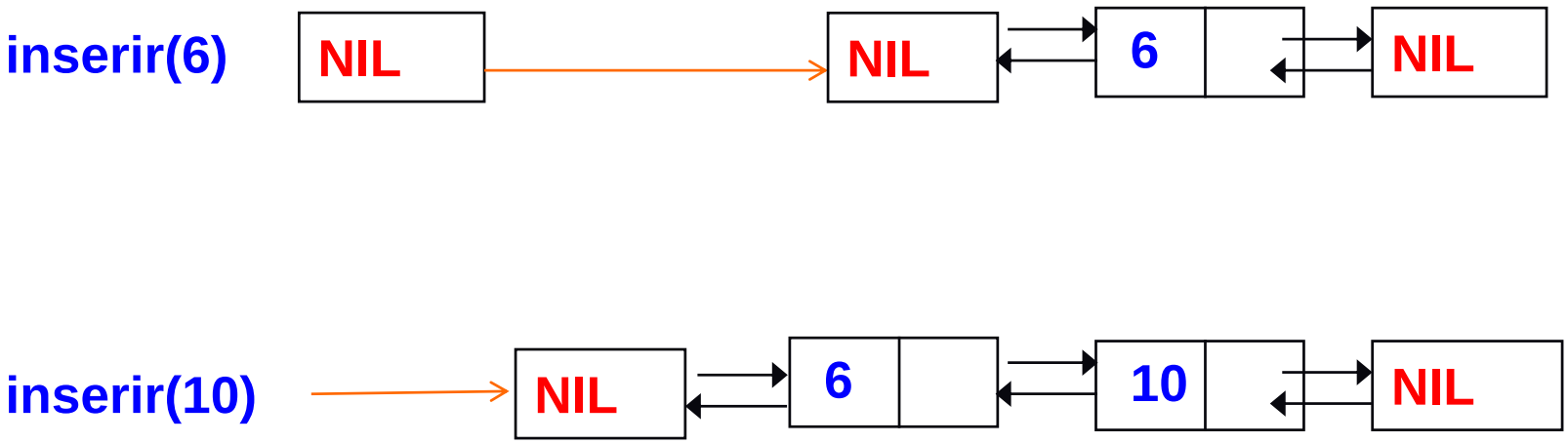
```
public class RecursiveDoubleLinkedListImpl<T>
   extends RecursiveSingleLinkedListImpl<T>
     implements DoubleLinkedList<T>{

  protected RecursiveDoubleLinkedListImpl<T> previous;

}
```

O que o construtor default origina?

Algum método herdado precisa ser sobrescrito?

insert,remove

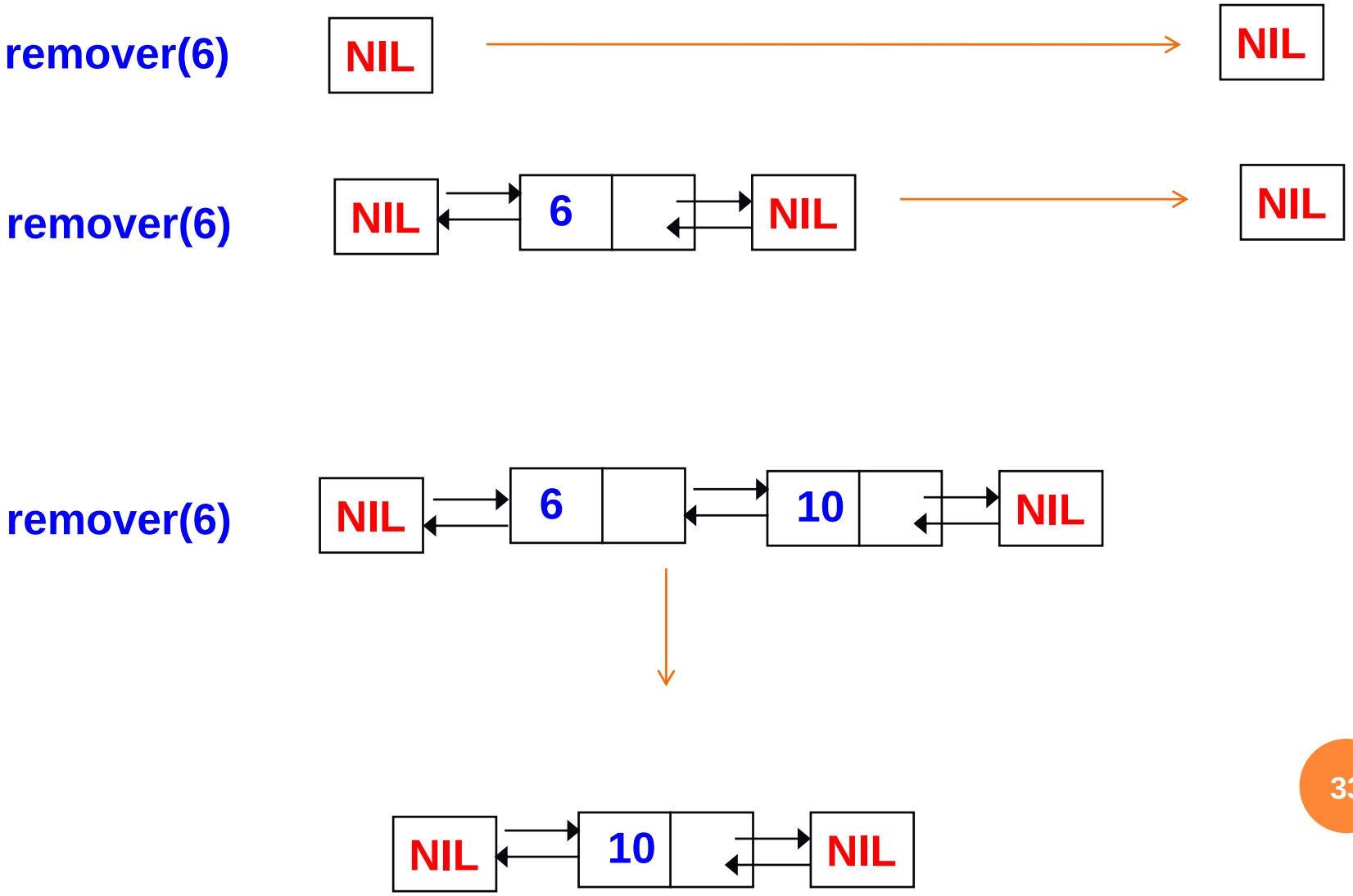
Lista Dupla(inserir no final=default)



Lista Dupla(Implementacao - Inserção)

```
//insere sempre no final
list-insert(item){
  if (isEmpty()){
    data = element
    next = NIL //previous do NIL precisa ser conectado!!!
    if (previous==null){
       previous = NIL
    }
  }else{
    next.list-insert(item)
  }
```

Lista Dupla(Remover pela Chave = default)



Lista Dupla(Implementacao – Remoção pela chave)

```
//remove pelo elemento (chave)
remove(item){
  if (isEmpty()){
     fazer nada
  }else{
     if (this.data == item){
        if(previous.isEmpty && next.isEmpty){
           data = next = previous = null
        } else{
            data = next.data
            next = next.next
            if(next != null)
               next.previous = this
        }
     } else{
        next.remove(item)
     }
  }
}
```

Exercício

- Implementar os seguinte métodos na lista duplamente encadeada recursiva.
  - `public void insertFirst(T element);`
  - `public void removeFirst();`
  - `public void removeLast();`

Referências

- Capítulo 11