

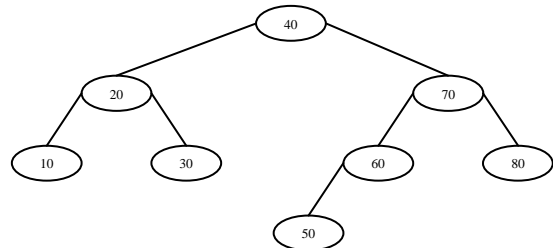
Examen TAD/PED diciembre 2003

- Normas:**
- Tiempo para efectuar el ejercicio: **2 horas**
 - En la cabecera de cada hoja **Y EN ESTE ORDEN** hay que poner: **Apellidos, Nombre**. Cada pregunta se escribirá en folios diferentes.
 - Se dispone de 20 minutos para abandonar el examen sin que corra convocatoria.
 - Las soluciones al examen se dejarán en el campus virtual.
 - Se puede escribir el examen con lápiz, siempre que sea legible
 - **Todas las preguntas tienen el mismo valor**. Este examen vale el 60% de la nota de teoría.
 - **Publicación de notas de exámenes:** 9 de diciembre. **ÚNICA fecha de revisión de exámenes:** 17 de diciembre. El lugar y la hora se publicará en el campus virtual.
- Los alumnos que estén en 5ª o 6ª convocatoria deben indicarlo en la cabecera de todas las hojas

1. Sea un vector de números naturales. Utilizando exclusivamente las operaciones *asignar* y *crear*, define la sintaxis y la semántica de la operación *eliminar* que borra las posiciones pares del vector marcándolas con "-1". Para calcular el resto de una división, se puede utilizar la operación MOD.

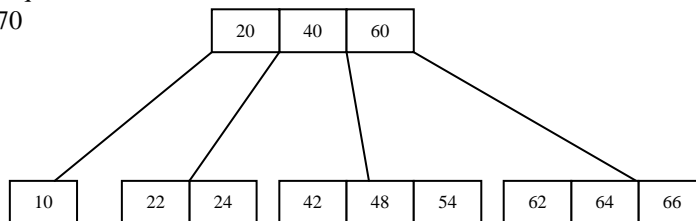
2. a) Dado el siguiente árbol AVL, realizar las siguientes operaciones en el siguiente orden, indicando en cada caso las transformaciones realizadas, y empleando los siguientes criterios: en caso de un nodo interior, sustituir por el mayor de la izquierda.

- Insertar las etiquetas 45, 85, 110, 5, 1
- Borrar las etiquetas: 60, 40



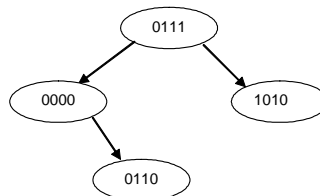
b) Dado el siguiente árbol 2-3-4, realizar las siguientes operaciones en el siguiente orden, indicando en cada caso las transformaciones realizadas, y empleando los siguientes criterios: tomar *r* como el hermano de la derecha, y en caso de un nodo interior, sustituir por el mayor de la izquierda.

- Insertar las etiquetas 50, 26, 56, 70
- Borrar las etiquetas: 22, 48, 24



NOTA: Si el árbol que se obtiene en cualquiera de los apartados no es el correcto, no se corregirán los siguientes apartados.

3. Insertar los ítems que vienen a continuación en el siguiente conjunto que está representado como un árbol digital: 1100, 1101, 1011, 1111, 1110, 0111, 0100.

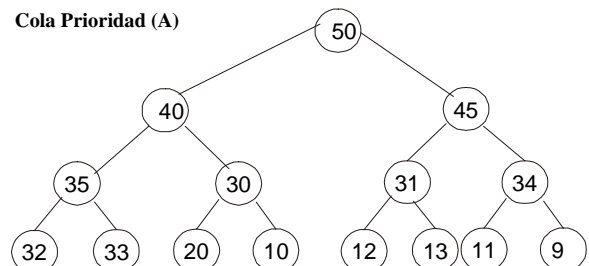


4. a) Sobre la cola de prioridad máxima A representada como un árbol leftist, realizar una operación de borrado.

b) Dada la clase *ArbolLFT*, escribir el código en C++ de la función *void ArbolLFT::cMIN(void)* que recalcula (a partir de los caminos mínimos ya almacenados en sus descendientes) y almacena (en *cmin*) el camino mínimo del nodo al que apunta *p*. Escribir una explicación del algoritmo elegido con una longitud máxima de 3 líneas. (NOTA: Los errores de compilación se puntuarán de forma negativa).

<pre> class ArbolLFT { ... void cMIN(void); ... private: TNode* p; }; </pre>	<pre> class TNode { friend class ArbolLFT; ... private: int clave; int cmin; ArbolLFT izq, der; }; </pre>
--	---

Cola Prioridad (A)



Examen TAD/PED diciembre 2003. Soluciones

1)

eliminar: vector \rightarrow vector

Var v:vector; i,x:natural;

eliminar(crear()) = crear()

si (i MOD 2) == 0

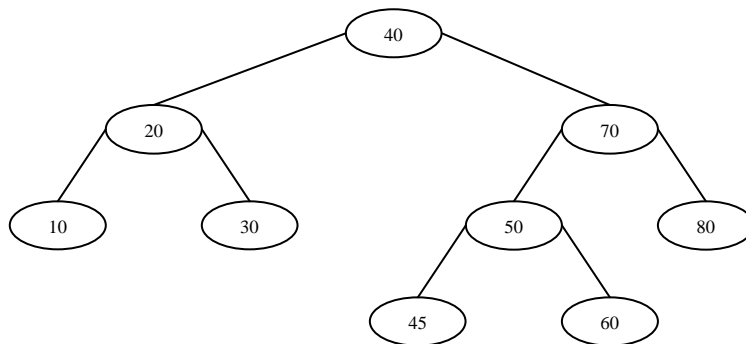
entonces eliminar(asignar(v,i,x)) = asignar(eliminar(v),i,-1)

sino eliminar(asignar(v,i,x)) = asignar(eliminar(v),i,x)

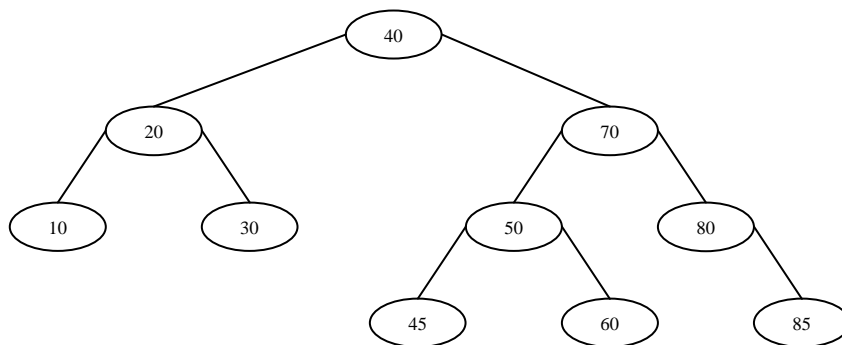
2)

a)

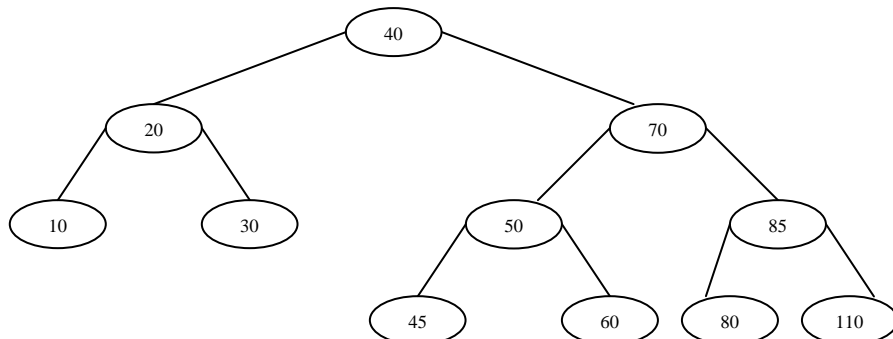
Insertar 45: ROTACIÓN II



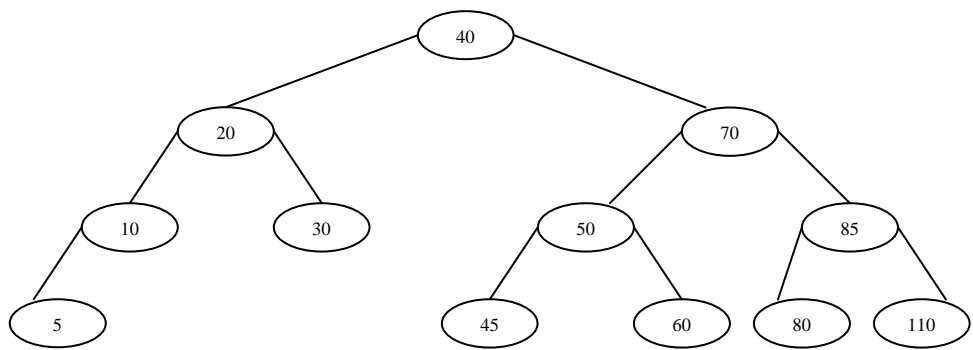
Insertar 85:



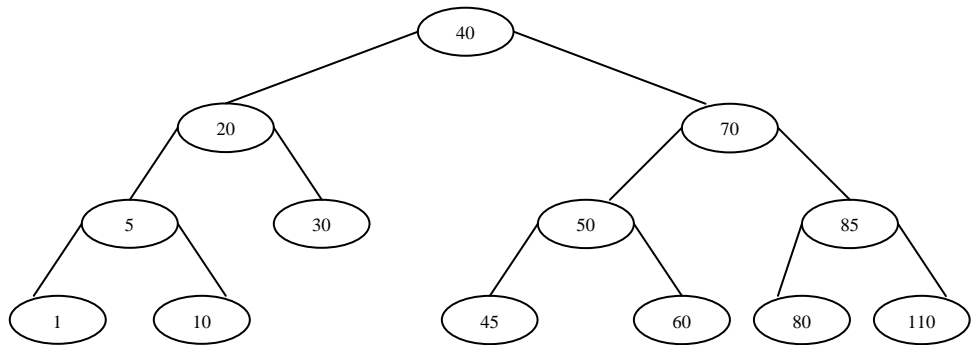
Insertar 110: ROTACIÓN DD



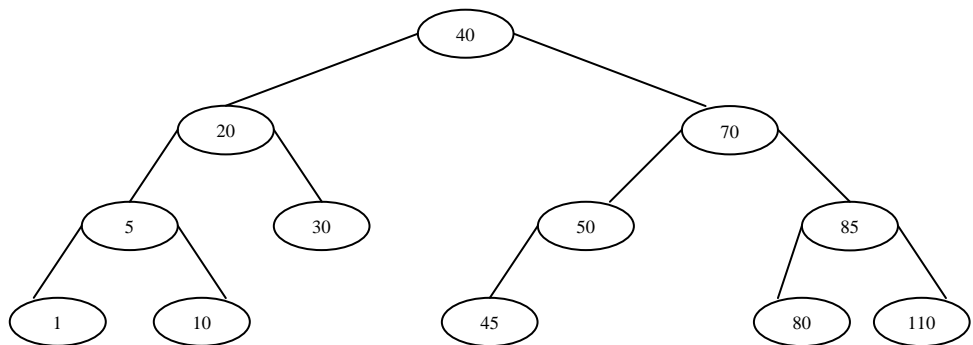
Insertar 5:



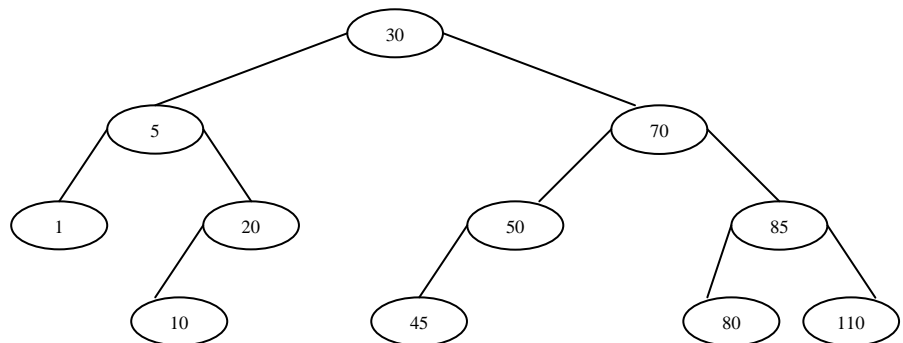
Insertar 1: ROTACIÓN II



PUNTO 2
Borrar 60:

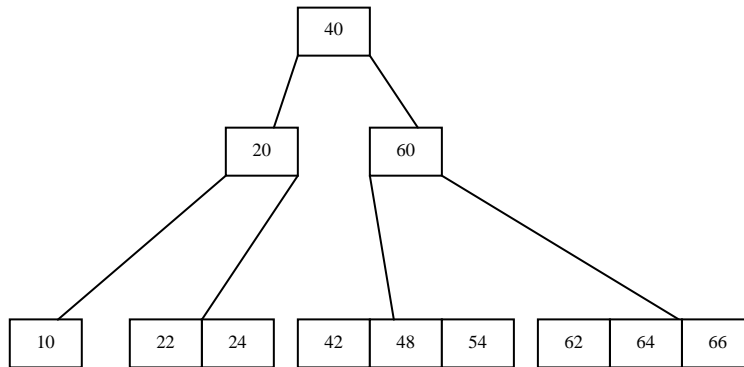


Borrar 40: ROTACIÓN II

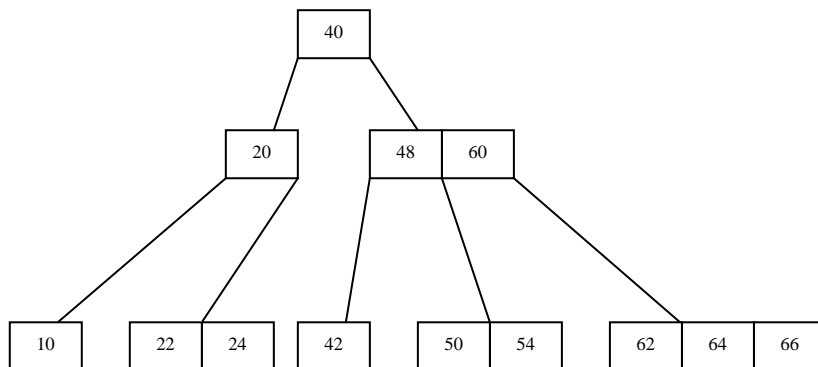


b)

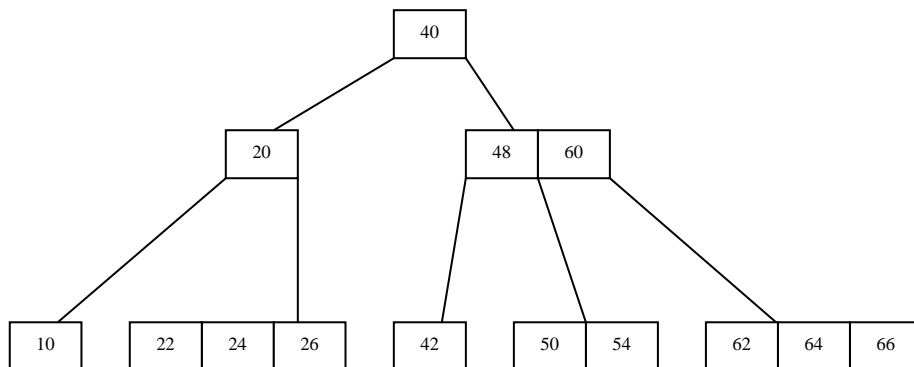
Insertar 50: DIVIDERAIZ



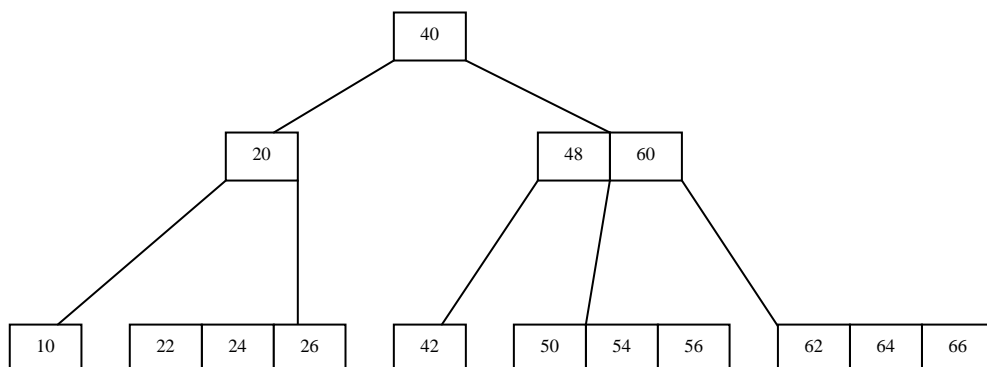
DIVIDEHIJODE2



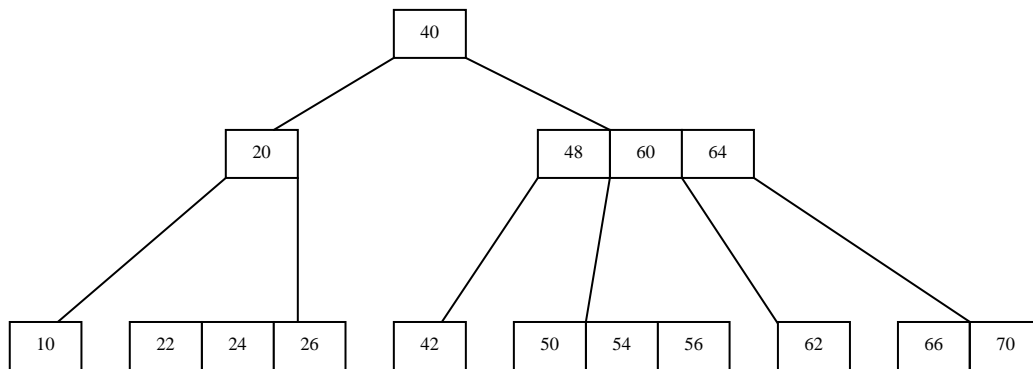
Insertar 26:



Insertar 56:

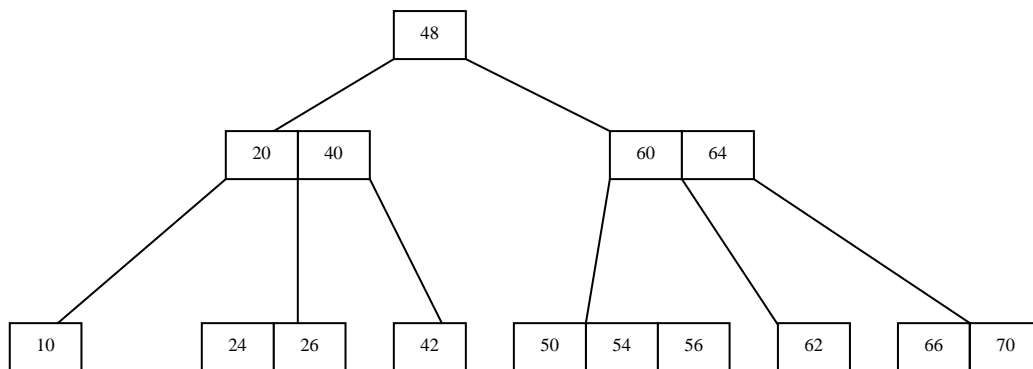


Insertar 70: DIVIDEHIJODE3

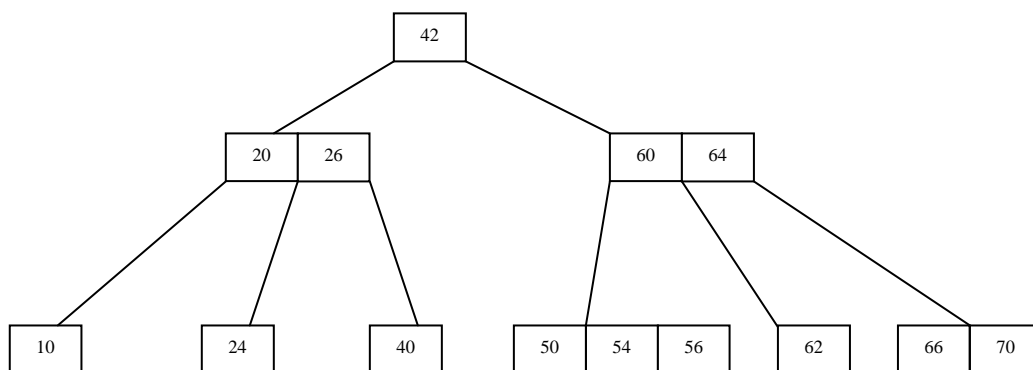


PUNTO 2

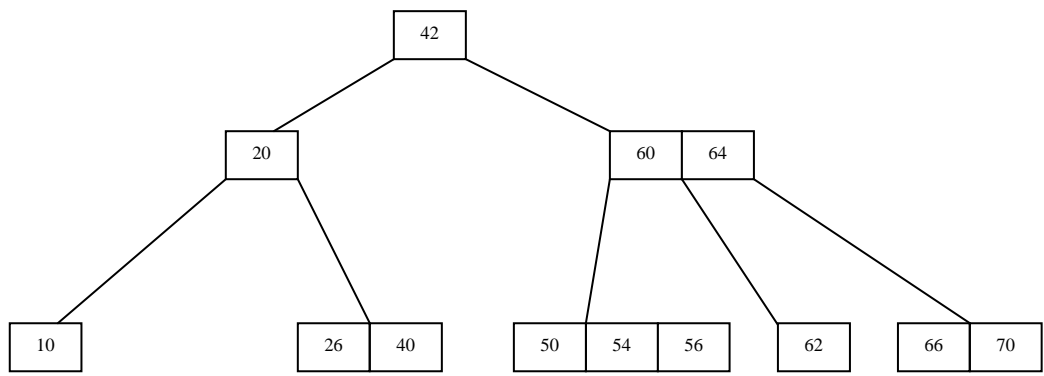
Borrar 22: q es 2-nodo y r es 3-nodo (ROTACIÓN)



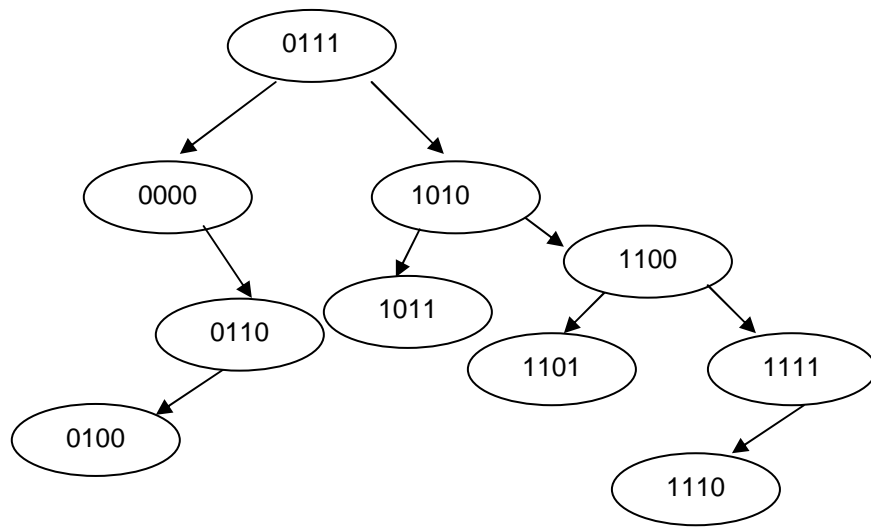
Borrar 48: q es 2-nodo y r es 3-nodo (ROTACIÓN)



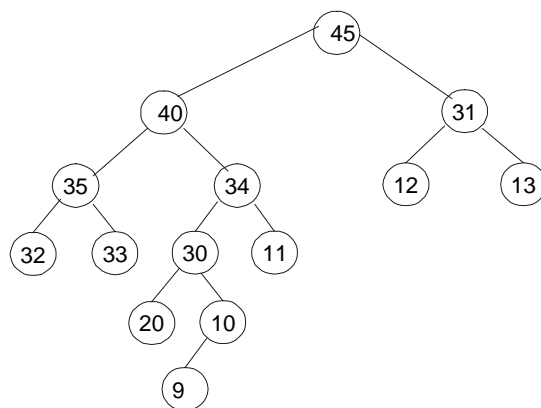
Borrar 24: q es 2-nodo y r es 2-nodo (COMBINACIÓN)



3) Resultado:



4) a)



b)

```

void
ArbolLFT::cMIN(void) {
    if (p != NULL) {
        if (p->der.p == NULL)
            p->cmin = 1;
        else
            p->cmin = p->der.p->cmin + 1;
    }
}
  
```

Apellidos:

Nombre:

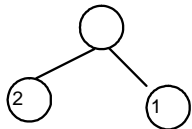
Convocatoria:

DNI:

Examen TAD/PED diciembre 2003

Modalidad 0

- Normas:**
- La entrega del test no corre convocatoria.
 - Tiempo para efectuar el test: **20 minutos**.
 - Una pregunta mal contestada elimina una correcta.
 - Las soluciones al examen se dejarán en el campus virtual.
 - **Una vez empezado el examen no se puede salir del aula hasta finalizarlo. A continuación comenzará el siguiente ejercicio.**
 - **El test vale un 40% de la nota de teoría.**
 - En la **hoja de contestaciones** el verdadero se corresponderá con la **A**, y el falso con la **B**.

	V	F	
La complejidad logarítmica aparece en algoritmos que descartan muchos valores (generalmente la mitad) en un único paso.	<input type="checkbox"/>	<input type="checkbox"/>	V 1.
En C++, la parte privada de una clase sólo es accesible por los métodos de la propia clase.	<input type="checkbox"/>	<input type="checkbox"/>	F 2.
El tipo posición en una lista con acceso por posición se puede instanciar a diferentes tipos de objetos.	<input type="checkbox"/>	<input type="checkbox"/>	V 3.
El recorrido en postorden es el inverso especular del recorrido en preorden para un árbol binario dado.	<input type="checkbox"/>	<input type="checkbox"/>	V 4.
En un árbol AVL las inserciones siempre se realizan en las hojas.	<input type="checkbox"/>	<input type="checkbox"/>	V 5.
Dado un árbol 2-3 de altura h con n items con todos sus nodos del tipo 3-Nodo: la complejidad de la operación de búsqueda de un ítem es $O(\log_3 h)$.	<input type="checkbox"/>	<input type="checkbox"/>	F 6.
En un árbol 2-3-4 las reestructuraciones se realizan desde la raíz hacia las hojas.	<input type="checkbox"/>	<input type="checkbox"/>	V 7.
En un árbol rojo-negro, el número de enlaces negros ha de ser mayor que el de enlaces rojos.	<input type="checkbox"/>	<input type="checkbox"/>	F 8.
La altura del árbol B m-camino de búsqueda es " $\log_m n$ ", con " n =número total de claves".	<input type="checkbox"/>	<input type="checkbox"/>	F 9.
La función de redispersión en una tabla hash abierta, para que se recorran todas las posiciones del vector, tiene que cumplir que el valor de B sea primo.	<input type="checkbox"/>	<input type="checkbox"/>	F 10.
El siguiente árbol es un montículo doble: 	<input type="checkbox"/>	<input type="checkbox"/>	F 11.
Todo árbol Leftist cumple las condiciones para ser un árbol binario de búsqueda.	<input type="checkbox"/>	<input type="checkbox"/>	F 12.
Al representar un grafo dirigido de N vértices y K aristas con una lista de adyacencia, la operación de hallar la adyacencia de entrada de un vértice, tiene una complejidad de $O(N^2)$.	<input type="checkbox"/>	<input type="checkbox"/>	V 13.
En C++, el constructor de copia sustituye al operador asignación.	<input type="checkbox"/>	<input type="checkbox"/>	F 14.