
Arbre binari. Crea arbre binari complet de n nivells X75469_ca

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

```
Abin(nat n);
```

que crea un arbre binari complet amb n nivells, on la informació de cada node de l'arbre és el nivell a on està situat.

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 de n , així com l'equació de recurrència que t'ha permès deduir el seu cost.

```
#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);

    Abin(nat n);
    // Pre: cert
    // Post: Crea un arbre binari complet amb n nivells, on la informació
    // de cada node de l'arbre és el nivell a on està situat

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 Abin(nat n) i privats addicionals

```

Per testejar la solució, judge.org ja té implementats la resta de mètodes de la classe *Abin* i un programa principal que llegeix un natural i després crida el mètode *Abin(nat n)*.

Entrada

L'entrada consisteix en un natural.

Sortida

El contingut de l'arbre binari després de cridar el mètode *Abin(nat n)*.

Observació

Només cal enviar la classe requerida, la implementació del mètode *Abin(nat n)* i el seu cost en funció de n , així com l'equació de recurrència que t'ha permès deduir el seu cost. Pots ampliar la classe amb mètodes privats. Segueix estrictament la definició de la classe de l'enunciat.

Exemple d'entrada 1

0

Exemple de sortida 1

.

Exemple d'entrada 2

1

Exemple de sortida 2

```
[1]
 \_
 \_
```

Exemple d'entrada 3

2

Exemple de sortida 3

```
[1]
 \_[2]
 | \_
 | \_
 \_[2]
   \_
   \_
```

Exemple d'entrada 4

3

Exemple de sortida 4

```
[1]
```

```

\_ [2]
|  \_ [3]
|  |  \_ .
|  |  \_ .
|  \_ [3]
|  \_ .
|  \_ .
|

```

Exemple d'entrada 5

4

```

\_ [2]
|  \_ [3]
|  \_ .
|  \_ .
|  \_ [3]
|  \_ .
|  \_ .
|

```

Exemple de sortida 5

```

[1]
\_ [2]
|  \_ [3]
|  |  \_ [4]
|  |  |  \_ .
|  |  |  \_ .
|  |  \_ [4]
|  |  \_ .
|  \_ [3]
|  \_ [4]
|  |  \_ .
|  |  \_ .
|  \_ [4]
|  \_ .
|  \_ .
|
\_ [2]
|  \_ [3]
|  |  \_ [4]
|  |  |  \_ .
|  |  |  \_ .
|  |  \_ [4]
|  |  \_ .
|  \_ [3]
|  \_ [4]
|  |  \_ .
|  |  \_ .
|  \_ [4]
|  \_ .
|  \_ .
|

```

Exemple d'entrada 6

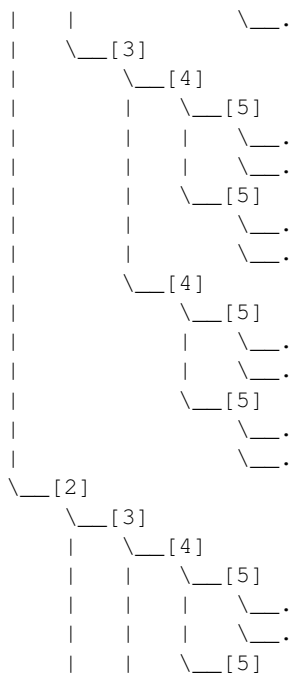
5

```

[1]
\_ [2]
|  \_ [3]
|  |  \_ [4]
|  |  |  \_ [5]
|  |  |  |  \_ .
|  |  |  |  \_ .
|  |  |  \_ [5]
|  |  |  \_ .
|  |  \_ [4]
|  |  \_ [5]
|  |  |  \_ .
|  |  |  \_ .
|  |  \_ [5]
|  |  \_ .
|

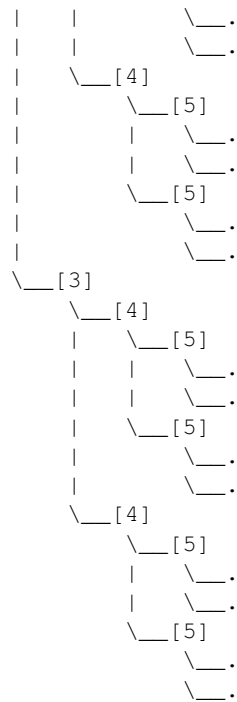
```

Exemple de sortida 6

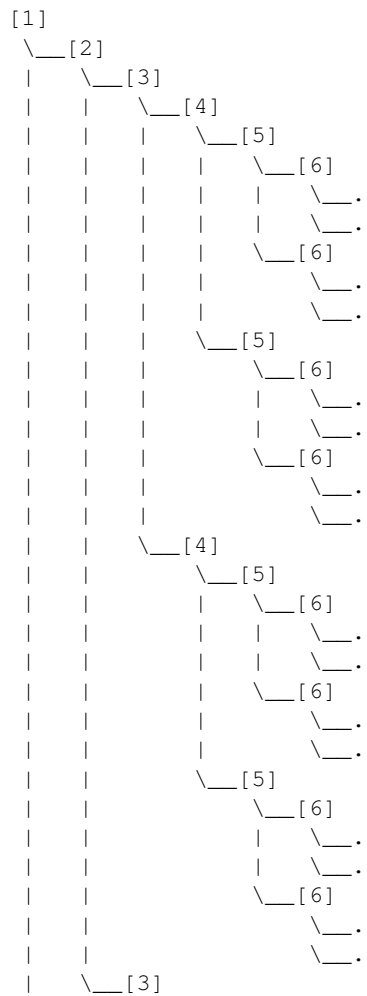


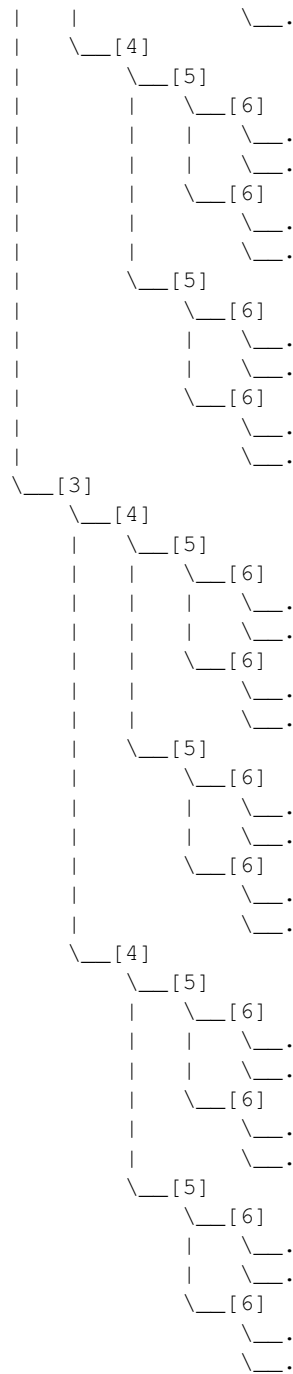
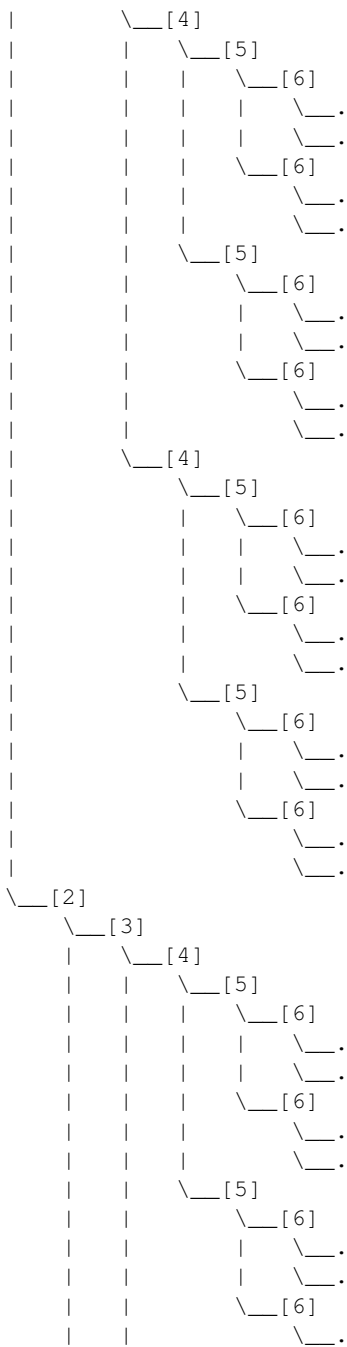
Exemple d'entrada 7

6



Exemple de sortida 7





Informació del problema

Autor : Jordi Esteve
 Generació : 2022-10-20 12:11:14

© *Jutge.org*, 2006–2022.
<https://jutge.org>