
Subseqüències en camins d'un arbre

X99792_ca

Heu d'implementar un programa que llegirà un arbre d'enters t i també vèries seqüències d'enters. Per a cada seqüència s , el programa haurà de calcular quantes fulles hi ha a t tals que els elements de s es troben en el camí des de l'arrel fins a aquella fulla, en el mateix ordre, i a on possiblement hi pot haver també altres valors. Recordeu que una fulla és un arbre amb un únic node, i que per tant els seus dos fills directes són arbres buits.

Per exemple, considereu el següent arbre i seqüència:

Entrada:

```
5 ( 1 ( , 5 ( 2 ( 3 , 1 ) , 3 ) ) , 2 ( 1 , 3 ( 2 , ) ) )
1 2
```

Sortida:

2

En total hi ha 2 fulles tals que en el camí des de l'arrel fins a qualsevol d'aquelles fulles ens hi apareix la seqüència s (en el mateix ordre, i també enmig d'altres valors). Mostrem a continuació quins serien els 2 camins, que indiquem amb els valors dels nodes visitats:

```
5, 1, 5, 2, 3
5, 1, 5, 2, 1
```

Fixeu-vos que l'enunciat d'aquest exercici ja ofereix uns fitxers que haureu d'utilitzar per a compilar: `Makefile`, `BinaryTree.hpp`. Us falta crear el fitxer `program.cpp` amb els corresponents `includes` i implementar-hi el programa que hem descrit. Quan pugueu la vostra solució al jutge, només cal que pugueu un tar construït així:

```
tar cf solution.tar program.cpp
```

Entrada

L'entrada té una primera línia amb un string que representa l'arbre t . Recordeu que podeu llegir l'string sobre una variable `tstring` i després usar `readStringtree(tstring, t)` per a transformar-lo en l'arbre t .

Cadascuna de les següents línies conté una seqüència d'enters. La podeu llegir i emmagatzemar com considereu convenient. Però penseu bé com ho feu (`queue`, `stack`, `list`, `vector`), doncs hi haurà maneres que faran més fàcil implementar un programa eficient.

Sortida

Per a cadascuna de les seqüències hi ha un valor de sortida en una línia, el nombre de fulles de t que compleixen la condició abans esmentada.

Exemple d'entrada 1

```
1 (5 (3 (3 (6, ), ), 0), 2)
1 0
1 5
3 3
1 2
1 3 3
2
1
```

Exemple de sortida 1

```
1
2
1
1
1
1
3
```

Exemple d'entrada 2

```
6 (2 (0 (1 (5 (4 (, 2 (8, )), 4 (10 (10, 3), 7 (10, )), 4 (10, 7)), ), 6), 7 (5 (4 (3, 10), 1), 9 (, 10 (6 (1, ), 3 (10 (6, ), )))))
6 2 1 10 3
6
6 2 0
7 5 1
7 9
6 10
6
6 7
6
5
6 0
6 2
7 4
6 1
6 2
6 2 6
6
6 10
9
```

Exemple de sortida 2

```
12
6
1
2
7
12
7
12
12
7
6
7
2
8
7
1
12
7
2
```

Observació

Aquest exercici requereix d'una bona optimització per tal de superar els jocs de proves privats.

Informació del problema

Autoria: PRO1

Generació: 2026-01-25T21:37:46.411Z

© Jutge.org, 2006–2026.

<https://jutge.org>