Separació dels elements menors i majors a un donat d'una llista doblement encadenada, circular i amb fantasma X79722_ca

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

void *separa*(*Llista* &*l2*, **int** *x*)

que a partir d'una llista l2 buida, separa els elements del paràmetre implícit quedant al paràmetre implícit els elements menors a *x* i a l2 els elements majors a *x* amb el mateix ordre. Cal enviar a jutge.org només la implementació del mètode *separa*. Al principi del mètode implementat, dins d'un comentari, cal indicar el cost temporal amb el raonament corresponent, incloent l'equació de la recurrència si fos necessari. La classe *Llista* té la següent especificació:

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

class Llista {
  // Llista doblement encadenada, circular i amb fantasma.
  private:
    struct node {
      int info; // Informació del node
      node *seg; // Punter al següent element
      node *ant; // Punter a l'anterior element
    };
    node *_prim; // Punter a l'element fantasma
    nat _long; // Nombre d'elements

  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 mostra_invertida() const;
    // Pre: True
    // Post: Mostra el p.i. en ordre invers pel canal estàndard de sortida.

    void separa(Llista &l2, int x);
    // Pre: l2 és buida
    // Post: S'han separat els elements del p.i., quedant al p.i. els elements
    // menors a x i a l2 els elements majors a x amb el mateix ordre.
};
```

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, llegeix un enter i després crida el mètode *separa*.

Entrada

L'entrada conté vàris parells de línies: La primera línia de cada parell conté una seqüència d'enters amb els elements que tindrà la llista i la segona línia conté un enter amb el valor x a usar per separar la llista.

Sortida

Per a cada parell de línies d'entrada, escriu dues línies amb el resultat després d'haver separat els elements menors a x i els majors a x: Per cada llista s'escriu el nombre d'elements de la llista seguit d'un espai, els elements de la llista entre claudàtors i separats per espais, i finalment aquests mateixos elements però amb ordre invers, també entre claudàtors i separats per espais.

Observació

Cal enviar la solució (el fitxer *solution.cpp*) comprimida en un fitxer *.tar*:

```
tar cvf solution.tar solution.cpp
```

Només cal enviar la implementació del mètode *separa*. Seguiu estrictament la definició de la classe de l'enunciat.
Al principi del mètode implementat, 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

```
1
```

Exemple de sortida 1

```
0 [] []
0 [] []
```

Exemple d'entrada 2

```
5
6
```

Exemple de sortida 2

```
1 [5] [5]
0 [] []
```

Exemple d'entrada 3

```
5
4
```

Exemple de sortida 3

```
0 [] []
1 [5] [5]
```

Exemple d'entrada 4

```
5
5
```

Exemple de sortida 4

```
0 [] []
0 [] []
```

Exemple d'entrada 5

```
9 7
9
```

Exemple de sortida 5

```
1 [7] [7]
0 [] []
```

Exemple d'entrada 6

```
3 -6 8 0 4 -2 0
1
```

Exemple de sortida 6

```
4 [-6 0 -2 0] [0 -2 0 -6]
3 [3 8 4] [4 8 3]
```

Exemple d'entrada 7

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

Exemple de sortida 7

```
2 [-6 -2] [-2 -6]
3 [3 8 4] [4 8 3]
```

Informació del problema

Autoria: Jordi Esteve

Generació: 2026-01-25T21:28:55.039Z