Problemes darrera sessió curs

R. Ferrer i Cancho

Universitat Politècnica de Catalunya

PRO2 (curs 2010-2011)
Versió 0.0

Avís: aquesta presentació no pretén ser un substitut dels apunts oficials de l'assignatura.

Diàmetre d'un arbre

- El diàmetre d'un arbre: màxim nombre de nodes de qualsevol camí (mínim) de l'arbre (o zero, si l'arbre es buit).
- Com a mètode public de la classe `Arbre`

```
int diametre() const {
/* Pre: cert */
/* Post: retorna el diametre del p.i. */
...
}
```

- Problema P51080 de *Jutge.org*.

Funció d'immersió

Mètode privat de la classe Arbre

```
pair<int,int> idiametre(node_arbre* n) const {
/* Pre: cert */
...
/* Post: retorna a "first" el diametre del subarbre apuntat per n i
         a "second" l'alçada del subarbre apuntat per n
}
```

Implementació de la funció d'immersió

```
pair<int,int> idiametre(node_arbre* n) const {
/* Pre: cert */
if (n == NULL) return make_pair(0, 0);
else {
   pair<int,int> c1 = idiametre(n->segE);
   pair<int,int> c2 = idiametre(n->segD);
   return make_pair(max(c1.first,max(c2.first, c1.second + c2.second + 1)),
          max(c1.second, c2.second) + 1);
   }
/* Post: retorna a "first" el diametre del subarbre apuntat per n i
         a "second" l'alçada del subarbre apuntat per n
}
```

Diàmetre d'un arbre

Com a mètode public de la classe `Arbre`

```
int diametre() const {
/* Pre: cert */
/* Post: retorna el diametre del p.i. */
return idiametre(primer_node).first;
}
```

Diàmetre d'un arbre

Com a mètode public de la classe `ArbreGen`

```
int diametre() const {
/* Pre: cert */
/* Post: retorna el diametre del p.i. */
return idiametre(primer_node).first;
}
```

Funció d'immersió

Mètode privat de la classe ArbreGen

```
pair<int,int> idiametre(node_arbre* n) const {
/* Pre: cert */
...
/* Post: retorna a "first" el diametre del subarbre apuntat per n i
         a "second" l'alçada del subarbre apuntat per n
}
```

Implementació de la funció d'immersió

```
pair<int,int> idiametre(node_arbreGen* n) const {
/* Pre: cert */
if (n == NULL) return make_pair(0, 0);
else {
     int i = 0;
     int mx = 0; // el màxim camí d'entre els fills [0..i-1]
     int ma1 = 0; // el màxim de les altures dels fills [0..i-1]
     int ma2 = 0; // el segon màxim de les altures dels fills [0..i-1]
     while (i < n->seg.size())
       pair<int,int> c = idiametre(n->seg[i]);
       if (c.first > mx) mx = c.first;
       if (c.second > ma1) { ma2 = ma1; ma1 = c.second; }
       else if (c.second>ma2) ma2 = c.second;
       ++i;
     }
     return make_pair(max(mx,ma1 + ma2 + 1), ma1 + 1);
  }
}
/* Post: retorna a "first" el diametre del subarbre apuntat per n i
         a "second" l'alçada del subarbre apuntat per n
}
```