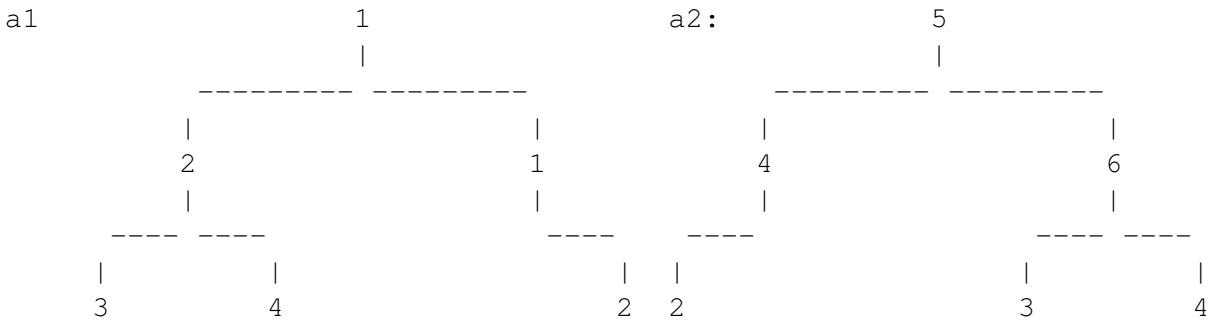
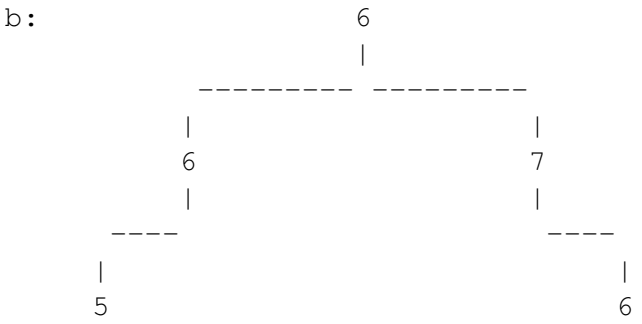
Avaluar expressions booleanes X80502_ca

En aquest exercici haurem de crear un nou mètode `suma` a la classe Arbre que rebrà un segon Arbre com a paràmetre, i retornarà un tercer Arbre, és a dir, es podran fer crides del tipus `a1.suma(a2)`, i el resultat també serà un Arbre. Aquest arbre resultant tindrà, com a posicions, les posicions comunes de `a1` i `a2`, i en cada node hi tindrà la suma dels nodes corresponents dels àrbres originals.
Com que el tipus `T` de la classe Arbre és genèric, no necessàriament té definida la operació de suma. Els jocs de proves treballen amb àrbres d'enters. Per tant, n'hi ha prou amb que implementeu la funció `suma` de manera que funcioni correctament quan `T` és `int`.
Exemple: si tenim els arbres a1 i a2



llavors la crida b=a1.suma(a2) haurà de retornar l'arbre b



D'entre els fitxers que s'adjunten en aquest exercici, trobareu `Arbre.hh`, a on hi ha una implementació de la classe genèrica Arbre binari. Haureu de buscar dins `Arbre.hh` les següents línies:

```
// Pre:  el p.i. i a són arbres de enters positius
// Post: Retorna la intersecció del p.i. i a, on cada node
//       conté la suma dels nodes corresponents del p.i. i a
// Descomenteu les següents dues línies i implementeu el mètode:
// Arbre suma (const Arbre& a) const{
// }
```

Descomenteu les dues línies que s'indiquen i implementeu el mètode, fent servir l'operació privada que trobareu just després i que també haureu d'implementar. No toqueu la resta de la implementació de la classe.

La implementació d'aquest mètode hauria de consistir en accedir a nodes mitçançant punters. De fet, possiblement qualsevol altra implementació produïrà error d'execució.
Observació: Per a superar els jocs de proves privats convindrà evitar recórrer nodes dels arbres originals que no pertanyin a la intersecció.
D'entre els fitxers que s'adjunten a l'exercici també hi ha `main.cc` (programa principal), i el podeu compilar directament, doncs inclou `Arbre.hh`. Només cal que pugeu `Arbre.hh` al jutge.

Entrada

Cada cas consisteix en una descripció de dos arbres binaris d'enters positius. La descripció d'un arbre consisteix en un recorregut en preordre del nodes de l'arbre, amb marques on hi anirien els arbres buits. 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 descripció de la corresponent intersecció sumada dels arbres, amb el mateix format que l'input. Fixeu-vos en que el programa que us oferim ja s'encarrega d'escriure aquesta avaluació. Només cal que implementeu la funció abans esmentada.

Exemple d'entrada 1

```
1 2 3 0 0 4 0 0 1 0 2 0 0
5 4 2 0 0 1 0 0 6 3 0 0 4 0 0
```

Exemple de sortida 1

```
6 6 5 0 0 5 0 0 7 0 6 0 0
```

Exemple d'entrada 2

```
6
 6
  6 0 0 6 0 0
 8
  8 0 0 8 0 0

5
 4 0 0
 6
  3 0 0 4 0 0
```

Exemple de sortida 2

```
11 10 0 0 14 11 0 0 12 0 0
```

Observació

La vostra funció i subfuncions que creeu han de treballar només amb arbres, punters i nodes d'arbre. Heu de trobar una solució **RECURSIVA** del problema.
Avaluació sobre 10 punts:

- Solució lenta: 5 punts.
- Solució ràpida: 10 punts.

Entenem com a solució ràpida una que és correcta, de cost lineal i que optimitza operacions booleanes, i capaç de superar els jocs de proves públics i privats. Entenem com a solució lenta una que no és ràpida, però és correcta i capaç de superar els jocs de proves públics.

Informació del problema

Autoria: PRO2

Generació: 2026-01-27T18:56:04.476Z