Nombre de subexpressions amb avaluació igual al paràmetreX97722_ca

INTRODUCCIÓ:

En aquest exercici considerarem arbres que representen expressions sobre els operadors `+`, `-`, `*`, i sobre operands naturals. Per exemple, l'arbre `-(+(3,*(4,2)),5)` representa l'expressió `3+4*2-5`.

EXERCICI:

Implementeu una funció que, donat un arbre binari `t` d'strings que representa una expressió correcta sobre naturals i operadors `+`, `-`, `*`, i un paràmetre enter `x`, retorna quantes subexpressions de `t` s'avaluen a `x`. Aquesta és la capcelera:

```
// Pre:  t és un arbre no buit que representa una expressió correcta
//       sobre els naturals i els operadors +,-,*.
//       Les operacions no produeixen errors d'overflow.
// Post: Retorna el nombre de subarbres de t que s'avaluen a x.
int numberSubtreesEvaluateToParam(const BinaryTree<string> &t, int x);
```

Aquí tenim un exemple de paràmetres d'entrada de la funció i la corresponent sortida:

```
numberSubtreesEvaluateToParam(*(+(1,2),-(6,3)), 3) = 3
```

Fixeu-vos que l'enunciat d'aquest exercici ja ofereix uns fitxers que haureu d'utilitzar per a compilar: `Makefile`, `program.cpp`, `BinaryTree.hpp`, `numberSubtreesEvaluateToParam.hpp` `utils.hpp`, `utils.cpp`. Us falta crear el fitxer `numberSubtreesEvaluateToParam.cpp` amb els corresponents `includes` i implementar-hi la funció anterior. Valdrà la pena que utilitzeu algunes de les funcions oferides a `utils.hpp`. Quan pugeu la vostra solució al jutge, només cal que pugeu un `tar` construït així:

```
tar cf solution.tar numberSubtreesEvaluateToParam.cpp
```

Entrada

L'entrada té un nombre arbitrari de casos. Cada cas consisteix en una primera línia amb un string describint un arbre binari d'strings, i una segona línea amb un enter. Fixeu-vos en que el programa que us oferim ja s'encarrega de llegir aquestes entrades. Només cal que implementeu la funció abans esmentada.

Sortida

Per a cada cas, la sortida conté el corresponent nombre de subarbres que s'avaluen a l'enter donat. Fixeu-vos en que el programa que us oferim ja s'encarrega d'escriure aquest nombre. Només cal que implementeu la funció abans esmentada.

Exemple d'entrada 1

```
-(+(+(1,2),-(3,4)),-(1,1))
2
```

```
-(-(*(2,3),3),-(+(-(2,2),1),3))
5
-(*(2,*(1,2)),+(1,2))
0
```

```
*(-(3,4),1)
4
+(3,-(4,4))
2
1
-10
+(3,-(+(-(4,2),-(1,1)),1))
-10
-(+(-(*(2,4),*(1,2)),-(4,4)),4)
1
-(+(4,-(1,4)),+(-(4,1),1))
-3
3
6
```

Exemple de sortida 1

```
3
1
0
1
0
0
0
1
2
0
```

Exemple d'entrada 2

```
*(-(*(-(15,14),8),+(4,4)),+(-(-(2,3),-(+(5,10),+(6,-(5,9)))),+(12,1)))
10
-(-(4,11),-(-(-(*(-(7,5),-(14,11)),-(-(9,10),12)),-(+(6,10),+(6,-(12,14)))),-(+(+(*(11,1),7),-
14
*(+(-(+(-(*(8,2),5),-(-(10,7),+(5,7))),-(12,-(-(7,9),-(11,15)))),+(-(11,8),-(-(-(15,4),6),-(12,*
10
-(+(8,-(8,8)),10)
-11
-(*(*(-(-(13,10),-(8,5)),*(-(13,10),+(2,2))),*(+(10,9),-(9,9))),-(+(-(+(-(5,15),7),-(-(5,9),-(
-7
+(+(-(-(4,+(-(4,15),+(2,10))),14),+(+(+(10,8),+(-(11,13),-(5,12))),-(+(-(13,6),4),11))),-(15,15)
-9
-(-(-(14,*(2,4)),+(5,3)),+(5,-(-(8,9),9)))
18
+(-(+(9,4),-(4,7)),-(-(-(-(-(10,1),13),+(-(5,11),+(6,9))),-(4,+(1,10))),-(*(1,*(15,1)),+(5,-(11,
10
+(4,+(-(5,-(*(-(12,5),-(+(3,9),13)),+(-(-(11,9),+(8,10)),5))),-(+(+(14,-(-(5,10),1)),+(+(13,-(7,
-13
-(+(2,8),+(-(+(-(10,2),3),+(-(14,5),-(14,7))),-(13,-(-(4,-(10,1)),1))))
-17
```

Exemple de sortida 2

```
1
5
4
0
2
1
0
3
1
0
```

Observació

La vostra funció i subfuncions que creeu han de treballar només amb arbres. Heu de trobar una solució **RECURSIVA** del problema. Possiblement necessitareu alguna funció auxiliar.

Informació del problema

Autor : PRO1
Generació : 2022-09-13 23:59:53