

---

## Método de la clase Lista para rotar hacia el sentido contrario del reloj tantas veces como indique n X24601\_es

---

Implemente un nuevo método de la clase List que mueva los elementos de la lista hacia el sentido contrario del reloj tantas veces como indique **n**. Es decir, el primer elemento pasará a ser el último, el segundo pasará a ser el primero y así sucesivamente **SOLO** tantas veces como indique **n**.

De entre los archivos que se adjuntan en este ejercicio, encontrará listCounterclockwise.old.hpp, donde hay una implementación de la clase genérica List. En primer lugar, tendrá que hacer:

```
cp listCounterclockwise.old.hpp listCounterclockwise.hpp
```

A continuación, tendrá que buscar dentro listCounterclockwise.hpp las siguientes líneas:

```
// Pre: 'n' indica la cantidad de posiciones que hay que desplazarse.
// Post: Devuelve una lista de string donde el primer elemento de la lista pasa
// a ser el último de la lista y el segundo pasará a ser el primer elemento
// de la lista, se repetirá tantas veces como indique 'n'. En resumen, mover
// tantos elementos de la lista como indique 'n' hacia el sentido contrario
// del reloj.
// Descomente las siguientes dos líneas e implemente la función:
// void listCounterclockwise(int n) {
// }
```

Descomente las dos líneas que se indican e implemente el método. No toque el resto de la implementación de la clase, excepto si, por algún motivo, considere que necesita añadir algún método auxiliar a la parte privada.

La implementación de este método debería consistir en modificar punteros. De hecho, posiblemente cualquier implementación produzca error de ejecución.

Entre los archivos que se adjuntan al ejercicio también hay program.cpp (programa principal) y Makefile para compilar. Para subir su solución, debe crear el archivo solution.tar así:

```
tar cf solution.tar listCounterclockwise.hpp
```

### Entrada

La entrada del programa es una secuencia de instrucciones del siguiente tipo que se irán aplicando sobre una lista que se supone inicialmente vacía y un iterador que se supone situado inicialmente al principio (y final) de esta lista:

```
push_front s (s es un string)
push_back s (s es un string)
pop_front
pop_back
it++
it--
*it
listCounterclockwise 3
```

Se supone que la secuencia de entrada será correcta (sin `pop_front` ni `pop_back` sobre lista vacía, ni `*it` teniendo `it` situado en el end de la lista). Tampoco habrá `pop_front` justo cuando el iterador esté apuntando al primer elemento de la lista, ni habrá `pop_back` justo cuando el iterador esté apuntando al último elemento de la lista (tenga en cuenta que el último elemento de la lista no es el end de la lista).

El programa principal que le ofrecemos ya se encarga de leer estas entradas y hacer las llamadas a los correspondientes métodos de la clase `List`. Sólo tiene que implementar los métodos anteriormente mencionados.

## Salida

Para cada instrucción `*it`, se escribirá el contenido apuntado por el iterador y mostrará la lista de `string`. El programa que le ofrecemos ya hace esto. Sólo tiene que implementar el método anteriormente mencionado.

### Ejemplo de entrada 1

```
listCounterclockwise 1
insert a
listCounterclockwise 1
it--
*it
push_back b
listCounterclockwise 1
it--
*it
push_back c
listCounterclockwise 2
it++
*it
push_back d
listCounterclockwise 2
it++
*it
push_back e
listCounterclockwise 3
it++
*it
```

### Ejemplo de entrada 2

```
insert n
listCounterclockwise 7
it--
push_front r
*it
push_front h
*it
push_front o
listCounterclockwise 7
push_back d
listCounterclockwise 8
it--
push_front j
listCounterclockwise 3
push_front s
it--
push_front b
```

### Ejemplo de salida 1

```
a a
b ba
a cba
d adcb
c beadc
```

```
push_back r
*it
push_back c
listCounterclockwise 5
push_front p
*it
push_back l
*it
push_back p
*it
push_front o
listCounterclockwise 6
push_back h
pop_front
push_back e
*it
*it
```

## Ejemplo de salida 2

```
n rn
n hrn
r bsnohjrdr
```

```
r pjrdrbsnoh
r pjrdrbsnohl
r pjrdrbsnohlp
r bsnohlpopjrdrhe
r bsnohlpopjrdrhe
```

## Observación

Evaluación sobre 10 puntos:

- Solución lenta: 6 puntos.
- solución rápida: 10 puntos.

## Información del problema

Autor : STUDENTS PRO1

Generación : 2022-12-24 02:32:06

© *Jutge.org*, 2006–2022.

<https://jutge.org>