

---

**Arbre binari. Calcula arbre amb factors de pes**

---

**X53343\_ca**

Donada la classe *Abin* que permet gestionar arbres binaris usant memòria dinàmica, cal implementar el mètode

```
void arbre_factors_pes ();
```

que modifica el contingut de l'arbre per tal de guardar a cada node el factor de pes (diferència entre la quantitat de nodes del fill esquerra i la quantitat de nodes del fill dret).

Cal enviar a jutge.org la següent especificació de la classe *Abin* i la implementació del mètode dins del mateix fitxer. Indica dins d'un comentari a la capçalera del mètode el seu cost en funció del nombre d'elements  $n$  de l'arbre.

```
include <cstdlib>
#include <iostream>
using namespace std;
typedef unsigned int nat;

template <typename T>
class Abin {
public:
    Abin(): _arrel (NULL) {};
    // Pre: cert
    // Post: el resultat és un arbre sense cap element
    Abin(Abin<T> &ae, const T &x, Abin<T> &ad);
    // Pre: cert
    // Post: el resultat és un arbre amb un element i dos subarbres

    // Les tres grans
    Abin(const Abin<T> &a);
    ~Abin();
    Abin<T>& operator=(const Abin<T>& a);

    // operador ;; d'escriptura
    template <class U> friend std::ostream& operator<<(std::ostream&, const Abin<U> &a);

    // operador << de lectura
    template <class U> friend std::istream& operator>>(std::istream&, Abin<U> &a);

    // Modifica el contingut de l'arbre per tal de guardar a cada node el factor
    // de pes (diferència entre quantitat nodes fill esquerra i quantitat nodes fill dret).
    void arbre_factors_pes ();

private:
    struct node {
        node* f_esq ;
        node* f_dret ;
        T info ;
```

```

};
node* _arrel ;
static node* copia_nodes (node* m);
static void esborra_nodes (node* m);
static void print_nodes (node* m, ostream &os, string d1);

// Aquí va l'especificació dels mètodes privats addicionals
};

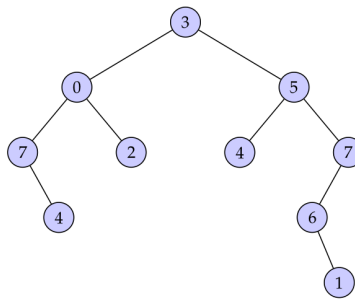
// Aquí va la implementació del mètode arbre_factors_pes

```

Per testejar la solució, jutge.org ja té implementats la resta de mètodes de la classe *Abin* i un programa principal que llegeix un arbre binari i després crida el mètode *arbre\_factors\_pes*.

## Entrada

L'entrada consisteix en la descripció d'un arbre binari d'enters (el seu recorregut en preordre, en el qual inclou les fulles marcades amb un -1). Per exemple, l'arbre (mira el PDF de l'enunciat)



es descriuria amb

```
3 0 7 -1 4 -1 -1 2 -1 -1 5 4 -1 -1 7 6 -1 1 -1 -1 -1
```

## Sortida

El contingut de l'arbre binari abans i després de cridar el mètode *arbre\_factors\_pes*.

## Observació

Només cal enviar la classe requerida i la implementació del mètode *arbre\_factors\_pes* amb el seu cost en funció del nombre d'elements  $n$  de l'arbre. Pots ampliar la classe amb mètodes privats. Segueix estrictament la definició de la classe de l'enunciat.

### Exemple d'entrada 1

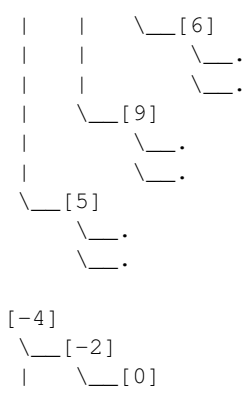
```
7 5 -1 -1 8 9 -1 -1 4 6 -1 -1 3 -1 -1
```

### Exemple de sortida 1

```

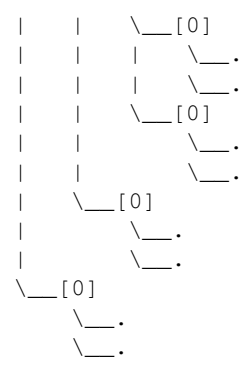
[7]
 \__[8]
  |  \__[4]
  |   |  \__[3]
  |   |   |  \__
  |   |   |   \__
  |   |   |   \__

```

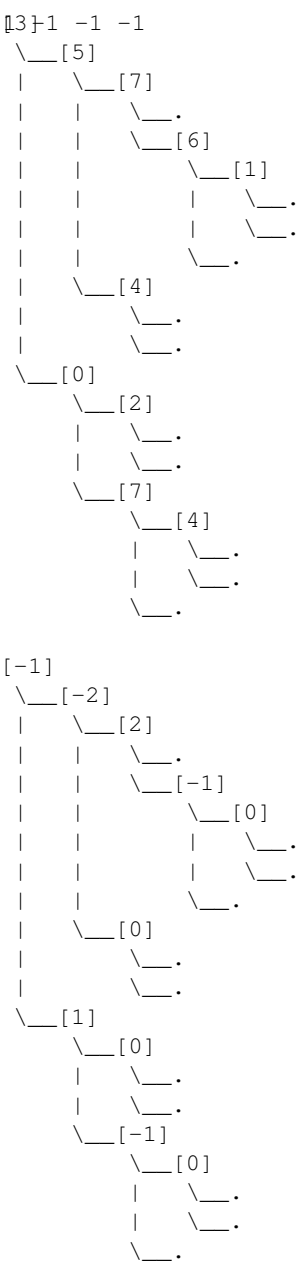


### Exemple d'entrada 2

3 0 7 -1 4 -1 -1 2 -1 -1 5 4 -1 -1 7 6 -1



### Exemple de sortida 2



### Exemple d'entrada 3

-1

### Exemple d'entrada 4

3 -1 -1

### Exemple d'entrada 5

3 2 -1 -1 -1

### Exemple d'entrada 6

3 -1 2 -1 -1

### Exemple d'entrada 7

-3 -2 -1 -1 -4 -1 -1

### Exemple de sortida 3

.  
.

### Exemple de sortida 4

[3]  
 \\_.  
 \\_.  
  
[0]  
 \\_.  
 \\_.

### Exemple de sortida 5

[3]  
 \\_.  
 \\_[2]  
 \\_.  
 \\_.  
  
[1]  
 \\_.  
 \\_[0]  
 \\_.  
 \\_.

### Exemple de sortida 6

[3]  
 \\_[2]  
 | \\_.  
 | \\_.  
 \\_.  
  
[-1]  
 \\_[0]  
 | \\_.  
 | \\_.  
 \\_.

### Exemple de sortida 7

[-3]  
 \\_[-4]  
 | \\_.  
 | \\_.  
 \\_-2]  
 \\_.  
 \\_.  
  
[0]  
 \\_[0]  
 | \\_.  
 | \\_.  
 \\_[0]  
 \\_.  
 \\_.

## **Informació del problema**

Autor : Jordi Esteve

Generació : 2025-10-21 20:51:32

© *Jutge.org*, 2006–2025.

<https://jutge.org>