
Esborrar 0s i 1s d'un arbre**X27662_ca**

Implementeu una funció **RECURSIVA** que, donat un arbre binari d'enters, retorna un altre arbre binari d'enters a base d'esborrar tots els seus 0's i 1's aplicant els següents passos tans cops com calgui:

- Si hi ha un subarbre amb arrel 0, llavors el reemplacem pel seu corresponent fill esquerre.
- Si hi ha un subarbre amb arrel 1, llavors el reemplacem pel seu corresponent fill dret.

Aquesta és la capcelera:

```
// Pre:
// Post: Retorna el resultat de reemplaçar tans cops com calgui en t
//       cada subarbre amb arrel 0 pel seu fill esquerre,
//       i cada subarbre amb arrel 1 pel seu fill dret.
BinaryTree<int> remove01(BinaryTree<int> t);
```

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

```
0 (1 (1, 7 (0, 8)), 8 (1 (0, 3), 1 (0, )))
=>
7 (, 8)
```

Fixeu-vos que l'enunciat d'aquest exercici ja ofereix uns fitxers que haureu d'utilitzar per a compilar: `Makefile`, `program.cpp`, `BinaryTree.hpp`, `remove01.hpp`. Us falta crear el fitxer `remove01.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 remove01.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 arbre resultant d'haver esborrat 0s i 1s. 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

```
0 (1 (1, 7 (0, 8)), 8 (1 (0, 3), 1 (0, )))
2 (1 (0, 4), 6 (3, ))
```

```
| 0 (, 2 (7 (, 5 (8, )), ))
| 0
| 0 (1, )
```

```

5 (0 (, 1) , 1 (5, 4) )
0 (5, 4)
3 (2 (2 (0, ) , 1 (1 (8, 1) , 6 (6, ) ) ) , 1 (2, 2) )
1 (7 (6 (1, ) , 2 (, 4) ) , 0)
6 (1 (0 (7, 0) , 0 (, 8) ) , 4 (, 8 (1 (5, 3) , 1) ) )

```

Exemple de sortida 1

```

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

```

Exemple d'entrada 2

```

-20 (1, -3)
1 (0 (, 6) , 0 (, 1) )
0 (, 17 (1 (7 (1, -11) , 1) , 20 (1 (9, -13) , 0 (, 5 (-3, 0) ) ) ) )
0 (-17 (, -20) , )
0 (-14 (16 (1, 1) , 1 (-20 (-8, ) , ) ) , -9 (3 (0 (6, -7) , -17 (16, ) ) , 18) )
1 (-9 (-12 (, -8) , 7 (1, -16) ) , -5 (-20 (-11, -18) , ) )
18 (8 (, -17) , -10)
0 (-19 (0 (-2, 0) , -6 (, -2) ) , -11 (-4 (1, ) , -11 (, 17) ) )
16 (-9 (, 0) , -11 (1, ) )
-8 (-9, 19)
-11 (, -12)
1
0 (-15, 9)
3
3 (1, -7)
0 (3, 1 (, -1) )
9 (-2, 8)
-7 (-4 (-15 (14, ) , 1 (1, 1) ) , )
14 (-8 (, 0) , 0 (, 0 (-15, ) ) )
0 (0 (-18, ) , )

```

Exemple de sortida 2

```

-20 (, -3)
()
-17 (, -20)
-17 (16, )
-5 (-20 (-11, -18) , )
18 (8 (, -17) , -10)
-19 (-2, -6 (, -2) )
16 (-9, -11)
-8 (-9, 19)
-11 (, -12)
()
-15
3
3 (, -7)
3
9 (-2, 8)
-7 (-4 (-15 (14, ) , ) , )
14 (-8, )
-18

```

Observació

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

Necessitareu crear alguna funció recursiva auxiliar amb més paràmetres per tal d'abordar més fàcilment el problema i també per a produir una solució més eficient capaç de superar tots els jocs de proves.

Informació del problema

Autoria: PRO1

Generació: 2026-01-25T21:06:44.810Z

© Jutge.org, 2006–2026.

<https://jutge.org>