

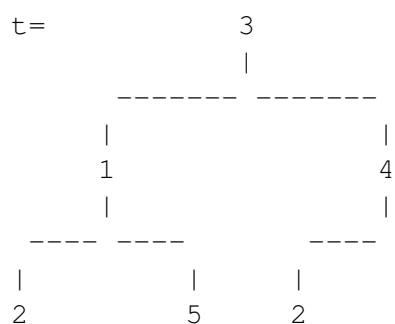
Arbre revessat

X33654_ca

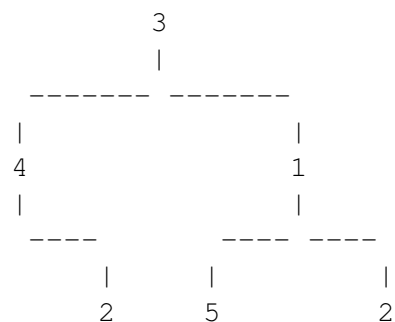
Implementeu una funció **RECURSIVA** que, donat un arbre binari d'enters t , retorna el revessat de t . Revessar un arbre és com enmirallar aquest arbre. Aquesta és la capcelera:

```
// Pre:
// Post: retorna el revessat de t.
BinaryTree<int> reverseTree(BinaryTree<int> t);
```

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



=>



Fixeu-vos que l'enunciat d'aquest exercici ja ofereix uns fitxers que haureu d'utilitzar per a compilar: `Makefile`, `program.cpp`, `BinaryTree.hpp`, `reverseTree.hpp`. Us falta crear el fitxer `reverseTree.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 reverseTree.cpp
```

Entrada

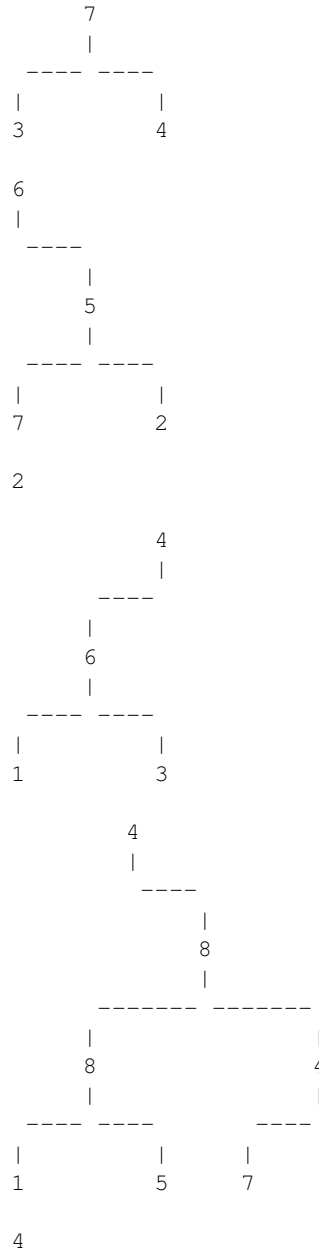
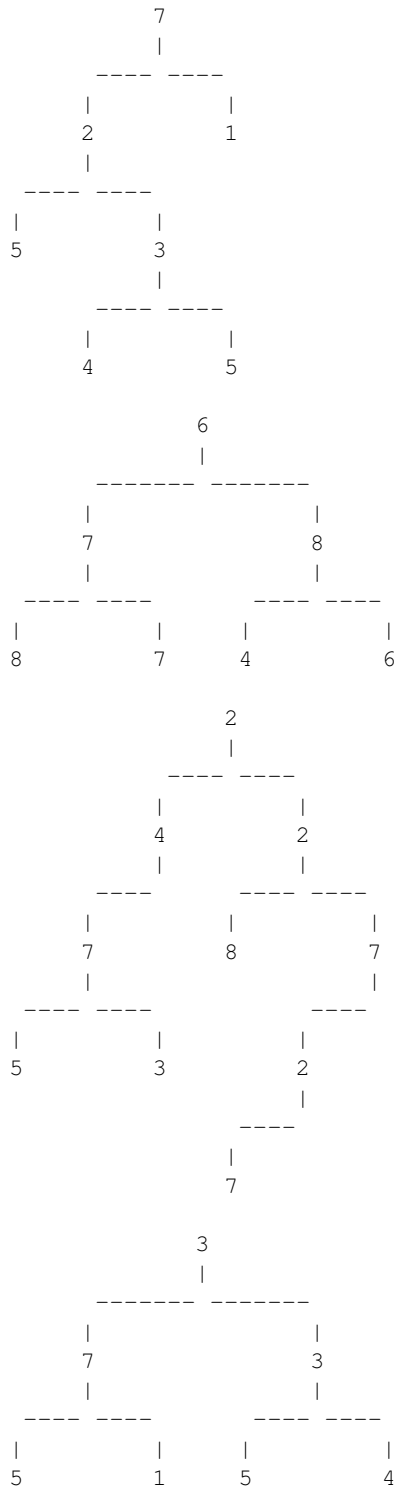
La primera línia de l'entrada descriu el format en el que es descriuen els arbres, o bé `IN-LINEFORMAT` o bé `VISUALFORMAT`. Després venen un nombre arbitrari de casos. Cada cas consisteix en una descripció d'un arbre un arbre binari d'enters. Fixeu-vos en que el programa que us oferim ja s'encarrega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

Sortida

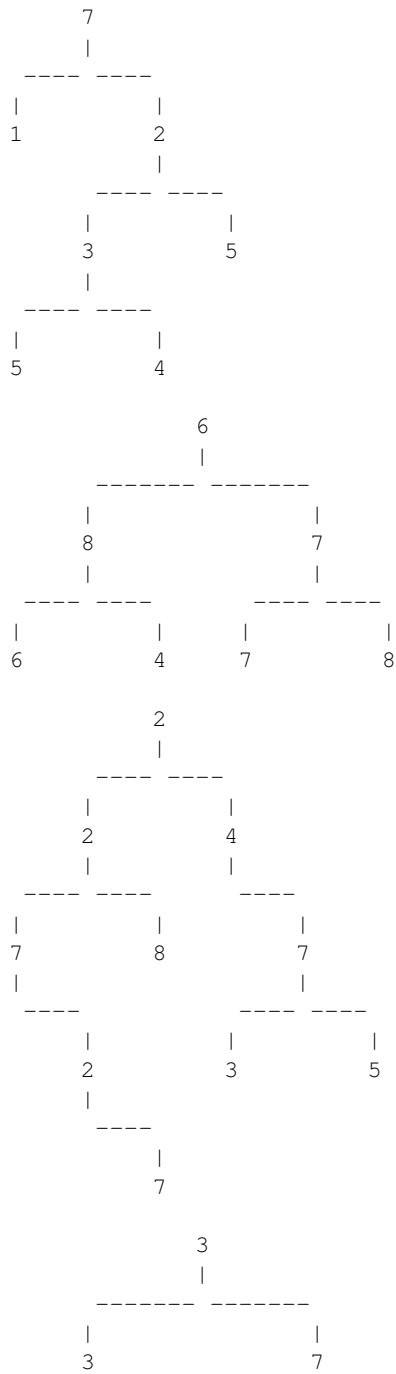
Per a cada cas, la sortida conté la descripció de l'arbre resultant d'aplicar el revers. Fixeu-vos en que el programa que us oferim ja s'encarrega d'escriure aquestes dades. Només cal que implementeu la funció abans esmentada.

Exemple d'entrada 1

VISUALFORMAT



Exemple de sortida 1

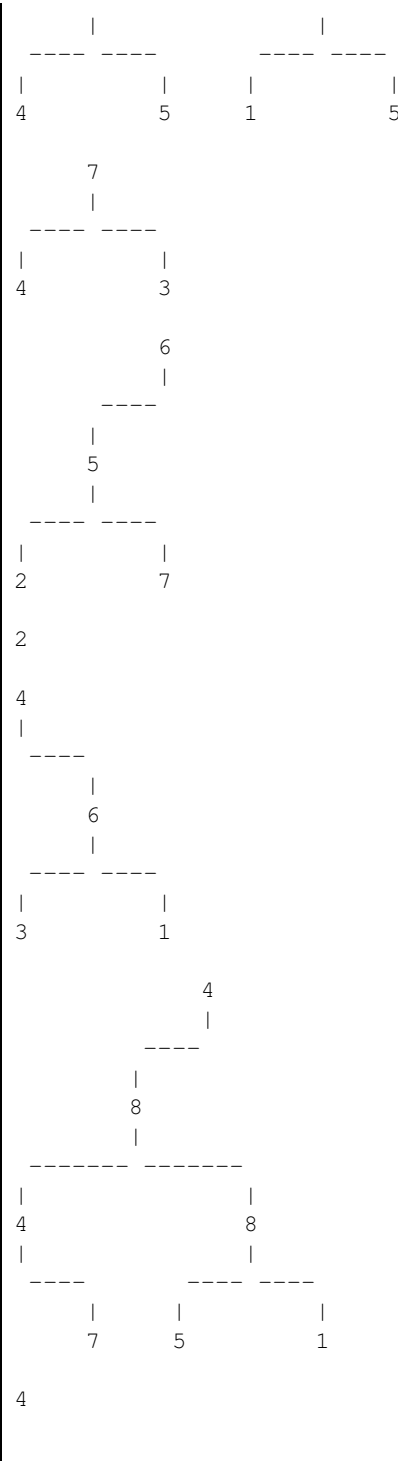


Exemple d'entrada 2

```

INLINEFORMAT
0 (55, 29)
72 (43 (, -73), -44 (-94, ))
-90 (61 (, -43), 54)
-13 (-80 (-29 (62 (-21, 2), 12 (-28, -20)), ), -67
-44
-38 (-73 (, -28 (30, 16)), 40 (35, ))
-46 (, 96 (51 (63, 41), -88 (-8, 59)))

```



```

-46 (-53, -48 (, -53 (98, 61)))
-74
17 (-56 (, -83), 9)
-22 (88 (31 (15 (-92, ), -47), 70 (-87 (82, -10), 4)), -72 (73, -93 (
-96 (86 (-74 (-20 (, -42 (-69, -62)), 98 (, 87))), ), 21 (-24 (52 (, -9
-74 (47 (73, ), )
-58 (-79, -56), -5))
-31 (-34, 32)
58 (-39 (, 53), 53 (-70, ))
60 (91 (55 (45, ), 27 (-16 (45, ), -53 (100, ))) , 80 (, 7 (5 (, -67), 78
-43 (-46 (-46, 84), -10 (, 45))

```

```
-54 (49 (78 (-10, -3), 52 (56, 39 (, 80 (, 24) ) ) ) , -48)
-16
95 (86 (-27, -52), 93 (-92, ))
```

Exemple de sortida 2

```
0 (29, 55)
72 (-44 (, -94), 43 (-73, ))
-90 (54, 61 (-43, ))
-13 (-67 (-5, -58 (-56, -79) ) , -80 (, -29 (12 (-20, -28), 62 (2, -21)
-44
-38 (40 (, 35), -73 (-28 (16, 30) , ))
-46 (96 (-88 (59, -8), 51 (41, 63) ) , )
-46 (-48 (-53 (61, 98) , ) , -53)
-74
17 (9, -56 (-83, ))
-22 (-72 (-93 (-91 (35, 89) , ) , 73) , 88 (70 (4, -87 (-10, 82) ) , 31 (-
-96 (21 (, -24 (28 (32 (-71, ) , -16 (-31, ) ) , 52 (-99, ) ) ) , 86 (, -74 (
-74 (, 47 (, 73) )
31 (32, -34)
58 (53 (, -70) , -39 (53, ))
60 (80 (7 (78 (-65, -96) , 5 (-67, ) ) , ) , 91 (27 (-53 (, 100) , -16 (, 45)
-43 (-10 (45, ) , -46 (84, -46) )
-54 (-48, 49 (52 (39 (80 (24, ) , ) , 56) , 78 (-3, -10) ) )
-16
95 (93 (, -92) , 86 (-52, -27) )
```

Observació

La vostra funció i subfuncions que creeu han de treballar només amb arbres. Heu de trobar una solució **RECURSIVA** del problema. En les crides recursives, incloeu la hipòtesi d'inducció, és a dir una explicació del que es compleix després de la crida, i també la funció de fita/decreixement o una justificació de perquè la funció recursiva acaba.

Molt possiblement, una solució directa serà lenta, i necessitareu crear alguna funció recursiva auxiliar per a produir una solució més eficient capaç de superar tots els jocs de proves.

Informació del problema

Autor : PRO1

Generació : 2023-03-11 21:33:36

© *Jutge.org*, 2006–2023.

<https://jutge.org>