L07: Reconstrucción de un árbol binario

Estructuras de Datos
Facultad de Informática - UCM

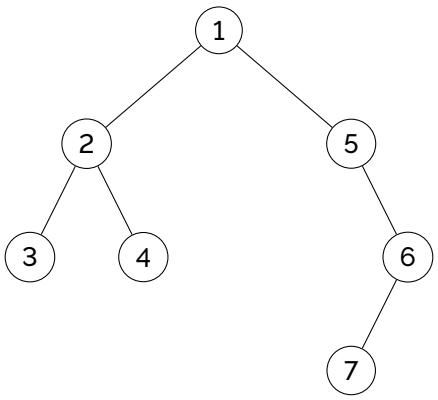
Para realizar este ejercicio, descarga la plantilla que se proporciona en este enunciado (icono del clip al lado del título) o a través del Campus Virtual.

La entrega de este ejercicio consiste en un único fichero `.cpp` que se subirá a *DOMjudge*. Podéis subir tantos intentos como queráis. Se tendrá en cuenta el último intento con el veredicto `CORRECT` que se haya realizado antes de la hora de entrega por parte de alguno de los miembros de la pareja.

No olvidéis poner el nombre de los componentes de la pareja en el fichero `.cpp` que entreguéis. Solo es necesario que uno de los componentes de la pareja realice la entrega.

Evaluación: La práctica se puntuará de 0 a 10 si ha obtenido el veredicto `CORRECT` por parte del juez. En caso contrario, la calificación será de 0. No se realizará evaluación por pares en esta práctica.

El semestre pasado estuvimos diseñando árboles binarios, con números enteros en los nodos. Los más bonitos eran aquellos que tenían todos los valores distintos, como este:



Cuando acabamos con ellos, guardamos los más bonitos en un archivo, serializados según su recorrido en preorden (ya sabes: raíz, hijo izquierdo, hijo derecho). Por aquello de la tolerancia a fallos, los guardamos en un segundo archivo, pero ahí serializados según su recorrido en inorden (hijo izquierdo, raíz, hijo derecho). Y *imenos mal!*. Creemos que con uno solo de los recorridos no habríamos podido ahora reconstruir los árboles, y tampoco estamos seguros de poderlo hacer teniendo los dos. ¿Nos ayudas?

1. Escribe una función:

   ```
   BinTree<int> reconstruir(const vector<int> &preorden,
                            const vector<int> &inorden)
   ```

   que devuelva el árbol correspondiente a los recorridos en preorden e inorden pasados como parámetro. Puedes definir las funciones auxiliares que sean necesarias.

2. Indica el coste de la función anterior.

3. Escribe un programa que lea de la entrada los recorridos en preorden e inorden de varios árboles, reconstruya el árbol correspondiente (llamando a la función `reconstruir` definida en el apartado 1) y haga un recorrido en postorden del árbol resultante, mostrando por pantalla los nodos visitados.

Entrada

La entrada está formada por una serie de casos de prueba. Cada caso consta de tres líneas. La primera contiene el número de nodos *N* del árbol. La segunda contiene *N* números con el recorrido en preorden de dicho árbol. La tercera contiene *N* números con el recorrido en inorden de ese mismo árbol. Todos los nodos del árbol son distintos y contienen valores comprendidos entre 1 y 10^6.

La entrada finaliza con un `0`, que no se procesa.

Salida

Para cada caso, se escribirá el recorrido en postorden del árbol reconstruido.

Entrada de ejemplo

```
7
1 2 3 4 5 6 7
3 2 4 1 5 7 6
3
2 4 6
2 4 6
8
1 2 4 8 5 3 6 7
8 4 2 5 1 6 3 7
0
```

Salida de ejemplo

```
3 4 2 7 6 5 1
6 4 2
8 4 5 2 6 7 3 1
```

Créditos

Adaptación del problema *Reconstrucción de un árbol binario* de Alberto Verdejo.