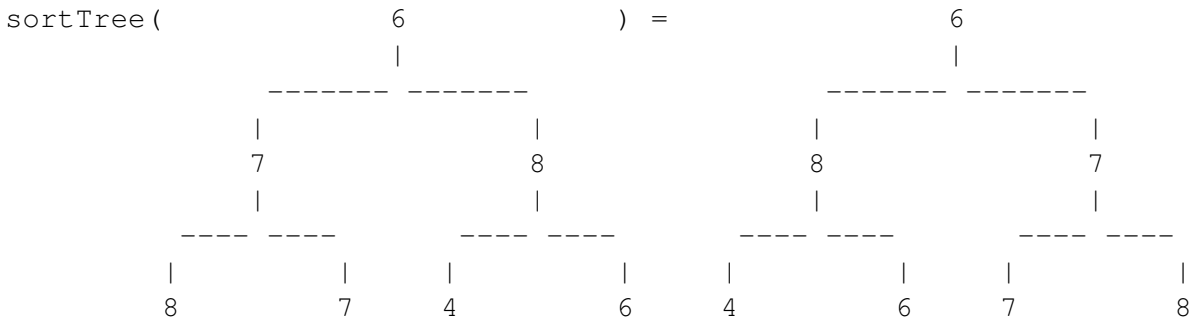
sortTree X92557_ca

Implementeu una funció **RECURSIVA** que, donat un arbre binari d'enters, el retorna ordenat. Ordenar l'arbre implica intercanviar els fills de cada node no buit en el cas en que la suma dels valors dels nodes del fill dret d'aquell node sigui menor estricte que la suma dels valors dels nodes del fill esquerra d'aquell node. Aquesta és la capcelera:

```
// Pre: sigui T el valor inicial de t.
// Post: el valor retornat s'obté a partir de T a base de
// swapejar els fills esquerra i dret dels nodes de T
// que cumpleixen que la suma dels valors que penjen del seu fill esquerra és m
// estricte que la suma dels valors que penjen del seu fill dret.
BinTree<int> sortTree(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, sortTree.hh`. Us falta crear el fitxer `sortTree.cc` amb els corresponents `includes` i implementar-hi la funció anterior. Només cal que pugeu `sortTree.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.

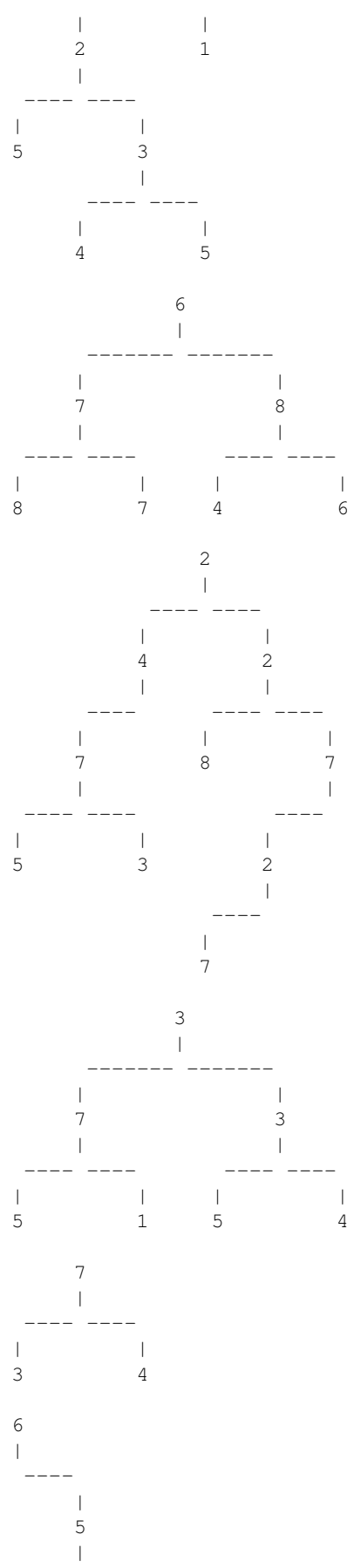
Sortida

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

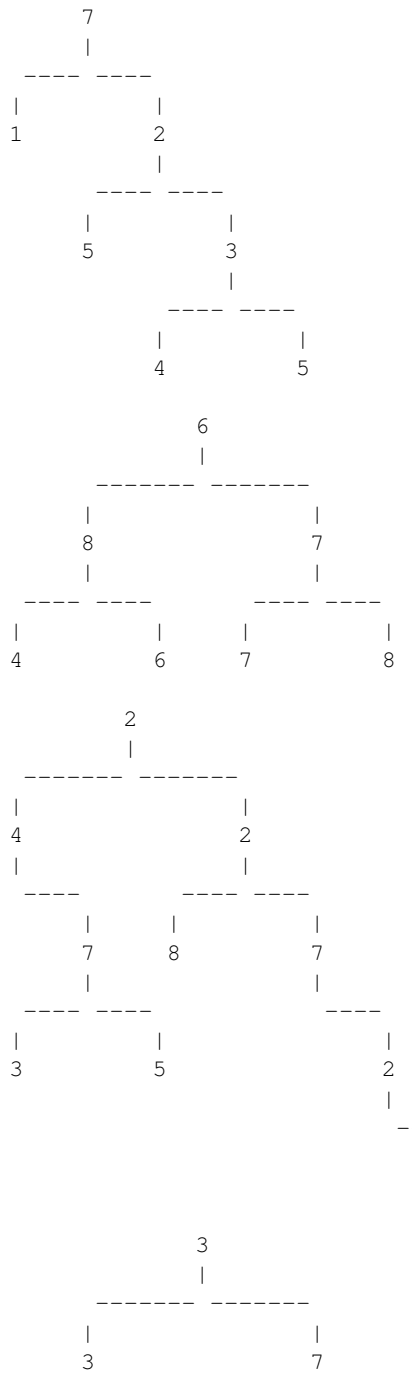
Exemple d'entrada 1

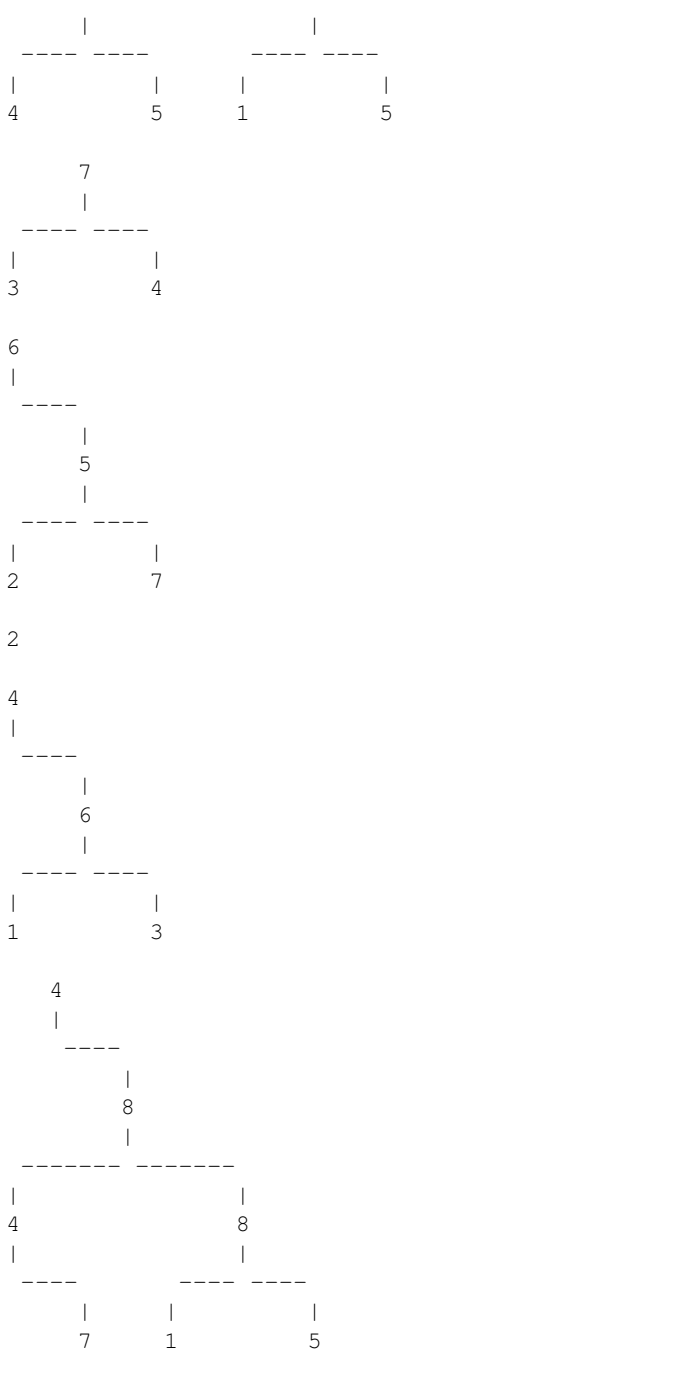
```
VISUALFORMAT
```

```
   7
   |
 ---- ----
```



Exemple de sortida 1





Exemple d'entrada 2

```
INLINEFORMAT
0(55(29,-47(-15,98)),-18)
-94(82(-21,80),-16(63,-85))
-27(-50(6(13,-56),),23(2,36(-2(-37,),)))
-56(-5(-100,-37),7(-70,-18))
5(-3,-32)
50(,-23(-17,91))
41
91(59(75,-46),)
```

```
55(,62(-31(-10,69),-74(67,)))
-56
12(96(-22(88,),31(15,-92)),-47(70,))
-58(4,-1(27,-35))
78
-91(89(35(-95,-24),-50(,77)),-95)
-69
89(-93(,-72),-31(-76,-91))
-25(93,76)
32(-71,73(-68(,-12(,-70)),-86(-61(-68,58),-39)))
```

```
68(-10(22,60),91)
89(-7(-20,37),)
```

Exemple de sortida 2

```
0(-18,55(29,-47(-15,98)))
-94(-16(-85,63),82(-21,80))
-27(-50(6(-56,13),),23(36(-2(-37,),),2))
-56(-5(-100,-37),7(-70,-18))
5(-32,-3)
50(,-23(-17,91))
41
91(,59(-46,75))
55(,62(-74(,67),-31(-10,69)))
-56
12(-47(,70),96(31(-92,15),-22(,88)))
-58(-1(-35,27),4)
78
-91(-95,89(35(-95,-24),-50(,77)))
-69
89(-31(-91,-76),-93(-72,))
-25(76,93)
32(73(-86(-61(-68,58),-39),-68(-12(-70,),)),-71)
68(-10(22,60),91)
89(,-7(-20,37))
```

Observació

La vostra funció i subfuncions que creeu han de treballar només amb arbres. Heu de trobar una solució **RECURSIVA** del problema. Molt possiblement, una solució directa serà lenta, i necessitareu crear alguna funció recursiva auxiliar per a produïr una solució més eficient capaç de superar tots els jocs de proves.

Informació del problema

Autor : PRO2
Generació : 2023-10-21 13:50:25