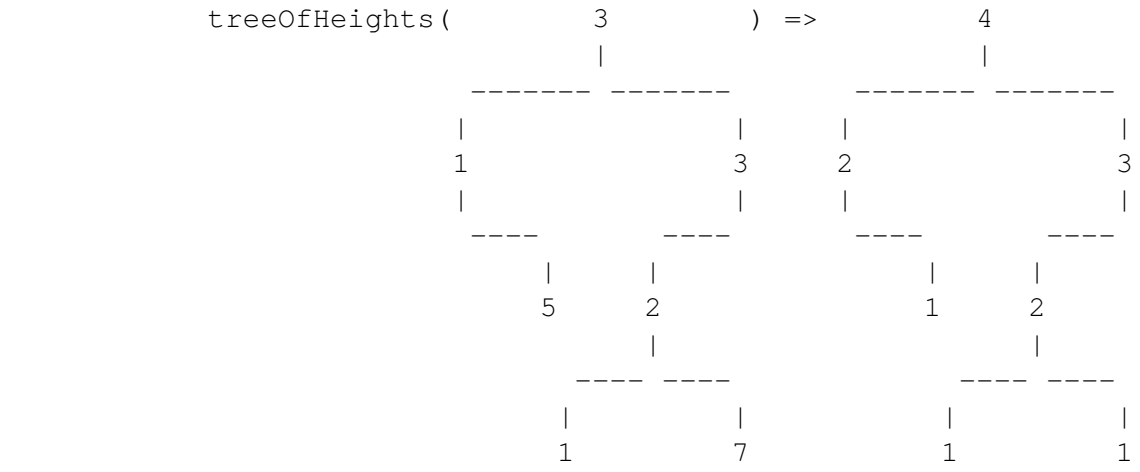
Arbre d'alçades X13384_ca

Implementeu una funció **RECURSIVA** que, donat un arbre binari d'enters, retorna un nou arbre amb la mateixa estructura, i a on cada posició conté un número que és l'alçada del subarbre que penja d'aquella posició. 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 un arbre d'enters amb la mateixa estructura que t,
//       i a on cada subarbre té com a arrel la seva alçada.
BinTree<int> treeOfHeights(BinTree<int> t);
```

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



Fixeu-vos que l'enunciat d'aquest exercici ja ofereix uns fitxers que haureu d'utilitzar per a compilar: `main.cc, BinTree.hh, treeOfHeights.hh`. Us falta crear el fitxer `treeOfHeights.cc` amb els corresponents `includes` i implementar-hi la funció anterior. Només cal que pugeu `treeOfHeights.cc` al jutge.

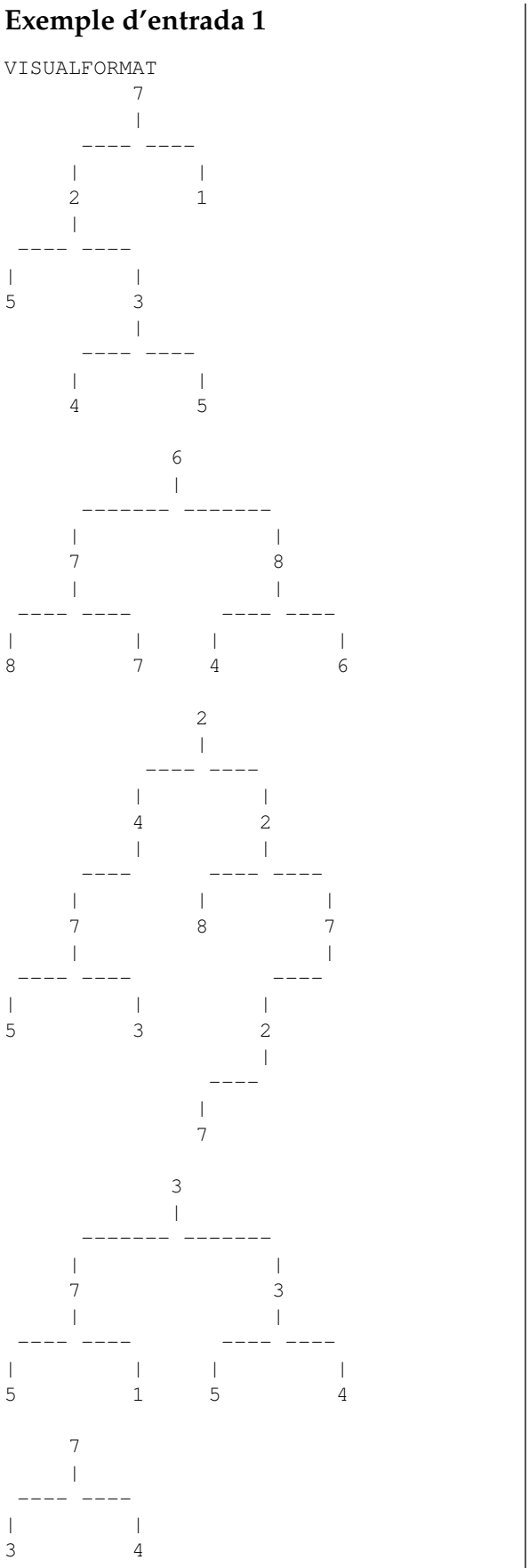
Entrada

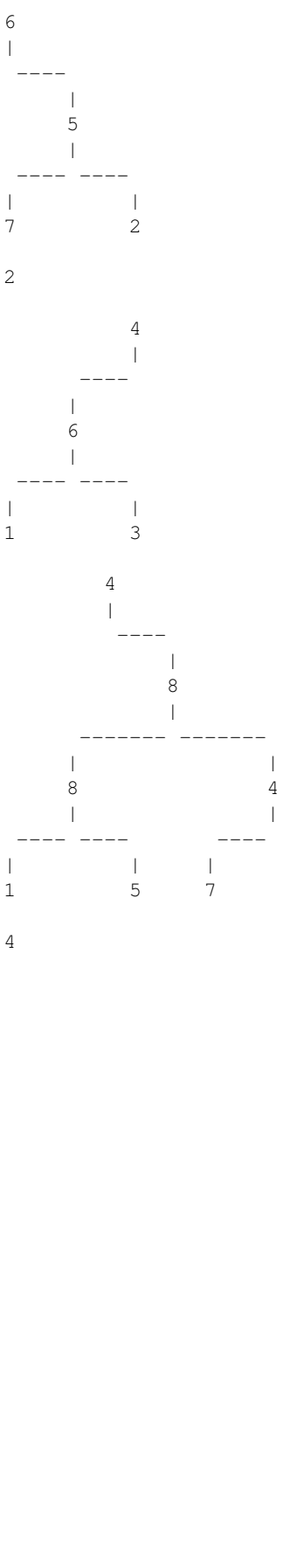
La primera linia de l'entrada descriu el format en el que es descriuen els arbres, o bé INLINEFORMAT o bé VISUALFORMAT. Després venen un nombre arbitrari de casos. Cada cas consisteix en una descripció d'un arbre un arbre binari d'enters. Fixeu-vos en que el programa que us oferim ja s'encarrega de llegir aquestes entrades. Només cal que implementeu la funció abans esmentada.

Sortida

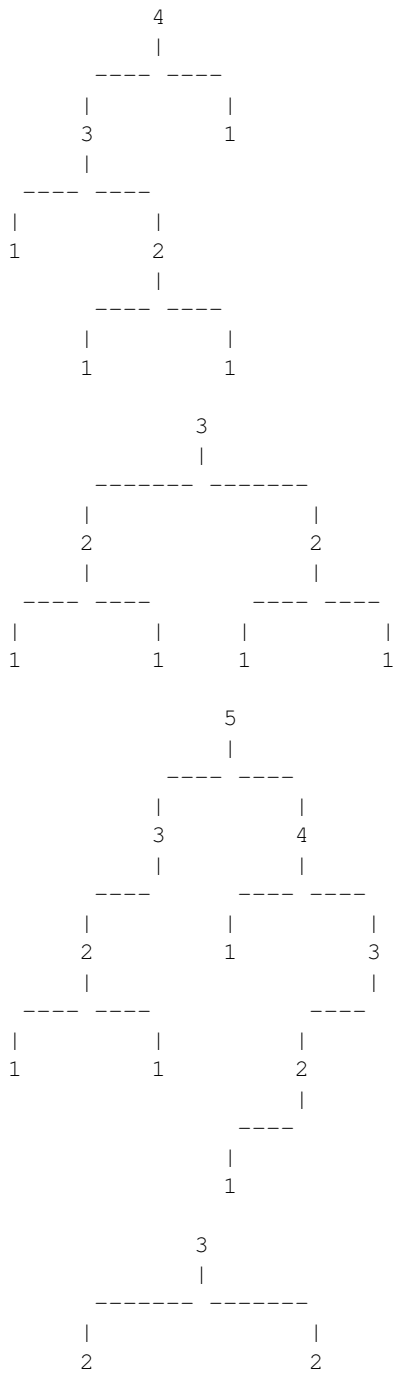
Per a cada cas, la sortida conté el corresponent arbre d'alçades. Fixeu-vos en que el programa que us oferim ja s'encarrega d'escriure aquesta sortida. Només cal que implementeu la funció abans esmentada.

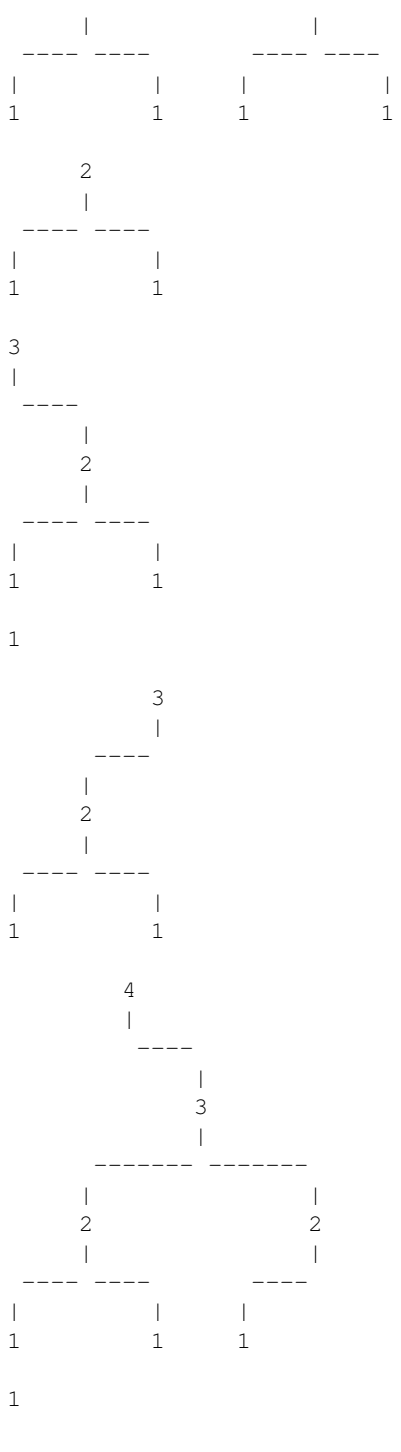
Exemple d'entrada 1





Exemple de sortida 1





Exemple d'entrada 2

```
INLINEFORMAT
0(55(29(-47(-15,98),),-18(86(-59(60(29(,-38),30),-13(-80,-29)),62(-21,2(12(-28,-20),-67(-58,-79)
75(-46(-53(-48,-53),98(,61)),-49)
67(25,-50)
9(-87,25(95,))
15(-92(-47(70,),-87),)
4(-1(27,-35),)
78(86(-5(,68),),46(88(-59,-9(68,83)),79(89(-93,-72),-31(-76,-91)))
-25(93(76(4,-8),-51(-22(-3,21),31(-34,32))),-95(-40(,53),93(,-81(16(-61,13(89,)),-7(-20,37)))))
94(37(,6),72(-90(,24(,-38(55(-65,22),46))),38(69(22(-65
58
-20(82,81(-19,37))
97(-45(53(87(-96(-16(-35,97(,-23)),65(97,52(56,)),59(2
-6(-10(,25(80,6(57,47))),-60(80,87))
40(-71(4(-17(90(,-4(,-57)),-67(,-87)),100),20(14(-28,80
-14(-95(-31(41(-30(59(-71(27,-4),-75(,-92)),),59),-42),
8(54(11(-99(67(7,),),-47(-10,-18)),82(9,-9)),43(16,-56)
-69(-15(25(57(38(-54,-13),80),-5),39(,-5(-28(-34,),74(-
-53(19,33(9(29(-5,87),-60(21(-7,-16),)),62(-37(90(47,28
```

```
40(-49(-36,-47(51(-22(-7(-67(74(33,-100),1
-9(-64(16,),49(-79,74))
```

Exemple de sortida 2

```
8(7(3(2(1,1),),6(5(4(3(2(,1),1),2(1,1)),4(1,3(2(1,1),2
4(3(2(1,1),2(,1)),1)
2(1,1)
3(1,2(1,))
4(3(2(1,),1),)
3(2(1,1),)
5(3(2(,1),),4(3(1,2(1,1)),3(2(1,1),2(1,1))))
7(4(2(1,1),3(2(1,1),2(1,1))),6(2(,1),5(,4(3(1,2(1,)),2
9(2(,1),8(5(,4(,3(2(1,1),1))),7(4(2(1,1),3(2(1,1),1)),6
1
3(1,2(1,1))
8(7(6(5(4(3(1,2(,1)),3(1,2(1,))),4(3(2(1,1),),1)),),4(2
5(4(,3(1,2(1,1))),2(1,1))
7(6(5(4(3(,2(,1)),2(,1)),1),3(2(1,1),2(1,1))),2(1,))
11(7(6(5(4(3(2(1,1),2(,1)),),1),1),6(2(,1),5(4(3(1,2(,1
6(5(4(3(2(1,),),2(1,1)),2(1,1)),2(1,1))
6(5(4(3(2(1,1),1),1),4(,3(2(1,),2(1,)))),3(2(1,),2(1,))
6(1,5(4(2(1,1),3(2(1,1),)),4(3(2(1,1),2(1,1)),1)))
11(10(1,9(6(5(4(3(2(1,1),1),2(1,)),4(3(1,2(1,1)),)),2(1
3(2(1,),2(1,1))
```

Observació

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

Informació del problema

Autoria: PRO2

Generació: 2026-01-25T13:41:13.620Z