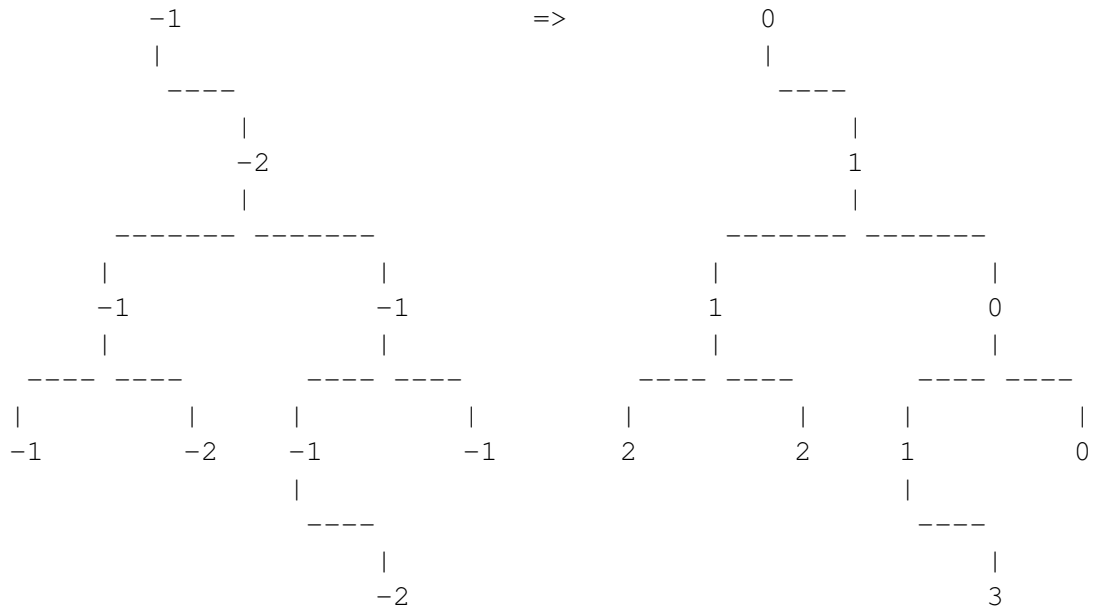
Nombre d'esquerra i dreta en un arbre binari — X86374_ca

Implementeu una funció **RECURSIVA** que, donat un arbre binari d'enters amb valors -1 i -2 als nodes, retorna un nou arbre amb la mateixa estructura que l'inicial, i a on per a cada posició `p` hi ha un valor d'acord al següent criteri. Si l'arbre inicial tenia -1 a posició `p`, llavors el nou arbre té a posició `p` el nombre de cops que s'escull fill-esquerra en el camí des de l'arrel fins a posició `p`. En canvi, si l'arbre inicial tenia -2 a posició `p`, llavors el nou arbre té a posició `p` el nombre de cops que s'escull fill-dret en el camí des de l'arrel fins a posició `p`.

```
// Pre:  Sigui T el valor inicial de l'arbre t que es rep com a paràmetre.
//       Els valors guardats a T son tots -1 o -2.
// Post: Sigui T' l'arbre retornat. T i T' tenen exactament la mateixa estructu
//       Sigui p una posició qualsevol de T'.
//       Si T té un valor -1 a posició p, llavors T' té el nombre de cops que
//       s'escull fill-esquerra en el camí des de l'arrel fins a posició p.
//       Si T té un valor -2 a posició p, llavors T' té el nombre de cops que
//       s'escull fill-dret en el camí des de l'arrel fins a posició p.
BinTree<int> numLeftRight(BinTree<int> t);
```

Aquí tenim un exemple de comportament de la funció:

```
numLeftRight(-1(,-2(-1(-1,-2),-1(-1(,-2),-1)))) = 0(,1(1(2,2),0(1(,3),0)))
```



Fixeu-vos que l'enunciat d'aquest exercici ja ofereix uns fitxers que haureu d'utilitzar per a compilar: `main.cc, BinTree.hh, numLeftRight.hh`. Només cal que creeu `numLeftRight.cc`, posant-hi els includes que calguin i implementant la funció `numLeftRight`. Només cal que pugeu `numLeftRight.cc` al jutge.

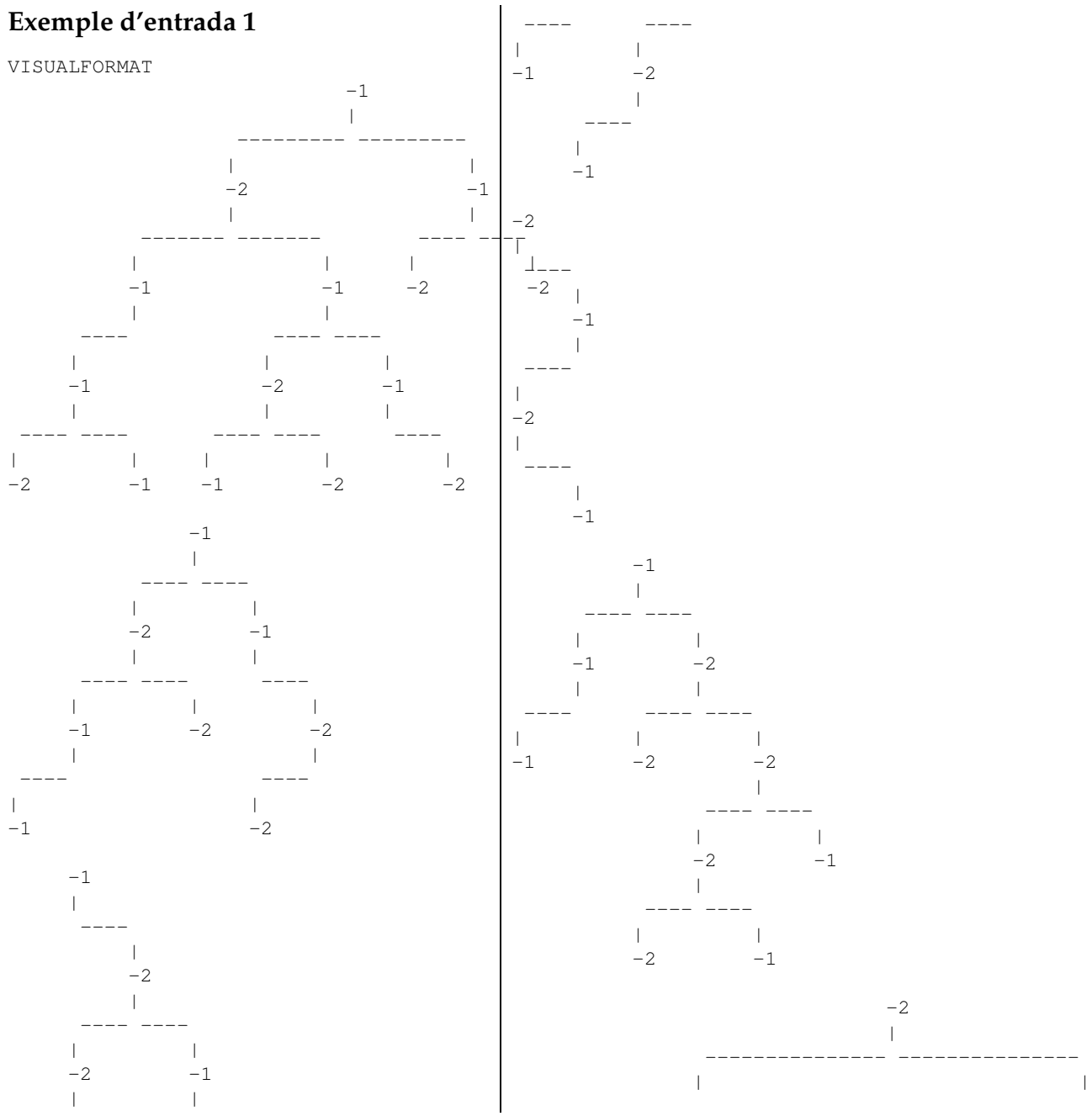
Entrada

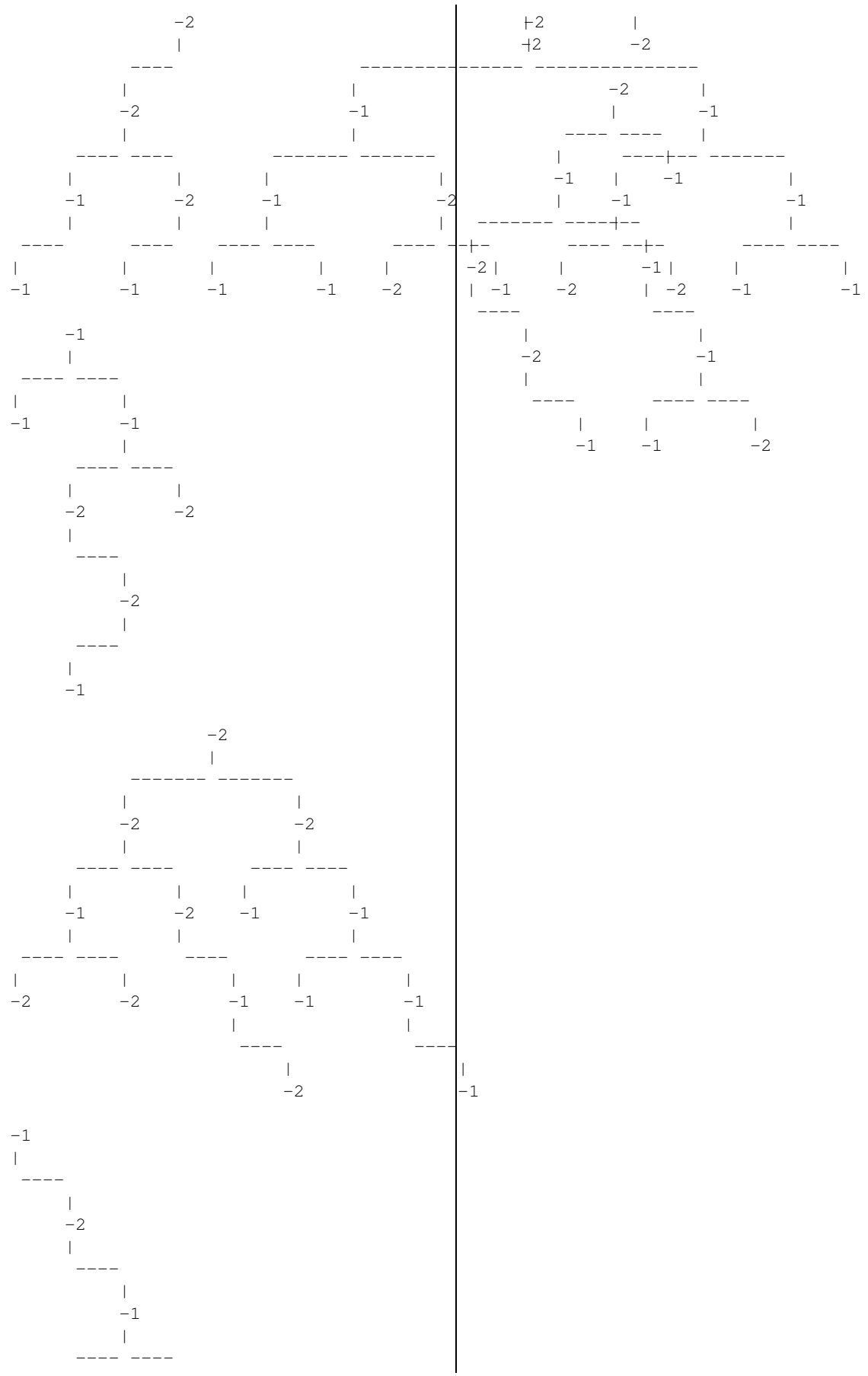
La primera linia de l'entrada descriu el format en el que es descriuen els arbres, o bé INLINEFORMAT o bé VISUALFORMAT. Després venen un nombre arbitrari de casos. Cada cas consisteix en una descripció d'un arbre un arbre binari d'enters amb valors -1 i -2. Fixeu-vos en que el programa que us oferim ja s'encarrega de llegir aquestes entrades. Només cal que implementeu la funció abans esmentada.

Sortida

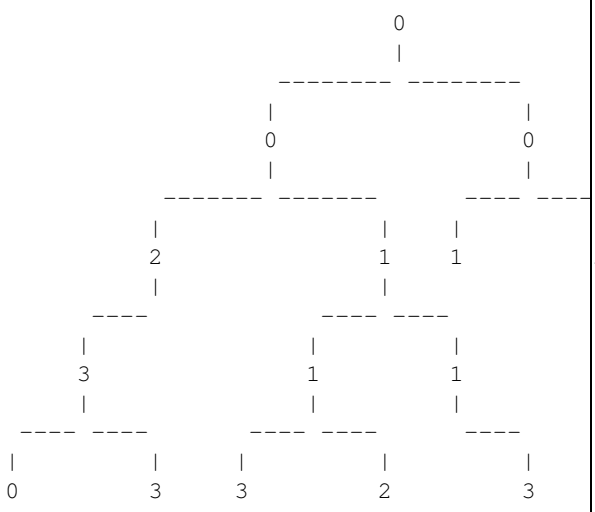
Per a cada cas, cal escriure l'arbre binari resultant de cridar a la funció abans esmentada amb l'arbre d'entrada. Fixeu-vos en que el programa que us oferim ja s'encarrega d'escriure aquesta sortida. Només cal que implementeu la funció abans esmentada.

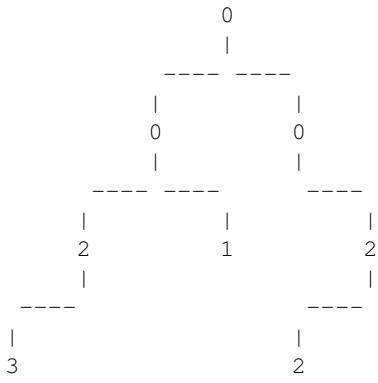
Exemple d'entrada 1

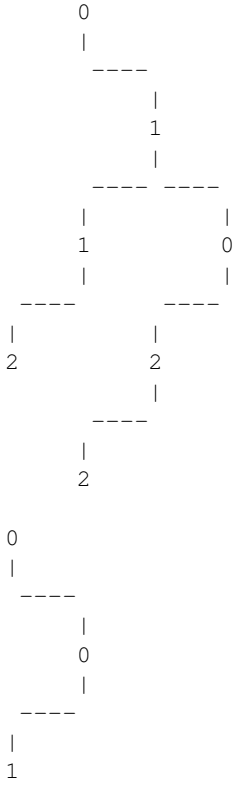


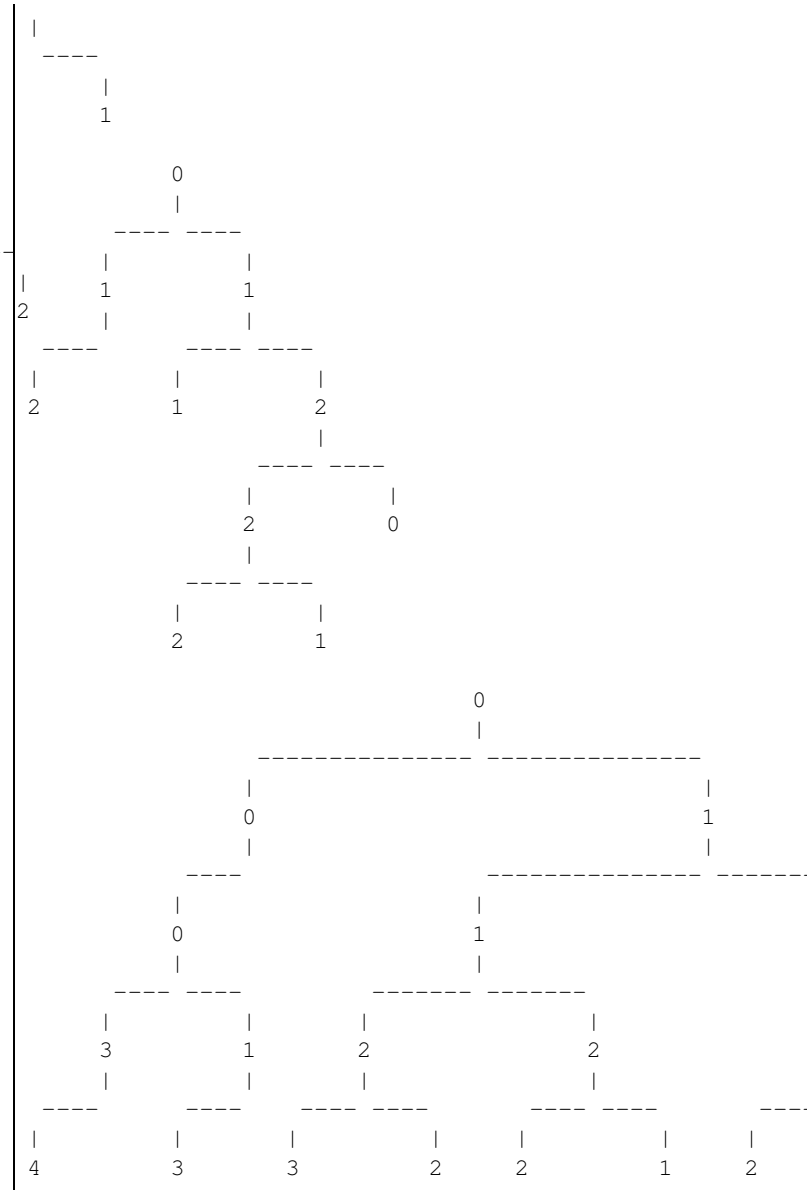


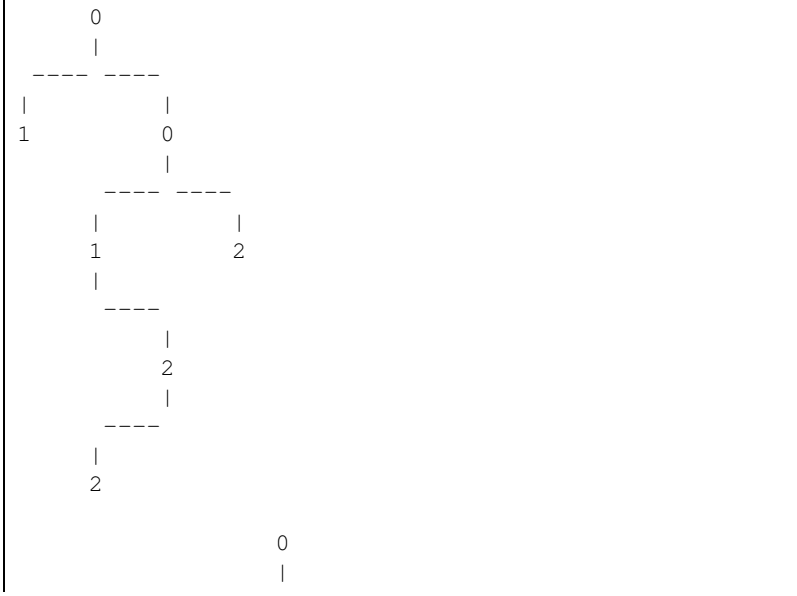
Exemple de sortida 1

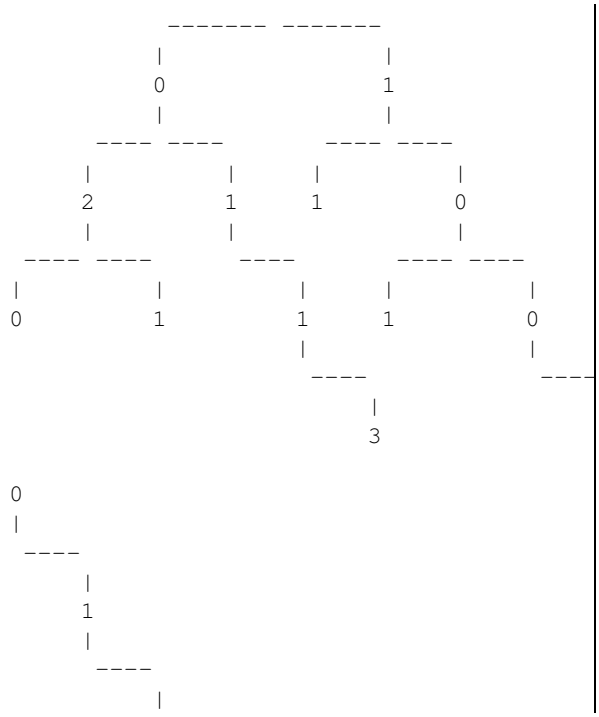


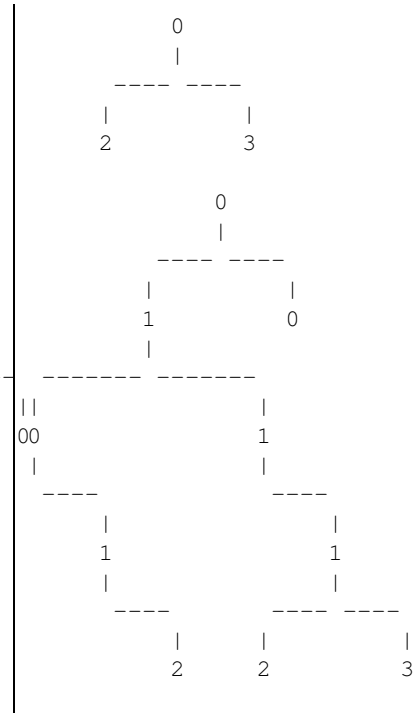












Exemple d'entrada 2

```
INLINEFORMAT
-1(-2(-1(-1(-2,-1),),-1(-2(-1,-2),-1(,-2))),-1(-2(-1(-1,),-2),-1(,-2(-2,)))
-1(-2(-1(-1,),-2),-1(,-2(-2,)))
-1(,-2(-2(-1,),-1(-2(-1,),)))
-2(,-1(-2(,-1),))
-1(-1(-1,),-2(-2,-2(-2(-2,-1),-1)))
-2(-2(-2(-1(-1,),-2(-1,)),),-2(-1(-1(-1,-1(-2,-2),-1(-1,-1)))))
-1(-1,-1(-2(,-2(-1,)),-2))
-2(-2(-1(-2,-2),-2(,-1(,-2))),-2(-1,-1(-1,-1(-1,-1(-1,-1)))))
-1(,-2(,-1(-2,-2)))
-2(-1(-2(,-2(,-1)),-1(,-1(-1,-2))),-1)
```

Exemple de sortida 2

```
0(0(2(3(0,3),),1(1(3,2),1(,3))),0(1,2))
0(0(2(3,),1(,2)),0(,2(2,)))
0(,1(1(2,),0(2(2,),)))
0(,0(1(,1),))
0(1(2,),1(1,2(2(2,1),0)))
0(0(0(3(4,),1(3,)),),1(1(2(3,2),2(2,1)),0(1(2,3),0(1,0
0(1,2(0(2,2),1(2,1)),2(-2,-2),-1(-1,-1))))
0(0(2(0,1),1(,1(,3))),1(1,0(1,0(,0))))
0(1,1(,1(2,3)))
0(1(0(,1(,2)),1(,1(2,3))),0)
```

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

Autor : PRO2
Generació : 2023-10-21 13:51:38