
Inserir i comptar claus repetides en un BST

X90310_ca

Donada la classe *dicc* que permet gestionar diccionaris usant arbres binaris de cerca (BST) on les claus poden estar **repetides**, cal implementar els mètodes

```
void insereix (const Clau &k);  
// Pre: Cert  
// Post: Insereix la clau k en el diccionari.
```

```
nat quantes(const Clau &k) const;  
// Pre: Cert  
// Post: Retorna el nombre de claus iguals a k
```

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 dels mètodes dins del mateix fitxer. La resta de mètodes públics i privats ja estan implementats.

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

```
template <typename Clau>
```

```
class dicc {
```

```
    // Diccionari implementat en un BST on les claus poden estar repetides.
```

```
    public:
```

```
        // Constructora per defecte. Crea un diccionari buit.
```

```
        dicc ();
```

```
        // Destructora
```

```
        ~dicc ();
```

```
        // Imprimeix el contingut del diccionari: Nombre d'elements i
```

```
        // totes les claus de més petita a més gran separades per un espai
```

```
        void print () const;
```

```
        void insereix (const Clau &k);
```

```
        // Pre: Cert
```

```
        // Post: Insereix la clau k en el diccionari.
```

```
        nat quantes(const Clau &k) const;
```

```
        // Pre: Cert
```

```
        // Post: Retorna el nombre de claus iguals a k
```

```
    private:
```

```
        struct node {
```

```

    Clau _k;        // Clau
    node* _esq;    // fill esquerre
    node* _dret;   // fill dret
};
node * _arrel ;
nat _n;          // Nombre d'elements del diccionari

// Elimina els nodes del subarbre apuntat per p
static void esborra_nodes (node* p);
// Imprimeix ordenades les claus del subarbre apuntat per p
static void print (node* p);

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

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

```

Degut a que `judge.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 dels mètodes `insereix` i `quantas` (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 comptar quantes claus són iguals a una donada.

Entrada

L'entrada conté varis blocs separats per línies amb 10 guions (———). Cada bloc consisteix en una línia que conté una seqüència d'enters, són els elements que tindrà originalment el diccionari. A continuació segueixen diverses comandes, una per línia, amb el següent format (clau és un enter):

- `quantas clau`

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 i totes les claus de més petita a més gran separades per un espai.
- 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 dels mètodes `insereix` i `quantas`. Pots ampliar la classe amb mètodes privats. Segueix estrictament la definició de la classe de l'enunciat.

Els mètodes `insereix` i `quantas` almenys han de tenir cost logarítmic (en el cas mig) per superar els jocs de prova privats.

Exemple d'entrada 1

quantas -6
quantas 0

Exemple de sortida 1

0:
quantas -6: 0
quantas 0: 0

Exemple d'entrada 2

-6
quantas -6
quantas 0

Exemple de sortida 2

1: -6
quantas -6: 1
quantas 0: 0

Exemple d'entrada 3

0 -6
quantas 0
quantas -6
quantas 6

Exemple de sortida 3

2: -6 0
quantas 0: 1
quantas -6: 1
quantas 6: 0

Exemple d'entrada 4

-6 -6
quantas 0
quantas -6
quantas 6

Exemple de sortida 4

2: -6 -6
quantas 0: 0
quantas -6: 2
quantas 6: 0

Exemple d'entrada 5

5 -3 8 2 -1 7 -7 -6
quantas 8
quantas 5
quantas -7
quantas 0

Exemple de sortida 5

8: -7 -6 -3 -1 2 5 7 8
quantas 8: 1
quantas 5: 1
quantas -7: 1
quantas 0: 0

Exemple d'entrada 6

5 -3 8 5 -3 7 5 -8
quantas 8
quantas 5
quantas -3
quantas 0

Exemple de sortida 6

8: -8 -3 -3 5 5 5 7 8
quantas 8: 1
quantas 5: 3
quantas -3: 2
quantas 0: 0

Exemple d'entrada 7

5 -5 -3 9 -5 5 -2 1 -3 9 -5 0 4 -5 5 -3 4
quantas -5
quantas -3
quantas -2
quantas 0
quantas 1
quantas 2
quantas 4
quantas 5
quantas 9

4 5 5 5 5 5 5 5 5 5 5 5 5
quantas 0
quantas 4
quantas 5

Exemple de sortida 7

17: -5 -5 -5 -5 -3 -3 -3 -2 0 1 4 4 5 5 9 9
quantas -5: 4
quantas -3: 3
quantas -2: 1
quantas 0: 1
quantas 1: 1
quantas 2: 0
quantas 4: 2
quantas 5: 3
quantas 9: 2

14: 4 5 5 5 5 5 5 5 5 5 5 5 5
quantas 0: 0
quantas 4: 1
quantas 5: 13

Informació del problema

Autor : Jordi Esteve

Generació : 2022-06-08 16:47:42

© *Jutge.org*, 2006–2022.

<https://jutge.org>