Trie primer fill-següent germà. Compta les claus que comencen per algun dels caràcters donats. X30408_ca

Donada la classe *dicc* que permet gestionar diccionaris on només hi guardem claus úniques usant tries implementats amb la tècnica d'arbres generals amb punters a primer fill i següent germà, cal implementar el mètode

```
nat quantes_comencen(string inicials) const;
// Pre: inicials no conté el char '#'
// Post: Retorna el nº de claus que comencen per algun dels caràcters que conté inicials
```

Les claus són del tipus string i els símbols utilitzats per construir el trie són els chars de les claus. S'ha usat el char especial '#' per indicar la fi de la clau. Els símbols dels nodes germans estan ordenats de menor a major.
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 {
  public:
    dicc (); // Constructora per defecte. Crea un diccionari buit.

    ~dicc (); // Destructora

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

    nat quantes_comencen(string inicials) const;
    // Pre: inicials no conté el char '#'
    // Post: Retorna el nº de claus que comencen per algun dels caràcters que conté inicials

  private:
    struct node {
      char _c; // Símbol posició i-èssima de la clau
      node* _pf; // Primer fill, apunta a símbols de la següent posició
      node* _sg; // Següent germà, apunta a símbols de la mateixa posició
      node(const char &c, node* pf = NULL, node* sg = NULL);
    };
    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 quantes_comencen i 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_comencen* (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 comencen per algun dels caràcters de diferents strings.

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: cada string conté els caràcters inicials de les claus que volem comptar del diccionari.

Sortida

Per a cada string d'entrada del segon bloc, escriu una línia amb el nombre de claus que comencen per algun dels caràcters que conté aquest string, el text " comencen per " i l'string d'entrada.

Observació

Només cal enviar la classe requerida i la implementació del mètode *quantes_comencen*. 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_comencen* ha de visitar només els nodes del trie imprescindibles.

Exemple d'entrada 1

```
DAU
DIT
AU
AVI
CASA
COP
CAP
CAPA
OU
OLA
UN
EXTRA
FUM
FOC
ILLA
ALA
AL
----------
A
E
I
O
U
AEIOU
BCDFGHJKLMNPQRSTVWXYZ
DMF
```

Exemple de sortida 1

```
4 comencen per A
1 comencen per E
1 comencen per I
2 comencen per O
1 comencen per U
9 comencen per AEIOU
8 comencen per BCDFGHJKLMNPQRSTVWXYZ
4 comencen per DMF
```

Exemple d'entrada 2

```
----------
A
AEIOU
BCDFGHJKLMNPQRSTVWXYZ
```

Exemple de sortida 2

```
0 comencen per A
0 comencen per AEIOU
0 comencen per BCDFGHJKLMNPQRSTVWXYZ
```

Exemple d'entrada 3

```
OCA
----------
O
AEIOU
BCDFGHJKLMNPQRSTVWXYZ
```

Exemple de sortida 3

```
1 comencen per O
1 comencen per AEIOU
0 comencen per BCDFGHJKLMNPQRSTVWXYZ
```

Exemple d'entrada 4

```
CASA
CAS
----------
C
AEIOU
BCDFGHJKLMNPQRSTVWXYZ
```

Exemple de sortida 4

```
2 comencen per C
0 comencen per AEIOU
2 comencen per BCDFGHJKLMNPQRSTVWXYZ
```

Informació del problema

Autor : Jordi Esteve
Generació : 2021-01-10 19:42:58