Duplica els elements d'una llista simplement encadenada i circular deixant-la capicua X97099_ca

Donada la classe *Llista* que permet guardar seqüències d'enters amb una llista simplement encadenada, sense fantasma i circular, cal implementar el mètode

void *duplica_capicua* ();

que duplica tots els elements del paràmetre implícit obtenint una llista capicua.
Per exemple, si inicialment tenim aquesta llista:

```
[2 5 3 8 4]
```

després de cridar a *duplica_capicua*, la llista quedarà així:

```
[2 5 3 8 4 4 8 3 5 2]
```

Cal enviar a jutge.org la següent especificació de la classe *Llista* i la implementació del mètode dins del mateix fitxer. Al principi de cada mètode implementat, dins d'un comentari, cal indicar el cost temporal amb el raonament corresponent, incloent l'equació de la recurrència si fos necessari.

```
#include <iostream>
#include <sstream>
#include <vector>
#include <cstddef>
using namespace std;
typedef unsigned int nat;

class Llista {
  // Llista simplement encadenada, sense fantasma i circular.

  public:
    Llista ();
    // Pre: True
    // Post: El p.i. és una llista buida.

    Llista (const vector<int> &v);
    // Pre: True
    // Post: El p.i. conté els elements de v amb el mateix ordre.

    ~Llista ();
    // Post: Destrueix els elements del p.i.

    nat longitud() const;
    // Pre: True
    // Post: Retorna el nombre d'elements del p.i.

    void mostra() const;
    // Pre: True
```

```
    // Post: Mostra el p.i. pel canal estàndard de sortida.

    void duplica_capicua ();
    // Pre: True
    // Post: Es dupliquen tots els elements del p.i. obtenint una llista capicua
    // Exemple: [2 5 3 8 4] => [2 5 3 8 4 4 8 3 5 2]

  private:
    struct node {
      int info;   // Informació del node
      node *seg;  // Punter al següent element
    };
    node *_prim; // Punter al primer element
    nat _long;   // Nombre d'elements

    // Aquí va l'especificació dels mètodes privats addicionals

};

// Aquí va la implementació del mètode duplica_capicua i dels privats addicionals
```

Per testejar la solució, jutge.org ja té implementats la resta de mètodes de la classe *Llista* i un programa principal que processa línies d'enters amb els que crea llistes i després crida el mètode *duplica_capicua*.

Entrada

L'entrada conté vàries línies formades per seqüències d'enters. Cadascuna d'elles són els elements que tindrà cada llista.

Sortida

Per a cada línia d'entrada, escriu una línia amb el resultat després d'haver duplicat els elements deixant-los capicua: El nombre d'elements de la llista seguit d'un espai, els elements de la llista entre claudàtors i separats per espais.

Observació

Només cal enviar la classe requerida i la implementació del mètode *duplica_capicua*. Pots ampliar la classe amb mètodes privats. Segueix estrictament la definició de la classe de l'enunciat. **No es poden usar estructures de dades auxiliars com els vectors o arrays.**
Al principi de cada mètode implementat i dins d'un comentari cal indicar el cost temporal amb el raonament corresponent, incloent l'equació de la recurrència si fos necessari.

Exemple d'entrada 1	Exemple de sortida 1
	`0 []`

Exemple d'entrada 2	Exemple de sortida 2
`2` `-1`	`2 [2 2]` `2 [-1 -1]`

Exemple d'entrada 3

```
2 -5
-1 4
```

Exemple de sortida 3

```
4 [2 -5 -5 2]
4 [-1 4 4 -1]
```

Exemple d'entrada 4

```
2 -5 3
1 6 -9
```

Exemple de sortida 4

```
6 [2 -5 3 3 -5 2]
6 [1 6 -9 -9 6 1]
```

Exemple d'entrada 5

```
2 5 3 8 4
3 -6 8 0 4 3
```

Exemple de sortida 5

```
10 [2 5 3 8 4 4 8 3 5 2]
12 [3 -6 8 0 4 3 3 4 0 8 -6 3]
```

Informació del problema

Autoria: Jordi Esteve

Generació: 2026-01-25T17:34:44.237Z