Cerca d'estudiants en un arbre binari d'estudiants X65608_ca

Heu d'implementar una cerca en un arbre d'estudiants. Com a entrada hi haurà els estudiants que hi ha en l'arbre (parells *dni, nota*) en preordre: primer l'arrel, després el subarbre esquerre i per últim el dret. Els arbres buits es representen amb el parell (0, 0). Cal dir que si la nota és un enter entre 0 i 10 llavors aquesta nota és la que té l'estudiant, mentre que si l'enter que segueix el dni no és dins d'aquest interval, caldrà afegir l'estudiant a l'arbre però no tindrà nota. Tot seguit hi haurà una llista d'enters positius que representen dni's d'estudiants.
Per a cadascun dels estudiants cal cercar si és a l'arbre o no. Si hi és i té nota, caldrà escriure la seva profunditat i la seva nota. Si l'estudiant hi és però no té nota, caldrà escriure la seva profunditat i -1. Si no hi és, caldrà caldrà escriure -1. En qualsevol cas, aquesta informació ha d'estar precedida pel dni de l'estudiant.
Si l'estudiant apareix més d'una vegada se n'ha d'obtenir la profunditat mínima i si n'hi ha dues aparicions a la mateixa profunditat, s'ha d'informar de la nota (o manca d'ella) de l'aparició més a l'esquerra.
Assumiu que la profunditat de l'arrel és 0.

Entrada

L'entrada és un arbre binari d'estudiants (parells *dni, nota*) en preordre, i una seqüència d'estudiants (només dni's) per cercar.

Sortida

La sortida per a un estudiant `x` pot ser:

- `x y z`, si l'estudiant hi és, la seva profunditat és `y` i la seva nota és `z`
- `x y -1`, si l'estudiant hi és, la seva profunditat és `y` i no té nota
- `x -1`, si l'estudiant no hi és

Observació

Cal fer servir les classes `BinTree` i `Estudiant` que us donem
Heu d'enviar tres fitxers en un sol .tar:

- `BinTreeIOEst.hh` amb les funcions:

```
void read_bintree_est(BinTree<Estudiant>& a);
// Pre:  a és buit; el canal estandar d'entrada conté una seqüència
// de parells <int, double> que representa un arbre binari d'estudiants
// en preordre, on un parell amb l'int = ``marca'' representa
// un arbre buit
// Post:  a conté l'arbre que hi havia al canal estandar d'entrada
```

`void write_bintree_est(const BinTree<Estudiant>& a);` (opcional)

```
// Pre:  a = A
// Post:  s'han escrit al canal estandar de sortida els elements
// d'a recorreguts en inordre, a = A
```

- `BinTreeIOEst.cc` amb la seva codificació
- `program.cc` amb el programa

Observeu que per compilar us donem el Makefile.

Informació del problema

Autor : Jaume Baixeries Juan Luis Esteban Borja Valles
Generació : 2020-10-06 18:11:40