Intersecció de dos BSTs — X94927_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

```
vector<Clau> interseccio(const dicc<Clau> &d2) const;
// Pre: True
// Post: Retorna un vector amb les claus de la intersecció del p.i. amb d2
// ordenades de menor a major
```

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. Indica dins d'un comentari a la capçalera del mètode el seu cost en funció del nombre d'elements $n1$ del diccionari del p.i. i nombre d'elements $n2$ del diccionari $d2$.

```
#include <iostream>
#include <vector>
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);

    vector<Clau> interseccio(const dicc<Clau> &d2) const;
    // Pre: True
    // Post: Retorna un vector amb les claus de la intersecció del p.i. amb d2
    // ordenades de menor a major

  private:
    struct node {
      Clau _k;        // Clau
      node* _esq;     // fill esquerre
      node* _dret;    // fill dret
      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);

    // 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 *interseccio* (el que normalment estarien separats en els fitxers *.hpp* i *.cpp*).
Per testejar la classe disposes d'un programa principal que llegeix dos diccionaris d'enters, després crida el mètode *interseccio* i finalment mostra el contingut del vector amb la intersecció dels dos diccionaris.

Entrada

L'entrada conté dues línies formades per seqüències d'enters, són els elements que tindran els dos diccionaris.

Sortida

A la sortida apareixen ordenats i separats per espais, els elements de la intersecció dels dos diccionaris.

Observació

Només cal enviar l'especificació de la classe *dicc*, la implementació del mètode *interseccio* i el seu cost en funció del nombre d'elements $n1$ i $n2$ dels dos diccionaris inicials. Pots ampliar la classe amb mètodes privats. Segueix estrictament la definició de la classe de l'enunciat.

Exemple d'entrada 1

```
5 -3 8 2 -1 7 -7 -6
7 -2 9 5 -3 2 -7
```

Exemple de sortida 1

```
-7 -3 2 5 7
```

Exemple d'entrada 2

```
5 -5 -3 9 -9 2 -2 1 -1 7 -7 0 4 -4 8 -8 6
2 10 4 12 8 14 0 10 14 6
```

Exemple de sortida 2

```
0 2 4 6 8
```

Exemple d'entrada 3

```
5 -3 8 2 -1 7 -7 -6
```

Exemple de sortida 3

Exemple d'entrada 4

```
7 -2 9 5 -3 2 -7
```

Exemple de sortida 4

Exemple d'entrada 5

Exemple de sortida 5

Informació del problema

Autoria: Jordi Esteve

Generació: 2026-01-25T17:26:18.131Z