Apellidos:
Nombre:
Convocatoria:
DNI:

Examen PED junio 2011
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.**
- En la **hoja de contestaciones** el verdadero se corresponderá con la **A**, y el falso con la **B**.

	V	F		
EsVacia: PILA -> BOOLEAN Si P y Q son pilas: Q = EsVacia (P), es una expresión sintácticamente correcta	❑	❑	1	F
En C++, cuando se sobrecarga un operador que no modifica al operando izquierdo (por ejemplo : "+") se debe crear un objeto temporal, que luego el método devuelve por valor	❑	❑	2	V

La complejidad temporal (en su caso promedio) del siguiente fragmento de código es $\Theta(n^2)$
`int i, length, n, i1, i2, k;`
`for (i = 0, length = 1; i < n-1; i++) {`
`  for (i1 = i2 = k = i; k < n-1 && a[k] < a[k+1]; k++, i2++);`
`  if (length < i2 – i1 + 1) length = i2 – i1 + 1;`
`}`
	❑	❑	3	F

En cualquier tipo de datos lineal cada elemento tiene como máximo un único sucesor y un único predecesor	❑	❑	4	V
El máximo número de nodos en un árbol binario de altura k-1 es $2^k - 1$, $k \geq 1$.	❑	❑	5	F
En la inserción, en el peor de los casos, las rotaciones realizadas en los árboles AVL para mantenerlos balanceados tienen un coste temporal lineal respecto al número de ítems del árbol	❑	❑	6	F
El borrado de un elemento en un árbol 2-3 se realiza en las hojas. Se pueden producir reestructuraciones del árbol en el camino de vuelta desde las hojas a la raíz del árbol.	❑	❑	7	V
En un árbol 2-3-4 de altura=2 y número de elementos=15, si se insertara un nuevo elemento se tendría que hacer un DIVIDERAIZ y un DIVIDEHIJODE2	❑	❑	8	V
Un árbol rojo-negro es un árbol B con m=2	❑	❑	9	F
Todo árbol binario de búsqueda es un árbol B con m=4.	❑	❑	10	F
La complejidad temporal en su peor caso de la operación de Unión entre 2 conjuntos con m elementos cada uno y representados con una lista desordenada es $0(m^2)$.	❑	❑	11	V
En dos tablas de dispersión cerrada y abierta con tamaños B=7 y B=6 respectivamente, siempre se cumple que el factor de carga en la abierta es mayor que en la cerrada.	❑	❑	12	F
Todo árbol binario que además es árbol mínimo es un Heap Mínimo	❑	❑	13	F
La complejidad temporal, en su peor caso, de la operación de PERTENECE de un elemento de tamaño N en un árbol de búsqueda digital es 0(N+1).	❑	❑	14	V

Examen PED junio 2011

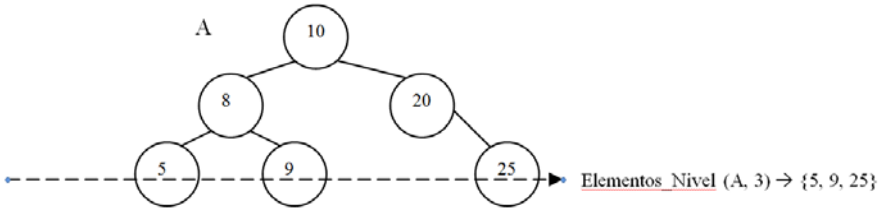
Normas:

- 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 hojas 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 notas**: 20 de junio. **Revisión del examen teórico**: 21 de junio en el aula LS13I POIV (planta sótano EPSIV) de 9:00 a 10:00 h.
- **Los alumnos que estén en 5ª o 6ª convocatoria deben indicarlo en la cabecera de todas las hojas**

1. Realiza la especificación algebraica de la función *Elementos_Nivel (arbin, natural).*

Esta función recibe como parámetros un árbol binario de números naturales y un número natural, devolviendo una lista de naturales. La lista devuelta contendrá los elementos del árbol que se encuentran en el nivel indicado por el número natural pasado como parámetro. Aclaraciones:

- El nivel de la raíz es el 1.
- Si el árbol está vacío se devolverá una lista vacía.
- Los elementos del nivel indicado deben almacenarse en la lista comenzando con el elemento situado más a la izquierda y acabando con el elemento situado más a la derecha, tal y como ilustra el siguiente ejemplo:

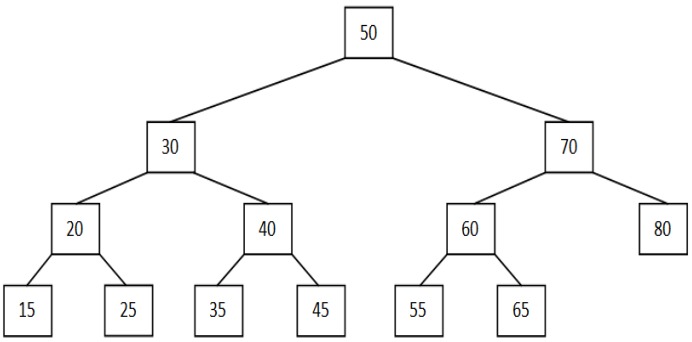


2. Dado el siguiente árbol, realiza estas operaciones (inserta 10, borra 25, inserta 52, borra 50, borra 30, inserta 5, inserta 37) en los siguientes tres apartados, comenzando siempre con este árbol:

a) Suponiendo que es un árbol binario de búsqueda
b) Suponiendo que es un árbol AVL.
c) Suponiendo que es un árbol 2-3-4.

Aplica los siguientes criterios cuando sean necesarios:

- En caso de 2 hijos, sustituid por el mayor de la izquierda.
- En caso de 2 hermanos, elegid el hermano de la izquierda.



3. Dado el grafo no dirigido representado por la lista de de adyacencia que se muestra a continuación:

a) Obtener el árbol extendido en profundidad partiendo del vértice 1 y la clasificación de las aristas.
b) ¿Es un grafo conexo? Justifica tu respuesta

Nota: La lista de adyacencia de cada vértice se recorre de menor a mayor vértice para todos los casos del ejercicio. Las listas están desordenadas y en las mismas puede aparecer información duplicada.

1 → 6→ 5 → 7 →8
2 → 4 → 6
3 → 5
4 → 1 → 5 → 6 → 7 → 8
5
6 → 1 → 2 → 4
7 → 1 → 2 → 4
8 → 1 → 2 → 4
9
10
11 → 10 → 9
12

4. Sea un árbol de búsqueda digital inicialmente vacio.

a) Inserta los siguientes elementos: 1001, 1111, 0010, 0101, 1110, 1001, 0011, 0111, 1010, 0000, 1101. Antes de insertar cada uno de los ítems, hay que calcular un código comprobador de errores (CCE o checksum) que añadiremos al final de la cadena. Este código comprobador de errores es el cálculo de la XOR (OR Exclusiva) de todos de los bits de la cadena. El primer bit de la cadena es el primer bit empezando por la izquierda de la cadena.

La tabla de verdad de la operación XOR es:

0 Xor 0 = 0
0 Xor 1 = 1
1 Xor 0 = 1
1 Xor 1 = 0

Ejemplo: Xor de la cadena 1011. 1 Xor 0 Xor 1 Xor 1 = 1

b) ¿Cuál será la altura máxima del árbol? Calcula la complejidad temporal en el peor de los casos de la función de inserción. Razona tu respuesta.

Examen PED junio 2011. Soluciones

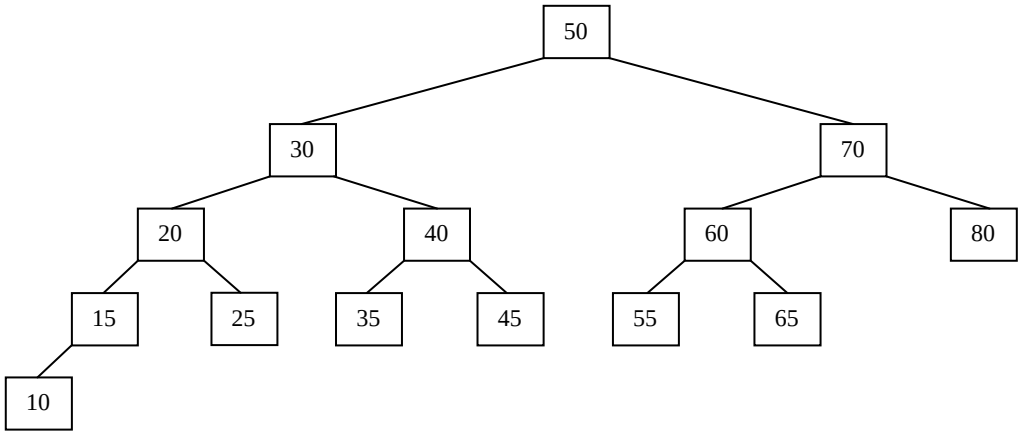
1.

```
MODULO arbol_binario
USA BOOL, NATURAL, LISTA

PARAMETRO TIPO item
        OPERACIONES
                Error_item() → item
FPARAMETRO

TIPO arbin

OPERACIONES
…..
Elementos_Nivel  (arbin, natural) → lista

VAR n,x: natural; i,d:arbin;
ECUACIONES
…
Elementos_Nivel ( crea_arbin(),  suc(n) )  =  crear_lista()
Elementos_Nivel ( crea_arbin(),  cero)  =  crear_lista()
Elementos_Nivel ( enraizar( i, x, d ), cero )  =  crear_lista()

Elementos_Nivel ( enraizar( i, x, d ), suc(n) )  =
si ( == (suc(n),  suc(cero) ) )
entonces
        inscabeza( crear_lista(),  x )
sino
        concatenar( Elementos_Nivel( i,  n),  Elementos_Nivel( d, n) )
fsi
```

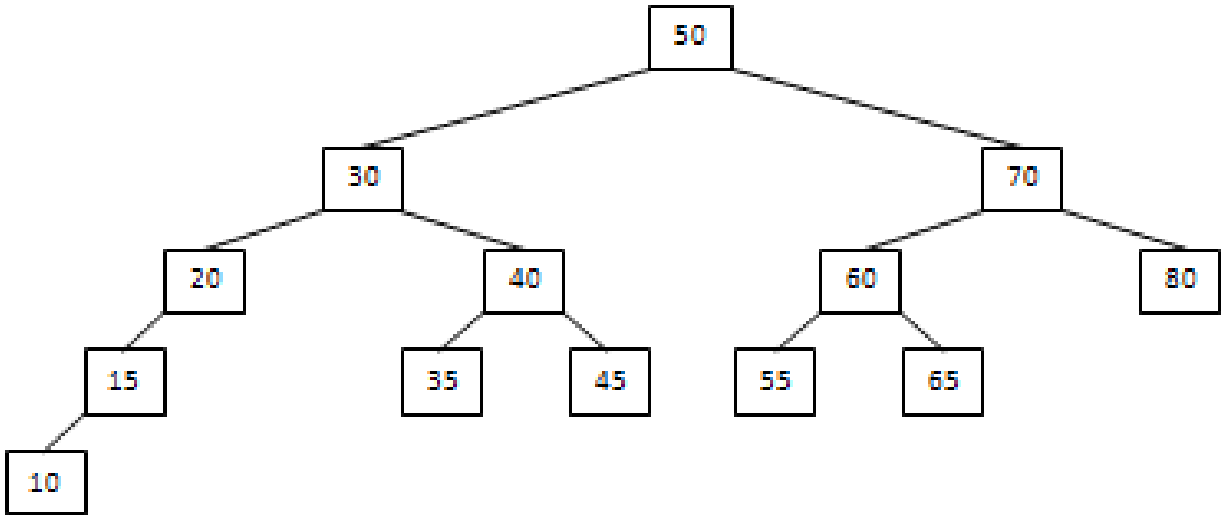
2.

a) Suponiendo que es un árbol binario de búsqueda.

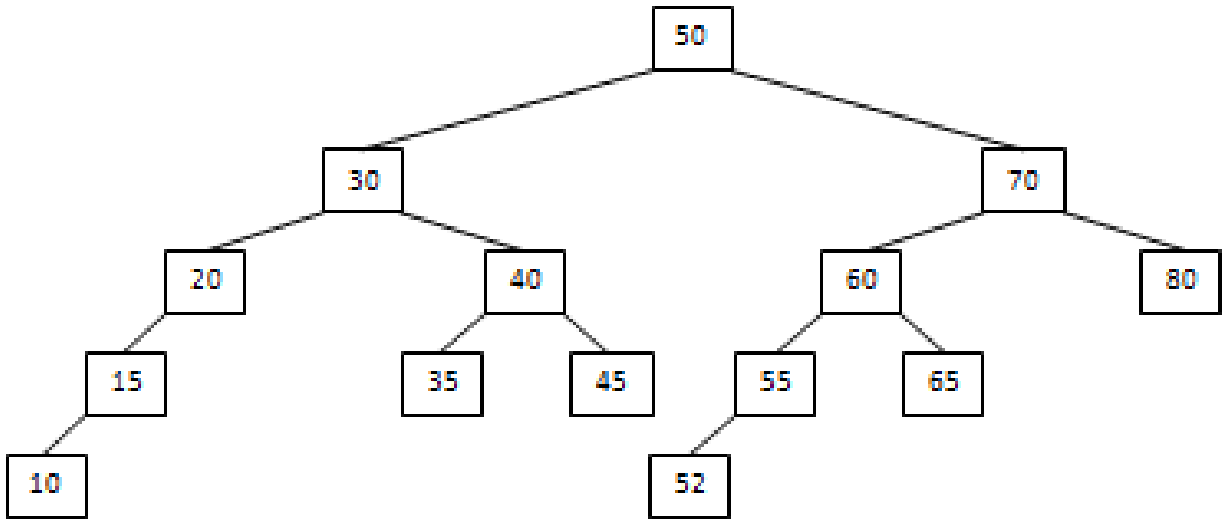
Inserta 10



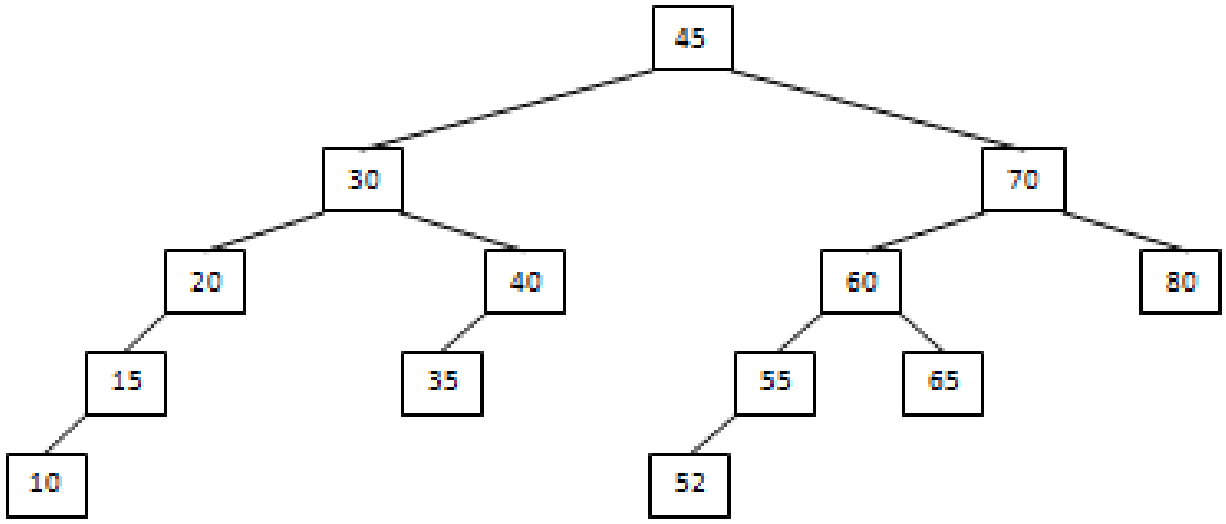
Borra 25



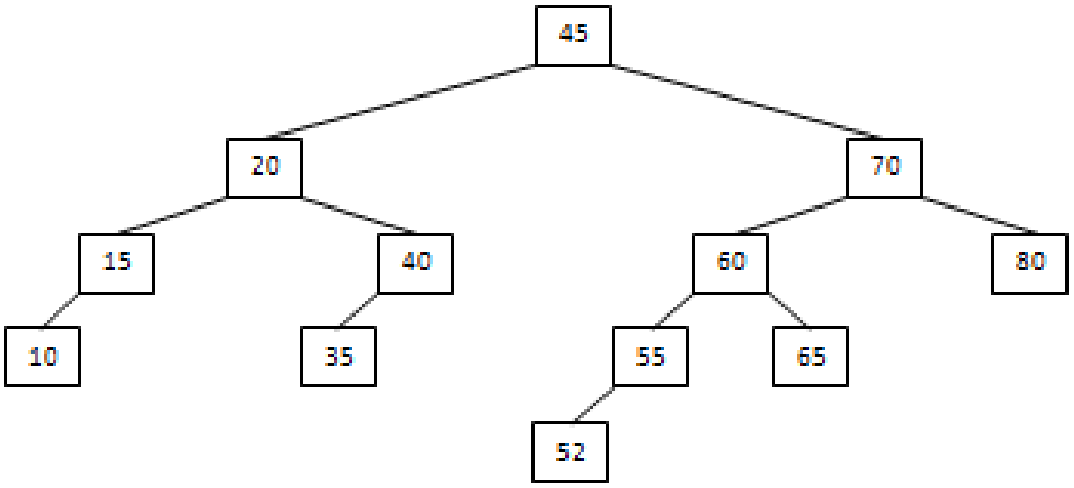
Inserta 52



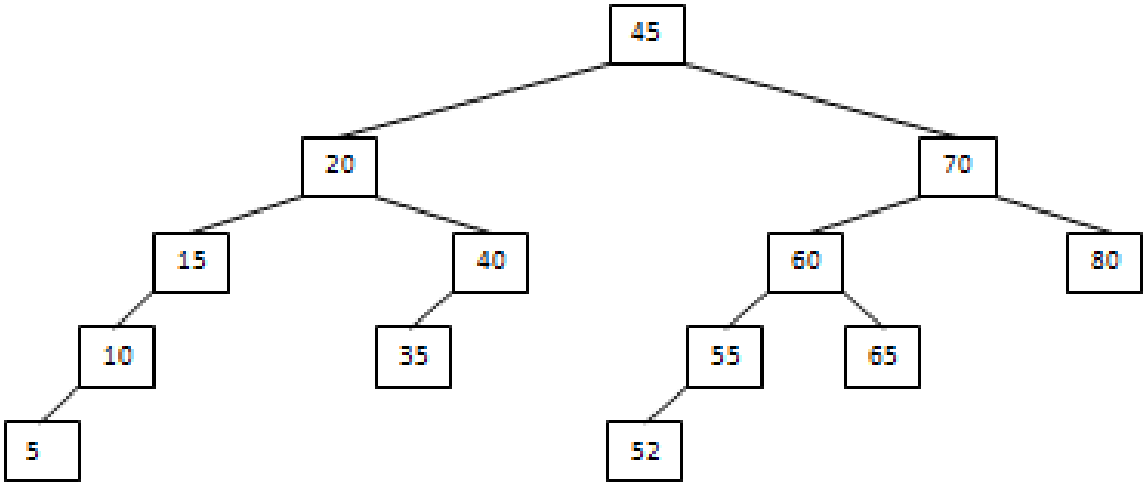
Borra 50



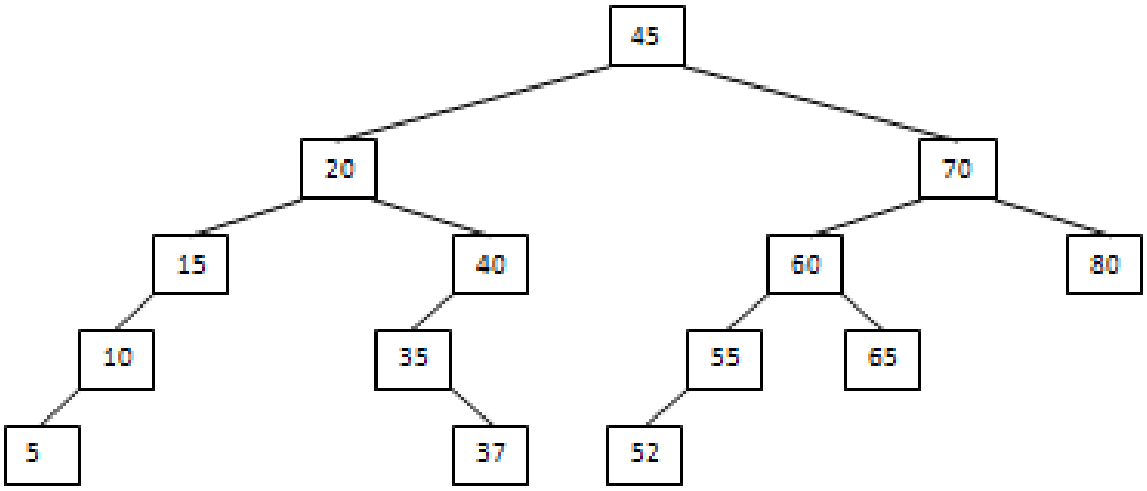
Borra 30



Inserta 5

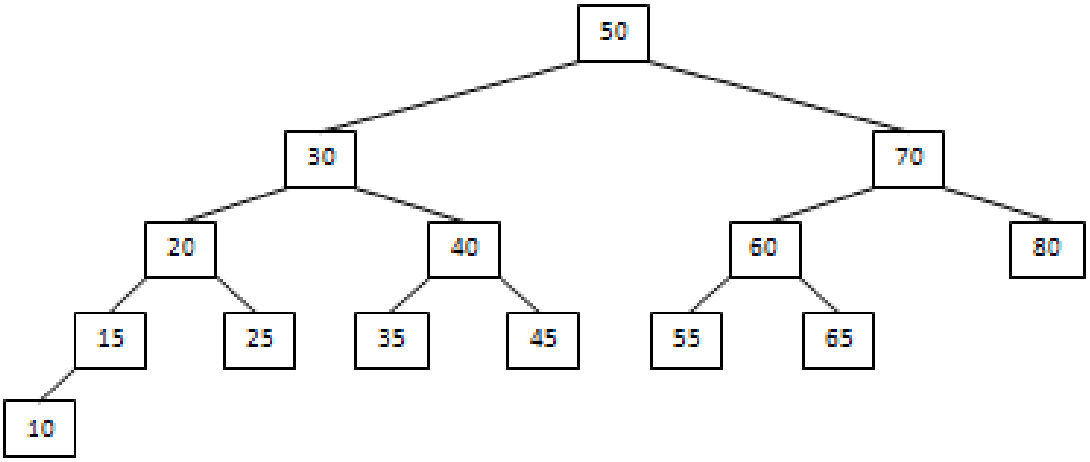


Inserta 37

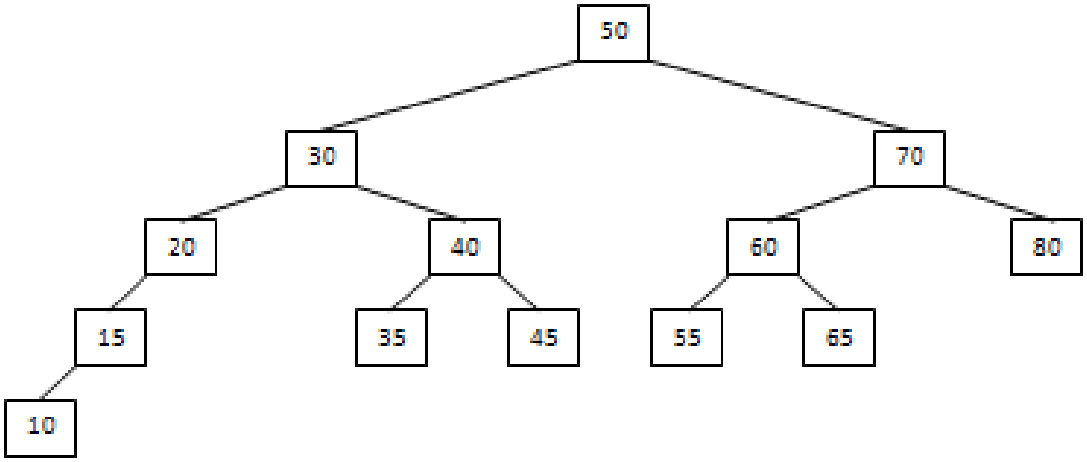


b) Suponiendo que es un árbol AVL.

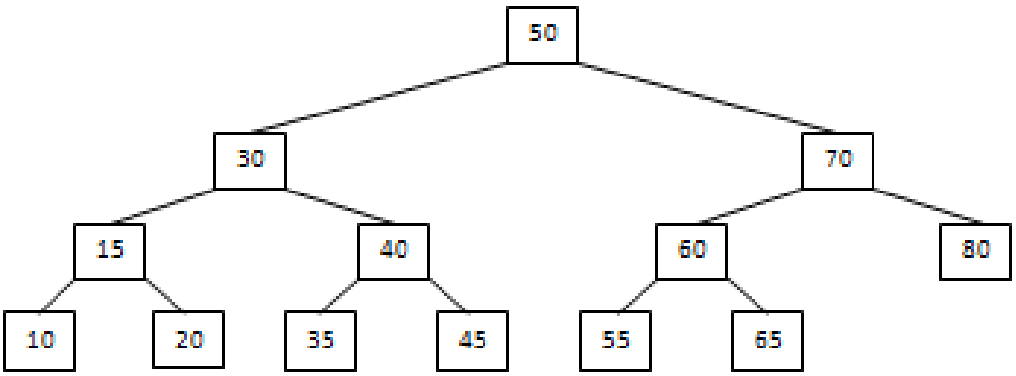
Inserta 10



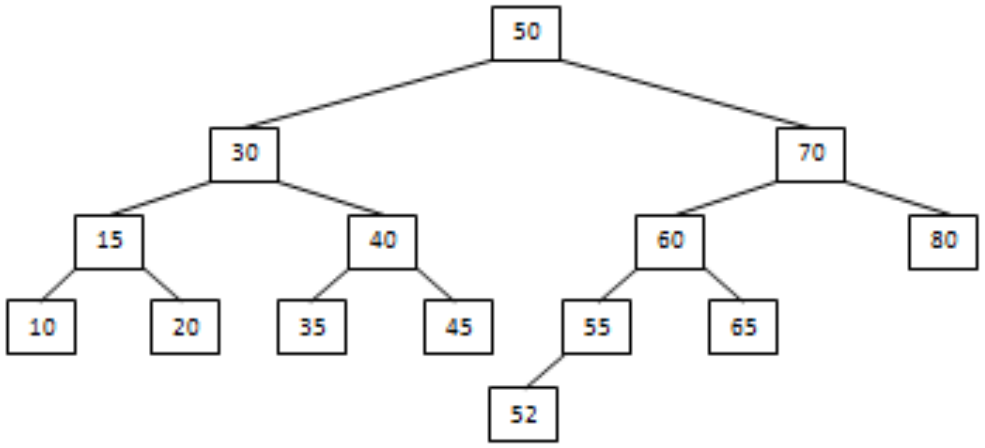
Borra 25



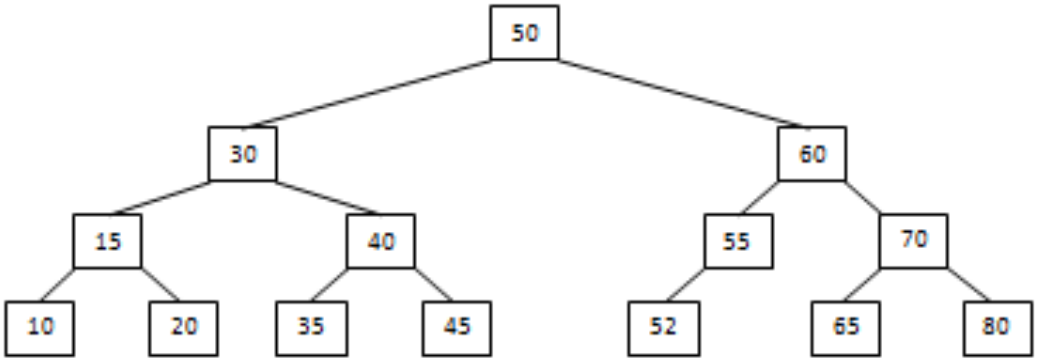
Al borrar 25 se produce un desequilibrio en el nodo 20 (FE=-2), que se resuelve con una rotación II.



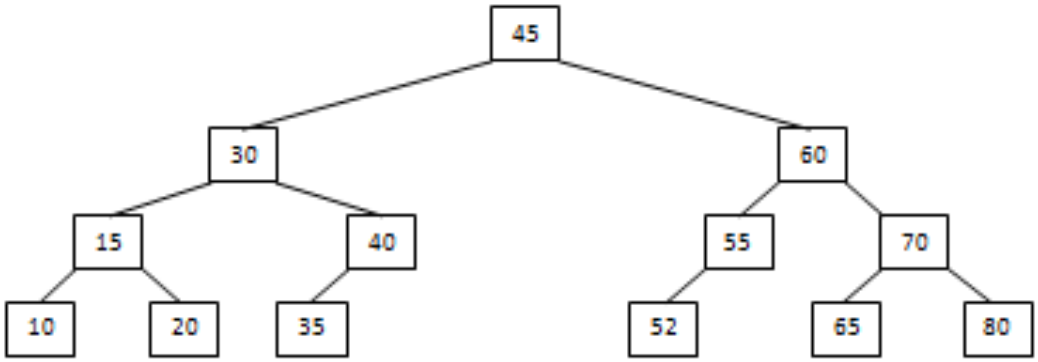
Inserta 52



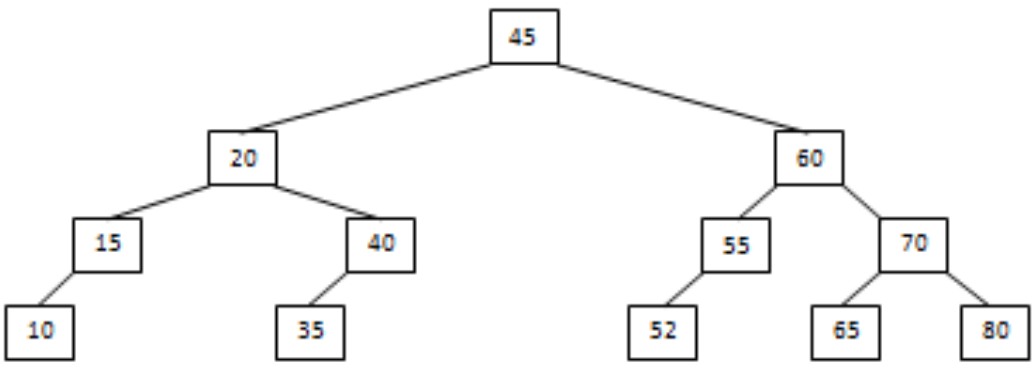
Al insertar el 52 se produce un desequilibrio en el nodo 70 (FE=-2), que se resuelve con una rotación II.



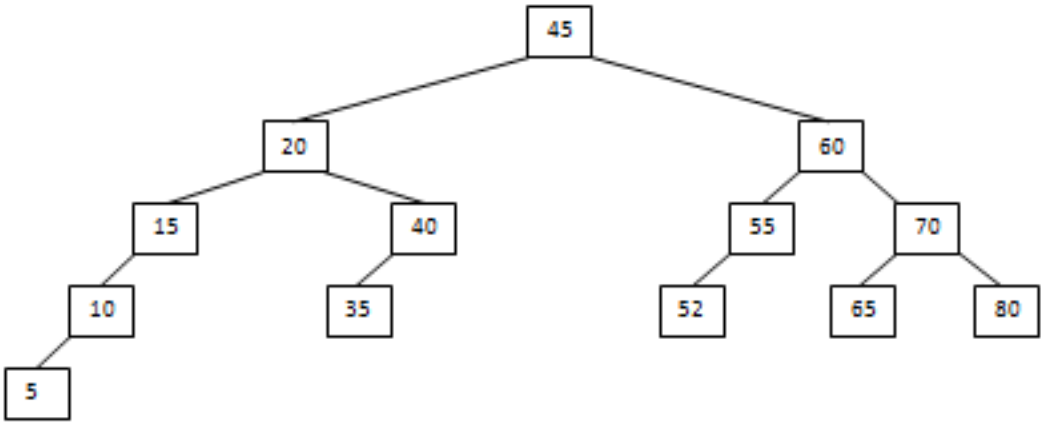
Borra 50



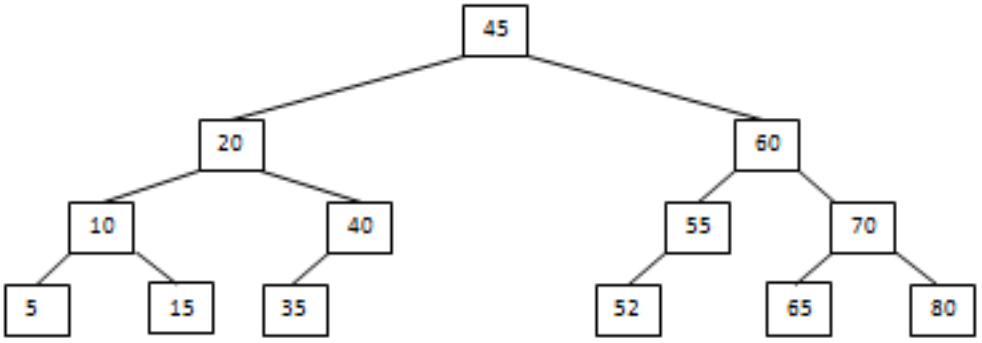
Borra 30



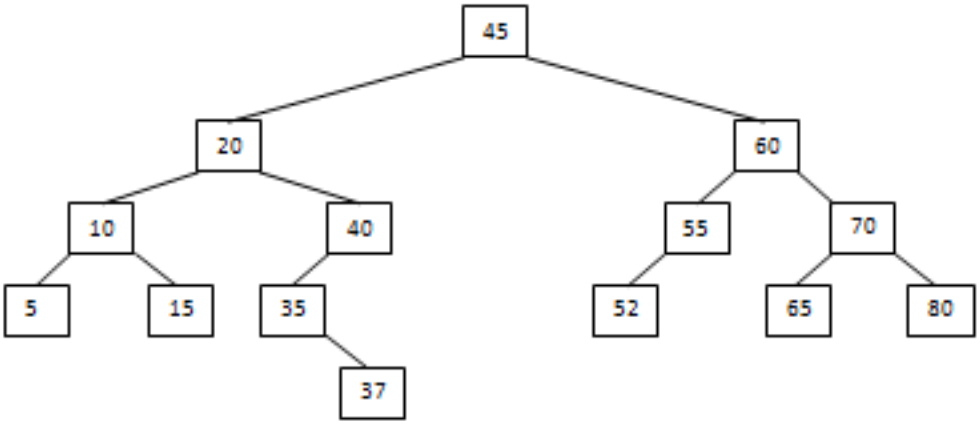
Inserta 5



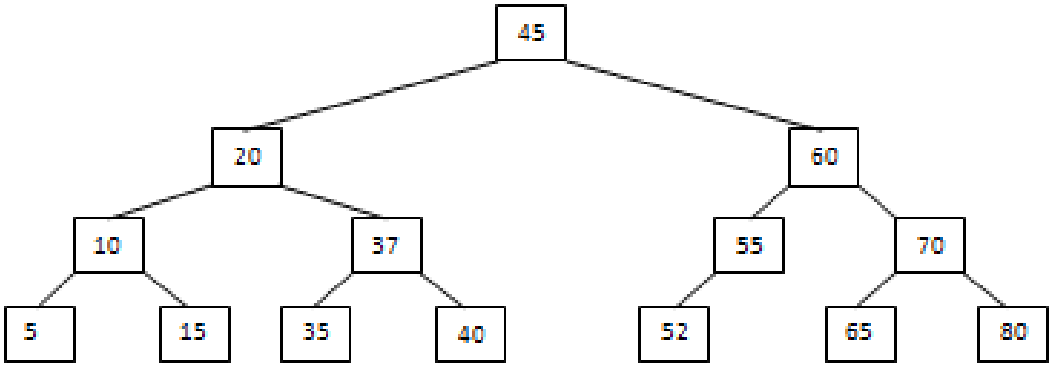
Al insertar el 5 se produce un desequilibrio en el nodo 15 (FE=-2), que se resuelve con una rotación II.



Inserta 37



Al insertar 37 se produce un desequilibrio en el nodo 40 (FE=-2), que se resuelve con una rotación ID.

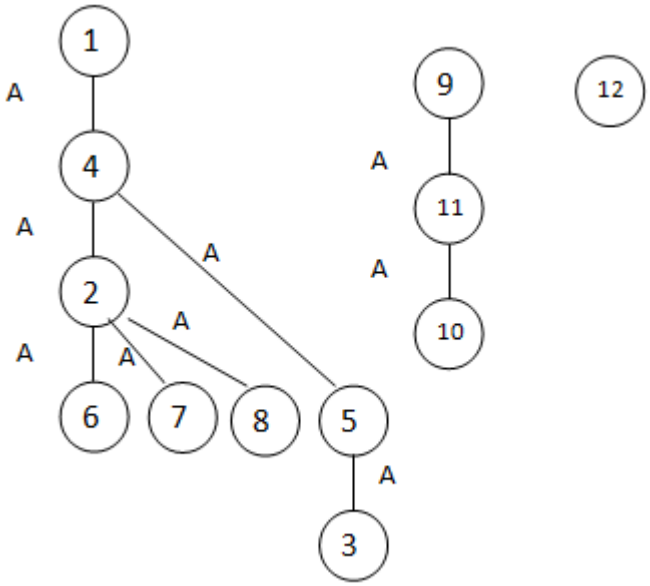


c) Suponiendo que es un árbol 2-3-4.

El árbol proporcionado no es un árbol 2-3-4, ya que todas las hojas no se encuentran en el mismo nivel. Por tanto, no tiene ningún sentido realizar las operaciones indicadas.

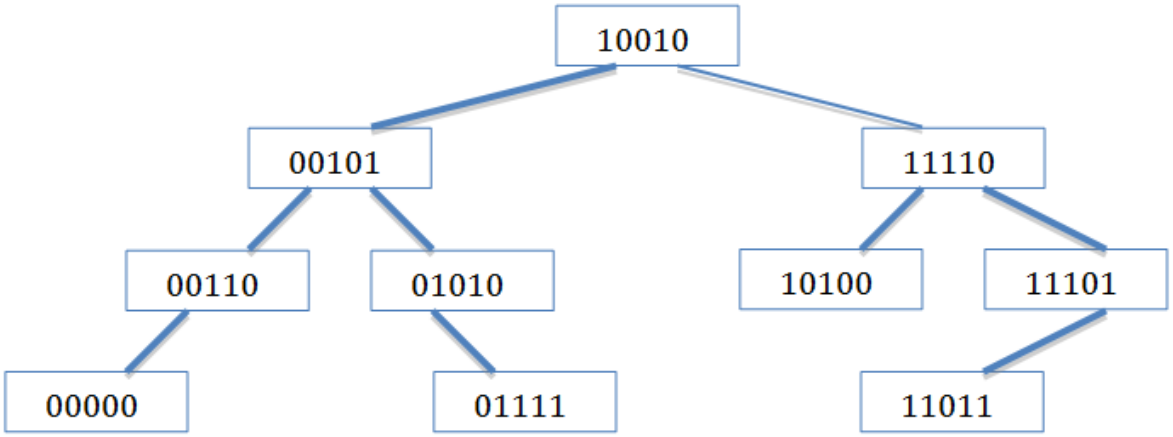
3.

a) Árbol extendido en profundidad. Las aristas marcadas son de árbol (A), el resto son de retroceso



b) No es un grafo conexo porque no hay camino entre cada par de vértices del grafo. Por ejemplo, del vértice 12 no hay camino a ningún vértice.

3. a)



XOR
10010
11110
00101
01010
11101
10010 - Repetido
00110
01111
10100
00000
11011

b)

La altura (h) máxima de un árbol de búsqueda digital es el numero de bits de la clave mas 1.

En este ejemplo concretamente, no nos tenemos que confundir y decir que la longitud de la clave es 5, porque el 5º bit es un bit sin información. Por lo tanto la altura máxima de este árbol de búsqueda digital es 4+1 = 5.

Por lo tanto, el checksum, no es un bit de información, sino calculado y no se tiene en cuenta a la hora de insertar elementos en el árbol.

En cuanto a la complejidad temporal O(n), es la altura porque cuando insertamos elementos en este tipo de arboles, hacemos tantas comparaciones como alturas tiene el árbol. La altura es el número de bits de la clave + 1. La complejidad es h.