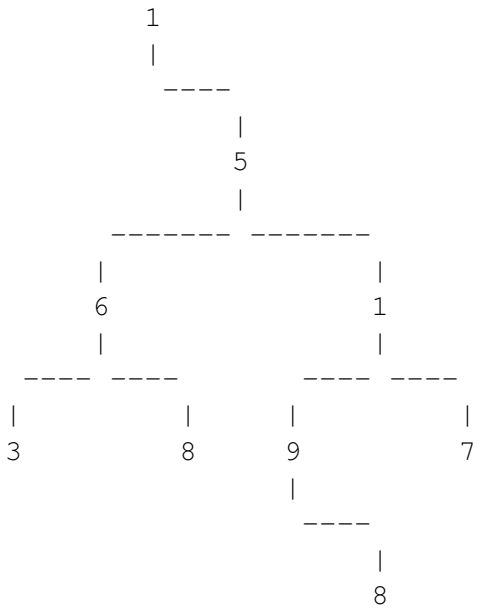
Trobar el valor més proper a un valor donat — T80023_ca

Implementa una funció **RECURSIVA** que, donat un arbre binari no buit d'enters i un enter x retorni el valor de l'arbre que està més proper a x. En cas que hi hagi dos valors que estan igual d'aprop a x s'ha de tornar el valor més petit.
La capçalera de la funció que has d'implementar és la següent:

```
// Pre: t no és buit
// Post: Torna el valor de t que és més proper a x. En cas que hi hagi dos valors
// igual de propers es torna el menor d'ells.
int mes_proper(const BinaryTree<int> &t, int x);
```

Exemple:
Donat el següent arbre binari, les següents crides a la funció tornarien:




```
mes_proper( 1(,5(6(3,8),1(9(,8),7))), 15 ) =  9
mes_proper( 1(,5(6(3,8),1(9(,8),7))), 2 ) =  1
mes_proper( 1(,5(6(3,8),1(9(,8),7))), 5 ) =  5
```

Fixa't que l'enunciat d'aquest exercici ja ofereix uns fitxers que has d'utilitzar per a compilar: `Makefile`, `program.cpp`, `BinaryTree.hpp`, `mes_proper.hpp`. Només cal que creïs `mes_proper.cpp`, posant-hi els includes que calguin i implementant la funció `mes_proper`. I quan pugis la teva solució al jutge, només cal que pugis un tar construït així:

```
tar cf solution.tar mes_proper.cpp
```

Entrada

La primera línia 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 binari d'enters i un enter. Fixa't que el programa

que t'oferim ja s'encarrega de llegir aquestes entrades. **Només cal que implementis la funció abans esmentada**.

Sortida

Per a cada cas, cal escriure el resultat de cridar a la funció abans esmentada amb l'arbre d'entrada. Fixa't que el programa que t'oferim ja s'encarrega d'escriure aquesta sortida. **Només cal que implementis la funció abans esmentada**.

Observació

La teva funció i subfuncions que creïs han de treballar només amb arbres binaris. Has de trobar una solució **RECURSIVA** del problema. En les crides recursives, inclou tant la **Hipòtesi d'inducció** com la **funció de fita/decreixement** de cada crida recursiva.

Exemple d'entrada 1

```
INLINEFORMAT
1(2(3,4),5(6(7(8,),9(10,11)),))
-1
1(2(3,4),5(6(7(8,),9(10,11)),))
0
1(2(3,4),5(6(7(8,),9(10,11)),))
1
1(2(3,4),5(6(7(8,),9(10,11)),))
2
1(2(3,4),5(6(7(8,),9(10,11)),))
3
1(2(3,4),5(6(7(8,),9(10,11)),))
4
1(2(3,4),5(6(7(8,),9(10,11)),))
5
1(2(3,4),5(6(7(8,),9(10,11)),))
6
1(2(3,4),5(6(7(8,),9(10,11)),))
7
1(2(3,4),5(6(7(8,),9(10,11)),))
8
1(2(3,4),5(6(7(8,),9(10,11)),))
9
1(2(3,4),5(6(7(8,),9(10,11)),))
10
1(2(3,4),5(6(7(8,),9(10,11)),))
11
1(2(3,4),5(6(7(8,),9(10,11)),))
12
1(2(3,4),5(6(7(8,),9(10,11)),))
13
1(,5(6(3,8),1(9(,8),7)))
2
1(,5(6(3,8),1(9(,8),7)))
1
1(,5(6(3,8),1(9(,8),7)))
3
1(,5(6(3,8),1(9(,8),7)))
4
1(,5(6(3,8),1(9(,8),7)))
0
```

```
10
1
10
10
10
100
10
-100
10
-10
```

Exemple de sortida 1

```
1
1
1
2
3
4
5
6
7
8
9
10
11
11
11
1
1
3
3
1
10
10
10
10
10
```

Exemple d'entrada 2

```
INLINEFORMAT
11(52(33,4),15(46(71(85,),19(10,101)),))
-1
11(52(33,4),15(46(71(85,),19(10,101)),))
0
11(52(33,4),15(46(71(85,),19(10,101)),))
10
11(52(33,4),15(46(71(85,),19(10,101)),))
20
11(52(33,4),15(46(71(85,),19(10,101)),))
30
11(52(33,4),15(46(71(85,),19(10,101)),))
40
11(52(33,4),15(46(71(85,),19(10,101)),))
50
11(52(33,4),15(46(71(85,),19(10,101)),))
60
11(52(33,4),15(46(71(85,),19(10,101)),))
70
11(52(33,4),15(46(71(85,),19(10,101)),))
80
11(52(33,4),15(46(71(85,),19(10,101)),))
90
11(52(33,4),15(46(71(85,),19(10,101)),))
100
11(52(33,4),15(46(71(85,),19(10,101)),))
110
11(52(33,4),15(46(71(85,),19(10,101)),))
120
11(52(33,4),15(46(71(85,),19(10,101)),))
130
1(10(100(1000(10000,),),),),)
-10
1(10(100(1000(10000,),),),),)
-5
1(10(100(1000(10000,),),),),)
-1
1(10(100(1000(10000,),),),),)
0
1(10(100(1000(10000,),),),),)
1
1(10(100(1000(10000,),),),),)
5
1(10(100(1000(10000,),),),),)
10
1(10(100(1000(10000,),),),),)
50
1(10(100(1000(10000,),),),),)
100
1(10(100(1000(10000,),),),),)
500
1(10(100(1000(10000,),),),),)
1000
```

Exemple de sortida 2

```
4
4
10
19
33
46
52
52
71
85
85
101
```

```
101
101
101
1
1
1
1
1
1
10
10
100
100
1000
```

Informació del problema

Autoria: Bernardino Casas

Generació: 2026-01-25T20:34:48.887Z