Determinar si un arbre binari està ordenat S94933_ca

Un arbre d'elements de tipus `T` està ordenat si, o bé és buit, o bé l'arrel és estrictament menor que tots els elements tant del fill esquerre com del fill dret, y els elements del fill dret són tots estrictament majors que els del fill esquerre. A més, els fills han de ser, també, arbres ordenats.
Implementa un *mètode públic* de la classe `Arbre<T>` que determini si un arbre està ordenat. La declaració és la següent:

```
/**
 * @brief Determina si un arbre està ordenat.
 *
 * L'arbre està ordenat si és buit, o bé si l'arrel és estrictament
 * menor que tots els elements fills, tant del fill esquerre com
 * del dret, i els elements del fill dret són tots estrictament majors
 * que els de l'esquerre. Alhora, els fills han de ser també arbres
 * ordenats.
 *
 * @returns `true` si l'arbre està ordenat, `false` altrament.
 */
bool ordered() const;
```

Observació

Per poder avaluar l'ús de punters, *no feu servir altres mètodes, ni públics ni privats,* de la classe per resoldre el problema, accediu sempre als membres privats directament.

Els fitxers públics (icona del gatet) contenen:

`Arbre.hh`	la classe `Arbre<T>`
`main.cc`	el programa principal (gestiona l'entrada i sortida)
`Makefile`	per compilar amb `make` al terminal
`.vscode`	per compilar i debuggar amb F5

Per entregar només cal **enviar el fitxer** `Arbre.hh` **modificat.**

Entrada

De l'entrada se n'encarrega ja el programa principal. L'entrada està formada per diferents cassos seguits. Cada arbre d'entrada és una línia de números o `#`s en preordre (un `#` indica un arbre buit).

Sortida

De la sortida també se n'encarrega el programa principal. La sortida mostra un 0 quan l'arbre no està ordenat o un 1 si no ho està, cadascún en una línia separada.

Exemple d'entrada	Exemple de sortida
1 2 # # 3 # #	1
1 2 # # 2 # #	0
3 1 # # 2 # #	0
1 2 3 # # 6 # # 7 # #	1
1 2 3 # # 7 # # 6 # #	0
3 4 6 8 10 # # # # # #	1
1 2 # 3 # 4 # 5 # 6 # # 9 # #	1
1 2 # 3 # 4 # 3 # 6 # # 9 # #	0
1 2 # 3 # 4 # 5 # 8 # # 6 # #	0
1 2 # # 3 # 4 # 6 # 8 # 10 # #	1
1 7 # # 3 # 4 # 6 # 8 # 10 # #	0
-2 -1 0 1 # # 2 # # 3 4 5 # 6 # # 8 9 # #	10 # # 12 14 # # 20 # # #
-2 -1 0 1 # # 2 # # 3 4 -5 # 6 # # 8 9 # #	#010 # # 12 14 # # 20 # # #
-12 # -10 -5 1 # # 2 # # 3 4 5 # 6 # # 8	91# # 10 # # 12 14 # # 20 # #
-12 # -20 -5 1 # # 2 # # 3 4 5 # 6 # # 8	90# # 10 # # 12 14 # # 20 # #
-12 # -10 -5 1 # # 2 # # 3 6 5 # 6 # # 8	90# # 10 # # 12 14 # # 20 # #
-12 # -10 -5 1 # # 2 # # 3 4 5 # 6 # # 8	90# # 10 # # 12 10 # # 20 # #
-12 # -10 -5 1 # # 2 # # 3 4 5 # 6 # # 8	90# # 10 # # 12 14 # # 10 # #

Informació del problema

Autor : Mª Lluïsa Bonet i Pau Fernández
Generació : 2025-06-11 10:04:04