Trie TST. Compta les claus que satisfan un cert patró. S35955_ca

Donada la classe *dicc* que permet gestionar diccionaris on només hi guardem claus úniques usant tries implementats amb la tècnica d'arbres ternaris de cerca (TST), cal implementar el mètode

```
nat quantes_satisfan_patro (string patro) const;
// Pre: patro conté símbols de la A a la Z o el símbol comodí *
// Post: Retorna el nº de claus que coincideixen amb el patro donat.
// El símbol * equival a qualsevol símbol de la A a la Z.
```

Les claus són del tipus string i els símbols utilitzats per construir el trie són els caràcters de les claus. S'ha usat el caràcter especial '#' per indicar la fi de la clau.
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;

class dicc {
  // Diccionari implementat amb un TST on les claus són strings i els
  // símbols de la clau són chars. El símbol indicador fi de clau és el #
  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 string &k);

    nat quantes_satisfan_patro (string patro) const;
    // Pre: patro conté símbols de la A a la Z o el símbol comodí *
    // Post: Retorna el nº de claus que coincideixen amb el patro donat.
    // El símbol * equival a qualsevol símbol de la A a la Z.

  private:
    struct node {
      char _c;      // Símbol posició i-èssima de la clau
      node* _esq; // Fill esquerra, apunta a símbols mateixa posició formant un BST
      node* _cen; // Fill central, apunta a símbols següent posició
      node* _dre; // Fill dret, apunta a símbols mateixa posició formant un BST
      node(const char &c, node* esq = nullptr, node* cen = nullptr, node* dre = nullptr);
    };
    node* _arrel;
```

```
    static void esborra_nodes(node* t);
    static node* insereix(node *t, nat i, const string &k);

    // Aquí va l'especificació dels mètodes privats addicionals

};

// Aquí va la implementació del mètode públic 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 del mètode *quantes_satisfan_patro* (el que normalment estarien separats en els fitxers *.hpp* i *.cpp*).
Per testejar la classe disposes d'un programa principal que insereix claus en un diccionari i després compta quantes coincideixen amb cadascun dels patrons donats.

Entrada

L'entrada conté dos blocs separats per una línia amb 10 guions (————). El primer bloc consisteix en una llista de strings: són les claus que tindrà el diccionari. El segon bloc consisteix en una altra llista de strings: són els patrons per comptar les claus del diccionari que els satisfan.

Sortida

Per a cada string d'entrada del segon bloc, escriu una línia amb el nombre de claus que satisfna aquest patró, el text " satisfan patró " i l'string d'entrada.

Observació

Només cal enviar la classe requerida i la implementació del mètode *quantes_satisfan_patro*.
Pots ampliar la classe amb mètodes privats. Segueix estrictament la definició de la classe de l'enunciat.
Per superar els jocs de prova privats, el mètode *quantes_satisfan_patro* ha de visitar només els nodes del trie imprescindibles.

Exemple d'entrada 1

```
----------
Z

*
```

Exemple de sortida 1

```
0 satisfan patró Z
0 satisfan patró
0 satisfan patró *
```

Exemple d'entrada 2

```
OCA
----------
O
OC
OCA
OCAS
****
*
*C
```

```
*CA
*CA*
O*
O*A
O*A*
**
**A
O**
***
****
```

Exemple de sortida 2

```
0 satisfan patró O
0 satisfan patró OC
1 satisfan patró OCA
0 satisfan patró OCAS
0 satisfan patró ****
0 satisfan patró *
0 satisfan patró *C
1 satisfan patró *CA
0 satisfan patró *CA*
0 satisfan patró O*
1 satisfan patró O*A
0 satisfan patró O*A*
0 satisfan patró **
1 satisfan patró **A
1 satisfan patró O**
1 satisfan patró ***
0 satisfan patró ****
```

Exemple d'entrada 3

```
CASA
CAS
----------
C
CA
CAS
CASA
CASAL
CA*
*AS
C**
**S
***
***A
***E
*AS*
C***
****L
****
*****
```

Exemple de sortida 3

```
0 satisfan patró C
0 satisfan patró CA
1 satisfan patró CAS
1 satisfan patró CASA
0 satisfan patró CASAL
1 satisfan patró CA*
1 satisfan patró *AS
1 satisfan patró C**
1 satisfan patró **S
1 satisfan patró ***
1 satisfan patró ***A
0 satisfan patró ***E
1 satisfan patró *AS*
1 satisfan patró C***
0 satisfan patró ****L
1 satisfan patró ****
0 satisfan patró *****
```

Exemple d'entrada 4

```
DAU
DEU
AU
AVI
CASA
COP
CAP
CAPA
OU
OLA
UN
EXTRA
FUM
FOC
ILLA
ALA
AL
----------
CAP
CAPA
CAPAR
CA*A
CA*AR
CA*
C*P
C*PA
C*P*
DAU
DAUS
D*U
D**
D**S
A**
**
***
****
*****
******
```

Exemple de sortida 4

```
1 satisfan patró CAP
1 satisfan patró CAPA
0 satisfan patró CAPAR
2 satisfan patró CA*A
0 satisfan patró CA*AR
1 satisfan patró CA*
2 satisfan patró C*P
1 satisfan patró C*PA
1 satisfan patró C*P*
1 satisfan patró DAU
0 satisfan patró DAUS
2 satisfan patró D*U
2 satisfan patró D**
0 satisfan patró D**S
2 satisfan patró A**
4 satisfan patró **
9 satisfan patró ***
3 satisfan patró ****
1 satisfan patró *****
0 satisfan patró ******
```

Exemple d'entrada 5

```
A

I
----------
A
E
I
*

**
```

Exemple de sortida 5

```
1 satisfan patró A
0 satisfan patró E
1 satisfan patró I
2 satisfan patró *
1 satisfan patró
0 satisfan patró **
```

Informació del problema

Autoria: Jordi Esteve

Generació: 2026-01-25T12:44:16.007Z