

---

## Mètode de llistes per a moure l'element apuntat per un iterador una posició cap al final X75139\_ca

---

Implementeu un nou mètode de la classe `List` per a moure el contingut apuntat per un iterador una posició cap al final. En cas que l'iterador ja apunti a l'últim element de la llista, llavors el mètode no farà res.

D'entre els fitxers que s'adjunten en aquest exercici, trobareu `list.hh`, a on hi ha una implementació de la classe genèrica `List`. Haureu de buscar dins `list.hh` les següents línies:

```
// Pre: it apunta a algun element de la llista implícita.
// Post: it continua apuntant al mateix element, el qual ha estat mogut una pos
//       cap al final. No s'ha creat ni eliminat memòria.
//       En el cas en que l'element apuntat per it ja era l'últim, res ha canvi
// Descomenteu les següents dues línies i implementeu el mètode:
// void moveTowardsEnd(iterator &it) {
// }
```

Descomenteu les dues línies que s'indiquen i implementeu el mètode. No toqueu la resta de la implementació de la classe, excepte si, per algun motiu, considereu que necessiteu afegir algun mètode auxiliar a la part privada.

D'entre els fitxers que s'adjunten a l'exercici també hi ha `main.cc` (programa principal), i el podeu compilar directament, doncs inclou `list.hh`. Només cal que pugueu `list.hh` al jutge.

### Entrada

La entrada del programa és una seqüència d'instruccions del següent tipus que s'aniran aplicant sobre una llista que se suposa inicialment buida i un iterador que se suposa situat inicialment al principi (i final) d'aquesta llista:

```
push_front s (s és un string)
push_back s (s és un string)
pop_front
pop_back
it++
it--
*it
moveTowardsEnd
```

Se suposa que la seqüència d'entrada serà correcta (sense `pop_front` ni `pop_back` sobre llista buida, ni `*it` ni `moveTowardsEnd` tenint `it` situat al end de la llista). Tampoc hi haurà `pop_front` just quan l'iterador estigui apuntant al primer element de la llista, ni hi haurà `pop_back` just quan l'iterador estigui apuntant a l'últim element de la llista (tingueu en compte que l'últim element de la llista no és el end de la llista).

El programa principal que us oferim ja s'encarrega de llegir aquestes entrades i fer les crides als corresponents mètodes de la classe `list`. Només cal que implementeu els mètodes abans esmentats.

## Sortida

Per a cada instrucció `*it`, s'escriurà el contingut apuntat per l'iterador. El programa que us oferim ja fa això. Només cal que implementeu el mètode abans esmentat.

### Exemple d'entrada 1

```
push_front a
it--
*it
moveTowardsEnd
*it
push_back b
it++
*it
moveTowardsEnd
*it
it--
*it
moveTowardsEnd
*it
it--
*it
push_front c
*it
it--
*it
moveTowardsEnd
*it
it--
*it
moveTowardsEnd
*it
it++
*it
moveTowardsEnd
*it
```

### Exemple de sortida 1

```
a
a
b
b
a
a
b
b
c
c
b
b
a
a
```

### Exemple d'entrada 2

```
push_front de
push_front eg
push_front gz
push_back aw
pop_back
push_front tf
it--
pop_front
*it
push_front rq
push_front s
it--
it++
it--
it--
push_back w
push_back ou
pop_front
push_front u
it++
pop_back
```

```
push_back gs
push_back ok
push_front s
it++
moveTowardsEnd
it++
*it
push_back j
pop_front
push_back xg
push_front lo
push_front j
pop_back
moveTowardsEnd
push_back i
push_front ir
moveTowardsEnd
*it
push_back wg
*it
```

## Exemple de sortida 2

de

| gs  
| gs  
| gs

## Informació del problema

Autor : PRO2

Generació : 2024-04-23 16:40:08

© *Jutge.org*, 2006–2024.

<https://jutge.org>