Mètode de llistes per intercanviar (swappejar) la posició de dos elements apuntats per iteradors X64186_ca

Implementeu un nou mètode de la classe List que rebi dos iteradors i intercanvii de posició els elements (items) apuntats per aquests iteradors. Això vol dir que cadascun d'aquests elements passarà a ocupar la posició dins la llista que ocupava l'altre element. Els iteradors no han de canviar. Per tant, cadascun ha de seguir apuntant a la mateixa memòria. Els valors dels elements tampoc han de canviar.
D'entre els fitxers que s'adjunten en aquest exercici, trobareu `list.old.hpp`, a on hi ha una implementació de la classe genèrica `List`. En primer lloc, haureu de fer:

```
cp list.old.hpp list.hpp
```

A continuació, haureu de buscar dins `list.hpp` les següents línies:

```
// Pre:  it1,it2 apunten a elements de la llista implícita.
// Post: it1,it2 continuen apuntant als mateixos elements,
//       i *it1, *it2 continuen valent el mateix que abans.
//       però *it1 ocupa la posició que ocupava abans *it2,
//       i *it2 ocupa la posició que ocupava abans *it1.
//       A part d'això, res més ha canviat.
// Descomenteu les següents dues linies i implementeu el mètode:
// void swapPositionsOfItems(iterator &it1, iterator &it2) {
// }
```

Descomenteu les dues linies 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.
La implementació d'aquest mètode hauria de consistir en modificar punters. De fet, possiblement qualsevol implementació alternativa serà massa lenta o produïrà error d'execució.
D'entre els fitxers que s'adjunten a l'exercici també hi ha `program.cpp` (programa principal) i `Makefile` per a compilar. Per a pujar la vostra solució, heu de crear el fitxer `solution.tar` així:

```
tar cf solution.tar list.hpp
```

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 dos iteradors it1, it2, que se suposen situats 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
it1++
```

```
it1--
it2++
it2--
*it1
*it2
swapPositionsOfItems
```

Se suposa que la seqüència d'entrada serà correcta (sense `pop_front` sobre llista buida ni amb algun dels iteradors apuntant al primer element de la llista, ni `pop_back` sobre llista buida ni amb algun dels iteradors apuntant a l'últim element de la llista, ni `++` ni `*` sobre iterador apuntant al `end` de la llista, ni `--` sobre iterador apuntant al principi de la llista, ni `swapPositionsOfItems` quan algun dels iteradors apunta al `end` de la llista. Fixeu-vos que el `end` de la llista no és l'últim element 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 `*it1` o `*it2`, 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
it1--
push_back b
it2--
*it1
*it2
swapPositionsOfItems
*it1
*it2
it1--
*it1
*it2
swapPositionsOfItems
*it1
*it2
push_back c
it2++
*it2
it2++
*it1
*it2
swapPositionsOfItems
*it1
*it2
swapPositionsOfItems
*it1
*it2
it2--
it2--
it1++
it1++
*it1
*it2
swapPositionsOfItems
*it1
*it2
it1++
*it1
it2--
*it2
swapPositionsOfItems
*it1
*it2
```

Exemple de sortida 1

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

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

Exemple d'entrada 2

```
push_front nk
push_back gn
push_front x
push_back o
it1--
pop_front
push_back ir
push_front wp
*it1
pop_back
push_back t
it1++
push_back ap
push_front v
pop_front
push_back v
it1++
pop_front
pop_front
it2--
it1--
it1--
swapPositionsOfItems
it2--
push_back p
*it2
push_back p
swapPositionsOfItems
it1++
push_back d
*it2
pop_front
it2--
*it1
push_back iz
swapPositionsOfItems
*it1
it1++
pop_back
push_front q
*it2
pop_back
it2++
swapPositionsOfItems
push_front l
```

```
*it1
push_front b
push_front g
push_back yo
pop_front
it1++
push_back y
it2--
push_front g
*it1
```

Exemple de sortida 2

```
o
gn
gn
```

```
v
v
ap
gn
v
```

Informació del problema

Autor : PRO1
Generació : 2023-05-07 09:09:09