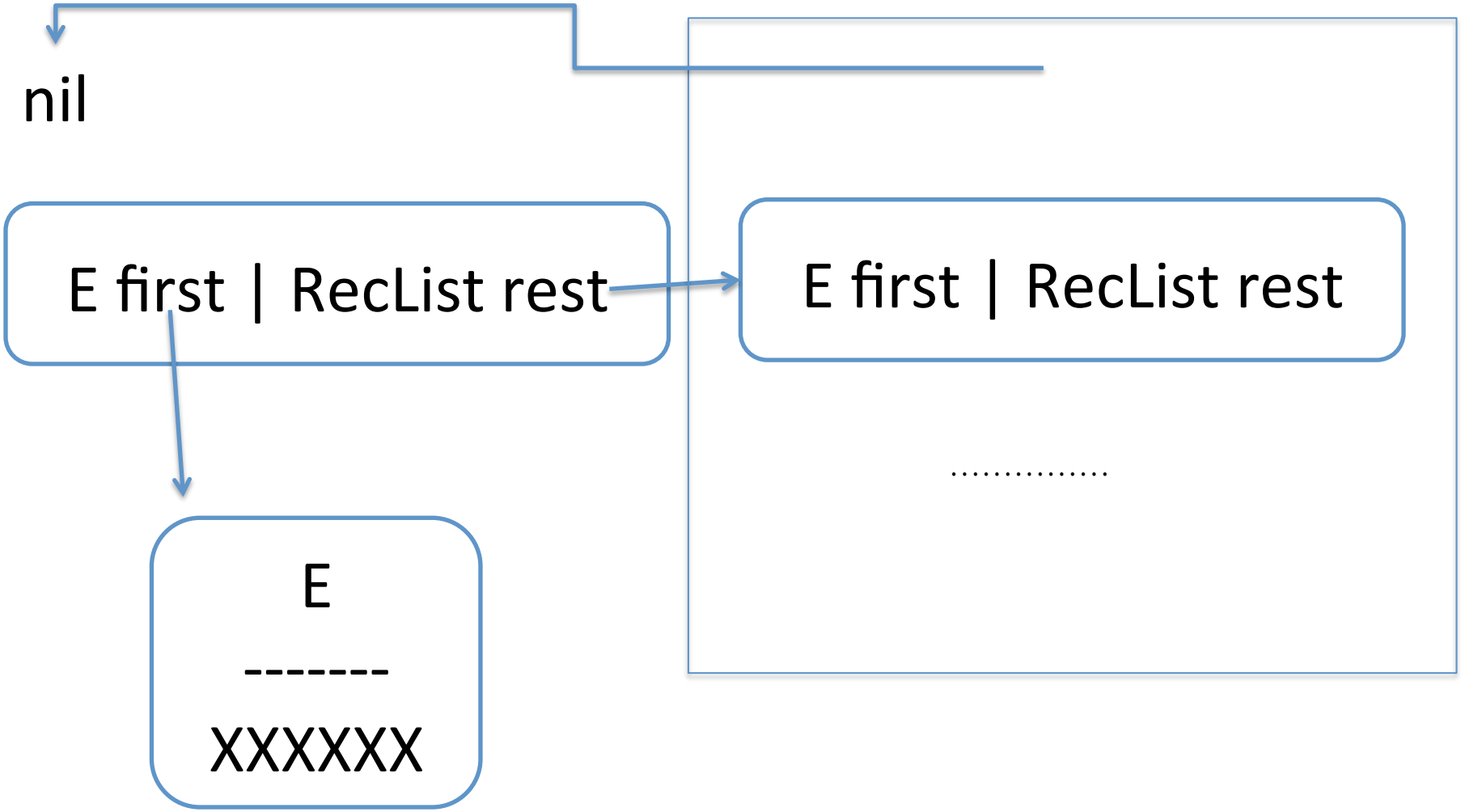
Liste e loro implementazioni

Implementazione con struttura dati ricorsiva

Liste con struttura dati ricorsiva



Liste con struttura dati ricorsiva

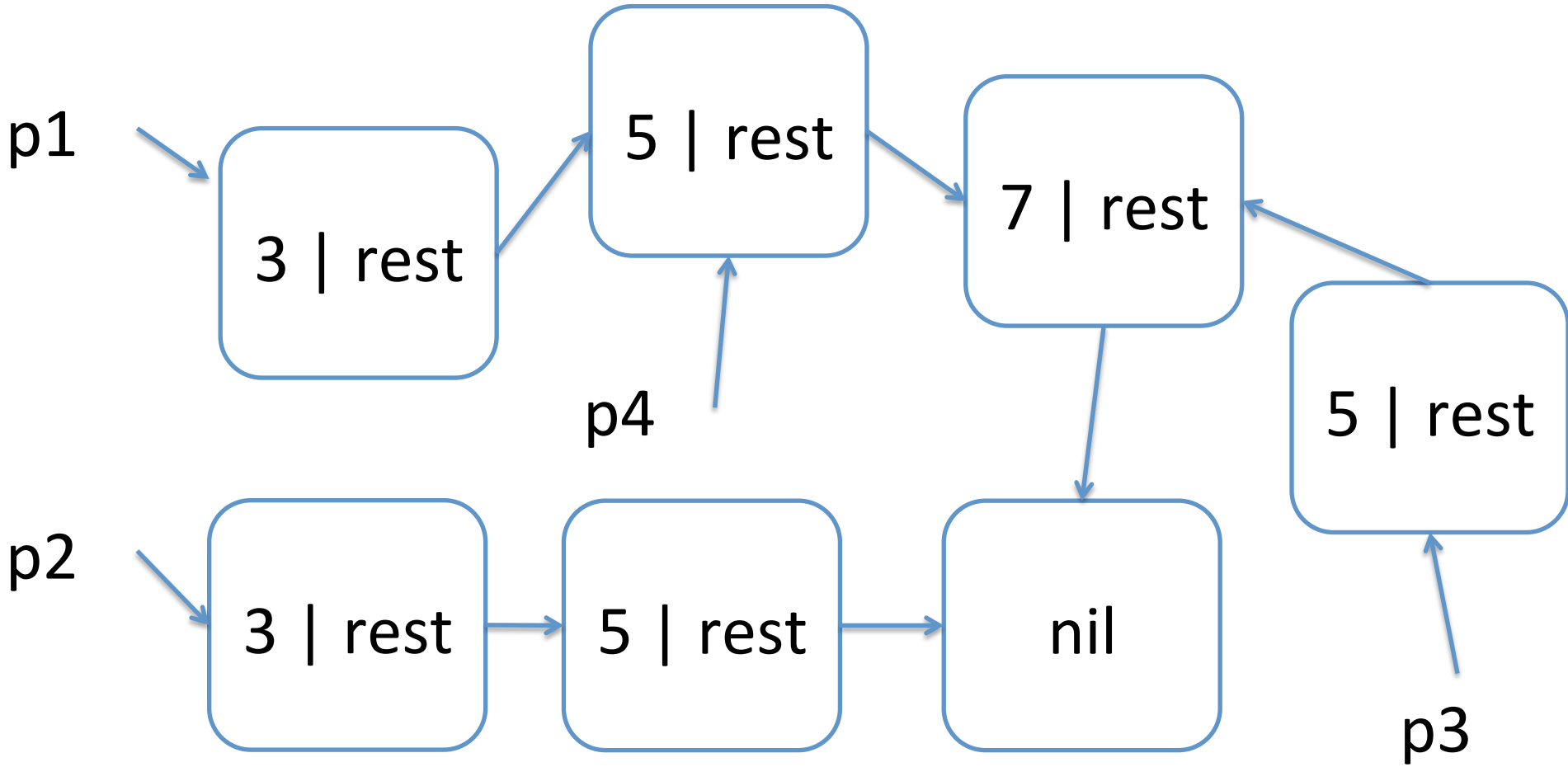
- **Visualizziamo le liste come “stringhe”:**
- [] = nil
- [1] = 1 -> nil
- [2,4,5] = 2 -> [4 –> [5 -> []]]]
- **Ulteriore alternativa:**
- Notazione cons : E x RecList -→ RecList
- Es cons(3,nil) = [3]
- Es cons(5,cons(3,nil)) = [5,3]

RecList<E> come tipo di dato astratto (approccio algebrico / funzionale)

- Partiamo da una classe E
- Esiste sempre la lista nil
- Operatore cons: E x RecList<E> → RecList<E>
- Operatore first: RecList<E> → E
- Operatore rest: RecList<E> → RecList<E>

- Questo è tutto quello che serve per definire qualsiasi lista di tipo E e qualsiasi operatore su liste di tipo E

Esempio equals



p1.equals(p2) è false p3.equals(p4) è true

Metodi ricorsivi

- Basati sulla struttura ricorsiva della struttura dati
- In questo caso
  - Caso Base: lista vuota nil
  - Caso Ricorsivo / Induttivo: lista cons(first,rest)
- Nel caso base una risposta immediata
- Nel caso ricorsivo, si decompone la lista nelle due parti, la chiamata ricorsiva si fa su rest e si compone il risultato della chiamata ricorsiva con una opportuna operazione sul first

Esempio: lunghezza

- size([2,5,7]) = 3
- size([]) = 0

- size(lista) restituisce la lunghezza della lista

Esempio: ricerca

- Scriviamo un metodo ricorsivo che presa una lista e un oggetto della classe E, risponde true se l'oggetto è presente nella lista e false in caso contrario.

- contains([2,3,4,5,4],4) = true
- contains([2,3,4,5,4],0) = false

Esempio: eliminazione

- Elimina il primo elemento presente nella lista che è uguale ad un oggetto della classe E passato. La lista risultato deve essere “uguale” alla lista iniziale, tranne l’assenza dell’elemento eliminato

Esempio removeFirst([1,2,1,3,4,3,5],3) = [1,2,1,4,3,5]

Esempio: eliminazione totale

- Elimina tutti gli elementi presenti nella lista che sono uguali ad un oggetto della classe E passato. La lista risultato deve essere “uguale” alla lista iniziale, tranne l’assenza degli elementi rimossi.

Esempio removeFirst([1,2,1,3,4,3,5],3) = [1,2,1,4,3,5]

Esercizio per casa: append

- append([3,4,6,1],[4,3,0]) = [3,4,6,1,4,3,0]

- Restituisce una lista che è la concatenazione delle due liste date
- Nota Bene: usare in maniera saggia la ricorsione, altrimenti la soluzione non si trova ☺

Esercizio per casa: get

- get([2,3,6,7],0) = 2
- get([2,3,6,7],2) = 3

- get(lista,i), restituisce l'elemento in posizione i, se i è compreso tra 0 e la size() − 1, in caso contrario lancia una eccezione IndexOutOfBoundException

Esercizio per casa: add

- add([2,3,6,7],0) = [2,3,6,7,0]
- add([],2) = [2]

- add(lista,elem), restituisce una nuova lista in cui l'elemento elem è stato inserito in fondo

- (suggerimento: usare append)