Inserir un element en una llista doblement encadenada i ordenada usant memòria dinàmica X98798_ca

Donada la classe *Llista* que permet guardar seqüències ordenades d'enters amb una llista doblement encadenada, sense fantasma, no circular i amb punt d'interès, cal implementar el mètode

```
void inserir_en_ordre (const int &x);
/* Pre: El p.i. es una llista L i esta ordenada de forma creixent */
/* Post: El p.i. s'ha modificat afegint x on li correspon per
   mantenir l'ordre creixent; si el punt d'interes de L
   referenciava a un element igual a x, passa a referenciar al
   primer element igual a x; altrament, el punt d'interes no canvia */
```

que insereix l'element x tot mantenint l'ordre creixent de la llista. Si el punt d'interès apuntava a un element diferent de x, no canvia; en cas contrari passa a apuntar al primer element igual a x.
Els nodes de la classe *Llista* estan doblement encadenats amb punters al següent (`seg`) i a l'anterior (`ant`). Una llista té quatre atributs: la `longitud` i tres punters a nodes, un pel primer element (`primer_node`), un per l'últim (`ultim_node`) i un altre per l'element actual (`act`), on tenim situat el punt d'interès de la llista.

Entrada

Com a entrada hi haurà una llista amb punt d'interès: el nombre de vegades que cal avançar el punt d'interès respecte el primer element, el nombre d'enters de la llista i els enters que la formen.
A continuació hi hauran un o més enters addicionals.
Per llegir les llistes, s'ha utilitzat l'operador >> que es troba definit a la classe `Llista`.

Sortida

Com a sortida es mostrarà la llista original. A continuació s'aniran inserint en ordre els enters addicionals de l'entrada, sempre a la llista modificada en el pas anterior. Si l'enter a inserir és positiu, després d'inserir-lo en ordre es mostrarà la llista actual recorrent-la del primer a l'últim i recorrent-la de l'últim al primer. Si no és positiu només s'insereix en ordre.
Per escriure les llistes, s'ha utilitzat l'operador << que es troba definit a la classe `Llista`.

Observació

Heu d'enviar la solució comprimida en un fitxer .tar:
`tar cvf program.tar llista_inserir_en_ordre.cpp`
Observeu que per compilar us donem el `Makefile`, la classe `Llista` amb tots els seus mètodes implementats excepte `inserir_en_ordre` i el programa principal `program.cpp`.

Exemple d'entrada 1

```
2 4
1 5 7 7
```

```
3
0
7
```

Exemple de sortida 1

```
[1,5,(7),7]>

[1,3,5,(7),7]>
[7,(7),5,3,1]<
```

```
[0,1,3,5,(7),7]>
[7,(7),5,3,1,0]<

[0,1,3,5,(7),7,7]>
[7,7,(7),5,3,1,0]<
```

Exemple d'entrada 2

```
2 4
2 5 5 7
3
5
6
7
9
```

Exemple de sortida 2

```
[2,5,(5),7]>

[2,3,5,(5),7]>
[7,(5),5,3,2]<

[2,3,(5),5,5,7]>
[7,5,5,(5),3,2]<

[2,3,(5),5,5,6,7]>
[7,6,5,5,(5),3,2]<

[2,3,(5),5,5,6,7,7]>
[7,7,6,5,5,(5),3,2]<

[2,3,(5),5,5,6,7,7,9]>
[9,7,7,6,5,5,(5),3,2]<
```

Exemple d'entrada 3

```
0 1
8
8
3
```

Exemple de sortida 3

```
[(8)]>

[(8),8]>
[8,(8)]<

[3,(8),8]>
[8,(8),3]<
```

Informació del problema

Autor : Neus Català - Jordi Esteve
Generació : 2020-05-10 20:48:28