PUBLICACIÓ DOCENT

MANUAL DE LABORATORI D'ESIN Sessió 6

AUTOR: Bernardino Casas, Jordi Esteve
ASSIGNATURA: Estructura de la Informació (ESIN)
CURS: Q3
TITULACIONS: Grau en Informàtica
DEPARTAMENT: Ciències de la Computació
ANY: 2019

Vilanova i la Geltrú, 12 d'octubre de 2019

6

Exercici

L'objectiu d'aquest exercici és resoldre problemes d'arbres generals usant un arbre general implementat per nosaltres mateixos usant memòria dinàmica.

Caldrà resoldre els següents problemes de la plataforma jutge.org; els trobareu en l'apartat Arbres del curs ESIN (Vilanova):

- https://jutge.org/problems/P66413_ca. Recorreguts recursius d'un arbre general.
- https://jutge.org/problems/P18593_ca. Suma d'arbres.

No heu de guardar l'arbre en un vector tal com s'explica en l'enunciat del primer problema, sinó que heu d'usar una classe que implementi els arbres generals en memòria dinàmica. L'especificació i implementació d'arbre general en memòria dinàmica la trobareu en els apunts de teoria; per evitar problemes copiant des de fitxers PDF la podeu copiar de la carpeta `/home/public/esin/sessio6`).

A més a més de l'especificació i implementació de la classe `Arbre` (fitxers `arbre.hpp` i `arbre.cpp`), cal implementar altres funcions i el programa principal que les cridarà (`main.cpp`).

Degut a que jutge.org només permet l'enviament d'un fitxer amb la solució del problema, en el mateix fitxer hi ha d'haver l'especificació i la implementació de la classe i la resta de funcions i programa principal. I també cal eliminar les directives `#include "arbre.t"` i `#include "arbre.hpp"` per no tenir problemes de precompilació. Ho pots fer tot a la vegada amb la comanda:

```
cat arbre.hpp arbre.t main.cpp | sed '/include "arbre./d' > solucio.cpp
```

i enviar a jutge.org el fitxer `solucio.cpp`.

6.1 Consells

Podeu solucionar aquests problemes fent funcions externes a la classe `Arbre` o fent nous mètodes dins de la classe `Arbre`.

Si feu funcions externes a la classe `Arbre` només podeu accedir als mètodes públics de la classe, per tant haureu d'usar iteradors per recórrer l'arbre general.

Si feu nous mètodes dins de la classe `Arbre` aquests també poden usar iteradors per recórrer l'arbre general. Però com que podem accedir directament als atributs privats, una solució més eficient i elegant és usar punters a node i visitar els nodes de l'arbre general seguint els enllaços primer fill i següent germà.

Us recomanem la primera opció per fer el primer problema; fixeu-vos que és un problema molt semblant al fet a la sessió 5, canviant arbres binaris per arbres generals.

I us recomanem la segona opció per fer el segon problema, així practiqueu amb la implementació amb punters dels arbres generals. Per exemple podríeu fer el mètode públic

```
Arbre operator+(const Arbre<T> &a);
// Retorna un arbre que és la suma del p.i. i l'arbre a
```

que crida al mètode privat recursiu

```
static node* suma_arbres(node *a1, node *a2);
// Retorna el punter a l'arrel de la suma dels arbres apuntats per a1 i a2
```

I fer el mètode públic

```
void preordre() const;
// Escriu una línia amb el recorregut en preordre de l'arbre general
// Cada element ha de sortir precedit d'un espai.
```

que crida al mètode privat recursiu

```
static void preordre(node *n);
// Escriu una línia amb el recorregut en preordre de l'arbre general
// que apunta n. Cada element ha de sortir precedit d'un espai.
```

6.2 Exercicis opcionals

Podeu practicar fent la resta dels exercicis de l'apartat Arbres del curs ESIN (Vilanova).

Tingueu en compte que per resoldre alguns problemes no és necessari reconstruir l'arbre, ja que es pot anar calculant el resultat a mida que es llegeix el recorregut en preodre de l'entrada. Per tant, en aquests casos, no caldrà incloure la classe `Arbre` dins del nostre codi.