

---

## Penúltima clau d'un diccionari BST

X35140\_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

```
Clau penultima() const;
// Pre: El diccionari té com a mínim dues claus.
// Post: Retorna la penúltima clau.
```

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.

```
#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 penúltima clau.
    // Pre: El diccionari té com a mínim 2 claus.
    Clau penultima() 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 `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 del mètode *penultima* (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 calcular la penúltima clau.

## 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:

- *penultima*

## 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 *penultima*. Pots ampliar la classe amb mètodes privats. Segueix estrictament la definició de la classe de l'enunciat. El mètode *penultima* almenys ha de tenir cost logarítmic (en el cas mig) per superar els jocs de prova privats.

### Exemple d'entrada 1

```

5 7
penultima

```

### Exemple d'entrada 2

```

7 5
penultima

```

### Exemple d'entrada 3

```

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

```

### Exemple de sortida 1

```

2
penultima: 5

```

### Exemple de sortida 2

```

2
penultima: 5

```

```

5 -3 2 -1 7 -7 -6
penultima
-----
5 -3 2 -1 -7 -6

```

```
penultima
-----
-3 2 -1 -7 -6
penultima
```

### Exemple de sortida 3

```
8
penultima: 7
-----
7
penultima: 5
-----
6
penultima: 2
-----
5
penultima: -1
```

### Informació del problema

Autoria: Jordi Esteve

Generació: 2026-01-25T15:22:04.933Z

© *Jutge.org*, 2006–2026.

<https://jutge.org>