Estructuras de Datos y Algoritmos
Grado en Ingeniería Informática
Universidad de Zaragoza – curso 2024-2025

Sesión de problemas 3
Árboles binarios

Objetivos

- Especificar e implementar (en pseudocódigo) diversas operaciones relativas a árboles binarios.

1. Implementación de función `nodosCaminoMin`

Diseña la función `nodosCaminoMin`, que tiene la siguiente cabecera:

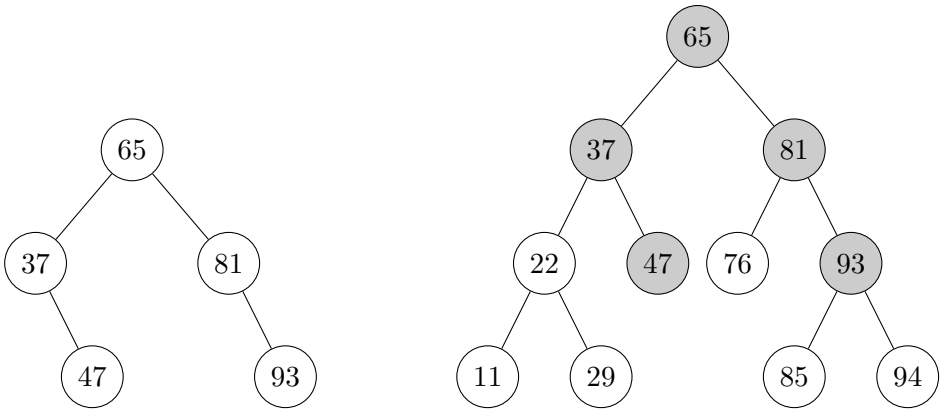
```
función nodosCaminoMin(a: arbin) devuelve natural
```

Dado un árbol binario no vacío, la función calcula el número de nodos existentes en el camino más corto desde la raíz a una hoja. Considera definidos los siguientes tipos:

```
tipos arbin = ↑nodo;
      nodo = registro
               dato: elemento;
               izq, der: arbin
             freg
```

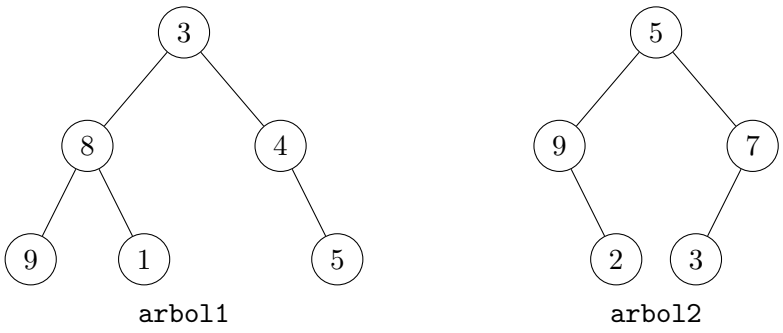
2. Implementación de función `esPrefijo?`

Implementa la función booleana `esPrefijo?` que, dados dos árboles binarios de elementos de tipo entero, `arbol1` y `arbol2`, devuelva `verdad` si `arbol1` es prefijo de `arbol2`. Se dice que un árbol binario `arbol1` es prefijo de otro árbol binario `arbol2` cuando `arbol1` coincide con la parte inicial del árbol `arbol2`, tanto en el contenido de los elementos como en su situación. En la siguiente figura de ejemplo, `arbol1` es prefijo de `arbol2`.



3. Especificación e implementación de la función `esInferior?`

Especifica e implementa una función booleana `esInferior?` que, dados dos árboles binarios `arbol1` y `arbol2`, indique si `arbol1` es *inferior* a `arbol2`. Diremos que *un árbol binario es inferior a otro si los elementos del primer árbol, en los nodos coincidentes en posición, son* ***menores*** *que los del segundo árbol.* En la siguiente figura de ejemplo, `arbol1` es inferior a `arbol2`.



4. Implementación de la función `esMonodistante?`

Un árbol *monodistante* de orden N es un árbol binario de números enteros en el que *la suma de los valores de los nodos de cada camino que va desde la raíz a un nodo hoja es igual a N*. Implementa una función booleana `esMonodistante?` que, dado un árbol binario de enteros y un N, determine si dicho árbol binario es monodistante de orden N.

5. Implementación del procedimiento `encender`

Se quiere utilizar un árbol binario para gestionar el encendido de las luces de un árbol de Navidad. Se supone que el árbol es completo, es decir, todos sus niveles están completos (o sea, todos los nodos tienen dos hijos, excepto los del último nivel). Cada una de las bolas se identifica por sus coordenadas, formadas por dos números: (nivel, posición en el nivel). Así, el nivel 0 tiene la bola (0,1); el nivel 1, las bolas (1,1) y (1,2); el nivel 2, las bolas (2,1), (2,2), (2,3), (2,4); etcétera. Implementa en pseudocódigo un procedimiento `encender` que reciba un árbol de Navidad y las coordenadas de una bola del árbol (nivel, posición en el nivel) y la encienda (si ya está encendida, la deja igual). Supón que se reciben siempre coordenadas correctas, es decir, de una bola existente en el árbol.

```
tipos estadoBombilla = (apagada, encendida);
        arbin = ↑nodo;
        nodo = registro
                 dato: estadoBombilla;
                 izq, der: arbin
               freg

procedimiento encender(e/s a: arbin; ent nivel: entero; ent posición: entero)
```

6. Implementación del procedimiento `eliminarApariciones`

Se quiere implementar una operación que, dado un árbol binario de números enteros, elimine del árbol todas las apariciones de un número n dado, sin eliminar ni introducir repeticiones de otros datos distintos a n, y conservando la forma del árbol original de acuerdo a las siguientes reglas:

- Si un dato a eliminar se encuentra en una hoja, esta se eliminará del árbol inmediatamente;
- Si se encuentra en un nodo no hoja con un solo hijo, el nodo será eliminado y sustituido por su hijo no vacío;
- Si se encuentra en un nodo no hoja con dos hijos, el nodo deberá ser marcado como borrado, para que pueda ser eliminado del árbol en cuanto sea posible hacerlo aplicando las reglas dadas (considerando para ello que un nodo marcado como borrado es un nodo a eliminar del árbol).

A efectos de la aplicación de dichas reglas, cualquier árbol formado únicamente con nodos marcados como borrados será considerado un árbol vacío.

Implementa el procedimiento `eliminarApariciones` que elimine del árbol todas las apariciones de un número n dado, eliminando además todos los nodos del árbol que estén marcados como borrados y que le sea posible eliminar del árbol, aplicando las reglas anteriores.

```
tipos arbin = ↑nodo;
       nodo = registro
                  dato: entero;
                  borrado: booleano; {verdad si se borró el dato que tenía el nodo,
                                      pero aún no se ha eliminado el nodo del árbol}
                  izq, der: arbin
              freg

procedimiento eliminarApariciones(e/s a: arbin; ent n: entero)
```

7. Implementación de la función `último`

Diseña la función `último` que, dado un árbol binario no vacío de elementos, devuelva el último de los elementos del árbol según el recorrido en anchura.

```
tipos arbin = ↑nodo;
       nodo = registro
                  dato: elemento;
                  izq, der: arbin
              freg

función último(a: arbin) devuelve elemento
```