
Posició d'un element en un arbre

X13115_ca

Implementeu una funció **RECURSIVA** que, donat un arbre binari d'enters t no repetits, i un enter x , retorna una llista amb els elements que es troben en el camí des de l'arrel fins la posició de x a t . En cas que x no pertanyi a l'arbre, la llista retornada serà buida. Aquesta és la capçelera:

```
// Pre: t no té repetits
// Post: retorna la llista de valors que es troben en el camí des de l'arrel
//       fins la posició de x en t. En cas que x no es trobi a t, retorna
//       una llista buida.
list<int> posInTree(const BinaryTree<int> t, int x);
```

Aquí tenim un exemple de paràmetres entrada i sortida de la funció:

```
t=
          3
          |
      -----
      |           |
      1           4
      |           |
  -----
  |       |   |   |
  6       5   2   7
  |
  -----
  |
  8

x = 2

=>

3, 4, 2
```

Fixeu-vos que l'enunciat d'aquest exercici ja ofereix uns fitxers que haureu d'utilitzar per a compilar: `Makefile`, `program.cpp`, `BinaryTree.hpp`, `posInTree.hpp`. Us falta crear el fitxer `posInTree.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 posInTree.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 binari d'enters sense elements repetits t , i un enter x . Fixeu-vos en que el programa que us oferim ja s'encarrega de llegir aquestes entrades. Només cal que implementeu la funció abans esmentada.

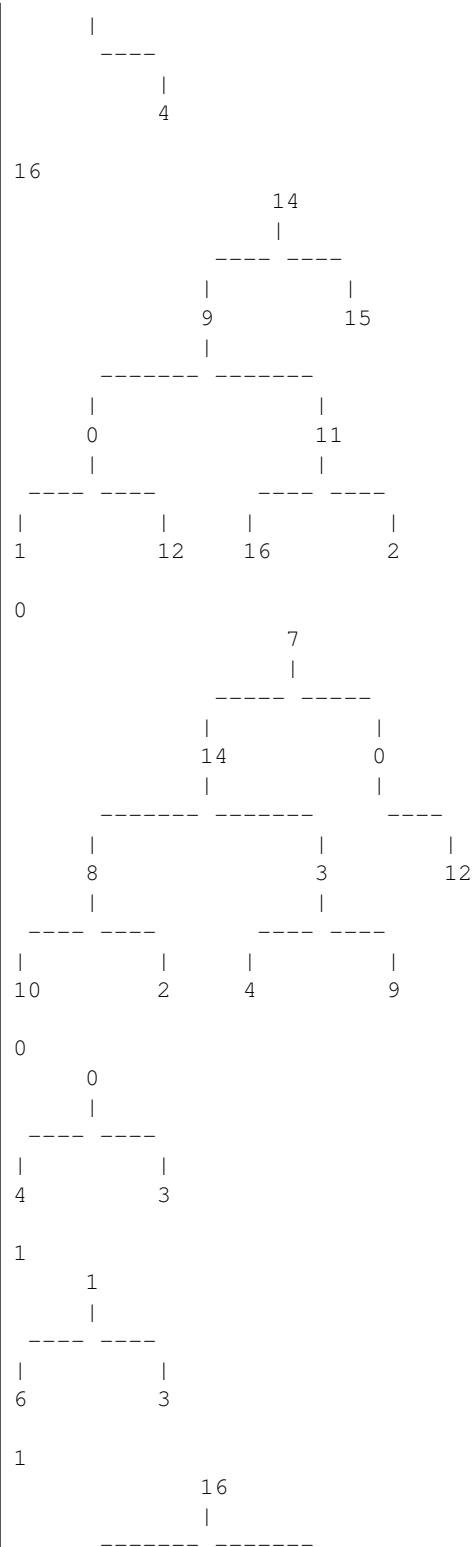
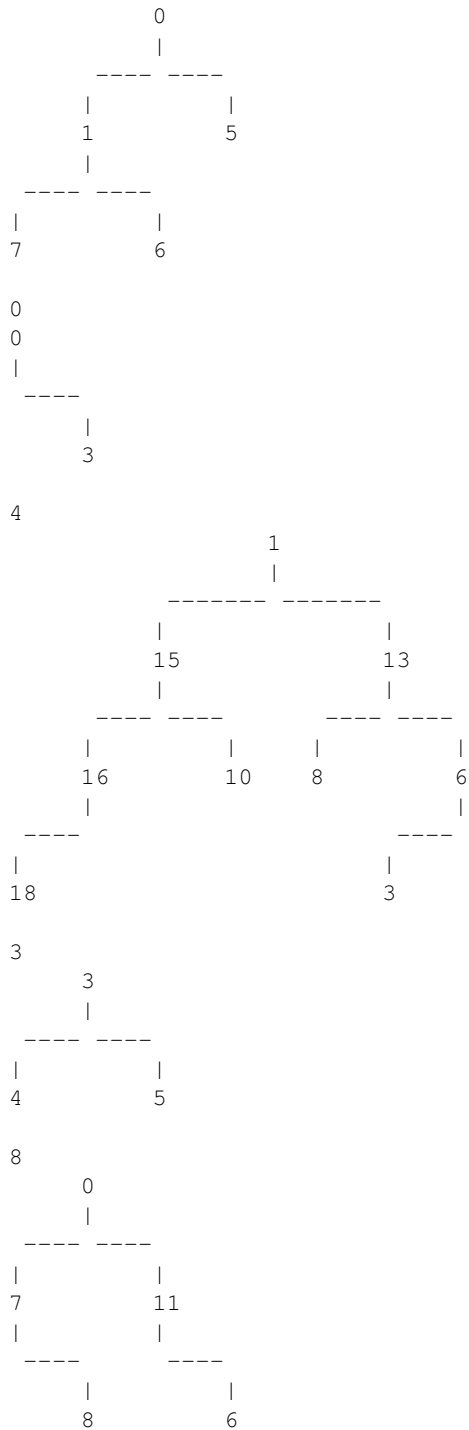
Sortida

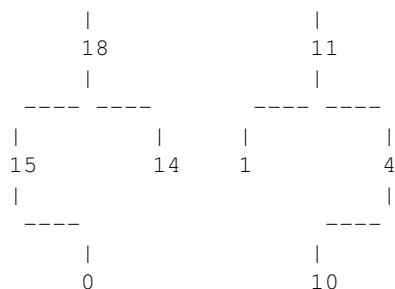
Per a cada cas, la sortida conté una línia amb la llista dels elements que es troben en el camí des de l'arrel fins la posició de x a t . La línia és buida si x no hi és a t . Fixeu-vos en que el programa que us oferim ja s'encarrega d'escriure aquestes dades. Només cal que implementeu la funció abans esmentada.

Exemple d'entrada 1

VISUALFORMAT

7





Exemple de sortida 1

```

0 1 7
0
3
0 7 8
14 9 11 16
7 0
0
1
16 11 1
  
```

Exemple d'entrada 2

INLINEFORMAT

```

4
13 (14 (7 (18, 2 (6, 0)), 10 (9, 17 (15, ))), 3)
2
8 (7 (, 2 (, 12 (, 3)), 9)
11
33 (30 (10, 27 (, 18)), 16 (39 (, 0 (20 (34, 36), 21 (4, 7))), 9 (11, 23 (12, 6 (40, )))))
27
21 (18 (5 (16 (, 11), 6 (8, )), ), 27 (14 (13 (15, 1 (3, )), 3 (10), 25))
1
1 (, 3)
18
14 (18 (13, 0 (22, )), 17 (8 (1, ), 9 (11, 19)))
4
0 (5, 6)
0
3 (, 0)
3
3 (2, )
3
4 (3, )
  
```

Exemple de sortida 2

```

8 7 2
33 16 9 11
21 27
1
14 18
3 0
3 (10), 25))
4 3
  
```

Observació

La vostra funció i subfuncions que creu 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

Autoria: PRO1

Generació: 2026-01-25T20:36:28.810Z

© Jutge.org, 2006–2026.

<https://jutge.org>