Apellidos:
Nombre:
Convocatoria:
DNI:

Examen TAD/PED julio 2009
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: 4 puntos.**
- En la **hoja de contestaciones** el verdadero se corresponderá con la **A**, y el falso con la **B**.

	V	F		
En C++, al declarar una clase "A" como AMIGA de otra clase "B", hay que declarar forzosamente todas las funciones de "A" como AMIGAS de "B"	☐	☐	1	F
En C++, si una clase "B" se construye por composición (layering) a partir de otra clase "A", definiendo un objeto de la clase "A" en su parte privada, cuando se invoca el destructor de "B", se invoca antes al destructor de "A" y luego al de "B"	☐	☐	2	V
La mejor complejidad temporal que se puede conseguir en un algoritmo es O(n), con "n" como la talla del problema.	☐	☐	3	F
La sintaxis y la semántica de la operación *quita_pares* que actúa sobre una lista y devuelve la lista original en la que se han eliminado los elementos que ocupan las posiciones pares es la siguiente: *quita_pares: lista → lista* *Var l1:lista; x,y:item;* *quita_pares(crear()) = crear()* *quita_pares(IC(crear(),x) = IC(crear(),x)* *quita_pares(IC(IC(l1,x),y)) = IC(l1,y)*	☐	☐	4	F
Dada la siguiente representación secuencial del árbol binario A, [14][8][19][5][ ][ ][21] el ítem 19 es el hijo derecha del ítem 8	☐	☐	5	F
Un árbol binario de búsqueda completo es un AVL	☐	☐	6	V
El numero de rotaciones que se nos pueden dar en un borrado de un AVL son como máximo 3 menos que la altura máxima del árbol	☐	☐	7	F
Todo AVL es un árbol 2-3.	☐	☐	8	F
En un árbol 2-3 con altura>3, la altura siempre disminuye si tras el borrado se producen al menos 2 combinaciones	☐	☐	9	F
En un árbol 2-3-4 de altura "h" y con (3^{h-1}) elementos, todos los nodos son del tipo 3-nodo.	☐	☐	10	F
En un árbol R-N, cualquier camino que se inicie en un nodo del segundo nivel tendrá el mismo número de hijos negros, ya que, en el árbol 2-3-4 equivalente todas sus hojas deben estar al mismo nivel.	☐	☐	11	F
En un árbol B con m=9 el número mínimo de hijos que tienen todos los nodos interiores, excepto la raíz, es 5.	☐	☐	12	V
La dispersión cerrada elimina el problema del clustering secundario.	☐	☐	13	F
La complejidad temporal, en su peor caso, de la operación de PERTENECE de una cadena de tamaño n en un trie es 0(n).	☐	☐	14	V

Examen PED julio 2009

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 y revisión exámenes:** se publicará lugar y hora en el campus virtual
- **Los alumnos que estén en 5ª o 6ª convocatoria deben indicarlo en la cabecera de todas las hojas**

1. A partir de la especificación algebraica de la lista, escribe la sintaxis y semántica de la operación SPP() que recibe una lista y devuelve una sublista en función de la posición inicial y la posición final indicadas; si la posición inicial no pertenece a la lista, se debe devolver una lista vacía; si la posición final no pertenece a la lista o es anterior a la posición inicial, se debe devolver una sublista desde la posición inicial hasta el final de la lista. Por ejemplo:

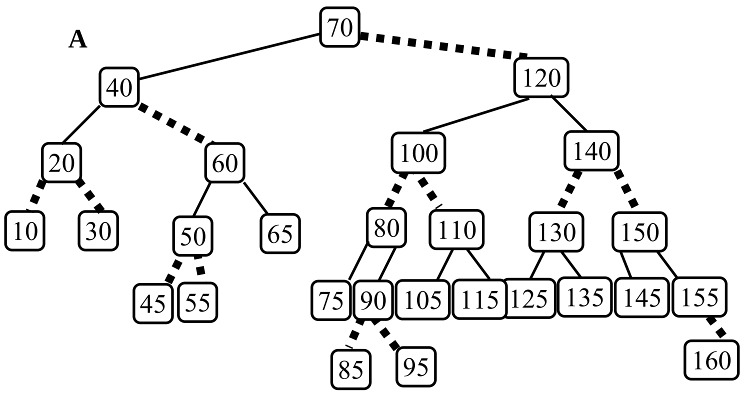
L = (e, a, f, b, c) p_e = posición letra e p_a = posición letra a p_b = posición letra b

SPP(L, p_e, p_b) = (e, a, f, b)
SPP(L, p_a, p_b) = (a, f, b)
SPP(L, p_a, p_e) = (a, f, b, c)

2. Dado el árbol Rojo-Negro A. Realizar las siguientes inserciones:

a) Sobre el árbol original A, insertar el 57.
b) Sobre el árbol original A, insertar el 97.
c) Sobre el árbol original A, insertar el 153.

Detallar los cambios de color y rotaciones realizadas, indicando los ítems implicados. **No será válido** realizar la inserción como si fuera un árbol 2-3-4 y realizar la transformación final a Rojo-Negro (en cuyo caso no se puntuará la pregunta).

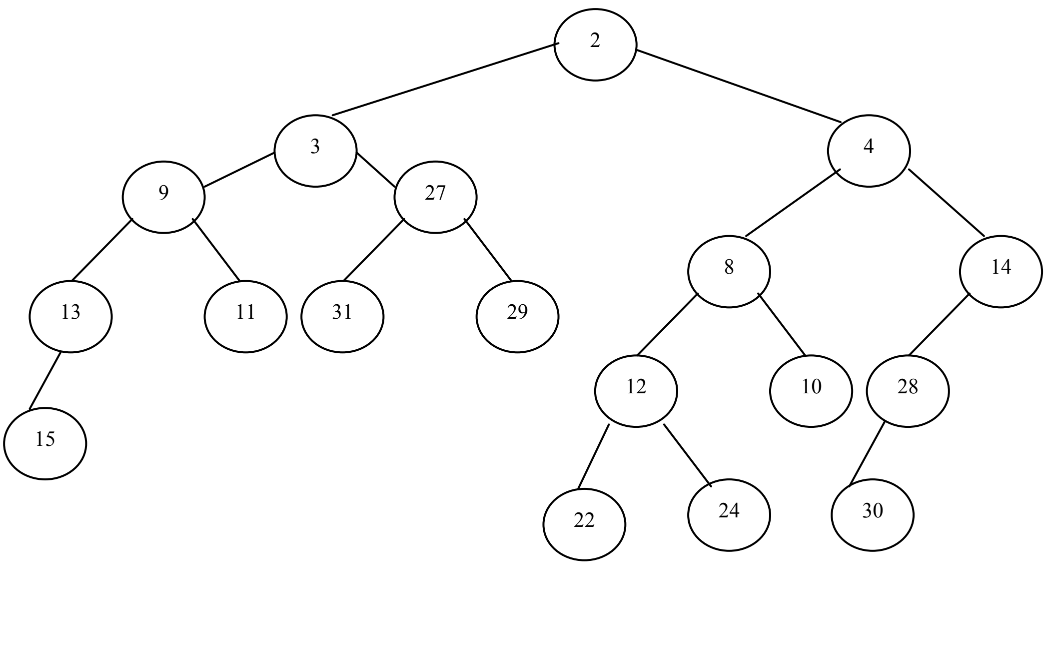


3. En un árbol B de orden **m** = 5 se insertan las claves 1, 2, 3, ..., **n**, en ese mismo orden:

a) ¿Qué claves originan la división de un nodo? ¿Cada cuántas inserciones se divide un nodo?
b) A la vista del resultado, ¿se puede generalizar el comportamiento para un árbol B de orden **m** cualquiera? ¿Cuántas divisiones se harían para un árbol B de orden m cualquiera y n inserciones de números enteros en orden creciente 1,2,3….n?
c) ¿Qué claves hacen que la altura del árbol crezca, para el caso de **m** = 5? Hallar la fórmula que obtiene las claves que hacen crecer la altura. Justificar la respuesta.

4. Dado el siguiente árbol Leftist mínimo (izquierdista mínimo):

a) Realiza el borrado del ítem mínimo.
b) Sobre el árbol leftist resultante del apartado a) inserta el ítem 1.



Examen PED julio 2009. Soluciones

1.

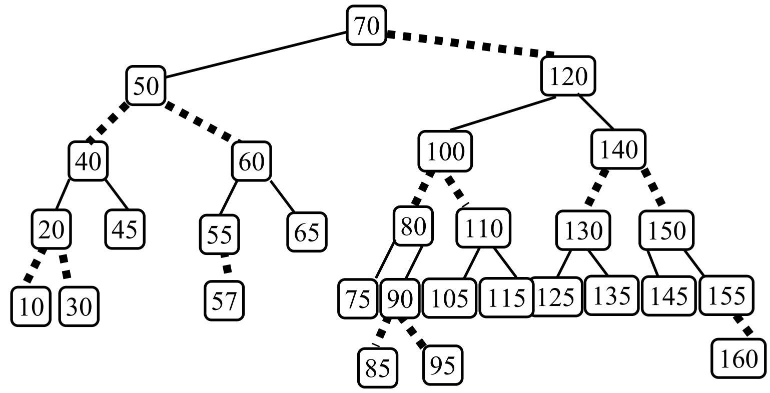
```
** Sintaxis:
SPP(lista, posición, posicion) □ lista

** Semántica:
Var L: lista; x: item; p1, p2: posición;

SPP(crear(), p1, p2) = crear()
si primera(inscabeza(L, x)) == p1 y p1 != p2  entonces
   SPP(inscabeza(L, x), p1, p2) = inscabeza(SPP(L, primera(L), p2), x)
/* También se puede escribir como:
   SPP(inscabeza(L, x), p1, p2) = inscabeza(SPP(L, siguiente(inscabeza(L, x), p1), p2), x)
*/
sino si primera(inscabeza(L, x)) == p1 entonces
   SPP(inscabeza(L, x), p1, p2) = inscabeza(crear(), x)
sino
   SPP(inscabeza(L, x), p1, p2) = SPP(L, p1, p2)
fsi
```

```
Solución (otra forma de escribir lo mismo):
** Sintaxis:
SPP(lista, posición, posicion) □ lista
SPPaux(lista, posición) □ lista

** Semántica:
Var L: lista; x: item; p1, p2: posición;

SPP(crear(), p1, p2) = crear()
si primera(inscabeza(L, x) == p1) entonces
   SPP(inscabeza(L, x), p1, p2)  = IC(SPPaux(L, p2), x)
sino
   SPP(inscabeza(L, x), p1, p2)  = SPP(L, p1, p2)
fsi

SPPaux(crear(), p2) = crear()
si primera(inscabeza(L, x) == p2) entonces
   SPPaux(inscabeza(L, x), p2)  = IC(crear(), x)
sino
   SPPaux(inscabeza(L, x), p2)  = IC(SPPaux(L, p2), x)
fsi
```

2.

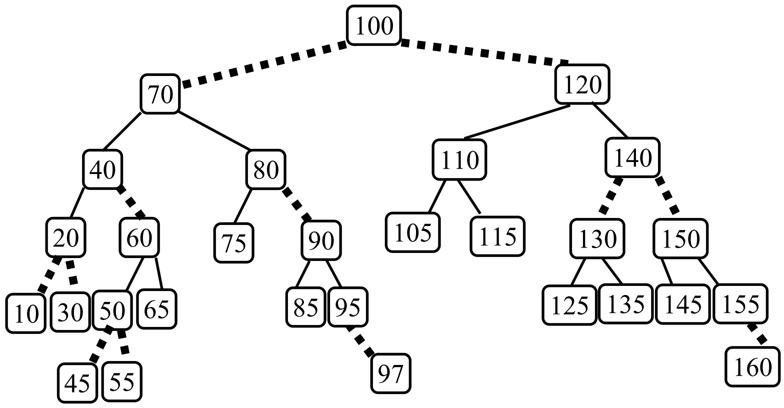
a)

Insertar 57
1 Cambio Color (45, 50, 55)
1 Rotación DI (40, 50, 60)



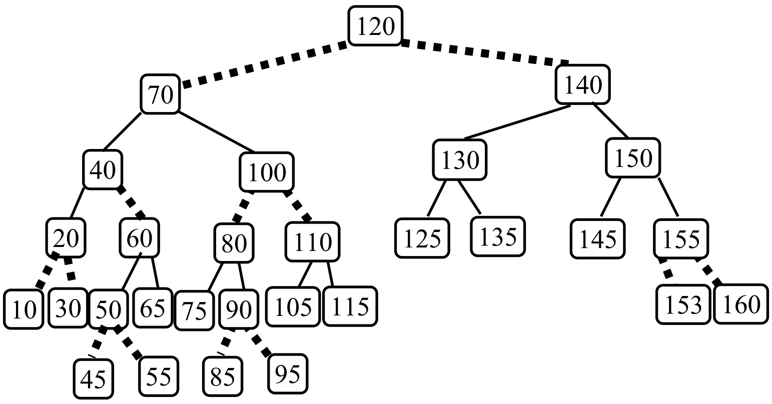
b)

Insertar 97
1 Cambio Color (80, 100, 110)
1 Rotación DI (70, 100, 120)
1 Cambio Color (85, 90, 95)

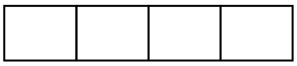


c)

Insertar 153
1 Cambio Color (130, 140, 150)
1 Rotación DD (70, 120, 140)



3.
Si el árbol es de orden m=5 quiere decir que cada nodo tendrá como máximo m-1 claves.



a)
5, 8, 11, 14, ….

m+(3*0), m+(3*1), m+(3*2) → La división i-esima se corresponde con la clave $m+\left(\left\lceil \frac{m}{2} \right\rceil * (i-1)\right)$

A partir de la quinta inserción cada $\left\lceil \frac{m}{2} \right\rceil$ inserciones se divide un nodo.

b)
Si m=5 → a partir de la quinta inserción se producen divisiones cada 3 inserciones: 5+(3*0), 5+(3*1)…

Si m=x → a partir de la inserción x se producirán divisiones cada $\left\lceil \frac{m}{2} \right\rceil$

Si m=6 (suponemos que para m=número_par la división de los nodos es: $\left\lceil \frac{m-1}{2} \right\rceil$ claves se quedan a la derecha y (m-1)/2 (parte entera de la división) claves se quedan a la izquierda). Las claves que producen divisiones son: 6, 9, 12, 15…

Si m=7. Las claves que producen divisiones son: 7, 11, 15, 19 …

Regla general: La división i-esima se corresponde con la clave $m+\left(\left\lceil \frac{m}{2} \right\rceil * (i-1)\right)$

Si tenemos n claves:

Nº de divisiones por insertar claves nuevas = $1+\dfrac{n-m}{\left\lceil \frac{m}{2} \right\rceil}$

En este caso, no se contabilizan las divisiones sucesivas debidas a la propagación de la clave intermedia hacia la raíz. Esto sucede cuando los nodos que reciben la clave intermedia también están llenos y por tanto, también deben dividirse.

Para contabilizar las divisiones debidas a la propagación de claves hacia la raíz habría que modificar la fórmula anterior de la siguiente forma:

Nº de divisiones totales $\simeq 1+\sum_{i=1}^{h-1}\dfrac{n-m}{\left\lceil \frac{m}{2} \right\rceil^{i}}$

Se debe cumplir que n-m > $\left\lceil \frac{m}{2} \right\rceil^{i}$

Nota: Se han dado por válidas cualquiera de estas dos aproximaciones.

c)
Para m=5 las claves que hacen que el árbol crezca son:
5 (h=2), 17 (h=3), 53 (h=4),…

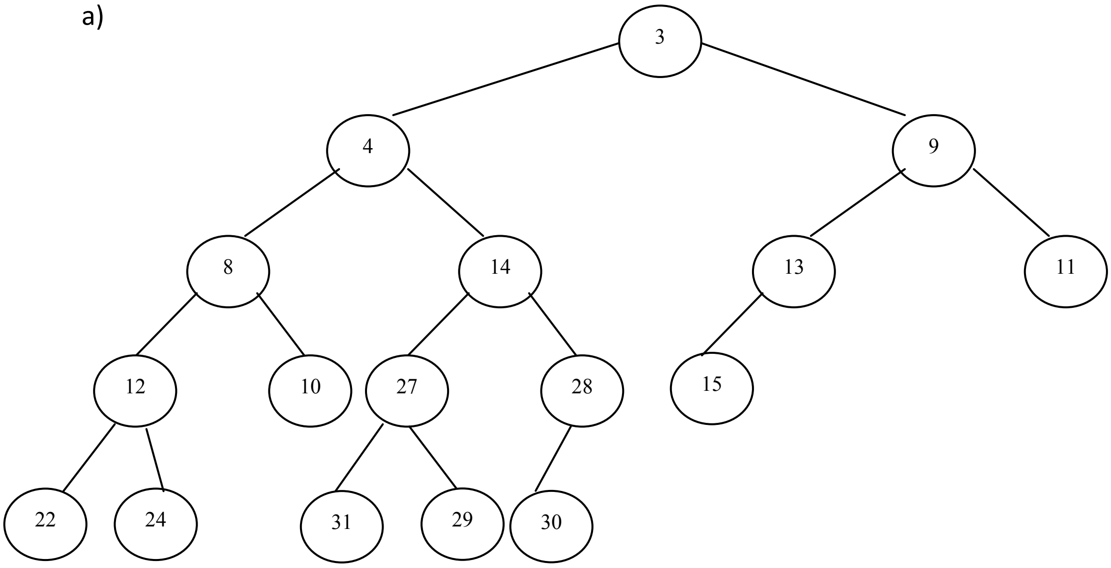
El árbol después de cada división queda con todos los nodos de tipo 3-nodo excepto la raíz, por tanto, la clave que hace aumentar la altura del árbol, se calcula de la siguiente forma:

Clave que hace aumentar la altura = $(3^{h}-1)-(3^{h-1}-1)-1$

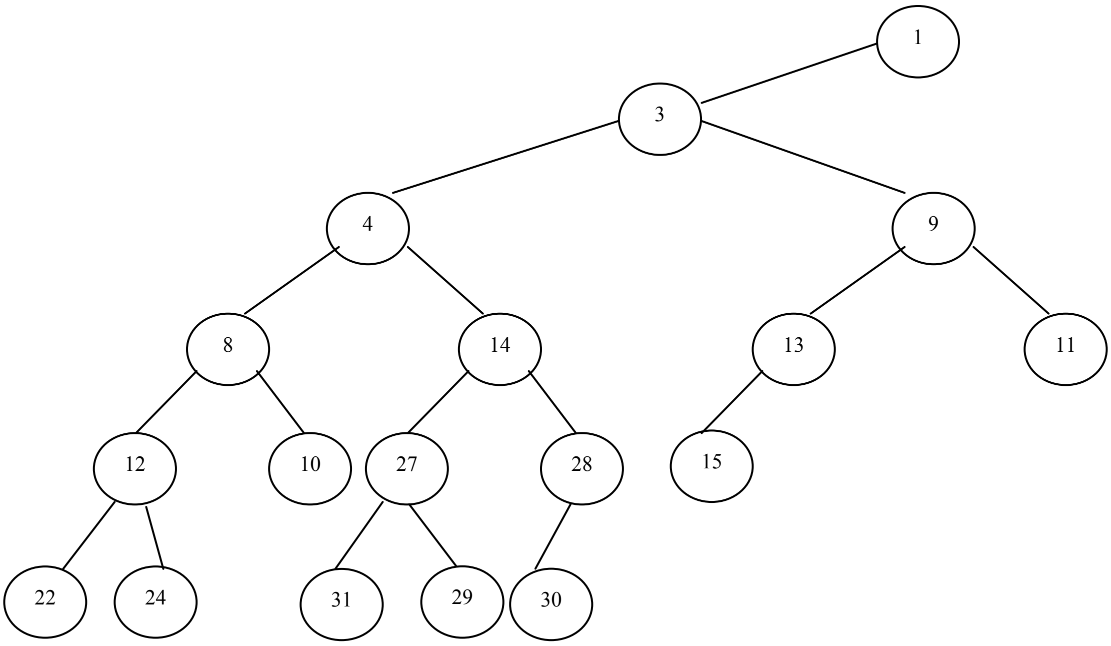
Nº de elementos si todos los nodos fueran de tipo 3-nodo = $(3^{h}-1)$

4.

a)


3
4
9
8
14
13
11
12
10
27
28
15
22
24
31
29
30

b)


1
3
4
9
8
14
13
11
12
10
27
28
15
22
24
31
29
30