N Elements endarrera a un BST — X59997_ca

Donada la classe *dicc* que permet gestionar diccionaris on només hi guardem claus úniques usant arbres binaris de cerca (BST), cal implementar el mètode

```
// Retorna la enésima clau del arbre comenza desde el final i va
// cap endarrera.
// Pre: El diccionari té com a mínim 1 clau.
Clau n_endarrera(int i) const;
```

Que retorna la enèsima clau del diccionari començant per el final i comptant cap a endarrere. D'aquesta manera, el mètode amb paràmetre 1 retorna la clau més gran del diccionari, amb paràmetre 2 la segona més gran i així successivament.
Les claus són del tipus *Clau* que admet una relació d'ordre total, és a dir, tenim una operació de comparació $<$ entre claus.
Cal enviar a jutge.org la següent especificació de la classe *dicc* i la implementació del mètode dins del mateix fitxer. La resta de mètodes públics i privats ja estan implementats.
Per resoldre el problema, pots editar el fitxer *solution.cc* posant-hi el teu codi on s'indica. Afegeix els mètodes privats que consideris necessaris.

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

template <typename Clau>
class dicc {

  public:
    // Constructora per defecte. Crea un diccionari buit.
    dicc();

    // Destructora
    ~dicc();

    // Insereix la clau k en el diccionari. Si ja hi era, no fa res.
    void insereix(const Clau &k);

    // Retorna quants elements (claus) té el diccionari.
    nat quants() const;

        // Retorna la enésima clau del arbre comenza desde el final i va
        // cap endarrera.
        // Pre: El diccionari té com a mínim 1 clau.
        Clau n_endarrera(int i) const;

  private:
    struct node {
      Clau _k;        // Clau
      node* _esq;     // fill esquerre
```

```
    node* _dret ;   // fill dret
    nat _n;         // Nombre de nodes del subarbre
    node(const Clau &k, node* esq = NULL, node* dret = NULL);
  };
  node * _arrel ;

  static void esborra_nodes (node* m);
  static node* insereix_bst (node *n, const Clau &k, bool &ins);

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

// Aquí va la implementació dels mètodes públics i privats
```

Degut a que jutge.org només permet l'enviament d'un fitxer amb la solució del problema, en el mateix fitxer hi ha d'haver l'especificació de la classe i la implementació del mètode *n_endarrera* (el que normalment estarien separats en els fitxers *.hpp* i *.cpp*).
Per testejar la classe disposes d'un programa principal que processa blocs que contenen un diccionari amb claus enteres seguit de comandes per cerca la clau enèsima del diccionari.

Entrada

L'entrada conté varis blocs separats per línies amb 10 guions (————). Cada bloc consisteix en una línia que conté una seqüències d'enters, són els elements que tindrà originalment el diccionari. A continuació segueixen vàries comandes, una per línea, amb el següent format (i es un natural major que 0):

- n_endarrera i

Sortida

Per a cada línia d'entrada, escriu una línia amb el resultat:

- Si la línia és un diccionari, mostra el nombre de claus del diccionari un cop inserit tots els seus elements.
- Si la línia és una comanda, mostra la comanda, el separador ": " i el resultat.
- Si la línia és el separador de blocs format per 10 guions, mostra els mateixos 10 guions.

Observació

Només cal enviar la classe requerida i la implementació del mètode *n_endarrera*. Pots ampliar la classe amb mètodes privats. Segueix estrictament la definició de la classe de l'enunciat. El mètode *n_endarrera* almenys ha de tenir cost logarítmic respecte el nombre de claus del diccionari (en el cas mig) per superar els jocs de prova privats.

Exemple d'entrada sample-1

```
17 6 3 1 2 5 7 8
n_endarrera 1
----------
17 6 3 1 2 5 7 8
n_endarrera 5
----------
1 2 3 5 6 7 8 17
n_endarrera 5
----------
15 2 25 26
n_endarrera 3
----------
5 2
n_endarrera 2
----------
5 -3 2 -1 7 -7 -6
n_endarrera 1
----------
5 -3 2 -1 -7 -6
n_endarrera 2
----------
-3 2 -1 -7 -6
n_endarrera 2
----------
-3
n_endarrera 1
```

Exemple de sortida sample-1

```
8
n_endarrera 1: és 17
----------
8
n_endarrera 5: és 5
----------
8
n_endarrera 5: és 5
----------
4
n_endarrera 3: és 15
----------
2
n_endarrera 2: és 2
----------
7
n_endarrera 1: és 7
----------
6
n_endarrera 2: és 2
----------
5
n_endarrera 2: és -1
----------
1
n_endarrera 1: és -3
```

Exemple d'entrada sample-2

```
5 7
n_endarrera 1
----------
7 5
n_endarrera 1
```

Exemple de sortida sample-2

```
2
n_endarrera 1: és 7
----------
2
n_endarrera 1: és 7
```

Informació del problema

Autor : Ignasi Gómez-Sebastià
Generació : 2021-04-24 21:50:37