

---

## Intersecció de taules de dispersió amb sinònims encadenats indirectes

X89802\_ca

---

Donada la classe *dicc* que permet gestionar diccionaris amb claus enteres, cal implementar els mètodes:

```
// Pre: Cert
// Post: Insereix la clau k en el diccionari. Si ja hi era, no fa res.
void insereix (const int &k);

// Pre: El diccionari res està buit
// Post: Omple res amb la intersecció entre el p.i. i d2
void interseccio (const dicc &d2, dicc &res) const;
```

Els diccionaris s'implementen amb taules de dispersió amb sinònims encadenats indirectes. Les llistes de sinònims estan ordenades per la clau. 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 ja estan implementats).

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

class dicc {
// Taula de dispersió sinònims encadenats indirectes
// Les llistes de sinònims estan ordenades per clau
public:
// Constructora per defecte. Crea un diccionari buit.
dicc ();

// Destructora
~dicc ();

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

// Impressió per cout de totes les claus del diccionari segons l'ordre
// en que estan a cadascuna de les llistes encadenades indirectes
void print () const;

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

// Pre: El diccionari res està buit
// Post: Omple res amb la intersecció entre el p.i. i d2
void interseccio (const dicc &d2, dicc &res) const;
```

```

private:
struct node_hash {
    int k; // Clau
    node_hash* seg;
};
node_hash ** taula; // Taula amb punters a les llistes de sinònims
nat M; // Mida de la taula
nat quants; // N° d'elements guardats al diccionari

static long const MULT = 31415926;

// Calcula un valor de dispersió entre 0 i LONG_MAX a partir de k
static long h(int k);

// Destruïx la llista de nodes apuntats per p
static void esborra_nodes (node_hash *p);

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

// Aquí va la implementació dels mètodes públics insereix, interseccio i
// dels mètodes privats addicionals
};

```

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 dels mètodes *insereix* i *interseccio* (el que normalment estarien separats en els fitxers *.hpp* i *.cpp*).

Per testejar la classe disposes d'un programa principal que llegeix elements i els insereix en un diccionari, després fa el mateix en un 2on diccionari i finalment crida el mètode *interseccio* per calcular la intersecció dels dos diccionaris llegits.

## Entrada

L'entrada conté dues línies amb enters separats amb espais. Cada línia conté els elements a inserir en un diccionari buit.

## Sortida

Escriu el contingut de 3 diccionaris: el 1er diccionari llegit, el 2on diccionari llegit i el diccionari intersecció dels dos primers. Per cada diccionari es mostra en diferents línies la quantitat d'elements que té i els elements que conté cadascuna de les llistes de sinònims encadenats indirectes (els elements de cada llista apareixen separats amb un espai i en el mateix ordre en que es guarden).

## Observació

Per calcular el valor de dispersió utilitza el mètode *h* que ja està implementat i que permet calcular un valor de dispersió entre 0 i *LONG\_MAX* (el valor long int més gran que permet el compilador) a partir d'una clau entera.

En aquest exercici s'ha limitat la mida de la taula de dispersió  $M$  a 13. En un cas real aquest valor seria molt més gran i/o s'implementaria la tècnica de redispersió per tal que la mida s'anés adaptant a les necessitats de cada moment.

Només cal enviar la classe requerida i la implementació dels mètodes *insereix* i *interseccio*. 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

```
Nº elem: 8
0:
1: -1
2:
3:
4: 2
5:
6:
7:
8:
9: -3
10: -7 -6 7
11:
12: 5 8
```

```
-----
Nº elem: 7
0:
1:
2:
3: 9
4: -2 2
5:
6:
7:
8:
9: -3
10: -7 7
11:
12: 5
```

```
-----
Nº elem: 5
0:
1:
2:
3:
4: 2
5:
6:
7:
8:
9: -3
10: -7 7
11:
12: 5
```

### Exemple d'entrada 2

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

### Exemple de sortida 2

```
Nº elem: 619
0: 0
1: -1 1
```

```
2:
3: -9 -4 4 9
4: -2 2
5:
6:
7:
8:
9: -3 3
10: -7 -6 6 7
11:
12: -8 -5 5 8
-----
Nº elem: 8
0: 0
1: 12 14
2:
3: 4
4: 2
5:
6:
7:
```

```
8:
9: 10
10: 6
11:
12: 8
-----
Nº elem: 5
0: 0
1:
2:
3: 4
4: 2
5:
6:
7:
8:
9:
10: 6
11:
12: 8
-----
```

## Informació del problema

Autor : Jordi Esteve

Generació : 2022-01-27 10:13:15

© *Jutge.org*, 2006–2022.

<https://jutge.org>