Alçada d'un arbre — X16362_ca

Implementeu una funció **RECURSIVA** que, donat un arbre binari d'enters, retorna la seva alçada. L'alçada d'un arbre és el nombre de nodes que es troben en el camí més llarg des de l'arrel fins a alguna de les fulles. Noteu que, si l'arbre és buit, llavors té alçada 0, i si l'arbre té un únic node (que serà arrel i fulla alhora), llavors té alçada 1. Aquesta és la capcelera:

```
// Pre:
// Post: Retorna l'alçada de t
int sumOfTree(BinaryTree<int> t);
```

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

```
3(1(,5),3(2(,1),)) => 4
```

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

```
tar cf solution.tar heightOfTree.cpp
```

Entrada

L'entrada té un nombre arbitrari de casos. Cada cas consisteix en una línia amb un string describint un arbre binari d'enters. 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 la corresponent suma de l'arbre. Fixeu-vos en que el programa que us oferim ja s'encarrega d'escriure aquesta suma. Només cal que implementeu la funció abans esmentada.

Exemple d'entrada 1

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

Exemple de sortida 1

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